Air Force Institute of Technology

AFIT Scholar

Theses and Dissertations | Student Graduate Works

3-23-2018

Integrity Monitoring for Automated Aerial Refueling: A Stereo Vision Approach

Thomas R. Stuart

Follow this and additional works at: https://scholar.afit.edu/etd

 Part of the Navigation, Guidance, Control and Dynamics Commons, and the Theory and Algorithms Commons

Recommended Citation

Stuart, Thomas R., "Integrity Monitoring for Automated Aerial Refueling: A Stereo Vision Approach" (2018). *Theses and Dissertations*. 1825.
https://scholar.afit.edu/etd/1825

This Thesis is brought to you for free and open access by the Student Graduate Works at AFIT Scholar. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of AFIT Scholar. For more information, please contact richard.mansfield@afit.edu.



INTEGRITY MONITORING FOR AUTOMATED AERIAL REFUELING

THESIS

Thomas R. Stuart, Maj, USAF

AFIT-ENG-MS-18-M-062

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views expressed in this document are those of the author and do not reflect the official policy or position of the United States Air Force, the United States Department of Defense or the United States Government. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT-ENG-MS-18-M-062

INTEGRITY MONITORING FOR AUTOMATED AERIAL REFUELING: A STEREO VISION APPROACH

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Thomas R. Stuart, B.S., M.P.P., M.S.

Maj, USAF

March 2018

DISTRIBUTION STATEMENT A
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AFIT-ENG-MS-18-M-062

INTEGRITY MONITORING FOR AUTOMATED AERIAL REFUELING: A STEREO VISION APPROACH

THESIS

Thomas R. Stuart, B.S., M.P.P., M.S.
Maj, USAF

Committee Membership:

Dr. Scott Nykl
Chair

Dr. John Raquet
Member

Dr. Sean Calhoun
Member

Abstract

Unmanned aerial vehicles (UAVs) increasingly require the capability to fly autonomously in close formation including to facilitate automated aerial refueling (AAR). The availability of relative navigation measurements and navigation integrity are essential to autonomous relative navigation. Due to the potential non-availability of the global positioning system (GPS) during military operations, it is highly desirable that relative navigation can be accomplished without the use of GPS. This paper develops two algorithms designed to provide relative navigation measurements solely from a stereo image pair. These algorithms were developed and analyzed in the context of AAR using a stereo camera system modeling that of the KC-46. Algorithms were analyzed in simulation and then in flight test using two C-12C aircraft at the United States Air Force Test Pilot School.

The first algorithm, the Vision and Bayesian Inference Based Integrity Monitor (V5), uses Bayesian inference and template matching to return a probability mass function (PMF) describing the position of an observed aircraft. This PMF provides a relative position estimate as well as a protection level—which characterizes the uncertainty of the relative position estimate—thus providing a degree of navigation integrity. Using both simulation and flight test data, mean V5 spherical error was less than one meter and protection levels reliably characterized algorithm uncertainty.

The second algorithm, relative pose estimation with computer vision and iterative closest point (R7), uses stereo vision algorithms and the iterative closest point algorithm to return relative position and attitude estimates. Using both simulation and flight test data, mean R7 spherical error was less than 0.5 meters. Additionally, in flight test, mean R7 attitude errors were less than 3° in all axes.

To my wife and daughter.

Acknowledgements

I would like to thank my advisor, Dr. Scott Nykl, for the countless hours of support and instruction he provided during the planning, research, and flight test process. His dedication and knowledge directly contributed to the successes of this research effort. I would also like to thank Dr. Sean Calhoun for the many hours he spent helping me to understand his work and how it could be adapted into this thesis, as well as for the valuable support during the flight test phase. Thanks also to Dr. John Raquet for his time and support during this research, in particular in putting me in touch with Dr. Sean Calhoun and for sharing invaluable insights garnered from his many years of navigation research and flight test experience.

This research and flight test effort would not have been possible without the support of Mr. Ba Nguyen, Mr. Dan Schreiter, and the rest of the AFRL/RQQC team. Many thanks for their sponsorship and encouragement throughout the process.

The staff of the USAF Test Pilot School ensured that the flight test portion of this research was a success. Countless hours were spent test and safety planning, designing and performing aircraft modifications, troubleshooting systems issues, coordinating for aircraft availability, etc. Their capabilities and dedication are truly remarkable, and I am indebted to everyone there who made the flight test program a success.

Thanks also to the members of the Have Vision team—Mr. Karl Major and my fellow Test Pilot School students Captain Shane Bellingham, Captain Timothy Grace, Captain Jacob Lambach, and Captain Daniel Welch—for planning and executing test program, and then analyzing the data from the flight test effort. Many hours were spent in front of a white board, in front of a computer, or in the cockpit of the mighty Huron. Without their hard work and dedication, flight testing would not have been possible.

Table of Contents

Page

Abstract . . . iv

Acknowledgements . . . v

List of Figures . . . xiii

List of Tables . . . xxiii

I. Introduction . . . 1

1.1 Problem Statement . . . 3
1.2 Scope of Analysis . . . 3
1.3 Overview . . . 4

II. Background . . . 7

2.1 Vision-Aided AAR Research . . . 7
2.1.1 Fravolini et al's Simulation Environment for Vision-Aided AAR . . . 7
2.1.2 Wilson et al's Unscented Kalman Filter . . . 10
2.1.3 Image Rendering Approaches to AAR . . . 11
2.1.3.1 Howard and Veth's Image Rendering Approach . . . 11
2.1.3.2 Calhoun, Curro, and Raquet's Image Rendering Flight Test . . . 14
2.1.4 Navigation Integrity Definitions . . . 17
2.1.5 Calhoun, Raquet, and Peterson's Vision-Based, Binary Hypothesis Test Integrity Monitor . . . 18
2.2 Calhoun's Bayesian Inference Integrity Monitor . . . 25
2.2.1 Overview of Calhoun's Bayesian Inference Integrity Monitor . . . 26
2.2.2 Image Processing . . . 30
2.2.2.1 Gradient-Based Edge Detection . . . 31
2.2.2.2 Prewitt Edge Detection . . . 31
2.2.2.3 Gaussian Blurring . . . 32
2.2.3 Template Matching . . . 34
2.2.4 Template Matching and Bayesian Inference . . . 36
2.2.5 Likelihood Function Determination . . . 37
2.2.6 Likelihood Score Computation . . . 39
2.2.6.1 Log-Likelihood Computation . . . 40
2.2.7 Posterior Probability Computation . . . 41
2.2.8 Protection Level Computation . . . 41

Page

2.2.9 Calhoun's Results . . . 43
2.3 Camera Model and Computer Vision . . . 46
2.3.1 Camera Coordinate Frame . . . 47
2.3.2 Pinhole Camera Model . . . 48
2.3.3 Normalized and Pixel Coordinates . . . 50
2.3.4 Intrinsic Camera Calibrations . . . 54
2.3.5 Extrinsic Camera Calibrations . . . 56
2.3.5.1 Direct Measurement . . . 58
2.3.5.2 Checkerboard Image Based Extrinsic Camera Calibration . . . 58
2.3.6 Stereo Vision . . . 59
2.3.6.1 Epipolar Geometry . . . 59
2.3.6.2 The Essential and Fundamental Matrices . . . 62
2.3.6.3 Stereo Camera Calibration . . . 65
2.3.6.4 Stereo Image Rectification . . . 68
2.3.6.5 Pixel Matching and Disparity Map Generation . . . 72
2.3.6.6 Obtaining Depth Information from Disparities . . . 77
2.3.6.7 Relating Points to an Arbitrary Frame . . . 81
2.3.6.8 Stereo Vision Summary . . . 82
2.4 The Iterative Closest Point Algorithm . . . 83

III. Methodology . . . 88

3.1 Frame and Coordinate System Definitions . . . 88
3.2 Variable Definitions and Notation . . . 91
3.3 Simulation Environment . . . 94
3.4 Vision & Bayesian Inference Based Integrity Monitor (V5) . . . 96
3.4.1 V5 Algorithm Overview . . . 96
3.4.2 Reference Image Database Generation . . . 101
3.4.3 Image Processing . . . 102
3.4.3.1 Prewitt Edge Detection . . . 103
3.4.3.2 Disparity Map Generation . . . 103
3.4.3.3 Depth Filter . . . 104
3.4.3.4 Gaussian Blurring . . . 105
3.4.3.5 Image Processing Block Diagram . . . 105
3.4.4 Likelihood Function Determination . . . 106
3.4.5 Likelihood Score Computation . . . 108
3.4.6 Probability Mass Function Computation . . . 109
3.4.7 Relative Position Estimation . . . 111
3.4.8 Protection Level Computation . . . 112

Page

3.5 Relative Pose Estimation with Computer Vision and ICP (R7) . . . 113
3.5.1 R7 Algorithm Overview . . . 114
3.5.2 Model Point Cloud Development . . . 115
3.5.3 Point Cloud Generation with OpenCV . . . 117
3.5.3.1 Stereo Camera Calibration . . . 119
3.5.3.2 Extrinsic Camera Calibration . . . 120
3.5.3.3 Image Capture . . . 121
3.5.3.4 Image Rectification . . . 121
3.5.3.5 Pixel Matching and Disparity Map Generation . . . 122
3.5.3.6 Projection into 3D . . . 123
3.5.3.7 Relate to v-frame . . . 123
3.5.4 Observed Point Cloud Filtering . . . 124
3.5.4.1 Boom Filtering . . . 125
3.5.4.2 Median Filtering . . . 130
3.5.4.3 Point Cloud Denoising . . . 132
3.5.4.4 Point Cloud Downsampling . . . 133
3.5.5 ICP Implementation . . . 133
3.5.5.1 Obtaining Measurements from ICP . . . 134
3.5.5.2 Model Point Cloud Positioning . . . 135
3.5.5.3 Implementation in MATLAB® . . . 136
3.5.5.4 Transforming ICP Outputs . . . 137
3.5.5.5 Initial Transformation for ICP . . . 138
3.5.5.6 Attitude Fed ICP . . . 140
3.6 Error Computations . . . 140
3.7 Tests of Statistical Significance . . . 141
3.7.1 One-Sample and Paired t-tests . . . 142
3.7.2 Two-Sample t-tests . . . 142
3.7.3 Two-Sample F-tests . . . 143

IV. Simulation Data and Analysis . . . 144

4.1 V5 Data and Analysis, Simulation . . . 145
4.1.1 Reference Image Database Generation . . . 145
4.1.2 Reference Image Database Attenuation . . . 146
4.1.3 Observation Image Generation . . . 147
4.1.4 Gaussian Blur Level Determination . . . 148
4.1.5 Likelihood Function Determination . . . 152
4.1.6 Pixel Intensity Threshold Determination . . . 159
4.1.7 V5 Simulation Results . . . 160
4.1.7.1 Relative Position Estimator Comparison . . . 162
4.1.7.2 Integrity Risk Level Effect . . . 168
4.1.7.3 Thresholding Comparison . . . 169

Page

4.1.7.4 Consequences of Thresholding . . . 174
4.1.7.5 Single Camera Versus Stereo Camera Results . . . 179
4.1.7.6 Prior Probability Distribution Comparison . . . 183
4.1.7.7 V5 Simulation Processing Times . . . 188
4.2 R7 Data and Analysis, Simulation . . . 190
4.2.1 Observation Image Generation . . . 190
4.2.2 R7 Simulation Results . . . 191
4.2.2.1 Case 1: No Boom in Field of View, No Attitude Feeding . . . 193
4.2.2.2 Case 2: No Boom in Field of View, Attitude Feeding . . . 201
4.2.2.3 Boom Filtering . . . 207
4.2.2.4 Case 3: Boom in Field of View, No Attitude Feeding . . . 209
4.2.2.5 Case 4: Boom in Field of View, Attitude Feeding . . . 217
4.2.2.6 Case 5: V5 Data Set . . . 221
4.2.2.7 R7 Simulation Processing Times . . . 225
4.3 R7 and V5 Comparison . . . 225

V. Flight Test Hardware and Methodology . . . 229

5.1 Have Vision Test Management Project Summary . . . 229
5.2 Flight Test Equipment . . . 230
5.2.1 LWIR Cameras . . . 232
5.2.2 EO Cameras . . . 232
5.2.3 Data Acquisition Computer . . . 233
5.2.4 Geodetics Geo-RelNav® System . . . 234
5.2.5 GLITE TSPI System . . . 235
5.2.6 Test Aircraft . . . 236
5.2.7 Have Vision System Block Diagram . . . 236
5.2.8 Hardware Limitations and Constraints . . . 237
5.3 Flight Test Data Collection Methodology . . . 238
5.3.1 Calibrations . . . 239
5.3.1.1 Extrinsic Camera Calibrations and System Boresighting . . . 240
5.3.1.2 Intrinsic Camera Calibrations . . . 240
5.3.1.3 Stereo Camera Calibrations . . . 242
5.3.2 Simulated Aerial Refueling In-Flight Methodology . . . 243
5.4 Flight Test Data Analysis Methodology . . . 247
5.4.1 Boresight Data Reduction . . . 247

Page

5.4.2 Truth Data Reduction . . . 249
5.4.3 Truth Data Uncertainty . . . 250
5.4.4 Relating Camera Data to the p-Frame . . . 253
5.4.5 V5 Algorithm Considerations . . . 253
5.4.5.1 Reference Image Database Generation . . . 253
5.4.5.2 Image Rectification and Scaling . . . 254
5.4.6 R7 Algorithm Considerations . . . 256
5.4.6.1 Shell Model Point Cloud . . . 256
5.4.6.2 Miscellaneous Data Processing Changes and Example . . . 257
5.4.7 Partial Autocorrelation Function and Data Decimation . . . 259

VI. Flight Test Data and Analysis . . . 261

6.1 Truth Data Resolution . . . 261
6.2 R7 Data and Analysis, Flight Test . . . 262
6.2.1 Data Set Decimation . . . 263
6.2.2 Case 1: No Attitude Feeding . . . 266
6.2.3 Case 2: Attitude Feeding . . . 273
6.2.4 R7 Flight Test Processing Times . . . 276
6.3 V5 Data and Analysis, Flight Test . . . 277
6.3.1 Case 1: Uniform Prior . . . 278
6.3.1.1 Flight Test PMF Analysis . . . 282
6.3.1.2 Test Case . . . 283
6.3.1.3 Bias Analysis . . . 284
6.3.1.4 Processing Times . . . 289
6.3.1.5 Integrity Risk Level Effect . . . 290
6.3.1.6 Maximum Likelihood Estimator . . . 290
6.3.2 Case 2: Gaussian Prior . . . 292
6.4 R7 and V5 Comparison . . . 296

VII. Conclusions and Recommendations . . . 299

7.1 V5 Algorithm Contributions and Recommendations . . . 299
7.1.1 Image Processing . . . 300
7.1.2 PMF Estimation . . . 301
7.1.3 Relative Position Estimates and Protection Level Computations . . . 301
7.1.4 Pixel Intensity Thresholds and Protection Level Computations . . . 302
7.1.5 Informative Priors . . . 303
7.1.6 Rectification Mapping and Flight Test Errors . . . 303
7.1.7 Algorithm Processing Times . . . 304
7.2 R7 Algorithm Contributions and Recommendations . . . 305

Page

7.2.1 Depth Error . . . 306
7.2.2 Attitude Fed ICP . . . 306
7.2.3 Boom Filtering . . . 307
7.2.4 Flight Test Attitude Errors . . . 307
7.2.5 R7 Error Variance . . . 308
7.2.6 R7 Processing Times . . . 308
7.3 Future Work: Leveraging ICP and Bayesian Inference . . . 309

Appendix A. Camera Calibration Results . . . 313

Appendix B. PACF Returns . . . 315

Appendix C. R7 Error Characterization . . . 316

Bibliography . . . 319

List of Figures

Figure Page

1 Hypothetical AAR System Block Diagram. 2

2 Calhoun's Depiction of Operating and Alert Regions. The Operating Region Falls Within the Blue Refueling Envelope, the Alert Region Falls Outside the Red Safety Boundary but Within the Green Validity Region. [1]. 19

3 Calhoun's Possible Results from the Binary Hypothesis Test [1]. 19

4 Calhoun's PDFs for the Silhouette Correspondence Metric; $p(H_0)$ is Depicted in Blue, $p(H_1)$ is Depicted in Red [1]. 21

5 Calhoun's PDFs for the SSD Correspondence Metric; $p(H_0)$ is Depicted in Blue, $p(H_1)$ is Depicted in Red [1]. 21

6 Calhoun's Depiction of P_{MD} and P_{FA} [1]. 22

7 Calhoun's ROC Curves for Both Image Correspondence Techniques Used in [1]. 22

8 Calhoun's Simulation Results with Observation Images Included from the Region Between the Refueling Envelope and Safety Boundary. (a) Silhouette PDFs, (b) GRD PDFs, (c) Silhouette ROC Curve, (d) GRD ROC Curve [1]. 24

9 Calhoun's ROCs for the Ratio Tests Compared to the Baseline Tests [1]. 25

10 Calhoun's Zoom-in of the ROC curve for the SIL Ratio Test [1]. 25

11 Representation of Pixels Used in Prewitt Edge Detector. 32

12 Calhoun's Gaussian fit for the pixel intensity difference likelihood function $p(g_b)$ or $p(\theta|\mathbf{x}_j)$ [2]. 38

13 Calhoun's Results for 10^{-6} Integrity Risk Level, 100 Sample Pixels, and Uniform Prior [2]. 44

Figure Page

14 Calhoun's Results for 0.05 Integrity Risk Level, 20 Sample Pixels, and Uniform Prior [2]. . . . 45

15 USAF Test Pilot School's EO Sensor Model [3]. . . . 47

16 Camera Coordinate Frame . . . 48

17 Pinhole Camera Model . . . 49

18 Raquet's Depiction of the Relationship Between Pixel and Normalized Coordinates [4] . . . 51

19 Extrinsic Camera Calibration Parameters. . . . 57

20 Depiction of Epipolar Geometry. . . . 60

21 Depiction of Frontal Parallel Cameras. . . . 69

22 Depiction of Stereo Camera Image Rectification. . . . 71

23 Left and Right Images Used to Generate the Disparity Map Shown in Figure 24. . . . 77

24 Disparity Map Generated from the Images in Figure 23. . . . 77

25 Relevant Formation-Level Coordinate Frames. . . . 90

26 Example Snapshot Taken of the 3DVW. . . . 95

27 Block Diagram Depicting the V5 Algorithm. . . . 101

28 Depiction of the Rendered Image Database. . . . 102

29 Depiction of the Image Processing Method Used in this Thesis. . . . 106

30 Random Pixel Sampling Example Shown For an Observation Image and the Reference Image that is Closest to Depicting the Same Relative Position. Both Images Were Generated in the 3DVW and Have Been Transformed to Edge Space and Been Blurred with a 5σ Gaussian Blur. . . . 107

31 Block Diagram of the R7 Algorithm. . . . 115

32 Model Point Cloud Used with Simulation Images. . . . 117

Figure Page

33 Depiction of the v-Frame and the B-Frame. Following the 3-2-1 Convention, Boom Elevation is the Required Rotation About the v-Frame's y-Axis and Boom Azimuth is the Required Rotation About the Resultant x-Axis to Align the Two Frames. 127

34 Depiction of the Tanker Boom and the Parameters Defining the Boom Filter. 129

35 Example of a Nuisance Point Cluster Manifesting After Model Point Cloud Generation. 131

36 From Left to Right, Examples of a 3DVW Left Camera Reference Image and Right Camera Reference Image. 146

37 Examples of a the Attenuated Reference Image Database for a Position Near the Center of the Database and for a Position Near the Boundary of the Database. 147

38 From Left to Right, Examples of a 3DVW Left Camera Observation Image and Right Camera Observation Image. 148

39 Histogram of Pixel Distance from all Edges in all Observation Images to the Nearest Edge Pixel in the Best Match from the Rendered Database. Data Depicted as Counts in Each Bin. 149

40 Histogram of Pixel Distance from all Edges in all Observation Images to the Nearest Edge Pixel in the Best Match from the Rendered Database. Data Depicted as a CDF at Each Bin. 149

41 Plots Depicting Mean or Median Edge Separation Distance of all Edges in all Observation Images to the Nearest Edge Pixel in the Best Match from the Reference Image Database. Data was generated when Using a "Spherical" Reference Image Database. 150

42 Plots Depicting Mean or Median Edge Separation Distance of all Edges in all Observation Images to the Nearest Edge Pixel in the Best Match from the Rendered Database. Data was generated when Using a "Square" Reference Image Database. 151

Figure Page

43 Candidate Fits for a Likelihood Function. The Histogram Depicts the Intensity Differences for the Entire Ensemble of Observation-Reference Image Pairs. 153

44 Simulation Results with Overbounded Gaussian Likelihood Function. 156

45 Simulation Results with Best Fit Gaussian Likelihood Function. 156

46 Simulation Results with Overbounded Cauchy Likelihood Function. 157

47 Simulation Results with Overbounded Laplacian Likelihood Function. 157

48 Pixel Intensity Difference Distributions for Two Sample Reference-Observation Image Pairs and an Ensemble of 150 Reference-Observation Image Pairs. A Total of 100 Pixels Were Randomly Sampled from Each Pair. 159

49 CDF of Pixel Intensities for 150 Processed Observation Images. 160

50 Spherical Error for the V5 Algorithm Using the Maximum Likelihood Estimator. 164

51 Spherical Error for the V5 Algorithm Using the Composite Position Estimator. 164

52 PMF Distribution Computed with a Pixel Intensity Threshold of Zero. 167

53 PMF Distribution Computed with a Pixel Intensity Threshold of Zero, Side and Top-Down Views. 167

54 Protection Level Comparison with Different Integrity Risk Levels. 169

55 Spherical Error When Using a Pixel Intensity Threshold of 0. 171

56 Spherical Error When Using Using a Pixel Intensity Threshold of 0.05. 171

Figure Page

57 PMF Distribution Computed with a Pixel Intensity Threshold of 0.05. 175

58 PMF Distribution Computed with a Pixel Intensity Threshold of 0.05, Side and Top-Down Views. 176

59 Composite Position Estimator Errors, Tanker Frame. 177

60 Composite Position Estimator Errors, Left Camera Frame. 177

61 Composite Position Estimator Errors Sorted by z-Position, Left Camera Frame. 178

62 Spherical Error When Using Only the Left Camera PMF. 182

63 Spherical Error When Using Only the Right Camera PMF. 182

64 Spherical Error When Combining Camera PMFs. 183

65 Spherical Error When Using a Uniform Prior Probability Distribution. 185

66 Spherical Error When Using Gaussian Prior Probability Distribution A. 185

67 Spherical Error When Using Gaussian Prior Probability Distribution B. 186

68 Time to Execute the V5 Algorithm on Pre-Processed Observation Images. 189

69 Depiction of 6,000 Randomly Generated Positions in Yellow and the Camera Fields of View in Red. 191

70 Spherical Position Errors, Case 1. 194

71 Spherical Position Errors, Case 1, Zoom In. 194

72 Position Errors, Case 1, Left Camera Frame. 195

73 Position Errors, Case 1, Left Camera Frame, Zoom In. 196

74 Spherical Position Error and Disparity Values. 198

Figure Page

75 Position Errors, Left Camera Frame, and Disparity Values. . . . 199

76 Attitude Errors, Case 1, Tanker Frame. . . . 200

77 Attitude Errors, Case 1, Tanker Frame, Zoom In. . . . 200

78 Spherical Position Errors, Case 2. . . . 202

79 Spherical Position Errors, Case 2, Zoom In. . . . 202

80 Position Errors, Case 2, Left Camera Frame. . . . 203

81 Position Errors, Case 2, Left Camera Frame, Zoom In. . . . 203

82 Position Errors, Cases 1 and 2, Left Camera Frame. . . . 205

83 Position Errors, Cases 1 and 2, Left Camera Frame, Zoom In. . . . 206

84 Effect of Boom Filtering, Observed Point Cloud Shown in Red, R7 Solution Shown in Blue. . . . 208

85 Spherical Position Errors, Boom in Field-of-View, No Boom Filter Applied. . . . 209

86 Spherical Position Errors, Case 3. . . . 210

87 Spherical Position Errors, Case 3, Zoom In. . . . 211

88 Spherical Position Errors, Cases 1 and 3. . . . 211

89 Spherical Position Errors, Cases 1 and 3, Zoom In. . . . 212

90 Simulation Images Captured on the Extreme Outlier. . . . 212

91 Boom Filtering Applied to the Outlier Point Cloud. . . . 213

92 Position Errors, Case 3, Left Camera Frame. . . . 214

93 Position Errors, Case 3, Left Camera Frame, Zoom In. . . . 215

94 Attitude Errors, Case 3, Tanker Frame. . . . 215

95 Attitude Errors, Case 3, Tanker Frame, Zoom In. . . . 216

96 Spherical Position Errors, Case 4. . . . 218

Figure Page

97 Spherical Position Errors, Case 4, Zoom In. 219

98 Position Errors, Case 4, Left Camera Frame. 219

99 Position Errors, Case 4, Left Camera Frame, Zoom In. 220

100 Spherical Error, R7 Applied to the V5 Simulation Data. 222

101 Position Errors, Left Camera Frame, R7 Applied to the V5 Simulation Data. 223

102 Attitude Errors, Tanker Frame, R7 Applied to the V5 Simulation Data. 224

103 Comparison of R7 and V5 Spherical Errors. 226

104 Comparison of R7 and V5 Processing Times. 228

105 Have Vision Camera Configuration, View from Aft of Aircraft. 231

106 Have Vision Camera Configuration, View from Side of Aircraft. 232

107 Stereo Camera Illustration and Field of View on Pseudo-Tanker Aircraft. 233

108 Close Up of the Have Vision Stereo Camera System. 234

109 Block Diagram of Flight Test Hardware and Data Processing Flow. 237

110 From Left to Right, EO and LWIR Images of the Checkerboard Used for Camera Calibrations. 241

111 Images from EO Cameras Used in Intrinsic and Stereo Camera Calibrations. Images Were Down-Sampled to Create this Figure. 242

112 Side View of Contact Position Used in Flight Test. 244

113 Formation Positions Flown During Flight Test (Not to Scale). 245

114 View from the Contact Position. 246

Figure Page

115 Background Environments Captured During Flight Test. 247

116 Experimental Results Used to Determine 95% Confidence Interval of Mean Truth Data System Spherical Error. 253

117 Methodology Used to Relate Reference Image Database to GLITE TSPI Data. 255

118 Shell Model Point Cloud Used with Flight Test Data. 257

119 Image Pair and Resultant Point Cloud Generated in 3DVW Viewed from Front and Side. 258

120 At Left, the Raw Point Cloud is Shown in Red. At Right, Blue Circles Depict the Raw Point Cloud and the Red Dots Depict the Results After Filtering. 258

121 From Left To Right, the ICP Solution and an Error Visualization. At Left, the Red Dots Depict the Raw Point Cloud and the Blue Circles Depict the Model Point Cloud. At Right, Red Dots Depict the R7 Estimate and the Blue Circles Depict the True Solution. 259

122 R7 Spherical Error in Flight Test, All Data Points, No Attitude Feeding. 264

123 R7 Spherical in Flight Test, All Data Points, No Attitude Feeding. 264

124 R7 Position Errors in Flight Test, All Data Points, No Attitude Feeding, Left Camera Frame. 265

125 R7 Position Errors in Flight Test, All Data Points, No Attitude Feeding, Left Camera Frame. 265

126 Partial Autocorrelation Function Results, Z Position Error, Left Camera Frame. 266

127 R7 Spherical in Flight Test, Decimated Data Set, No Attitude Feeding. 267

128 R7 Position Errors in Flight Test, Decimated Data Set, No Attitude Feeding, Left Camera Frame. 268

Figure Page

129 At Left, Left Camera Frame Depths for the Non-Decimated Data Set. At Right, the Ratio of True Depth to Estimated Depth for the Non-Decimated Data Set. . . . 270

130 R7 Attitude Errors in Flight Test, Decimated Data Set, Tanker Frame. . . . 271

131 R7 Spherical in Flight Test, Decimated Data Set, Attitude Feeding. . . . 274

132 R7 Position Errors in Flight Test, Decimated Data Set, Attitude Feeding, Left Camera Frame. . . . 276

133 V5 Spherical Position Error on Flight Test Data Using a Uniform Prior. . . . 279

134 V5 Position Error on Flight Test Data Using a Uniform Prior, Left Camera Frame. . . . 280

135 V5 Position Error on Flight Test Data Using a Uniform Prior, Zoomed In, Left Camera Frame. . . . 281

136 Example V5 PMF and V5 Position Estimates Returned from Flight Test Data. . . . 283

137 At Left, Reference Image Corresponding to the Observation Image Shown at Right. . . . 285

138 True Best Match Reference Images Superimposed Onto Observation Images. Reference Images are Shown in Red, Observation Images are Shown in Blue. . . . 286

139 Reference Images Most Closely Depicting the Composite Position Estimate Superimposed Onto Observation Images. Reference Images are Shown in Red, Observation Images are Shown in Blue. . . . 286

140 Reference Images for the Maximum Likelihood Estimate Superimposed Onto Observation Images. Reference Images are Shown in Red, Observation Images are Shown in Blue. . . . 287

141 V5 Processing Time on Pre-Processed Flight Test Images. . . . 289

Figure Page

142 Integrity Risk Level Effect on the V5 Protection Level in Flight Test. . . . 290

143 V5 Spherical Position Error on Flight Test Data Using a Uniform Prior and the Maximum Likelihood Estimator. . . . 291

144 V5 Spherical Position Error on Flight Test Data Using a Gaussian Prior. . . . 293

145 V5 Position Error on Flight Test Data Using a Gaussian Prior, Left Camera Frame. . . . 295

146 V5 Position Error on Flight Test Data Using a Gaussian Prior, Zoomed In, Left Camera Frame. . . . 296

147 Comparison of R7 and V5 Spherical Errors in Flight Test, Left Camera Frame. . . . 298

148 Hypothetical AAR System Block Diagram. . . . 310

149 From Top to Bottom, PACF Results for X, Y, and Z Position Errors, Left Camera Frame. . . . 315

150 From Top to Bottom, PACF Results for Roll, Pitch, and Yaw Attitude Errors, Tanker Frame. . . . 315

List of Tables

Table		Page
1	Frames and Their Corresponding Superscripts and Subscripts.	92
2	Variable Definitions. Vectors and Matrices are Bolded. Point Cloud Variables are Arrays Comprised of Columns of Three-Dimensional Coordinate Vectors Wherein Each Column is a Distinct Point.	93
3	R7 Implementation with OpenCV and MATLAB®.	119
4	Overview of Camera Calibration Routines. Items Marked with a † Are Quantities That Must Be Provided as Estimates. Items Marked with a * Are Optional Outputs. Deducible Outputs Are Not Directly Returned but Can Be Estimated from the Outputs.	121
5	Boom Filter Parameterization. For Simulation Data, L_4 Was Set to Equal the Total Length of the Boom, as Rendered, in Meters. At Full Extension, $L_4 = 18$ meters.	130
6	Configuration of the V5 Algorithm Used to Compare Likelihood Functions.	154
7	Candidate Likelihood Function Spherical Errors and Integrity Violations for a 0.05 Integrity Risk Level.	155
8	Baseline Configuration of the V5 Algorithm Identified for Use on Flight Test Data.	162
9	Configuration of the V5 Algorithm Used for Position Estimator Comparison.	163
10	Maximum Likelihood Estimator and Composite Position Estimator Errors, Left Camera Frame.	166
11	Configurations of the V5 Algorithm Used for Integrity Risk Level Analysis.	168
12	Configurations of the V5 Algorithm Used for Pixel Intensity Threshold Analysis.	170
13	Threshold Comparison, Composite Position Estimator Errors, Protection Levels, Left Camera Frame.	174

Table Page

14 Configuration of the V5 Algorithm Used to Compare Single Camera PMF and Combined Camera PMF Performance. . . . 179

15 Comparison of V5 Algorithm Errors When Using a the Left Camera PMF, the Right Camera PMF, and the Combined PMF. . . . 181

16 Configurations of the V5 Algorithm Used for Prior Probability Distribution Analysis. . . . 183

17 Prior Probability Distribution Comparison, Composite Position Estimator Errors, Left Camera Frame, Pixel Intensity Threshold of 0.05. . . . 188

18 R7 Simulation Cases. . . . 192

19 Position Errors in the Left Camera Frame and Attitude Errors in the Tanker Frame for Samples at Less than 40 Meters of Depth, Case 1. . . . 201

20 Position Errors in the Left Camera Frame for Samples at Less than 40 Meters of Depth, Cases 1 and 2. . . . 206

21 Position Errors in the Left Camera Frame and Attitude Errors in the Tanker Frame for Samples at Less than 40 Meters of Depth, Cases 1 and 3. . . . 217

22 Position Errors in the Left Camera Frame, Cases 3 and 4. . . . 221

23 Position Errors in the Left Camera Frame, Attitude Errors in the Tanker Frame, R7 Applied to V5 Data Set. . . . 224

24 Configuration of the V5 Algorithm Used for Simulation Comparison with the R7 Algorithm. . . . 225

25 Comparison of V5 and R7 Position Errors, Left Camera Frame. . . . 227

26 Difference in Processed Image Sizes Between Reference Images and Flight Test Observation Images 255

27 Truth Data System Confidence Intervals, GLITE TSPI System. . . . 262

Table Page

28 Flight Test Results for the R7 Algorithm Using the 221 Decimated Samples. . . . 272

29 Mean Error 95% Confidence Intervals for the R7 Algorithm on Flight Test Data. Intervals Indicated with a "*" Overlap with the 95% Confidence Interval of the Truth Data Mean Error. . . . 273

30 Configurations of the V5 Algorithm Used on Flight Test Data. . . . 278

31 Flight Test Results for the V5 Algorithm Using the 221 Decimated Samples, Left Camera Frame. . . . 281

32 Mean Error and Mean Protection Level 95% Confidence Intervals for the V5 Algorithm on Flight Test Data, Left Camera Frame. Intervals Indicated with a "*" Overlap with the 95% Confidence Interval of the Truth Data Mean Error. . . . 282

33 Comparison of Maximum Likelihood Estimator and Composite Position Estimator Errors, Left Camera Frame, Flight Test Data. . . . 292

34 Statistical Comparison of R7 and V5 Error Distributions, Flight Test. . . . 298

35 V5 Algorithm Simulation and Flight Test Results, Left Camera Frame. . . . 304

36 Configuration of the V5 Algorithm for Data Shown in Table 35. . . . 305

37 R7 Algorithm Case 5 Simulation and Flight Test Results, Left Camera Frame. . . . 309

INTEGRITY MONITORING FOR AUTOMATED AERIAL REFUELING:
A STEREO VISION APPROACH

I. Introduction

This thesis and the associated research at the Air Force Institute of Technology (AFIT), the Air Force Research Laboratory (AFRL), and the United States Air Force (USAF) Test Pilot School seek to demonstrate the feasibility of using computer vision algorithms to develop a robust relative navigation capability for the purpose of automated aerial refueling (AAR). The key features of the envisioned vision-based relative navigation system include (1) the system is passive, (2) the system need not be reliant on the Global Positioning System (GPS) constellation, and (3) the system is deliberately designed to leverage the camera system implemented on the KC-46.

An AAR capability is highly desirable from a strategic standpoint for a number of reasons. Primarily, the ability to refuel unmanned aerial vehicles (UAVs) would enable the fielding of more capable UAV systems. These systems would be able to fly longer duration missions, would be capable of flying more fuel intensive missions (such as those requiring the ability to rapidly maneuver in relation to threat systems), and could carry greater payloads. Additionally, an AAR capability with no GPS dependency would provide valuable support during contested, degraded operations (CDO). Overall, a reliable and routinized AAR capability would strengthen the combat power available to combatant commanders. Vision-based relative navigation systems have the potential to provide this capability in GPS-denied and CDO environments.

The key components of a hypothetical vision-based AAR system are depicted in Figure 1. This thesis addressed a subset (the pose estimation and state measurement

components which are outlined in red) of the overall system depicted. The system is iterative and begins with image capture. Captured images are used to obtain a relative position and/or attitude measurement with an associated uncertainty. These measurements are fused with measurements from other systems, such as inertial measurement units (IMUs), in a Kalman filter (or other recursive estimation algorithm) to obtain an updated relative navigation state estimate. This result is fed into the formation control laws for either or both aircraft, which results in a new relative navigation state between the tanker and UAV. At this point, the process continues with the next image capture.

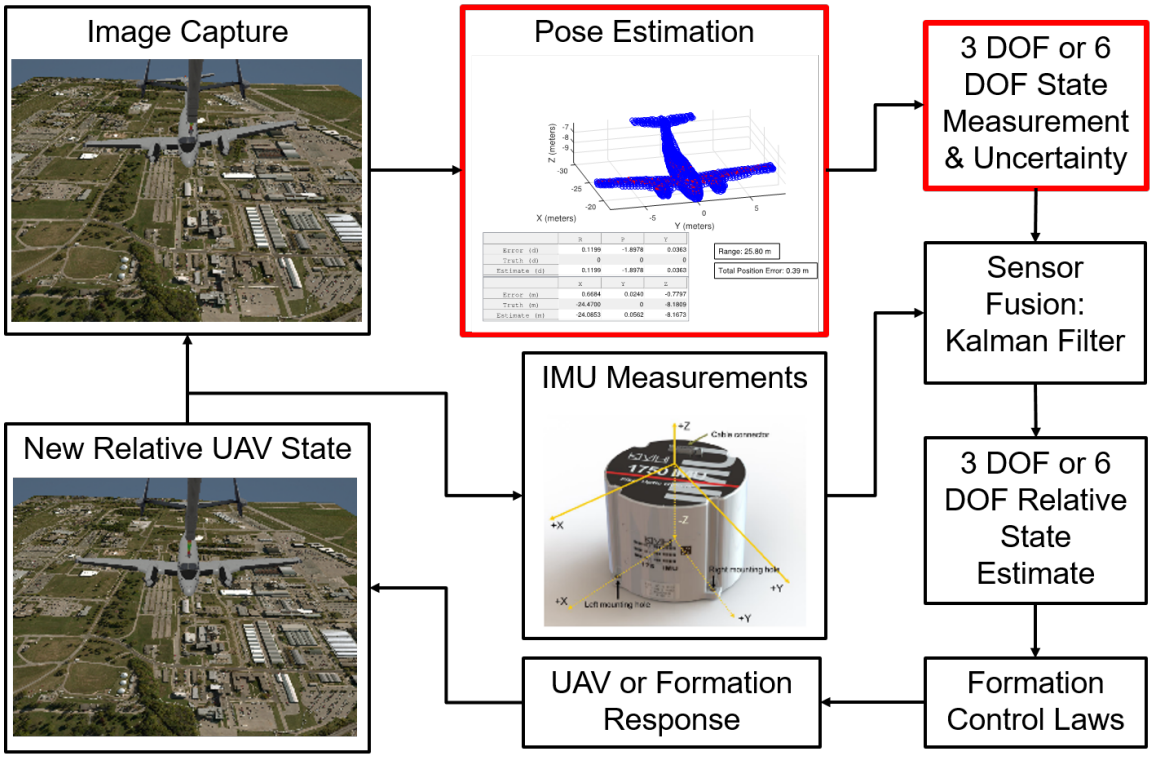


Figure 1. Hypothetical AAR System Block Diagram.

1.1 Problem Statement

This thesis developed two algorithms designed to provide the relative pose estimates depicted in Figure 1. The first algorithm, the Vision and Bayesian Inference Based Integrity Monitor (V5), uses a stereo image pair to estimate a probability mass function (PMF) describing the relative position of a receiver aircraft. This PMF is constructed by comparing the observation image pairs to a reference image database with Bayesian inference. This PMF is then used to compute both a relative position measurement and a protection level applicable to the precise time at which the stereo image pair was observed. Hence, the algorithm is designed to provide an independent measure of navigation integrity in addition to a relative navigation measurement. This algorithm was based upon the work of Calhoun in [2] and [5].

The second algorithm, Relative Pose Estimation with Computer Vision and Iterative Closest Point (R7), utilizes a stereo image pair to obtain a three-dimensional point cloud representing the receiver aircraft. This observation point cloud is then matched to a model point cloud with the iterative closest point (ICP) algorithm to obtain a relative position and relative attitude measurement. This algorithm is closely related to the work of Parsons [6], but utilized different point cloud filtering methods, ICP implementations, and data analysis methods.

Both these algorithms were designed to facilitate AAR on the KC-46. In order to meet this design objective, the test effort sought to reproduce the camera configuration of a KC-46 in both simulation and in flight test. However, the algorithms developed in this thesis could also be applied in other relative navigation applications.

1.2 Scope of Analysis

Algorithms were analyzed with both simulation and flight test data. The simulation environment captured simulated images of a C-12C aircraft. These images were

generated by a simulated stereo camera pair. The stereo cameras were separated by a 0.5 meters and were mounted on the simulated tanker such that their optical axes had a 25° downlook angle with respect to the tanker's longitudinal axis. This geometric configuration mimics that of the KC-46. Additionally, the camera focal lengths, image sizes, and fields of views were defined such that they modeled the cameras found on the KC-46. The generated simulated images were provided to both algorithms and the resulting algorithm errors and performance were evaluated.

Flight test data was collected as part of the Have Vision test management project at the USAF Test Pilot School [7]. Testing was accomplished by mounting two stereo camera pairs on a C-12C pseudo-tanker in a manner designed to emulate the KC-46 camera configuration. These stereo camera pairs were mounted in the same geometric configuration used in simulation and the cameras had the same properties (image size, focal length, field of view, spectral range, etc.) as those found on the KC-46. With this configuration installed, the C-12C pseudo-tanker flew a series of maneuvers typical in aerial refueling (AR) while the cameras captured images of a second C-12C aircraft flying maneuvers typical of an AR receiver. A subset of the captured images was processed post-flight through the R7 and V5 algorithms. Algorithm returns were compared to relative navigation truth data to compute algorithm errors and to facilitate analysis of algorithm performance.

1.3 Overview

The remainder of this thesis covers applicable background information, algorithm development and methodology, and algorithm analysis. It is structured as follows.

Chapter II covers background information and is comprised of four primary components. First, vision-aided AAR research is reviewed. Attention is paid to techniques focusing on template matching and navigation integrity. Second, Calhoun's Bayesian

inference integrity monitor is covered in detail [2], [5]. The V5 algorithm is a direct follow on from these efforts, so special attention is paid to his work. Third, the camera model and computer vision techniques applicable to this thesis are discussed in detail. Special attention is paid to stereo vision techniques. Fourth, the ICP algorithm is covered in detail. The ICP algorithm is essential to the operation of the R7 algorithm. This discussion summarizes the work and Besl and McKay in [8].

Chapter III covers the methodology used for data analysis in this thesis and is comprised of seven primary parts. First, the applicable reference frames and coordinate systems are defined. Second, a set of variable and notational definitions are made to facilitate discussion in the remainder of the thesis. Third, the simulation environment used for simulation data analysis is described. Fourth, the V5 algorithm is developed. Fifth, the R7 algorithm is developed. Sixth, the method used to compute algorithm errors is described. Seventh, the statistical tests used to analyze algorithm performance are described.

Chapter IV covers simulation data analysis. The performance of the V5 and R7 algorithms are examined separately and then compared to one another.

Chapter V covers the methodology and hardware specific to the flight test portion of this thesis and is comprised of four parts. First, the Have Vision test program is summarized. Second, the flight test equipment is described. Third, the flight test methodology, to include camera calibration methodology and formation flight profiles, is described. Fourth, data analysis methodology specific to flight test data is developed.

Chapter VI covers flight test data analysis. First, the resolution of the truth data system used in flight test is presented. Then, the performance of the R7 and V5 algorithms are examined separately and then compared to one another.

Chapter VII presents the conclusions and recommendations for this thesis. The primary contributions stemming from both algorithms are presented. Additionally, specific recommendations are made on how to improve both V5 and R7 performance. Finally, an outline for new algorithm that capitalizes on the strengths of both the V5 and R7 algorithms is proposed.

II. Background

This chapter has four primary components. First, previous vision-aided AAR research efforts and previous work on vision-based integrity monitors are reviewed. Second, special attention is paid to Calhoun's Bayesian Inference based integrity monitor developed in [2] and [5], since this V5 algorithm developed in this thesis builds upon Calhoun's efforts. Third, the pinhole camera model, camera calibrations, and stereo vision are discussed. Fourth, the Iterative Closest Point (ICP) algorithm is outlined. These four background elements are foundational to the V5 and R7 algorithms developed in Chapter III.

2.1 Vision-Aided AAR Research

Numerous researchers have worked on vision-based or vision-aided approaches to AAR. However, to date only a handful of researchers have examined the concept of integrity in a vision-based AAR or a vision-based precision relative navigation context. This section reviews relevant work on both vision-based AAR and on efforts to quantify the integrity of such systems. First, the work of researchers on various vision-based AAR approaches and methods are reviewed. Second, special attention is payed to image rendering approaches to precision relative navigation. Finally, research on vision-based integrity monitors is discussed in greater detail.

2.1.1 Fravolini et al's Simulation Environment for Vision-Aided AAR.

In [9] Fravolini et al developed a comprehensive simulation environment designed specifically to investigate the use of computer vision systems in the AAR context. This paper was the culmination of work done by these authors and other colleagues primarily at West Virginia University in [10], [11], [12], [13], [14], [15], and [16].

The authors framed the AAR problem in terms of a UAV refueling with a manned tanker. As envisioned by the authors, the overall goal of an AAR system is to obtain an accurate relative pose estimate of the UAV-tanker formation, to use that pose estimate to successfully guide the UAV from the pre-contact to the contact position, and finally to maintain the contact position once established in the AR envelope.

To ensure the realism of their virtual environment, Fravolini et al incorporated realistic models of tanker and UAV dynamics, the refueling boom, atmospheric turbulence, wind gusts, tanker wake turbulence, sensor noise, and UAV docking control laws. Moreover, the environment allowed for the variation of initial conditions, the location and orientation of the camera on the UAV, and the location of additional objects such as passive markers on the tanker. The net result of their work was a highly realistic, comprehensive AAR simulation environment that enabled the authors to investigate several candidate computer vision algorithms and sensor fusion methods in subsequent research efforts.

For the purposes of their research, the authors assumed that the UAV was equipped with a single digital camera mounted such that it captures images of the tanker. Once the UAV captured a tanker image in the simulation, the authors' virtual environment was designed to analyze any comp algorithm of interest via four step process:

1. Feature extraction: the computer vision algorithm identified features in the tanker image. The authors considered corners to be features of interest and identified these with a corner detector algorithm.

2. Feature matching: the identified corners were matched to a set of projected corner locations.

3. Pose estimation: based on the geometry of the identified corners, the position and orientation of the UAV relative to the tanker was estimated.

4. Sensor fusion: the computer vision estimate was fused with data from other available simulated sensors such as inertial measurement units (IMUs) and the global positioning system (GPS).

At the end of these four steps, the relative pose estimate was passed to the UAVs control systems thereby facilitating simulated AAR.

Implicit in the authors' method was the assumption that the tanker had passive markers affixed to its body in known locations. In the feature extraction stage, when a corner detector algorithm was applied to the captured imagery, these passive markers manifested as strong corners. In the real-world these markers would be retro-reflective tape or some other physical feature designed to yield greater pixel intensity in an image.

For the sensor fusion portion of their algorithm, Fravolini et al applied an extended Kalman filter (EKF) to combine differential GPS (DGPS) position data with the output of their computer vision algorithm. The authors evaluated the accuracy of the relative navigation solution output from the EKF as the receiving aircraft moved from a pre-contact to a contact position using their computer simulation environment. The authors also conducted a robustness analysis, subjecting the receiving aircraft to randomly generated position and attitude perturbations. Results showed that the authors' EKF was had typical total position errors of less than 0.003 meters in the absence of sensor noise and bias. In the presence of simulated sensor noise and bias, typical errors were on the level of approximately 0.2 meters These results were an order of magnitude or more better than results obtained with a linear interpolation sensor fusion scheme applied by the same researchers in [10].

The accuracy levels obtained by this group of researchers would likely be sufficient to enable AAR were they to be obtained in a real world flight test. This research group did not conduct any form of integrity analysis on their developed system. Nonethe-

less, the robustness of the Fravolini et al's simulation environment and sensor fusion techniques represent a standard against which other AAR efforts can be measured. Additionally, the general architecture of their computer vision-based approach, where computer vision pose estimates were fused with pose estimates from other sensors in an EKF, was a significant contribution.

2.1.2 Wilson et al's Unscented Kalman Filter.

Wilson et al [17] built upon the work of Mammarella et al in [14] and Williamson et al in [18]. The authors conducted sensor fusion with an Unscented Kalman Filter (UKF) rather than an EKF and incorporated measurements from magnetic and atmospheric sensors in addition to GPS and inertial measurements. Much like previous work, the authors assumed that a single camera was mounted on the receiving aircraft and obtained their results in simulation. Within this context, they developed a UKF that incorporated DGPS data, INS data, and computer vision data. The computer vision data was obtained using the same feature extraction, feature matching, and pose estimation algorithms that Mammarella et al developed in [16]. The authors improved the feature matching portion of the algorithm by including an additional check that helped reject incorrect point matching.

Simulation results showed that the authors' UKF outperformed the EKF developed by Mammarella et al in [14]. Moreover, the algorithm was shown to be somewhat robust against visual occlusions. Again, the authors did not conduct any form of integrity analysis their system's errors and only performed 100 simulations. More significantly, in [19], Wilson et al conducted a flight test of their system using two small drone aircraft.

During flight tests, the follower aircraft was commanded to remain 7 meters in trail of and 1 meter below the lead aircraft. During the test of their system, their

trail aircraft was able to maintain a position within 1.5 meters of horizontal error and and 1 meter vertical error during most of the flight test. No large divergences were encountered. Moreover, during deliberately forced visual sensor outages, the system degraded gracefully to INS-only performance levels. The flight test results of Wilson et al demonstrated the potential for a computer vision-aided Kalman filter to effectively maintain close formation flight.

2.1.3 Image Rendering Approaches to AAR.

The work of researchers like Wilson et al and Fravolini et al has done much to develop the use of feature matching techniques in AAR relative pose estimation. However, passive markers are problematic in a military application as they could highlight friendly assets to an adversary. Thus, in a military application, it is much more desirable that vision-based pose estimation is accomplished without the use of either passive or active markers. As a a result, this thesis is centered upon using purely passive pose estimation techniques. One method of passive pose estimation that has promise is an image rendering approach. In this section, research efforts on image rendering in the AAR context are discussed.

2.1.3.1 Howard and Veth's Image Rendering Approach.

Rather than rely on a feature matching scheme, Howard and Veth [20] utilized a predictive rendering approach to derive a computer vision-based relative navigation state estimate for AAR. The predictive rendering scheme was utilized to overcome many of the challenges associated with feature matching. The authors noted that feature matching algorithms tend to struggle with changes in aircraft attitudes, variable lighting conditions, and image occlusion. Additionally, they noted that feature

matching algorithms can encounter difficulties if mismatches occur or if a feature is not identified.

The authors incorporated the vision-based state estimate into a Kalman filter for integration with INS data. Their algorithm is summarized below.

1. The propagated relative navigation states from a Kalman filter were converted into the camera frame, resulting in a direction cosine matrix (DCM) representing the relative attitude of the two aircraft and a vector representing the relative position vector of the two aircraft.

2. INS updates from the trail aircraft were incorporated into the filter, resulting in updated relative attitude and relative position estimates. These estimates were used to generate the first rendered image.

3. A three-dimensional model of the lead aircraft with the relative attitude and position specified in step 2 was rendered. That model was used to create a template image.

4. The rendered image from step 3 and several other rendered images depicting position and attitude perturbations were compared to the observed image via a template comparison process. The specifics of the template comparison process used by Howard and Veth is discussed in detail below.

5. The best matching rendered template image was used to estimate the relative attitude and position captured in the observation image.

6. This estimate was provided to the Kalman filter in the form of a measurement vector. The authors did not discuss how they quantified the uncertainty associated with this measurement.

The most interesting result from Howard and Veth's research was the specifics of how they identified a best matching rendered template image. The search for the best rendered image consisted of two primary steps:

1. The initial rendered image was translated along the x and y axes of the observed image until a best match was identified. This was done with the OpenCV function `cvMatchTemplate`. The best match was declared based upon which template translation yielded the best normalized correlation coefficient. The normalized correlation coefficient was based upon the degree of similarity between corresponding pixel intensities in the template and observation image.

2. The best matching rendered image was perturbed in the yaw, pitch, and roll axes until a best match was identified via the same process. The magnitude of the search perturbations was heuristically determined based on observed lead aircraft movement during data collection as well as the experience of Air Force pilots during manned AR.

At the end of this process, the resulting best match was used as the measurement update for the Kalman filter as described above. Of particular note, the authors were able to significantly reduce the number of rendered images that had to be searched by making several key assumptions. First, they assumed that most changes in the relative navigation state were presumed to be along the longitudinal translation axis and in roll. Second, they bounded the perturbation search space as described above. Third, they assumed that any changes in yaw and pitch would be slow and gradual, so checks for these changes were only performed every third measurement. The net result was that only 9 or 13 rendered images were searched during the perturbation phase.

To test their system, the authors were able to make use of flight test data collected at the USAF Test Pilot School. In their approach, the authors mounted a single

camera inside the cockpit of an LJ-24 Learjet to collect observation images; this aircraft flew as the receiving aircraft. During the majority of flight tests, the camera collected images at 10 Hz. A T-38A was flown to act as the tanker aircraft. Flight test results yielded maximum position component errors on the order of 1 meter and maximum attitude errors of 2° in roll during benign test conditions.

While flying into the Sun, the algorithm required significant modification to achieve desirable results. In order to accomplish a relatively accurate match, the observed and rendered images had to be processed prior to executing the matching phase of the algorithm. The processing consisted of contouring the images, thus highlighting curves of consistent intensities within the image. This modification improved performance, but still resulted in up to 3 meters of position error.

Overall, this research demonstrated the plausibility of using an image-rendering approach to AAR during an actual flight test. The authors did not systematically characterize the error present in their image rendering approach, but did recommend future research examine the integrity of the system.

2.1.3.2 Calhoun, Curro, and Raquet's Image Rendering Flight Test.

In [21] Calhoun, Curro, and Raquet conducted a flight test of an image rendering relative navigation approach to AAR. In the flight test a Learjet was flown in trail of a KC-135. The Learjet was equipped with an Electro-Optical (EO) camera with which to capture images of the tanker. Additionally, both aircraft were equipped with a Novatel embedded global positioning system/inertial navigation system (EGI) to determine precise "truth data" of the relative position and attitude of the two aircraft.

Similar to Howard and Veth's work in [20], rendered image intensities were compared with observed image intensities on a pixel-by-pixel basis. However, instead of

examining the correlation coefficient as Howard and Veth did in [20], the authors instead examined the sum-squared difference (SSD) of pixel intensities between the observed and rendered image. Additionally, they first applied an edge filter to both the rendered and observed images prior to applying the SSD matching algorithm. The intent of the using edge features was to make the SSD comparison environmentally invariant. The SSD of the transformed observation image (I_o) and transformed rendered image (I_r) was computed as:

$$\text{SSD} = \sum_{i=1}^{n} (I_o - I_r)^2 \tag{1}$$

Where n is the number of pixels.

The three-dimensional tanker model used to generate the rendered imagery was a high-fidelity model developed by Science Applications International Corporation for the AFRL. For each observed image, a set of rendered images was searched to find the best match in an iterative fashion:

1. The algorithm used an initial position estimate and defined the uncertainty in this estimate.

2. The algorithm divided the search space into a set of discrete rendered images.

3. The algorithm searched over this set of images and identified the best match. The best match was considered to be the rendered image with the smallest SSD between it and the observation image.

4. The algorithm collapsed the search space around this best match and repeated the process a user-defined number of times in order to refine the relative navigation estimate.

Algorithm parameters for this sequence of operations were defined by four modifiable software variables: initial position uncertainty, number of divisions (essentially the number of rendered images to create in each search space), reduction factor (determines by what factor the search space is reduced during each state space search iteration), and maximum search level (determines the number of iterations that will be performed).

Using this approach, the authors were able to attain a good level of environmental invariance in flight test due to two factors. First, the camera used during the flight test divided the image into at least four different quadrants of different exposure settings. This enabled the camera to handle diverse lighting conditions within a single image observation. When flying into the Sun, this feature of the camera was able to mitigate the masking effect the Sun would otherwise have caused. Second, the use of an edge filter minimized noise effects arising from variability in lighting conditions. As a result, the flight tests revealed good results when flying into the Sun—a significant improvement upon previous work done by Howard and Veth in [20]. No resultant divergences or biases in the relative navigation solution were noted during flight tests, even when the camera was looking directly into the Sun.

A total of eight flight tests were flown with this system. At the astern position (approximately 20 meters in trail of the tanker), the image rendering algorithm was able to maintain accuracy within 35 centimeters of the Novatel data with a mean error of 16 centimeters. Forward translational error (i.e. toward or away from the tanker's longitudinal axis) was determined to be the biggest error contributor. Interestingly, performance was slightly worse in the contact position (which is much closer to the tanker at 13 meters in trail), maintaining accuracy within 60 centimeters of the Novatel data with a mean error of 17 centimeters The authors assessed this was due to a combination of worsening formation geometry and occlusion of tanker features

relative to the camera. Occasional error divergences were also noted. These were attributed to the possibility of locking onto a local minimum during the algorithm's search over the rendered image space. The authors suggested this possibility could be eliminated by increasing search resolution, but noted that this would come at a computational cost.

This research directly contributed to Calhoun, Raquet, and Peterson's work on a vision-based integrity monitor [1] which is discussed in Section 2.1.5.

2.1.4 Navigation Integrity Definitions.

Prior to discussing previous work on AAR integrity monitors, this section defines several key terms of relevance to these research efforts. Since this thesis in part closely follows the work of Calhoun [2] the definitions adopted are the same as those used in Calhoun's work.

First, the Radio Technical Commission for Aeronautics, an advisory committee chartered by the Federal Aviation Administration, defines navigation integrity as, "The ability of a system to provide timely warnings to users when the system should not be used for navigation." Second, an alert limit is defined as the maximum acceptable navigation system error tolerance. For example, an alert limit could be a position error tolerance of three meters. Third, the integrity risk level is the probability that the navigation system error exceeds the alert limit without providing a warning. Thus, an integrity risk level must be a real number between zero and one, inclusive. For example, an integrity risk level of 0.001 would mean that there is a 0.1% chance that the system navigation error exceeds the alert limit. Fourth, a protection level is a real-time navigation system output that statistically bounds the navigation system error to the required integrity risk level. For example, if the integrity risk level were 0.001, then a protection level of two meters would mean there was a 0.1% chance

that the navigation system error was actually greater than two meters. Finally, a loss of integrity is defined to be a situation in which navigation system error exceeds the alert limit without notifying the operator in a timely fashion [22], [23], and [24].

2.1.5 Calhoun, Raquet, and Peterson's Vision-Based, Binary Hypothesis Test Integrity Monitor.

In part following from the work done by Howard and Veth in [20], Calhoun, Raquet, and Peterson [1] developed an integrity monitor for vision-based precision relative navigation applications in a simulation environment. This integrity monitor was designed in the context of an AAR scenario. In the simulations, the receiving aircraft was equipped with a single camera which captured images of the tanker flying in the lead position. The authors processed the resulting imagery to make a determination of whether or not the receiving aircraft was at a formation position acceptable for AR. In turn, they used this information as the basis for an integrity monitor. The integrity monitor developed in [1] can be thought of as a binary scheme which assesses the confidence that the system will correctly declare an aircraft inside or outside the AR envelope.

The relative navigation system developed by the authors considered two hypotheses. First, that the receiving aircraft was inside the refueling envelope, also called operating region (X_{OR}). This was termed the H_0 hypothesis. Second, that the receiving aircraft was dangerously far outside the refueling envelope, in an area called the alert region (X_{AR}). This was termed the H_1 hypothesis. This situation is depicted in Figure 2.

With these definitions, the authors noted that four distinct conditions can result: detection, false alarm, rejection, and missed detection. Each condition has an associated probability: the probability of detection (P_D), the probability of false

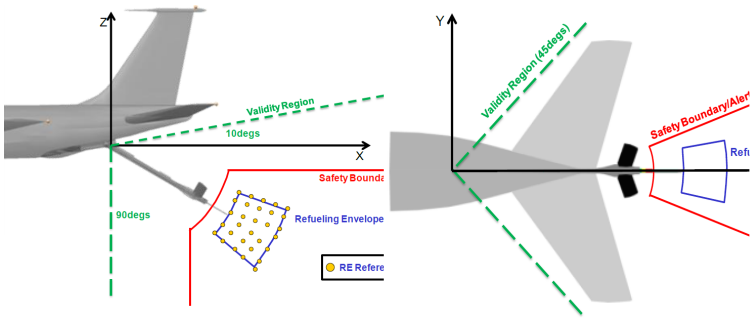


Figure 2. Calhoun's Depiction of Operating and Alert Regions. The Operating Region Falls Within the Blue Refueling Envelope, the Alert Region Falls Outside the Red Safety Boundary but Within the Green Validity Region. [1].

alarm (P_{FA}), the probability of rejection (P_R), and the probability of missed-detection (P_{MD}). The authors' depiction of the relationship between these conditions and the two hypotheses is shown in Figure 3.

Truth / Hypothesis	$x \in X_{OR}$	$x \in X_{AR}$
$H_0 : x \in X_{OR}$	P_R	P_{MD}
$H_1 : x \in X_{AR}$	P_{FA}	P_D

Figure 3. Calhoun's Possible Results from the Binary Hypothesis Test [1].

Next, the authors showed that with knowledge of the probability density functions (PDFs) describing H_0 and H_1 hypotheses, the probabilities of the P_{FA} and P_{MD} can be computed as:

$$P_{FA} = \int_{\delta}^{\infty} p_{H_0}(x)dx, \tag{2}$$

$$P_{MD} = \int_{-\infty}^{\delta} p_{H_1}(x)dx \tag{3}$$

Where δ is the appropriate parameter for the specific binary detection scheme being implemented and x is the dependent variable for the PDFs. P_D and P_{MD} are com-

plements as are P_{FA} and P_R. Therefore, to develop a binary hypothesis test, only P_D and P_{FA} needed to be assessed.

Correct PDF construction was vital to the authors' process. As a basis for PDF construction, the authors used an image rendering approach. Initially, their rendered database only included simulated images within the operating region. Using their approach, the PDF $p_{H_0}(x)$ describes the probability of observing an image correspondence metric x between the best matching rendered image and the observation image when the observed aircraft is actually in the operating region. Similarly, $p_{H_1}(x)$ describes the probability of observing an image correspondence metric x between the best matching rendered image and the observation image when the observed aircraft is actually in the alert region.

The authors examined two different image correspondence techniques as means to obtaining these PDFs. In the first case, they transformed each image into a binary silhouette in which pixels were assigned a 1 or 0 based upon being greater or less than a user-defined intensity threshold. For this technique, the authors computed an image correspondence metric ($\mathrm{SIL}_{\mathrm{CORR}}$) as the percentage of overlap between the observed and rendered silhouette for all rendered images evaluated:

$$\mathrm{SIL}_{\mathrm{CORR}_i}(I_O, I_{R_i}) = \frac{|I_{O_{\mathrm{SIL}}} \cap I_{R_{\mathrm{SIL}_i}}|}{|I_{O_{\mathrm{SIL}}} \cup I_{R_{\mathrm{SIL}_i}}|} \tag{4}$$

Where $I_{O_{\mathrm{SIL}}}$ represents the observed image transformed into a silhouette and $I_{R_{\mathrm{SIL}_i}}$ represents the rendered image i transformed into a silhouette. Then, the best matching rendered image of all rendered images i was determined by identifying which rendered image yielded the maximum value of $\mathrm{SIL}_{\mathrm{CORR}_i}$. This result served as a image correspondence data point (X_{SIL_j}) for the development of the PDFs $p(H_0)$ and $p(H_1)$:

$$X_{SIL_j} = \min_i \left[\mathrm{SIL}_{\mathrm{CORR}_i}(I_O, I_{R_i})\right] \tag{5}$$

Where "SIL" signifies the use of the silhouette image processing technique.

This process was repeated thousands of times in simulation. In order to construct the $p(H_0)$ PDF, observation images were used that were known to be in the operating region. In order to construct $p(H_1)$ PDF, observation images were used that were known to be in the alert region. The final result obtained by the authors with this process is depicted in Figure 4.

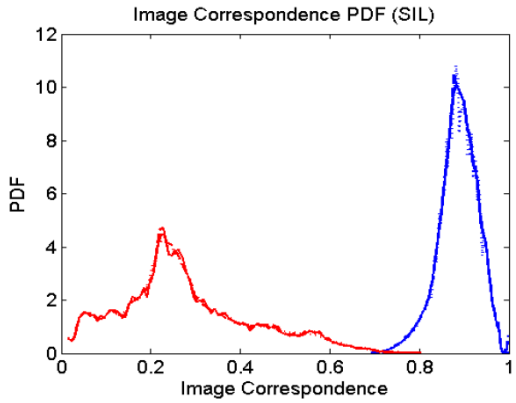


Figure 4. Calhoun's PDFs for the Silhouette Correspondence Metric; $p(H_0)$ is Depicted in Blue, $p(H_1)$ is Depicted in Red [1].

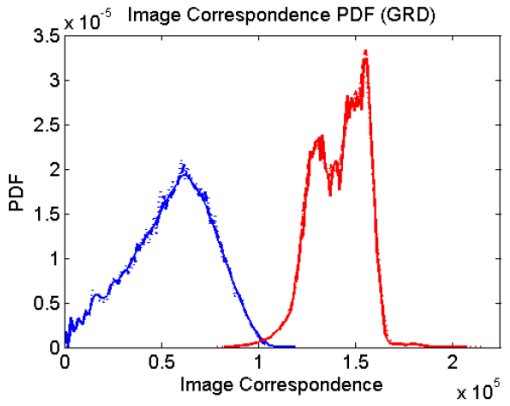


Figure 5. Calhoun's PDFs for the SSD Correspondence Metric; $p(H_0)$ is Depicted in Blue, $p(H_1)$ is Depicted in Red [1].

In the second case, rendered and observation images were put through a gradient-based edge detector and then Gaussian blurring was applied. The authors applied a Gaussian blur because the reference image set was discrete. Without Gaussian blurring, insufficient overlap between observed and reference images would cause the correspondence between reference-observation image pairs to collapse toward zero. Thus, applying Gaussian blurring enabled a match to be determined between the observed image and the discrete reference image set.

Once this transformation was complete, the SSD between the observation image and all rendered images was computed as in Equation (1). Again, the rendered image i that yielded the best match for the observation image was identified. In this case, the authors identified which rendered image i yielded the smallest SSD_i. This SSD

value was used as a data point X_{GRD_j} for PDF construction, where "GRD" signifies the use of the gradient-based edge detection and Gaussian blurring image processing technique:

$$X_{\text{GRD}_j} = \min_i \left[\text{SSD}_i\right] \tag{6}$$

Using the points X_{GRD_j}, the PDF construction process was identical to that used in the silhouette case. The final result of this process is depicted in Figure 5.

In all cases, during PDF construction the authors assumed that the prior probability of the true receiver location was uniformly distributed throughout a finite region within and outside the refueling envelope. As the authors noted, the region of overlap between the two PDFs characterizes the trade space between P_D and P_{FA}. Since P_D and P_{MD} are complements, as are P_R and P_{FA}, this trade space, depicted in Figure 6, completely characterizes probabilistic system behavior.

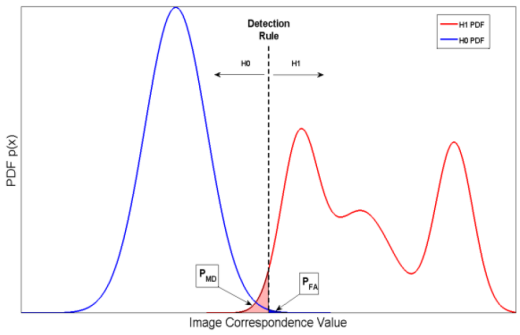


Figure 6. Calhoun's Depiction of P_{MD} and P_{FA} [1].

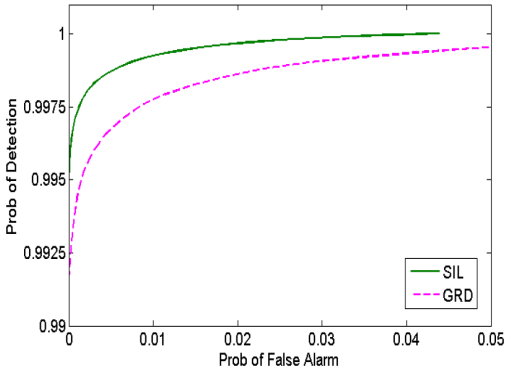


Figure 7. Calhoun's ROC Curves for Both Image Correspondence Techniques Used in [1].

Using each H_0 and H_1 PDF pair, Calhoun, Raquet, and Peterson constructed a receiver operating characteristics (ROC) curve for each feature matching technique used. Figure 7 shows both ROC curves obtained by the authors. These curves specify the system's P_D given a P_{FA}. These curves were constructed by setting a threshold (specifically, the detection rule depicted in Figure 6) for the image correspondence

metric above or below which the system will declare the receiving aircraft to be within or outside the refueling envelope. In other words, using this technique the threshold image correspondence metric dictates for what range of image correspondence values the system hypothesizes H_0 and H_1. Analysis of the ROC determined that the system could achieve a 10^{-3} P_{MD} (equivalent to a 0.999 P_D) with a corresponding P_{FA} of less than 5%. In this manner, the authors demonstrated that the ROC curve can be used to quantify system integrity for a given image correspondence threshold.

Using this work as a baseline, the authors performed several sensitivity studies and examined several modifications to the integrity monitor. Sensitivity analysis revealed substantial trade space in the size of the reference image set, and some level of robustness against image distortions, lower pixel resolutions, and changes in lighting conditions. As their first modification, the authors analyzed what would happen if observation images were also generated in the region between the operating region and safety boundary depicted in Figure 2. Observation images in this region were redefined in this case to be part of the alert region. As would be expected, included observation images drawn from this region diminished integrity monitor performance– achieving a 96% detection rate with a 5% false alarm rate. The authors results for this case are shown in Figure 8.

As a second modification, the authors expanded the reference image set to include a finite set of images within the alert region. Using this reference set, the authors redefined the test statistic to be the ratio of the best image correspondence metric between the observed image (I_S) and a rendered image within the operating region ($I_{R_{OR}}$) to the best image correspondence metric between the observed image and a rendered image within the alert region ($I_{R_{AR}}$). The authors applied this modification to simulations conducted with both the silhouette processing approach as well as the

gradient-based edge detection approach:

$$X(IC_{\text{SIL},j}) = \frac{\max_i \left[\text{CORR}_{\text{SIL}}(I_S, I_{R_{OR,i}})\right]}{\max_i \left[\text{CORR}_{\text{SIL}}(I_S, I_{R_{AR,i}})\right]} \tag{7}$$

$$X(IC_{\text{GRD},j}) = \frac{\min_i \left[\text{SSD}(I_S, I_{R_{OR,i}})\right]}{\min_i \left[\text{SSD}(I_S, I_{R_{AR,i}})\right]} \tag{8}$$

Where $X(IC_{\text{SIL},j})$ is the ratio correspondence metric when using silhouette detection as image processing, and $X(IC_{\text{GRD},j})$ is the ratio correspondence metric when using edge detection and Gaussian blurring as image processing.

These ratios were used to develop the ROC curves shown in Figure 9. Use of the silhouette ratio test with the extended imagery set yielded superior results—achieving

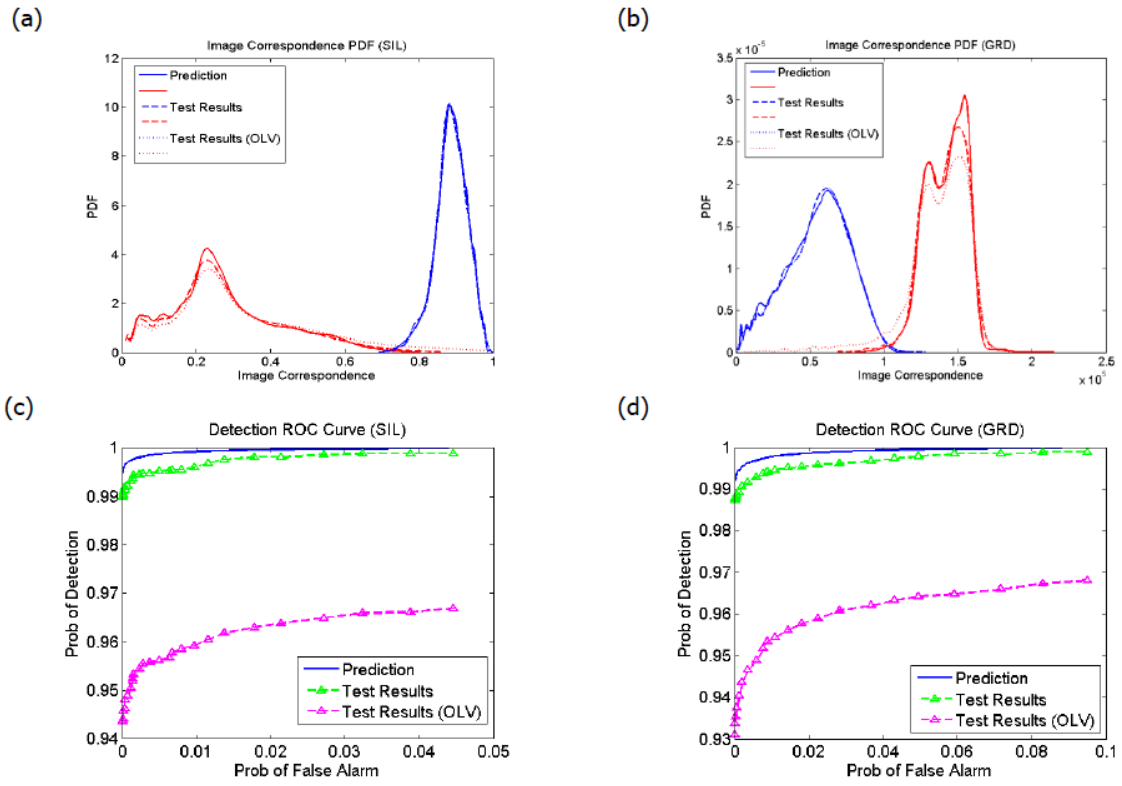


Figure 8. Calhoun's Simulation Results with Observation Images Included from the Region Between the Refueling Envelope and Safety Boundary. (a) Silhouette PDFs, (b) GRD PDFs, (c) Silhouette ROC Curve, (d) GRD ROC Curve [1].

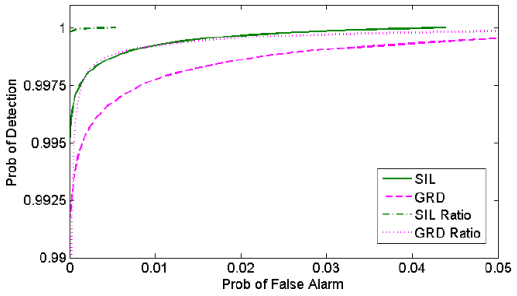


Figure 9. Calhoun's ROCs for the Ratio Tests Compared to the Baseline Tests [1].

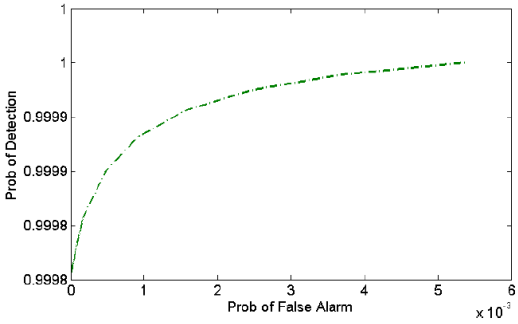


Figure 10. Calhoun's Zoom-in of the ROC curve for the SIL Ratio Test [1].

a 10^{-5} integrity level performance with a false alarm probability of approximately 0.5%.

Overall, this work developed a viable method by which to construct a binary integrity monitor for vision-based precision relative navigation. Calhoun utilized many of the lessons learned from this research effort in subsequent work on vision-based integrity monitors for AAR. Notable takeaways included the use of edge detection as a potentially viable means to make template images environmentally invariant and the construction of a reliable simulation environment.

2.2 Calhoun's Bayesian Inference Integrity Monitor

Calhoun built upon the work discussed in Section 2.1.5 in his dissertation [2] and in a paper with Raquet [5]. Much of the V5 algorithm developed in this thesis is a direct follow-on from Calhoun's work in these papers. As a result, much attention is paid to his work in this section. First, a general overview of his Bayesian inference integrity monitor algorithm is presented. Second, Sections 2.2.2 through 2.2.8 discuss the mechanics and theory underlying his methodology in greater detail. Third, Section 2.2.9 summarizes his results.

2.2.1 Overview of Calhoun's Bayesian Inference Integrity Monitor.

Calhoun utilized Bayesian inference techniques to derive a probability mass function (PMF) describing the relative navigation state of a tanker-receiver formation. This PMF was obtained solely by comparing observation images to a reference image database. Based on the PMF obtained via Bayesian inference, Calhoun was able to estimate both the relative position of the formation and a protection level for that estimate. By contrast, the binary hypothesis test integrity monitor he developed in [1] only assessed whether or not the receiver was inside or outside of the alert region and assigned a confidence level to that assessment. Calhoun's Bayesian inference integrity monitor is outlined below:

1. **Likelihood Function Determination**:

   (a) **Reference Image Database Generation**: Generate a reference image database comprised of images rendered in a simulation environment. These reference images must depict the reference aircraft at a level attitude at known, discrete points in space relative to the observing aircraft. In Calhoun's work the reference aircraft was the tanker and the observing aircraft was the receiver. Calhoun's database consisted of simulated images captured at a rectangular box of receiver positions. Each of these positions was spaced from its nearest four neighbors by a set distance. In most of his analysis this distance was 0.5 meters.

   (b) **Observation Image Collection**: Next, collect a set of observation images in simulation. The relative position captured in the observation image must be randomly sampled from the geometric space containing the reference images. The relative position depicted in each observation image must be known in order to enable likelihood function determination. In

Calhoun's work these observation images depicted the reference aircraft in level flight.

(c) **Image Processing**: Process both the reference and observation images in the following fashion.

i. **Edge Detection**: Apply a Prewitt edge detector to all images, resulting in depictions of only the edges detected in each image. For convenience these are referred to as "edge-space images."

ii. **Blurring**: Apply Gaussian blurring to all the resultant edge-space images.

(d) **Identify Best Reference-Observation Image Pair**: Identify the reference image generated in step 1(a) that most closely represents the same relative position captured by each observation image generated in step 1(b).

For instance, assume there are only three reference images. These depict the reference aircraft located at three-dimensional positions (1,2,3), (4,5,6), and (7,8,9) relative to the observing aircraft. If there is an observation image whose relative position is known to be (1,2,4), then the first reference image is the best match and would be selected for comparison to the observation image.

(e) **Compare Best Reference-Observation Image Pairs**: Compare the features depicted in the best reference-observation image pair. Calhoun simply differenced the pixel intensities of the reference and observation images at randomly selected pixel locations. These pixel locations were randomly selected from the population of sample image edge pixels.

(f) **Construct the Likelihood Function**: Identify a PDF that models the feature differences between the best reference-observation image pairs.

Enough reference-observation image pairs must be used to adequately represent the sample space of interest. This PDF serves as the likelihood function used in all subsequent steps. Calhoun selected a PDF to model the ensemble of all edge pixel intensity differences he observed when performing the comparisons described in step 1(e).

2. **Observation Image Collection**: Collect a set of observation images either in the real-world or in simulation. Calhoun's work was confined to simulation. These observation images must fall within the confines of the reference image database. However, now that a likelihood function has been determined, the relative position of these observation images need not be precisely known.

3. **Prior Probability Determination**: Determine the prior probability for each relative position represented in the reference image database generated in step 1(a). Note that the possible relative positions form a discrete set. Calhoun had the best results when assuming a uniform prior. However, in practice, this prior probability could be the output of an EKF, an UKF, or some other form of a recursive estimation algorithm.

4. **Reference Database-Observation Image Comparison**:

   (a) **Reference Database Image Attenuation**: If possible, confine the reference image database to be searched to a subset of the entire reference image database generated in step 1(a). Calhoun confined his search space to a 6 meter-by-6 meter-by-6 meter box centered near the true location of the observation image.

   (b) **Image Comparison**: Compare the observation image to all reference images identified in step 4(a) according to the same metric used in step 1(e). Compare multiple pixels to make the ultimate results more robust.

For his metric, Calhoun differenced the pixel intensities of the reference and observation images at randomly selected pixel locations. These pixel locations were randomly selected from the population of sample image edge pixels.

(c) **Compute Likelihood Scores**: Based on the results from 4(b) and the likelihood function determined in step 1(f), compute a likelihood score for each reference image-observation image pair. For example, in Calhoun's work the likelihood of observing edge pixel intensity differences was computed.

5. **Bayesian Inference**: For each point analyzed in the attenuated the reference image database obtain in step 4(a), determine the posterior probability that the observation image depicts the formation at that relative position. This is done using each computed likelihood score and each prior probability with Bayes' Law. Since the reference image database is discrete and the prior probabilities form a PMF, the posterior probability obtained from Bayes' Law will be a PMF.

6. **Relative Position Estimate**: The relative position estimate is given by the location of the reference image with the highest posterior probability of depicted the observation image.

7. **Protection Level Determination**: Starting from the relative position estimate, geometrically move outward from the that position in the reference image database, summing posterior probabilities as you move outward. Continue this process until reaching the desired integrity risk level. The distance moved away from the relative position estimate when the desired integrity risk level is reached constitutes the protection level. Calhoun split this into vertical and horizontal protection level components.

2.2.2 Image Processing.

In the image rendering approach used in Calhoun's work in [2] and [5], comparison of a sample image to the reference image database requires that the features being compared are consistent in different environmental conditions and that these features are able to span the gaps in the reference image database. For these reasons, Calhoun applied both a Prewitt edge detector and a Gaussian blur to his observation images and his reference image database.

Calhoun applied edge detection in order to make the image features to be compared environmentally invariant. Calhoun applied Gaussian blurring for two primary reasons. First, the application of blurring can help correct for template modeling errors. Second, the rendered image database contains reference images of the aircraft at discrete points in space. Contrastingly, observed images are drawn from a continuous, infinite set of relative positions. These possible locations exist along the entire continuum of points located both at and between the discrete locations of the reference image database. This is true both in simulation and in flight. If no Gaussian blurring were applied, then the only time an observed image would be a good match with a reference image would be when the observed aircraft happens to be located at or very close to a point depicted in the discrete reference image database. Hence, Gaussian blurring must be applied to either the reference image database, the observed images, or both.

Moreover, the level of Gaussian blurring must be tuned such that the discrete reference image database sufficiently overlaps with all possible sample images. This is done by changing the standard deviation parameter, σ. Alternatively, blurring functions other than Gaussian could be applied. Calhoun chose Gaussian blurring due to its simplicity (only the standard deviation parameter, σ, must be tuned)

and low-pass filtering effects. [2]. Edge detection is discussed in Section 2.2.2.1 and Gaussian blurring in Section 2.2.2.3.

2.2.2.1 Gradient-Based Edge Detection.

Image gradients are commonly used to find edges in an image. The gradient vector of a function f is [25]:

$$\nabla \mathbf{f} = [g_x, g_y]^T = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]^T \tag{9}$$

The magnitude of the gradient vector is [25]:

$$\nabla f = \left[g_x^2 + g_y^2\right]^{1/2} \tag{10}$$

A gradient vector points in the direction of maximum rate of change. Applied to pixel intensities in an image, the gradient at a given pixel identifies the direction of maximal intensity change. Moreover, the magnitude of the gradient at each pixel can be used to identify pixels where the rate of intensity change surpasses a given threshold value [25]. Points where this occurs are deemed edge points, the collection of which are used to identify edges [26]. In practice one of several approximations to Equation (10) can be used to identify pixels in an image where the intensity change exceeds a given threshold. This paper uses a Prewitt edge detector.

2.2.2.2 Prewitt Edge Detection.

As previously mentioned, Calhoun [2], [5] used a Prewitt filter to identify edges within each image. A Prewitt edge detector takes the difference in pixel intensities in a three-by-three region to approximate the partial derivatives used in Equation (9). Gonzalez, Woods, and Eddins [25] provide an excellent visual depiction of this

process which is replicated in Figure 11. The figure and commensurate discussion summarizes their description.

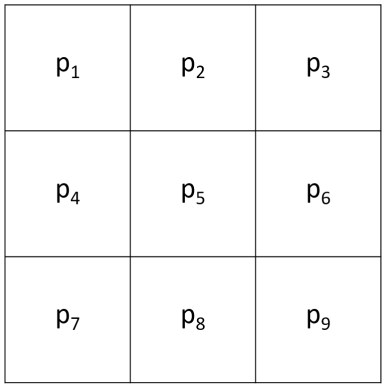


Figure 11. Representation of Pixels Used in Prewitt Edge Detector.

Consider the pixel, p_5. In a Prewitt edge detector, the partial derivatives in the x and y directions, g_x and g_y, are approximated as [25],

$$g_x = (p_7 + p_8 + p_9) - (p_1 + p_2 + p_3), \tag{11}$$

$$g_y = (p_3 + p_6 + p_9) - (p_1 + p_4 + p_7). \tag{12}$$

Equation (10) is then applied and the resulting magnitude of gradient is compared against a threshold value, T. If the magnitude of gradient exceeds the threshold value, then pixel p_5 is declared as an edge point.

2.2.2.3 Gaussian Blurring.

Blurring of an image can be used to simulate sensor noise. As outlined by Calhoun [2], [5], blurring is also useful when performing template matching and/or when estimating object poses with templates pulled from a discrete rendered image database. In this context, appropriately tuned blurring of an observed image and/or the discrete database will help to account for template modeling errors. Moreover,

when using a discrete template dataset for pose estimation, blurring helps to ensure that the discrete dataset "covers" the spatial gaps between each discrete position represented in the database.

The most common method of blurring involves the use of a Gaussian filter. In [26], Forsyth and Ponce outline how convolving the partial derivative of the Gaussian blurring function with the image results in the derivative of a Gaussian blurred image along the direction of interest. Their discussion is summarized below:

$$\frac{\partial I_B}{\partial x} = \left(\frac{\partial G_\sigma}{\partial x}\right) * I \tag{13}$$

Here, I_B represents the blurred image, I the raw image, G_σ represents the Gaussian smoothing function parameterized by its standard deviation, σ, and x specifies that blurring is along the x pixel direction. The mean of the Gaussian smoothing function is zero. The Gaussian smoothing function itself, with x and y expressed in units of pixels, is given in [26] as,

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right). \tag{14}$$

Recall from Equation (9), that a vector of the partial derivatives in x and y comprises an image gradient. Hence, using Equation (13) one can obtain the image gradient of a Gaussian blurred image , I_B:

$$\nabla I_B = \left[\frac{\partial I_B}{\partial x}, \frac{\partial I_B}{\partial y}\right]^T \tag{15}$$

In practice, the normalizing coefficient can be ignored since we are only concerned with relative differences in blurred image pixel intensity and not in absolute blurred

image pixel intensity. Forsyth and Ponce describe several important properties of Gaussian blurring [26]:

1. The net effect of Gaussian blurring is to create a weighted average of pixel intensity that places more weight on nearby pixels.

2. Increasing the standard deviation, σ, of the Gaussian blur filter increases the weight of further afield pixels.

3. Gaussian blurring acts as a low pass filter; the effect of high spatial frequency intensity changes is attenuated.

As a result of these properties, performing Gaussian blurring on an edge-space image tends to spread out image edges [2].

2.2.3 Template Matching.

Template matching is fundamental to Calhoun's Bayesian inference-based integrity monitor. In the context of his work an this thesis, template matching is a computer vision technique that matches an observed image to an explicit template of the object or feature of interest. In general, the purpose of this technique is object recognition. Brunelli noted that when an observation image and template image are not a perfect match, a template matching algorithm is faced with comparing a template to a modified version of itself, plus additional content, plus noise [27]. In this circumstance, a statistical quantification of the goodness of a template's match to an observed image is necessary in order to draw meaningful conclusions from any template matching process. In the context of this thesis and Calhoun's work [2], [5], wherein a probabilistically quantifiable relative position estimate is the ultimate goal, a rigorous statistical quantification of the quality of match between the observed image and the template is essential.

Brunelli [27] frames template matching as "a two-person statistical game between nature and a computational agent." This game can be thought of as an algorithm's attempt to achieve a set of goals. The goal of Calhoun's Bayesian inference integrity monitor is to estimate a PMF describing the relative position of the formation based solely on images captured by a single camera. As described above, this enables both relative position and protection level estimation [2]. For this thesis, the algorithm goals include the relative pose of two aircraft given stereo camera imagery. The goal of the V5 algorithm would include PMF and protection level estimation while the goal of the R7 algorithm would include attitude estimation. As described by Brunelli, the game proceeds as follows:

1. Nature selects a physical state $x \in \mathbb{X}$. In the context of this thesis and Calhoun's work, x is the relative pose of the two aircraft and $\mathbb{X}$ represents the set of all possible relative poses.

2. Based on the state x selected by nature, a set of observations θ are made according to a conditional probability distribution $p(\theta|x)$. In the context of the V5 algorithm and Calhoun's work, the observation θ is the difference between the observation image and the relevant images in the reference image database. In part, these differences depend upon the image processing algorithms used to detect features of interest. These observations are the consequence of an unknown conditional probability distribution that depends on the state of interest, x, and on the construction of the image processing algorithm itself. Properly estimating $p(\theta|x)$ is critical if the overall algorithm is to probabilistically estimate the true state x and its confidence in that estimate.

3. Based on the observations θ, the algorithm estimates the unknown true state of nature, x, according to its decision function $\phi(\theta) = \delta$, where δ is the algorithm's best estimate of the true state x. In the context of the V5 algorithm and

Calhoun's work, the estimate δ is attained via likelihood computations and Bayesian inference as discussed in Section 2.2.4. The algorithm's statistical confidence in its relative state estimate δ is required to return a protection level.

In the following section, Bayesian inference and its relation to template matching is introduced. Next, the details of the template matching process outlined above are discussed in the context of Calhoun's work in [2] and [5].

2.2.4 Template Matching and Bayesian Inference.

According to Bayes' Law, for a discrete set of mutually exclusive and mutually exhaustive events, $x_1, x_2, ..., x_N$:

$$p(\mathbf{x}_j|\theta) = \frac{p(\theta|\mathbf{x}_j)p(\mathbf{x}_j)}{\sum_1^N p(\theta|\mathbf{x}_j)p(\mathbf{x}_j)} \tag{16}$$

Where $p(\cdot)$ represents the probability of an event. Here, $p(\theta|\mathbf{x}_j)$ is commonly referred to as the likelihood function, $p(\mathbf{x}_j)$ as the *a priori* probability of event $\mathbf{x}_j$, and $p(\mathbf{x}_j|\theta)$ as the *a posteriori* probability of event $\mathbf{x}_j$ given the observations θ [28], [29].

As described in steps 2 and 3 of Brunelli's template matching process (outlined in Section 2.2.3), the conditional probability distribution $p(\theta|\mathbf{x}_j)$ is needed in order to probabilistically obtain an estimate of the true state $\mathbf{x}_j$. Calhoun demonstrated in [2] and [5] that Bayesian Inference can be used to accomplish this task. Hence, in the context of Bayes' Law, Brunelli's conditional probability distribution, $p(\theta|\mathbf{x}_j)$, can be thought of as the likelihood function. The next section describes the method Calhoun used to determine an appropriate likelihood function for his algorithm.

2.2.5 Likelihood Function Determination.

In [2] and [5], Calhoun used the difference in pixel intensities between a processed observation image and a processed reference image as his set of observations, θ. To recap, processed images were obtained from raw images in the following manner:

1. Raw images were transformed into edge-space with a Prewitt edge detector.
2. Gaussian blurring was applied to the resulting images at a heuristically determined level.

Once these processed images were generated, Calhoun computed the difference in intensities at pixel i as:

$$g_b(x_i, y_i) = I_{GB_O}(x_i, y_i) - I_{GB_R}(x_i, y_i) \tag{17}$$

Where $I_{GB_O}(x_i, y_i)$ is the intensity of the processed observation image at pixel i, $I_{GB_R}(x_i, y_i)$ is the intensity of the processed reference image at pixel i, and $g_b(x_i, y_i)$ is the difference of these intensities at pixel i. Calhoun only examined pixel intensity differences at pixels identified as edge features in the processed observation image, I_{GB_O}.

To determine the likelihood function, Calhoun generated a probability density function (PDF) describing $p(g_b(x_i, y_i))$, the probability of observing any given pixel intensity difference between the observed and rendered images at pixel i. This quantity will be denoted as $p(g_{b_i})$. In the context of Bayesian Inference, $p(g_{b_i})$ constitutes the likelihood function $p(\theta|\mathbf{x}_j)$ if the observations are confined to a single pixel. Extension to multiple pixel observations will be discussed below.

In order to generate the PDF $p(g_{b_i})$, Calhoun evaluated many observation image-reference image pairs. From each of these pairs, multiple pixels were randomly se-

lected. The intensity difference between these images at these pixels, as described in Equation (17), was used to construct the PDF [2], [5].

Critically, in order to generate $p(g_{b_i})$, Calhoun deliberately selected I_{GB_R} such that the relative navigation state captured by I_{GB_R} was known to be the best match for the relative navigation state captured by I_{GB_O} out of all images in the reference image database. For the purposes of likelihood function generation, the best match reference image was chosen since the algorithm's objective is to accurately estimate the relative navigation state. In simulation, the researcher has perfect knowledge of the relative navigation state of every reference and observation image. Hence, determining the best match reference image for the observation image is trivial in simulation.

Calhoun used an unspecified number of randomly generated observation image-reference image pixel pairs in order to obtain an estimate of $p(g_{b_i})$. His results are depicted in Figure 12.

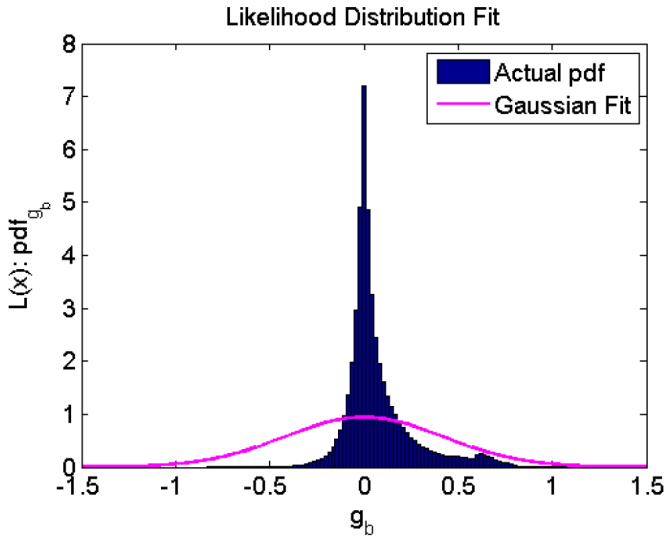


Figure 12. Calhoun's Gaussian fit for the pixel intensity difference likelihood function $p(g_b)$ or $p(\theta|\mathbf{x}_j)$ [2].

As can be seen in the Figure 12 and as noted by Calhoun, the actual PDF closely resembles a Laplace distribution with one major caveat—there is a small local maximum on the positive tail of the distribution.

Calhoun asserted that the local maximum was mostly attributable to the fact that g_b was calculated based on differencing pixel intensities at pixels identified as edges in the observation image. Non-edge pixels have an intensity value of zero. Recall that $g_b(x_i, y_i) = I_{GB_O}(x_i, y_i) - I_{GB_R}(x_i, y_i)$ and pixels were deliberately selected such that the value of $I_{GB_O}(x_i, y_i)$ is non-zero. Hence, unless the observation and reference images are identical, there will almost certainly be cases where pixels in I_{GB_O} are edges while the corresponding pixels in I_{GB_R} are not. Moreover, the discrete nature of the reference image database and the level of Gaussian blurring applied would affect both the location and magnitude of this small intensity difference peak. As a result of this positive tail artifact, Calhoun chose to overbound the distribution with a Gaussian fit.

However, while this fit does better account for the local maximum in the data, Figure 12 clearly shows that the overall fit is poor. Additionally, Calhoun noted that overbounding was necessary to achieve desired performance. Likelihood function analysis performed in this thesis for the V5 algorithm, discussed in Section 4.1.5, helps to demonstrate why overbounding is necessary with this sort of method.

2.2.6 Likelihood Score Computation.

Using the likelihood PDF $p(g_{b_i})$, an observation image can be compared to any image in the reference image database. If the observation image is compared to a set of reference images, then each reference image will have its own likelihood score. Theoretically, the reference image that is truly the best match for the observation image will typically have the highest likelihood score. However, Calhoun observed

that if only one pixel were evaluated it is likely that "bad matches" in the reference image database might score higher than "good matches." Hence, it is advantageous to use multiple pixels.

In order to utilize multiple pixels, Calhoun assumed that the intensity difference g_b observed at each pixel i is independent of all others. He noted that similar assumptions were made in related work by Olson [30] and Wells [31]. Calhoun asserted that the independence assumption made the likelihood score computation tractable and increased the chances of identifying a "good match." Based on this assumption, Calhoun's used his Gaussian fit of $p(g_{b_i})$ to compute an overall likelihood function according to the equation:

$$p(\theta|\mathbf{x}_j) = \prod_{i=1}^{N} p(g_{b_i}) \tag{18}$$

Where θ represents the set of all N pixel observations and i denotes pixel i. Recall that the likelihood function $p(g_{b_i})$ is applicable at the individual pixel level. Contrastingly, the likelihood score $p(\theta|\mathbf{x}_j)$ applies to a set of pixels obtained from a single observation-reference image pair.

Using the fit that Calhoun applied to $p(g_{b_i})$, he showed that the overall likelihood function can be computed as:

$$p(\theta|\mathbf{x}_j) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^N \exp\left[-\frac{1}{2\sigma^2}\sum_{i=1}^{N}(g_{b_i} - \mu)^2\right] \tag{19}$$

Where μ is the mean intensity difference parameterizing Calhoun's Gaussian fit and σ is the standard deviation parameterizing Calhoun's Gaussian fit.

2.2.6.1 Log-Likelihood Computation.

When dealing with multiple pixels, Calhoun noted that computing a product of probabilities in this manner will quickly converge toward zero and could run into

machine-precision issues [2], [5]. Hence, he deemed use of a log-likelihood metric to be preferable. By taking the logarithm of the likelihood function $p(\theta|\mathbf{x}_j)$ given in Equations (18) and (19), a product is turned into a summation, thereby avoiding potential machine precision issues:

$$\log(p(\theta|\mathbf{x}_j)) = \sum_{i=1}^{N} \log(p(g_{b_i})) \tag{20}$$

$$\log(p(\theta|\mathbf{x}_j)) = N \log(\frac{1}{\sqrt{2\pi}\sigma}) - \sum_{i=1}^{N} \frac{(g_{b_i} - \mu)^2}{2\sigma^2} \tag{21}$$

In this manner, a log-likelihood can be computed for every observation image-reference image pair of interest.

2.2.7 Posterior Probability Computation.

Once obtaining likelihood scores, Calhoun leveraged Bayes' Law as shown in Equation (16), in order to compute a probability distribution describing the relative position of the aircraft. Calhoun examined the use of both a normal distribution and a uniform distribution as prior probability functions. Obtaining a probability distribution describing the aircraft state with Bayes' Law enabled Calhoun to garner information on the uncertainty of the relative position estimate returned by his algorithm and to compute a protection level [2], [5].

2.2.8 Protection Level Computation.

As described in the previous section, the posterior probability that the relative navigation state is x_i can be computed via Bayesian inference. Every reference point examined (that is, every relative position $\mathbf{x}_j$ in the search space) has a posterior probability. The sum of all these probabilities is one. Calhoun took the state $\mathbf{x}_j$ with the highest posterior probability as the system's best estimate for the relative pose

of the two aircraft. Based on this assumption, Calhoun determined a protection level for the estimate by expanding outward from this best estimate until achieving the desired integrity risk level [2], [5]:

$$[i, j, k] = \arg \min_{i,j,k} \sum_i \sum_j \sum_k p(x_i, y_j, z_k|\theta) < 1 - \text{IRL}, \tag{22}$$

$$\text{PL} = \sqrt{(x_i - x_o)^2 + (y_j - y_o)^2 + (z_k - z_o)^2}, \tag{23}$$

$$\text{VPL} = |z_i - z_o|, \tag{24}$$

$$\text{HPL} = \sqrt{(x_i - x_o)^2 + (y_i - y_o)^2}. \tag{25}$$

Above, (x_o, y_o, z_o) is the relative position estimate, the acronym IRL stands for integrity risk level, the acronym PL stands for protection level, the acronym VPL stands for vertical protection level, and the acronym HPL stands for horizontal protection level. Correspondingly, the position (x_i, y_i, z_i) is the closest position to (x_o, y_o, z_o) for which the required integrity risk level was met. Note that the quantity $1 - IRL$ must be a probability. Recall that using Calhoun's relative position estimation method, the relative position estimate was by necessity a position depicted in the reference image database.

For the purposes of his analysis, Calhoun defined the horizontal and vertical directions relative to the body frame of the tanker aircraft—thereby making these directions consistent with the geometry of the refueling envelope. Recall that protection levels are distances, typically expressed in meters or feet depending upon the user's preference.

2.2.9 Calhoun's Results.

Calhoun conducted simulation analysis of his Bayesian inference integrity monitor in [2] and [5]. To conduct his analysis, Calhoun simulated images of a tanker taken from 181 distinct receiver positions both within and just outside the refueling envelope. Both aircraft were simulated to be at level flight for both the observation images and the images in the reference database.

As described in Section 2.2.1, Calhoun attenuated the reference image database when conducting comparisons to observation images. The attenuated reference database consisted of a 6 meter-by-6 meter-by-6 meter box of images taken from evenly spaced receiver positions. The center of the database was chosen as the true receiver location plus a random error in all dimensions. The random error was chosen to be normally distributed with a mean of zero and a standard deviation of 0.5 meters. Each of the images in the reference database was spaced 0.5 meters from its nearest neighbors, thereby generating a database of 2,197 images centered in the proximity of the receiver's true relative position. One can visualize the rendered image set as forming a cube. Thus, each of the 181 unique observation images was compared to a different database of 2,197 reference images.

As described in Section 2.2.5, Calhoun computed a likelihood score for image in the rendered image database using an overbounded Gaussian distribution. In order to do this, he randomly sampled 100 pixels identified as edges in the processed observation image. As previously mentioned, he had best results when using a uniform distribution as the prior probability distribution. When using 100 pixels with a uniform prior, typical navigation position errors returned by his algorithm were less than approximately 0.5 meters—a figure equivalent to the spacing of the reference images in his database. This indicates that his algorithm was typically able to converge precisely or very close to the best possible approximation of the true navigation solution

in his simulation environment. Calhoun analyzed results using an integrity risk level of 10^{-6}. The navigation position error never exceeded the protection level during his simulations. Typical protection levels returned by the algorithm were on the order of 1-2.5 meters in the horizontal dimension and 0.5-1.5 meters in the vertical dimension. Figure 13 presents these results.

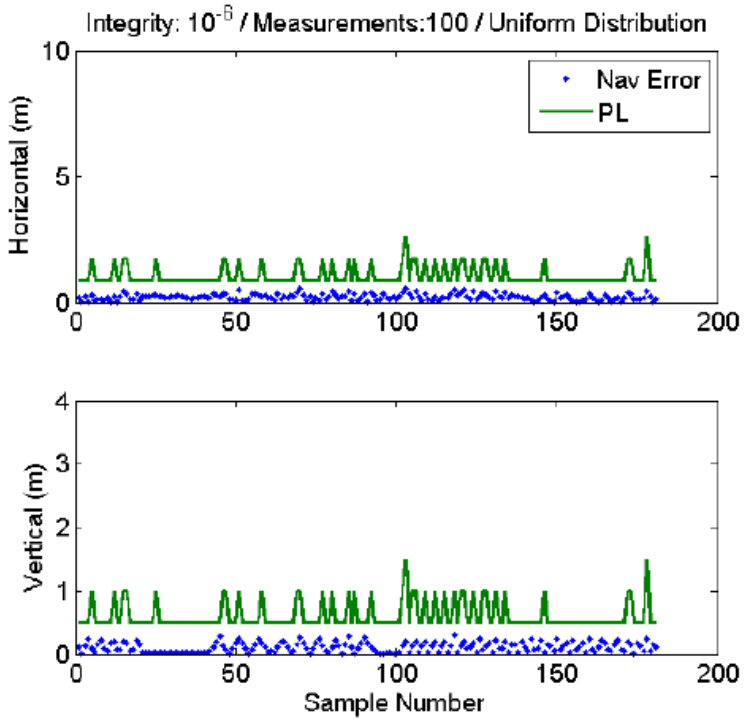


Figure 13. Calhoun's Results for 10^{-6} Integrity Risk Level, 100 Sample Pixels, and Uniform Prior [2].

Calhoun also conducted several sensitivity studies to analyze the effect of using fewer sample pixels and/or less restrictive integrity risk levels. His analysis revealed the potential to use fewer than 100 pixels while still achieving satisfactory results. Additionally, it revealed, as expected, that a less restrictive integrity risk level results in a protection levels that are smaller in magnitude. For instance, using a less restrictive integrity risk level of 0.05 and sampling only 20 pixels from each image, typical protection levels were on the order of 0.5 meters, with an occasional excursion

to greater than 1 meter. When using this lower number of sample pixels, typical position errors were still less than 0.5 meters. These results are depicted in Figure 14.

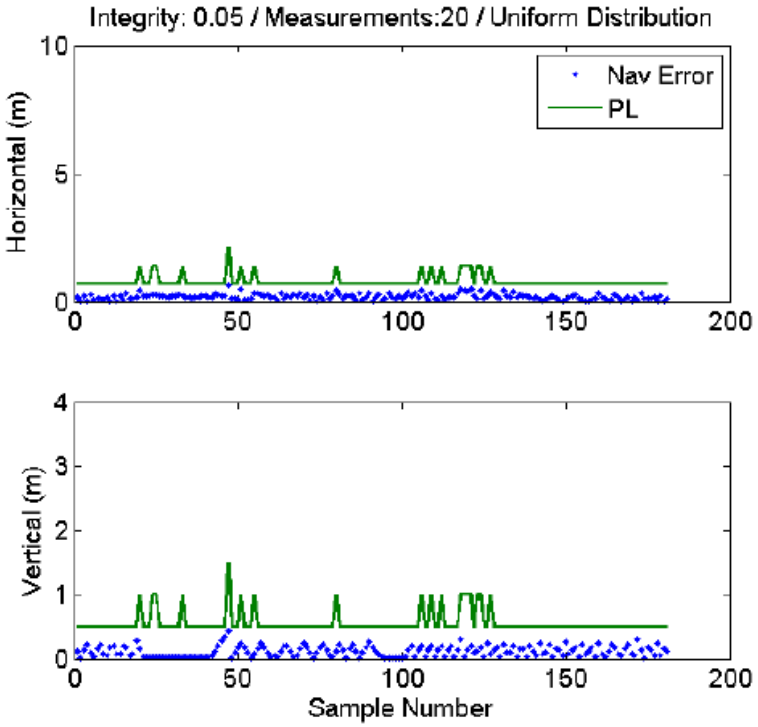


Figure 14. Calhoun's Results for 0.05 Integrity Risk Level, 20 Sample Pixels, and Uniform Prior [2].

Overall, Calhoun's work demonstrated that a relatively simple computer vision algorithm has the potential to provide a degree of navigation integrity to close formation flight. This V5 algorithm developed in this thesis attempts to build upon his work by (1) taking advantage of a stereo camera system, (2) analyzing the effect of a tanker-based, rather than a receiver-based vision system, (3) conducting a more in depth analysis of appropriate likelihood functions, (4) analyzing algorithm errors based upon the geometry of the vision system rather than the geometry of the tanker, (5) using a more realistic and robust simulation environment, (6) exploring the effect

of non-level attitudes, and (7) more aggressively exploring areas of potential integrity violations. These items are discussed in detail in Chapters III, IV and VI.

2.3 Camera Model and Computer Vision

The V5 and R7 algorithms developed in this thesis are designed to take advantage of the images output by electro-optical (EO) sensors, specifically digital cameras. While EO sensors can target various portions of the electro-magnetic (EM) spectrum, the flight test analysis performed in this thesis utilized EO cameras that targeted the visible spectrum.

The USAF Test Pilot School's generalized EO sensor model is summarized here, and depiction of the model is shown in Figure 15 [3]. In the figure, the EO sensor has a specific target which has associated EM radiation. However, background clutter present in the sensor's field of view will also emit or reflect radiation. Radiation from the scene must pass through the atmosphere which will scatter and absorb EM radiation in a manner primarily dependent upon frequency resulting in signal losses. Radiation which reaches the sensor first passes through an objective lens, then a reticle, and then a field lens. These components condition the EM signal for acquisition by the detector. The detector transforms the EM radiation into an electrical signal suitable for digital processing, such as a digital image. The ultimate objective of the system is to capture information on the target.

In practice, a simplified camera model can be used in vision-aided navigation. This section outlines the model used in this thesis. In the forthcoming discussion, capitalized vectors are used to represent three-dimensional points, lowercase vectors are used to represent two-dimensional points.

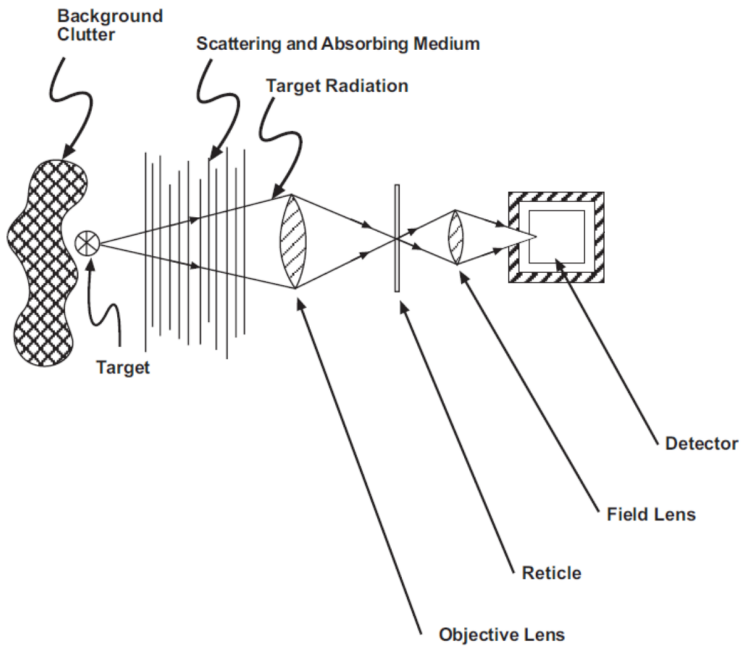


Figure 15. USAF Test Pilot School's EO Sensor Model [3].

2.3.1 Camera Coordinate Frame.

Throughout this thesis, the following convention is used to define a camera-centered coordinate frame:

1. The origin is located at the center of projection of the camera. In a pinhole camera model (described in Section 2.3.2), this corresponds to the pinhole.

2. The positive z-axis extends directly out from the origin, through the center of the lens along the optical axis of the camera.

3. As viewed form the behind the camera, the positive x-axis extends to the right, parallel to the image plane of the camera.

4. As viewed form the behind the camera, the y-axis extends downward, parallel to the image plane of the camera. Hence, this is a right-handed coordinate system.

Figure 16 depicts this convention, where $\mathbf{X}^c = [x'_c, y'_c, z'_c]^T$ represents the three-dimensional coordinates of a real-world point in the camera frame.

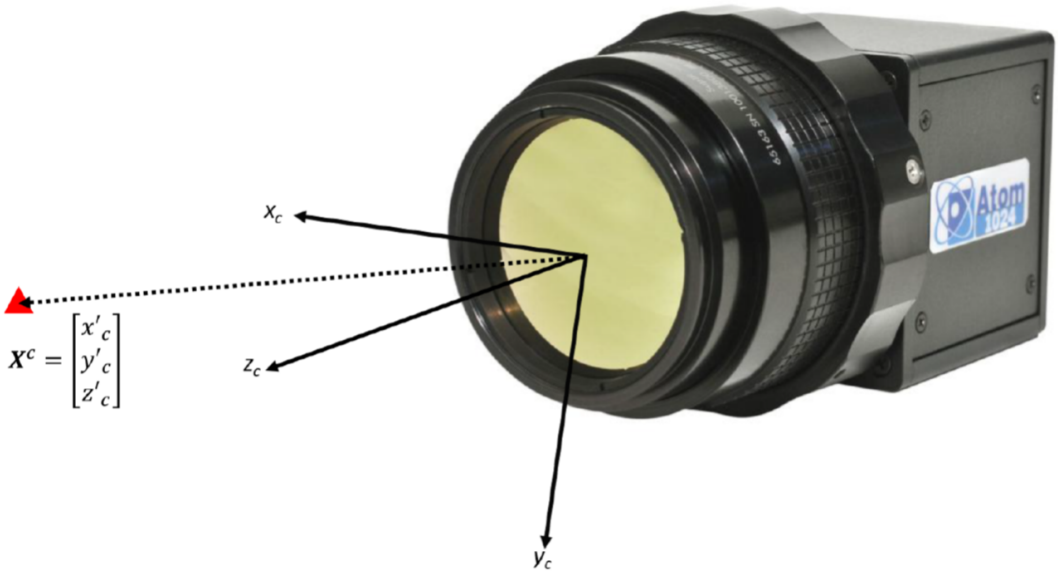


Figure 16. Camera Coordinate Frame

Another parameter of interest to cameras is the field of view. The field of view describes, in terms of angles, how much of a scene is visible to the camera at an given instant in time. The horizontal field of view (HFOV) describes the angular coverage with respect to the z-axis in the xz-plane of the camera coordinate frame. The vertical field of view (VFOV) describes the angular coverage with respect to the z-axis in the yz-plane of the camera coordinate frame.

2.3.2 Pinhole Camera Model.

In computer vision, cameras are often modeled as a pinhole camera; this convention is referred to as the pinhole camera model. In the pinhole camera model, light rays enter the camera through an infinitesimally small hole (a pinhole). These rays are then projected onto an image plane. The image plane is located behind

the pinhole at a distance equal to the physical focal length of the camera. The net result is an inverted image of the scene being observed. To ease visualization, it is common practice to define a virtual image plane that is located in front of the pinhole at a distance equal to the focal length. The image formed on this plane is not inverted [32], [33], [4].

Figure 17 depicts this model. In the figure, F is the physical focal length of the camera, which is typically expressed in millimeters. The optical axis passes through the center of both the image plane and the virtual image plane. The point $\mathbf{X}^c = [x_c, y_c, z_c]^T$ represents a real-world point in the camera frame while the point $\mathbf{x}^{VI} = [x'_{VI}, y'_{VI}]^T$ represents the corresponding point on the virtual image plane and $\mathbf{x}^I = [-x'_{VI}, -y'_{VI}]^T$ represents the point on the image plane. In both planes, the coordinate origin is located at the center of the image plane (also called the principal point), with the positive x-axis defined in the right direction and the positive y-axis defined in the down direction.

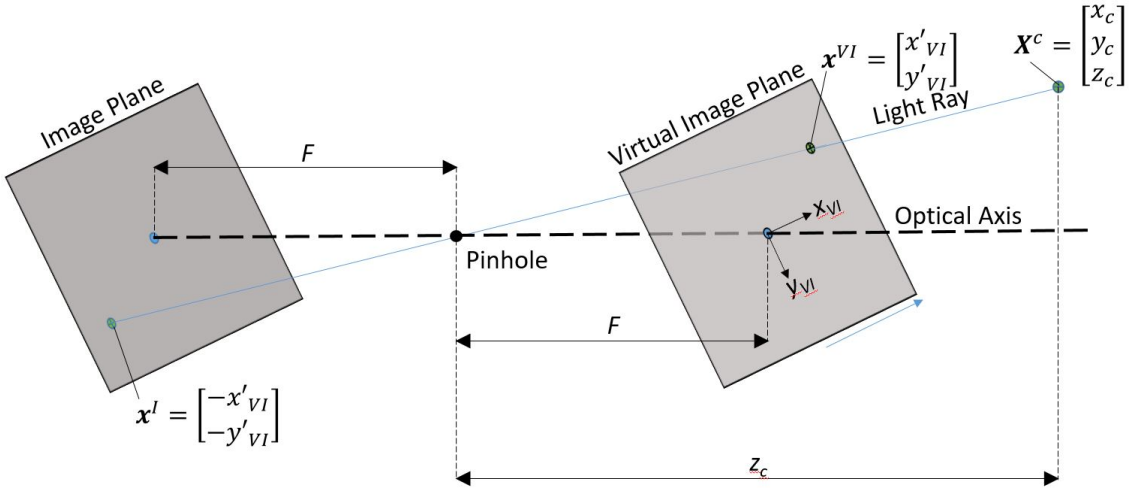


Figure 17. Pinhole Camera Model

It is evident from similar triangles that [32], [33], [4]:

$$\mathbf{x}^{VI} = \begin{bmatrix} x'_{VI} \\ y'_{VI} \end{bmatrix} = F \begin{bmatrix} x_c/z_c \\ y_c/z_c \end{bmatrix} \tag{26}$$

Based on this relationship, the location of a real-world point in the camera frame can be directly related to a corresponding point on the virtual image plane. However, as Equation (26) makes clear, the three-dimensional real-world location of a known point on the virtual image plane can not be directly deduced without knowledge of the physical focal length of the camera, and, critically, the z-coordinate of that point in the camera frame. When the value of z_c is unknown, this problem is commonly referred to as depth ambiguity.

2.3.3 Normalized and Pixel Coordinates.

Since digital images are most commonly expressed in pixels, it is much more convenient to express the coordinates of our virtual image plane in units of pixels. This subsection outlines how one obtains pixel coordinates for an image. Dividing Equation (26) by the focal length results in what is termed normalized coordinates [32], [33], [4]:

$$\mathbf{x}^{n} = \begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} x_c/z_c \\ y_c/z_c \end{bmatrix} \tag{27}$$

Where $[x_n, y_n]^T$ is the resulting normalized coordinate. The origin of the normalized coordinates is located at the center of the virtual image plane.

In practice, these normalized coordinates are typically converted to pixel coordinates for convenience. This is done by shifting the origin to the top left corner of the virtual image plane, as depicted in Figure 18. In the figure, $\mathbf{x}^p$ represents the point

of interest in the image in pixel coordinates, and the terms n_x and n_y represent the number of pixels along the x and y pixel axes of the virtual image plane [4].

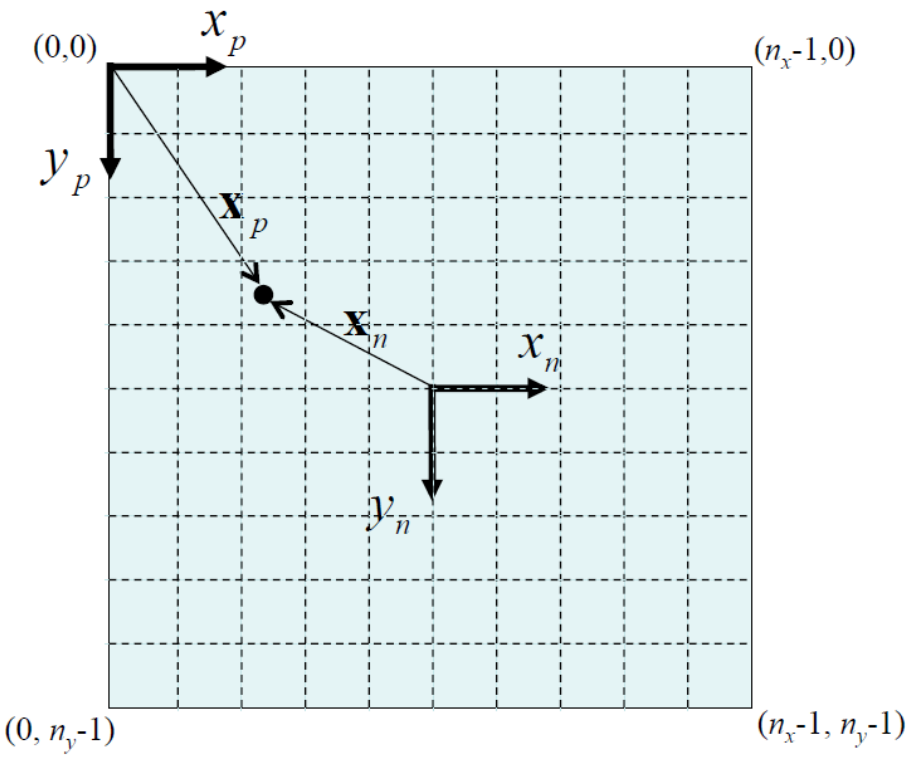


Figure 18. Raquet's Depiction of the Relationship Between Pixel and Normalized Coordinates [4]

.

Mathematically, pixel coordinates and normalized coordinates are related by the focal length of the camera, f (expressed in units of pixels), and the location of the optical center of the image, $[n_x, n_y]^T$ (expressed in units of pixels) [4]:

$$\mathbf{x}^p = \begin{bmatrix} x_p \\ y_p \end{bmatrix} = \begin{bmatrix} f & 0 \\ 0 & f \end{bmatrix} \begin{bmatrix} x_n \\ y_n \end{bmatrix} + \begin{bmatrix} n_x/2 \\ n_y/2 \end{bmatrix} \tag{28}$$

The focal length in pixels, f, is directly related to the physical focal length of the camera in units of millimeters (referred to as F in the preceding description of the

pinhole camera model) according to the relationship [32]:

$$f = Fs \tag{29}$$

Where s represents the number of pixels per millimeter on the actual camera sensor. When dealing with real-world cameras, it is not uncommon that s will differ between the x and y axes of the camera sensor. Hence, two different pixel focal lengths are defined where f_x is the focal length in pixels along the x-axis and f_y is the focal length in pixels along the y-axis.

This same relationship can be expressed in homogeneous coordinates. Homogeneous coordinates merely add an additional unitary dimension to the coordinate definitions described above, but enable one to express the relationship given in Equation (28) as a single multiplication instead of a multiplication and an addition. Incorporating this change, Equation (28) becomes [4]:

$$\underline{\mathbf{x}}^p = \begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \alpha_c f_x & x \\ 0 & f_y & y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} x_n \\ y_n \\ 1 \end{bmatrix} \tag{30}$$

Where α_c is a skew coefficient corresponding to the angle between the x and y pixel axes [34], [35], [4]. Typically this coefficient can be assumed to be zero in modern cameras, but it can also be estimated with most commonly implemented camera calibration tools to be described later [34]. This coefficient is related to ϕ, the angle between the x and y axes of the physical sensor, by the Equation $\alpha_c = \tan(\pi/2 - \phi)$ [34], [35]. In the Equation, the terms x and y are the x and y pixel coordinates for the principal point in the image plane, where $x = n_x/2$ and $y = n_y/2$ for an ideal camera.

By combining the Equation (27) and Equation (30), the relationship between a homogeneous set of three-dimensional camera frame coordinates and homogeneous pixel coordinates can be defined as [4]:

$$\underline{\mathbf{x}}^p = \frac{1}{z_c}\mathbf{K}\left[\mathbf{I}_{3x3}|\mathbf{0}_{3x1}\right]\underline{\mathbf{X}}^c \tag{31}$$

For clarity, the homogeneous world coordinates are $\underline{\mathbf{X}}^c = [x_c, y_c, z_c, 1]^T$ and the homogeneous pixel coordinates are $\underline{\mathbf{x}}^p = [x_p, y_p, 1]$. Again, z_c is typically unknown when examining an image produced by a camera. Therefore, in absence of knowledge of the value of z_c, Equation (31) does not allow one to relate pixel coordinates directly to a unique three-dimensional point in the camera frame.

The relationship between the camera frame coordinates of a point of interest and the coordinates of the same point in an arbitrary world frame is given by the equation [4]:

$$\mathbf{X}^c = \mathbf{R}_w^c\left(\mathbf{X}^w - \mathbf{C}^w\right) \tag{32}$$

Where $\mathbf{X}^c$ is the point's camera frame coordinates, $\mathbf{R}_w^c$ is the DCM from the world frame to the camera frame, $\mathbf{X}^w$ is the point's world frame coordinates, and $\mathbf{C}^w$ is the location of the camera frame origin in the expressed in the world frame. Converting this result to homogeneous coordinates and combining it with Equation (31) one can obtain:

$$\underline{\mathbf{x}}^p = \frac{1}{z_c}\mathbf{K}\left[\mathbf{R}_w^c|\mathbf{t}^c\right]\underline{\mathbf{X}}^w \tag{33}$$

Where $\mathbf{t}^c = -\mathbf{R}_w^c\mathbf{C}^w$ is the location of the world origin expressed in the camera frame [4].

The product $\mathbf{K}\left[\mathbf{R}_w^c|\mathbf{t}^c\right]$ can be combined into a single matrix called the projection matrix:

$$\mathbf{P} = \mathbf{K}\left[\mathbf{R}_w^c|\mathbf{t}^c\right] \tag{34}$$

The projection matrix, $\mathbf{P}$, describes how a real-world point (aside from scale) is projected into the image plane.

Based on this equation, one can relate the three-dimensional coordinates of a point in an arbitrary world frame to the pixel coordinates of the same point in a camera image. However, when the problem is reversed (going from pixel coordinates to world coordinates) the problem of depth ambiguity remains in the absence of additional information.

2.3.4 Intrinsic Camera Calibrations.

In equations (30), (31), and (33), the matrix $\mathbf{K}$ is called the camera calibration matrix:

$$\mathbf{K} = \begin{bmatrix} f_x & \alpha_c f_x & x \\ 0 & f_y & y \\ 0 & 0 & 1 \end{bmatrix} \tag{35}$$

Each element in the matrix is typically not directly observable or known (due to camera manufacturing defects, error tolerances, etc.) but is instead attained from an intrinsic camera calibration.

Additionally, real-world cameras will distort an image such that it differs from the image an ideal camera would produce. There are two components of distortion, radial and tangential. Radial distortion describes image distortions caused by lens effects while tangential distortion describes image distortions cause by the lens and imaging plane not being perfectly parallel [34], [32], [4]. Intrinsic camera calibration routines return parameters describing these two components of distortion.

For the intrinsic camera calibration methods used in this thesis, the radial and tangential distortion model is:

$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = \left(1 + k_1 r^2 + k_2 r^4 + k_5 r^6\right) \begin{bmatrix} x_n \\ y_n \end{bmatrix} + \begin{bmatrix} 2k_3 x_n y_n + k_4 \left(r^2 + 2x_n^2\right) \\ k_3 \left(r^2 + 2y_n^2\right) + 2k_4 x_n y_n \end{bmatrix} \tag{36}$$

Where x_d and y_d represent the undistorted normalized coordinates of a point of interest, the parameters k_1, k_2, and k_5 are radial distortion parameters, the parameters k_3 and k_4 are tangential distortion parameters, $r = \sqrt{x_n^2 + y_n^2}$, and x_n and y_n are the raw (i.e. distorted) normalized coordinates of a point in the raw image. A common convention for non-wide angle, modern cameras is to set the k_5 term and tangential distortion terms to zero since such cameras typically lack higher order radial distortion and tangential distortion [34].

Jean-Yves Bouguet of the California Institute of Technology developed a tool to perform intrinsic camera calibrations utilizing these models in MATLAB® and his process has also been implemented in OpenCV (OpenCV is an open source computer vision toolbox implemented in C and C++) [34], [32]. The MATLAB® implementation was used in this thesis. The algorithm utilizes multiple images of a checkerboard of known dimensions. These images are taken at various poses. Bouget's calibration algorithm is based primarily on the work of Brown [36], Heikkila [37], and Zhang [38] and returns the following:

1. An estimate and an associated uncertainty for each element in the camera calibration matrix given in Equation (35).

2. An estimate and an associated uncertainty for each distortion coefficient from Equation (36).

3. A DCM, $\mathbf{R}_c^b$, describing the rotation from the camera frame to the checkerboard frame, and a translation vector $\mathbf{B}^c$ describing the location of the checkerboard origin in the camera frame.

The specifics of how these parameters are attained is not discussed in this thesis. However, Bradski presents a very useful discussion of the algorithm in [32].

With these camera calibration parameters in hand, a raw image can be undistorted and three-dimensional world camera coordinates can be related to undistorted pixel coordinates in an image captured by the camera.

2.3.5 Extrinsic Camera Calibrations.

To make imagery useful for navigation, the camera frame often must be related to an arbitrary world frame as described by Equation (32). Typically, this frame would be the most useful frame for describing the position and orientation of the body to which the camera is attached. The quantities of interest are the DCM describing the relative orientation of the camera frame and the world frame, $\mathbf{R}_c^w$, and the position of the camera origin in the world frame, $\mathbf{C}^w$.

Figure 19 shows the translation vectors and DCMs of interest to a stereo camera configuration where the p-frame is the world frame of interest. In the figure, the two DCMs, $\mathbf{R}_L^p$ and $\mathbf{R}_R^p$, characterize the rotation between the p-frame and the left and right camera frames, respectively. The two translation vectors, $\mathbf{C}_L^p$ and $\mathbf{C}_R^p$, specify the locations of the left and right camera origins, respectively, in the p-frame. The image planes for the left and right cameras are depicted in red and blue, respectively. The three-dimensional point $\mathbf{X}$ corresponds to the pixel points $\mathbf{x}_p^L$ and $\mathbf{x}_p^R$. The relationship between the various coordinizations of $\mathbf{X}$ ($\mathbf{X}^L$, $\mathbf{X}^R$, and $\mathbf{X}^p$) can be deduced from the extrinsic camera calibration parameters.

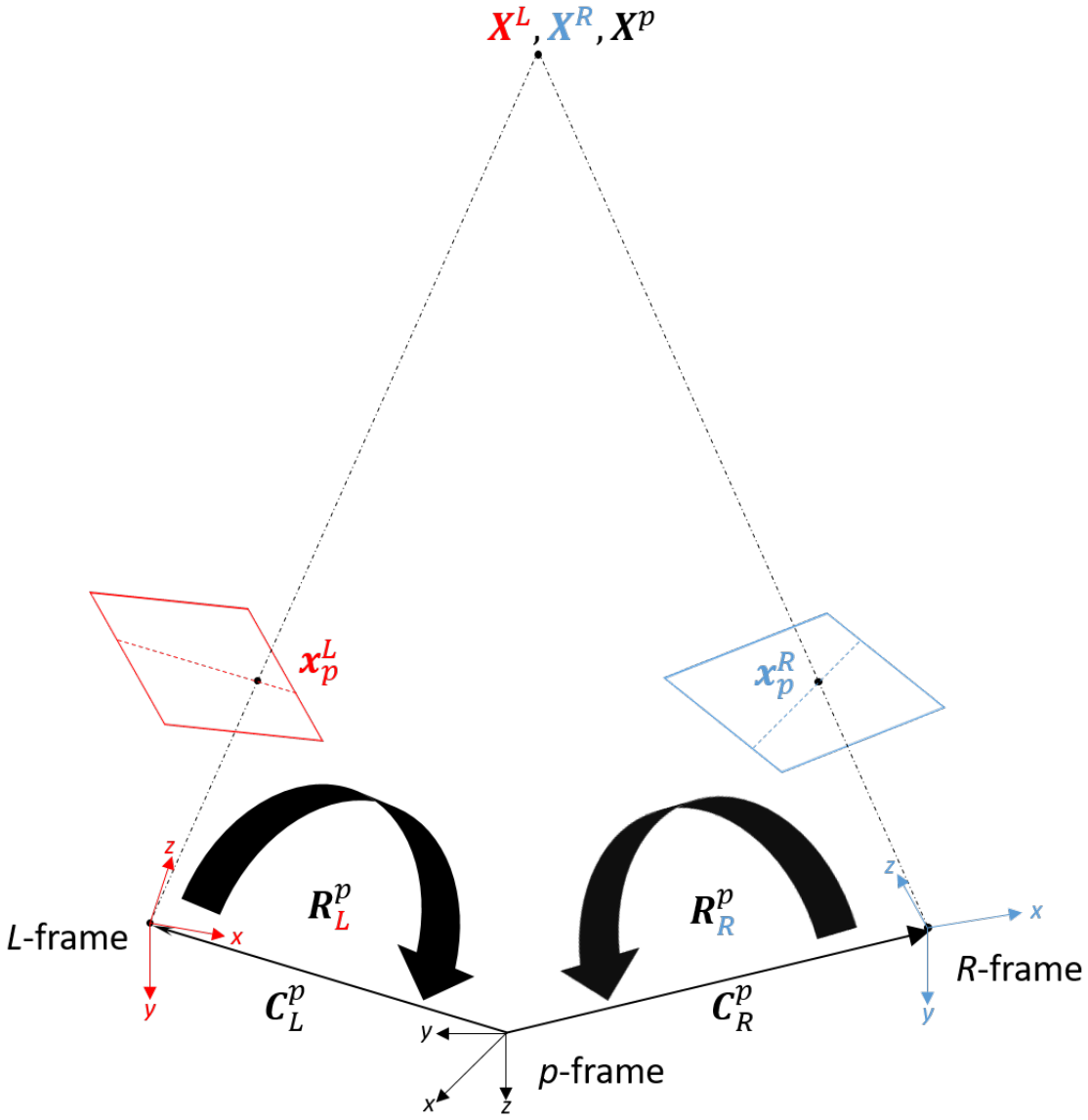


Figure 19. Extrinsic Camera Calibration Parameters.

Numerous approaches exist to conducting extrinsic camera calibrations, two of which are described below. In flight testing, this thesis utilized the direct measurement approach.

2.3.5.1 Direct Measurement.

In this approach the lever arm in the world frame, $\mathbf{C}^w$, from the origin of the world frame to the origin of the camera frame is measured directly as is the DCM from the world frame to the camera frame, $\mathbf{R}_w^c$. This method relies upon the assumption that the external camera body corresponds ideally to the imaging plane (i.e. the imaging plane is normal to the sides of the camera body) and that the imaging plane is located precisely where the camera manufacturer says it is located. Hence, the accuracy of this method is dependent upon the precision of the manufacturing process used to construct the cameras.

2.3.5.2 Checkerboard Image Based Extrinsic Camera Calibration.

This approach takes advantage of the Bouguet's intrinsic camera calibration algorithm to deduce the lever arm, $\mathbf{C}^w$, and the DCM, $\mathbf{R}_w^c$. Recall that in addition to returning the camera calibration matrix and the distortion coefficients, Bouguet's algorithm also returns the translation between the camera and the checkerboard in the camera frame, $\mathbf{B}^c$, and the DCM from the camera frame to the checkerboard frame, $\mathbf{R}_c^b$ for each image used in the calibration. Therefore, if one of these images is taken at a known position and orientation of the checkerboard relative to the world frame ($\mathbf{B}^w$ and $\mathbf{R}_w^b$, respectively), then $\mathbf{C}^w$ and $\mathbf{R}_w^c$ can be deduced as follows:

$$\mathbf{R}_c^w = \mathbf{R}_b^w \mathbf{R}_c^b \tag{37}$$

$$\mathbf{C}^w = \mathbf{B}^w - \mathbf{R}_c^w \mathbf{B}^c \tag{38}$$

In practice, the position and orientation of the checkerboard in the world frame ($\mathbf{T}^w$ and $\mathbf{R}_w^b$, respectively) can be determined either by placing the checkerboard at a precisely known location and orientation. This process will inherently involve taking

direct measurements similar to those described above. However, these measurements may be less prone to error since one need not directly measure the position and orientation of a camera frame.

2.3.6 Stereo Vision.

As discussed in Section 2.3, the depth of a three-dimensional point corresponding to a pixel in a single image is unknown. However, given the following information from two images, one can deduce depth information:

1. Two images of the same scene, taken from different vantage points.
2. Knowledge of which specific pixels in the first image match with which specific pixels in the second image.
3. Knowledge of the relative translation and rotation between the two cameras.

Based on this information, one can obtain the three-dimensional points locations for each pair of matched pixels. Note that more than two images could be employed in a similar fashion, but this research only focuses on the results attainable from a stereo camera pair. This section describes this process.

2.3.6.1 Epipolar Geometry.

Images captured by stereo camera pairs obey epipolar geometry. Szeliski's discussion of epipolar geometry in [39] is summarized in this section as is information from Raquet found in [4]. Figure 20 depicts epipolar geometry in the same fashion as these two authors. Epipolar geometry can be usefully employed to derive depth information about scenes captured by stereo camera pairs. Essentially, epipolar geometry is a special case of planar geometry pertaining to a stereo camera pair.

In Figure 20, the left camera origin is $\mathbf{L}_0$, and right camera origin is $\mathbf{R}_0$. The image planes of the respective cameras are depicted in front of the camera centers. The projection of $\mathbf{R}_0$ in the left camera's image plane is called an epipole (depicted as $\mathbf{e_L}$ in Figure 20). Likewise, the epipole $\mathbf{e_R}$ is the projection of $\mathbf{L}_0$ in the right camera's image plane. The vector $\mathbf{T^R}$ specifies the translation from the right camera origin to the left camera origin expressed in the right camera frame. The magnitude of this vector is also referred to as the baseline. The matrix $\mathbf{R_L^R}$ is the DCM specifying the rotation from the left camera to the right camera [39].

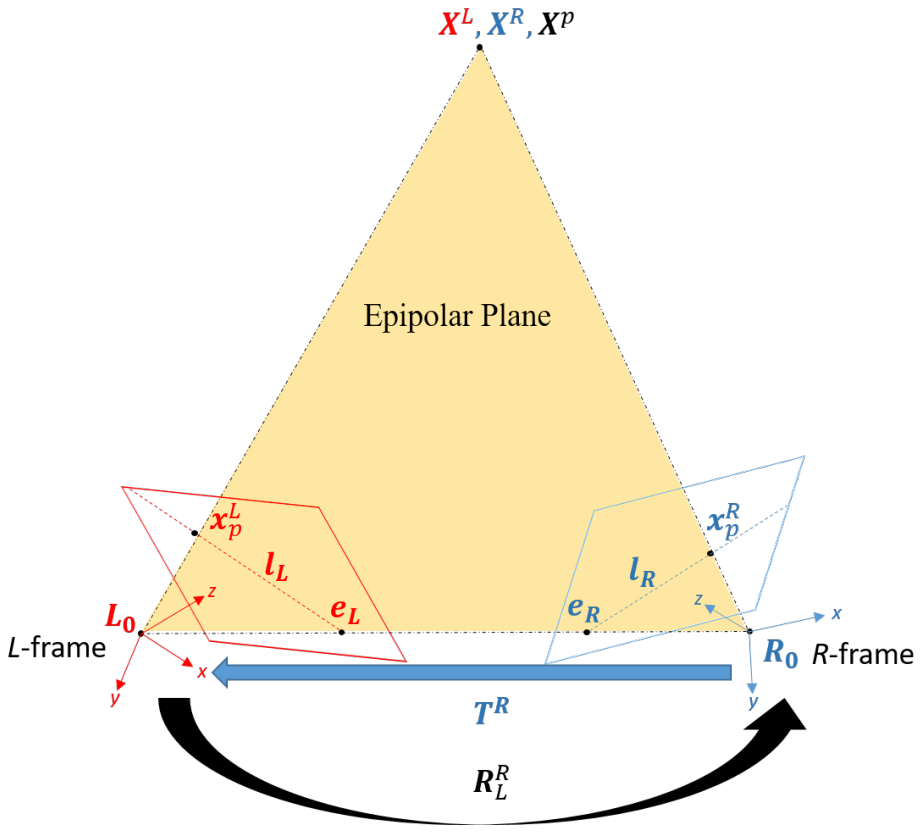


Figure 20. Depiction of Epipolar Geometry.

Assume that there is a real-world point of interest, $\mathbf{X^p}$, common to both images expressed in the arbitrary p-frame. This point could also be expressed in the left and right camera frames as $\mathbf{X^L}$ and $\mathbf{X^R}$, respectively. This point manifests as the

pixel $\mathbf{x_p^L}$ in the left camera's image plane. The epipolar plane is then defined by the baseline (the line segment formed by the vector $\mathbf{T^R}$), and the ray extending from the camera center $\mathbf{L}_0$, through the pixel $\mathbf{x_p^L}$, through the point of interest $\mathbf{X^p}$ to infinity. Projecting the epipolar plane into the image plane of the right camera results in the epipolar line $\mathbf{l_R}$. Epipolar geometry dictates that the pixel corresponding to $\mathbf{X^p}$ must fall along the epipolar line segment $\mathbf{l_R}$ in right camera's image plane. The epipolar line segment is bounded on one end by the epipole $\mathbf{e_R}$ and on the other end by the edge of the right camera's image plane.

The exact location of where this pixel falls on the epipolar line segment directly depends upon the geometry of the point of interest $\mathbf{X^p}$ relative to the cameras, the relative geometry of the stereo camera pair as defined by $\mathbf{R_L^R}$ and $\mathbf{T^R}$, and the intrinsic properties of the cameras. In the specific situation shown in Figure 20, the pixel $\mathbf{x_p^R}$ is the pixel in right camera's image plane that corresponds to $\mathbf{X^p}$ as a result of the particular geometry depicted [39].

It is important to note that the precise location of $\mathbf{x_p^R}$ can not be known unless the precise location of $\mathbf{X^p}$ is known. Lacking this data, one again is confronted with the problem of depth ambiguity discussed in Section 2.3. However, in the case of a stereo camera system, even if one only knows where $\mathbf{x_p^L}$ falls in the left camera's image plane, one does know with certainty that $\mathbf{x_p^R}$ must fall on the epipolar line $\mathbf{l_R}$ in the right camera image [4], [39].

This is the key takeaway from the foregoing discussion of epipolar geometry. If one knows that the matching pixel to $\mathbf{x_p^L}$ must fall on $\mathbf{l_R}$ and knows the geometry defining $\mathbf{l_R}$, then one can much more easily employ a pixel matching algorithm to identify the correct pixel (i.e. $\mathbf{x_p^R}$). The following three points from Bradski and Kaehler nicely summarize this particular utility of epipolar geometry [32]:

1. Every real-world, three-dimensional point that is in view of both cameras is located in an epipolar plane. This epipolar plane intersects both image planes along an epipolar line.

2. The epipolar constraint dictates that the matching pixel in one image for a pixel of interest in the other image must fall along the corresponding epipolar line.

3. Based on the epipolar constraint, one can much more easily employ a pixel matching algorithm to correctly identify matching pixel pairs. This is because one only needs to search along the proper epipolar line instead of through the entire image.

2.3.6.2 The Essential and Fundamental Matrices.

Mathematically, the epipolar constraint is applied with the essential and fundamental matrices. The essential matrix, **E**, and fundamental matrix, **F**, both relate the epipolar geometry of a stereo camera system. They differ in that the fundamental matrix also incorporates the intrinsic properties of the two cameras in the system, while the essential matrix relates only the geometry between the two cameras. Bradski and Kaehler's discussion of these properties is summarized in this section [32]. One obtains these matrices via application of the following definition of a plane:

$$(\mathbf{x} - \mathbf{a}) \cdot \mathbf{n} = 0 \tag{39}$$

Where $\mathbf{x}$ is any point in the plane, $\mathbf{n}$ is a vector normal to that plane, and $\mathbf{a}$ is the point at which $\mathbf{n}$ passes through the plane.

Applying Equation (39) to the special case of epipolar geometry, one can assert [32], [4]:

$$\left(\mathbf{X}^L - \mathbf{T}^L\right)^T \cdot \left(\mathbf{T}^L \times \mathbf{X}^L\right) = 0 \tag{40}$$

Where $\mathbf{X}^L$ is the three-dimensional coordinates of the point of interest in the left camera frame, and $\mathbf{T}^L$ is the vector from the left camera origin to the right camera origin expressed in the left camera frame. Note that the endpoints of both $\mathbf{X}^L$ and $\mathbf{T}^L$ fall in the same epipolar plane by definition.

A three-dimensional point in the right camera frame, $\mathbf{X}^R$, is related to a point in the left camera frame according to:

$$\mathbf{X}^R = \mathbf{R}_L^R \left(\mathbf{X}^L - \mathbf{T}^L \right) \tag{41}$$

Therefore,

$$\mathbf{X}^{R^T} \mathbf{R}_L^R = \left(\mathbf{X}^L - \mathbf{T}^L \right)^T \tag{42}$$

Thus, one can relate the the planar definition of epipolar geometry given in Equation (40) with reference to the point of interest expressed in both the left and right camera frames [32]:

$$\mathbf{X}^{R^T} \mathbf{R}_L^R \left(\mathbf{T}^L \times \mathbf{X}^L \right) = 0 \tag{43}$$

Applying the skew symmetric matrix form of the cross product in Equation (43):

$$\mathbf{X}^{R^T} \mathbf{R}_L^R \mathbf{T}_\times^L \mathbf{X}^L = 0 \tag{44}$$

Equation (44) can be converted to homogeneous normalized coordinates via division by $Z_L Z_R / (F_L F_R)$, where Z_L and Z_R are the z-coordinates of the points $\mathbf{X}^L$ and $\mathbf{X}^R$, respectively, and F_L and F_R are the physical focal lengths (expressed in units of distance) of the left and right cameras, respectively. Applying this divisor:

$$\underline{\mathbf{x}}_n^{R^T} \mathbf{R}_L^R \mathbf{T}_\times^L \underline{\mathbf{x}}_n^L = 0 \tag{45}$$

Where $\underline{\mathbf{x}}_n^R$ and $\underline{\mathbf{x}}_n^L$ are the homogeneous normalized coordinates of a point in the right and left image planes, respectively.

Now, the essential matrix, $\mathbf{E}$, can now be defined to be:

$$\mathbf{E} = \mathbf{R}_L^R \mathbf{T}_\times^L \tag{46}$$

Thus, the epipolar constraint can be defined in terms of the essential matrix as:

$$\underline{\mathbf{x}}_n^{R^T} \mathbf{E} \underline{\mathbf{x}}_n^L = 0 \tag{47}$$

Again, the essential matrix, as the above derivation makes clear, only relates the planar geometry of the stereo camera system. Specifically, it is a product of the DCM describing the rotation between the two camera frames and the skew symmetric matrix corresponding to the translation between the two camera frames.

By applying the relationship between pixel and normalized coordinates, one can obtain the fundamental matrix which also contains information contained in the camera calibration matrices of both the left and right cameras. From Equation (30) recall that:

$$\underline{\mathbf{x}}_p = \mathbf{K} \underline{\mathbf{x}}_n \tag{48}$$

Where $\underline{\mathbf{x}}_p$ is a set of homogeneous pixel coordinates, $\mathbf{K}$ is the camera calibration matrix, and $\underline{\mathbf{x}}_n$ is a set of homogeneous normalized coordinates. Applying this definition to Equation (47):

$$\underline{\mathbf{x}}_p^{R^T} \left(\mathbf{K}_R^{-1}\right)^T \mathbf{E} \mathbf{K}_L^{-1} \underline{\mathbf{x}}_p^L = 0 \tag{49}$$

Where $\underline{\mathbf{x}}_p^R$ and $\underline{\mathbf{x}}_p^L$ are the homogeneous pixel coordinates of a point in the right and left image planes, respectively, and $\mathbf{K}_R$ and $\mathbf{K}_L$, are the right and left camera calibration matrices, respectively.

From this the fundamental matrix, $\mathbf{F}$, can now be defined as:

$$\mathbf{F} = \left(\mathbf{K}_R^{-1}\right)^T \mathbf{E}\mathbf{K}_L^{-1} \tag{50}$$

Thus the epipolar constraint can now be expressed in terms of the fundamental matrix:

$$\underline{\mathbf{x}}_p^{R^T} \mathbf{F} \underline{\mathbf{x}}_p^L = 0 \tag{51}$$

The fundamental matrix relates the planar geometry of the stereo camera system as well as the camera calibration matrices of both cameras. Specifically, it is a product of the DCM describing the rotation between the two camera frames, the skew symmetric matrix corresponding to the translation between the two camera frames, and the camera calibration matrices of both cameras. Hence, almost all information about the stereo camera system, aside from the distortion coefficients described in Section 2.3.4, is contained in the fundamental matrix, $\mathbf{F}$. To use $\mathbf{F}$ in the context of a camera with non-zero distortion coefficients, one would merely undistort both images as described in section 2.3.4, and apply these undistorted pixel coordinates in Equation (51).

In summation, Equations (47) and (51) state that the product of matching feature image coordinates with the essential or fundamental matrix ought to be zero. In reality this property will not hold perfectly when using real cameras and real images, but one should expect the products described by these two equations to be very near zero.

2.3.6.3 Stereo Camera Calibration.

In order to usefully employ the properties of epipolar geometry discussed in Section 2.3.6.1, one must have knowledge of the relationship between the left and right camera

frames. Mathematically, as the foregoing discussion has stated, the DCM $\mathbf{R}_L^R$ and the translation vector $\mathbf{T}^L$ describe this relationship. Solving for these parameters can be accomplished in one of two ways: either by applying the eight point algorithm to solve for $\mathbf{E}$ and then deducing $\mathbf{R}_L^R$ and $\mathbf{T}^L$ [4] , or with an algorithm, implemented by Bouguet, which makes use of the same procedure utilized for Bouguet's intrinsic camera calibration [32]. This thesis utilizes the second method because it yields only one solution (utilizing the eight point algorithm yields a translation vector of ambiguous sign), is less susceptible to outliers, and has been implemented in both OpenCV and MATLAB® [34], [32], [4].

The chosen method makes use of the fact that Bouguet's intrinsic camera calibration algorithm returns a translation vector, $\mathbf{B}^c$, to the checkerboard origin in the camera frame and a DCM, $\mathbf{R}_c^b$, from the camera frame to the checkerboard frame for each image (see Section 2.3.4). For the left and right cameras, the translation vectors are denoted as $\mathbf{B}^L$ and $\mathbf{B}^R$, respectively. Likewise, the DCMs are denoted as $\mathbf{R}_L^b$ and $\mathbf{R}_R^b$, respectively.

With these terms, one can solve for the DCM between the left and right camera frames, $\mathbf{R}_L^R$, and the translation between the two frames, $\mathbf{T}^R$ or $\mathbf{T}^L$, per the following [32]:

$$\mathbf{R}_L^R = \mathbf{R}_b^R \mathbf{R}_L^b \tag{52}$$

$$\mathbf{T}^R = \mathbf{B}^R - \mathbf{R}_L^R \mathbf{B}^L \tag{53}$$

$$\mathbf{T}^L = -\mathbf{R}_R^L \mathbf{T}^R \tag{54}$$

Thus, for any pair of intrinsic camera calibration images taken from the left and right cameras (note that these must be images of the same scene), one can solve for the stereo parameters $\mathbf{R}_L^R$ and $\mathbf{T}^R$. In Bouguet's application, any number of image pairs can be utilized. The Levenberg-Marquadt algorithm is applied iteratively to

the image set to return a solution for $\mathbf{R}_L^R$ and $\mathbf{T}^R$ which minimizes the error in the estimate (note that Levenberg-Marquadt returns a local minimum so the algorithm must be properly initialized) [32].

Additionally, applying this algorithm to stereo image pairs optionally enables simultaneous estimation of the camera calibration matrices for both cameras and the distortion coefficients for both cameras [34], [32]. To accomplish this, Bouguet's algorithm forces the absolute pose of the checkerboard to be the same in both the left and right camera images, and then optimizes estimated parameters based on this assumption [34]. To aid with convergence, it helps to seed the algorithm with an initial guess of both the camera calibration matrices and the distortion coefficients for both cameras [34], [32].

As with the discussion of intrinsic camera calibrations in Section 2.3.4, the details of the stereo calibration algorithm are not discussed here. However, it is worth emphasizing that this routine returns the following:

1. Optionally, the routine returns an estimate and an associated uncertainty for each element in the both the left and right camera calibration matrices parameterized as in Equation (35).

2. Optionally, the routine returns an estimate and an associated uncertainty for each distortion coefficient for both the left and right cameras parameterized as in Equation (36).

3. The routine returns a DCM describing the rotation from the camera frame to the checkerboard frame for both the left and right cameras, and a translation vector describing the location of the checkerboard origin in the camera frame for both the left and right cameras.

4. Critically, the routine returns an estimate of the DCM from the left camera frame to the right camera frame, $\mathbf{R}_L^R$, and the translation vector from the origin of the right camera frame to the origin of the left camera frame expressed in the right camera frame, $\mathbf{T}^R$.

If one already is confident of their estimates of the camera calibration matrices and distortion coefficients, then the algorithm can hold other parameters fixed and only return an estimate of $\mathbf{R}_L^R$ and $\mathbf{T}^R$.

2.3.6.4 Stereo Image Rectification.

The goal of stereo image rectification is to transform the left and right camera image such that corresponding epipolar lines fall along the same pixel rows (scan lines) in both rectified images. As discussed at the end of Section 2.3.6.1, pixels corresponding to the same feature in both images must fall along the corresponding epipolar lines. Hence, if images are rectified, these matching pixels must fall along the same pixel row in both images, and the search process to identify matching pixels will be greatly simplified [39].

Physically, a real world stereo camera pair would generate rectified imagery if the cameras were frontal parallel, had the same focal lengths, had the same principal points, and were both free of distortion [32]. In frontal parallel cameras, the image planes are row aligned and parallel, and both optical axes are parallel as well. This situation is depicted in Figure 21.

In the figure, cameras are depicted as gray boxes and virtual image planes are shown as blue rectangles. The principal points of the two cameras are $(c_{x_{R_r}}, c_{y_{R_r}})$ and $(c_{x_{L_r}}, c_{y_{L_r}})$. Epipolar lines are depicted as dashed lines. Corresponding epipolar lines fall along the same pixel row in both images. The vector $\mathbf{T}^{R_r}$ is the baseline between

the cameras. The baseline falls completely along the x-axis of both cameras. The DCM between the two cameras, $\mathbf{R}_{R_r}^{L_r}$, is equal to the identity matrix, $\mathbf{I}_3$.

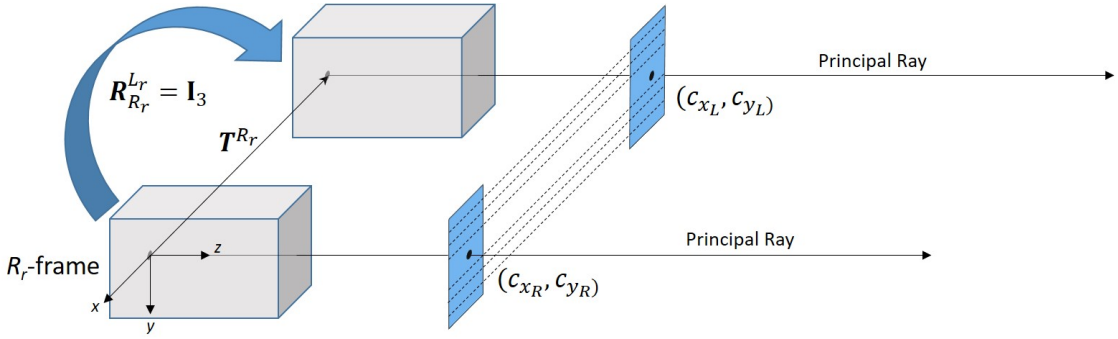


Figure 21. Depiction of Frontal Parallel Cameras.

When using actual cameras, it is likely impossible to definitively place a stereo camera system in a frontal parallel arrangement due to imperfections in manufacturing, the difficultly in definitively determining the location of a camera's imaging plane, etc. Instead, it is much more feasible to correct for these deficiencies with a rectification algorithm.

Like the camera calibration algorithms discussed above, this thesis utilizes a rectification algorithm developed by Bouguet [34], [32]. The precise details of the process are not covered in this thesis, but a more detailed discussion of the algorithm can be found in Bradski and Kaehler's work [32]. Other implementations have been developed including by Fusiello, Trucco, and Verri [40]. Bouguet's is selected for its efficiency and readily available implementation in both OpenCV and MATLAB®.

Based on the stereo calibration parameters, $\mathbf{R}_L^R$ and $\mathbf{T}^R$, Bouguet's rectification algorithm seeks to reproject the left and right camera images into a rectified image pair. In doing so, the algorithm is designed to minimize the change in the images required to do so (thereby reducing consequent distortions) and to maximize the common viewing area of the resulting rectified images [32].

Essentially, the algorithm operates by ensuring the optical axes of both cameras are parallel and that the x-axes of both cameras are parallel with the baseline. This is accomplished via computation of a rectification DCM for the left and right images. Additionally, the algorithm returns a DCM for each camera that relates each camera frame to the frame of each rectified (frontal parallel) camera. In this thesis, these DCMs are called $\mathbf{R}_L^{L_r}$ and $\mathbf{R}_R^{R_r}$. This notation specifies a rotation from the camera frame to the rectified camera frame for the left and right camera frames, respectively. These DCMs are computed as [32]:

$$\mathbf{R}_L^{L_r} = \mathbf{R}_{\text{rect}}\mathbf{R}_l \tag{55}$$

$$\mathbf{R}_R^{R_r} = \mathbf{R}_{\text{rect}}\mathbf{R}_r \tag{56}$$

Here, $\mathbf{R}_{\text{rect}}$ is a rectification DCM applicable to both the left and right cameras. The DCM $\mathbf{R}_l$ applies to the left camera and accounts for half of the rotation specified by the stereo calibration DCM $\mathbf{R}_L^R$. The DCM $\mathbf{R}_r$ applies to the right camera and accounts for half of the rotation specified by the stereo calibration DCM $\mathbf{R}_R^L$. Of importance for the forthcoming discussion in Section 2.3.6.6, applying $\mathbf{R}_L^{L_r}$ and $\mathbf{R}_R^{R_r}$ to the left and right cameras would result in the camera images becoming rectified.

Figure 22 shows an example of the rectification DCMs for a pair of stereo cameras. In the figure, the actual virtual image planes of the cameras are depicted in red. The rectified virtual image planes are depicted in black. Epipolar lines are depicted as dashed lines. In the non-rectified images, corresponding epipolar lines are not parallel. In the rectified images, corresponding epipolar lines fall along the same pixel row (scan line) in both images. The rectification DCMs, $\mathbf{R}_L^{L_r}$ and $\mathbf{R}_R^{R_r}$, describe the relationship between the non-rectified and rectified camera frames. The vector $\mathbf{T}^R$ is the baseline between the cameras expressed in the non-rectified right camera fame. It does not

necessarily fall only along the x-axis of the right camera frame since the cameras are not frontal parallel. The stereo camera calibration DCM, $\mathbf{R}_L^R$, describes the rotation between the non-rectified left and right cameras, and is not necessarily equal to the identity matrix. The point $\mathbf{X}$ could be expressed in any of the four frames depicted and would have different coordinizations in each frame. Similarly, the pixel points, such as $\mathbf{x}_p^L$ and $\mathbf{x}_p^{L_r}$, have different coordinates in the rectified and non-rectified images.

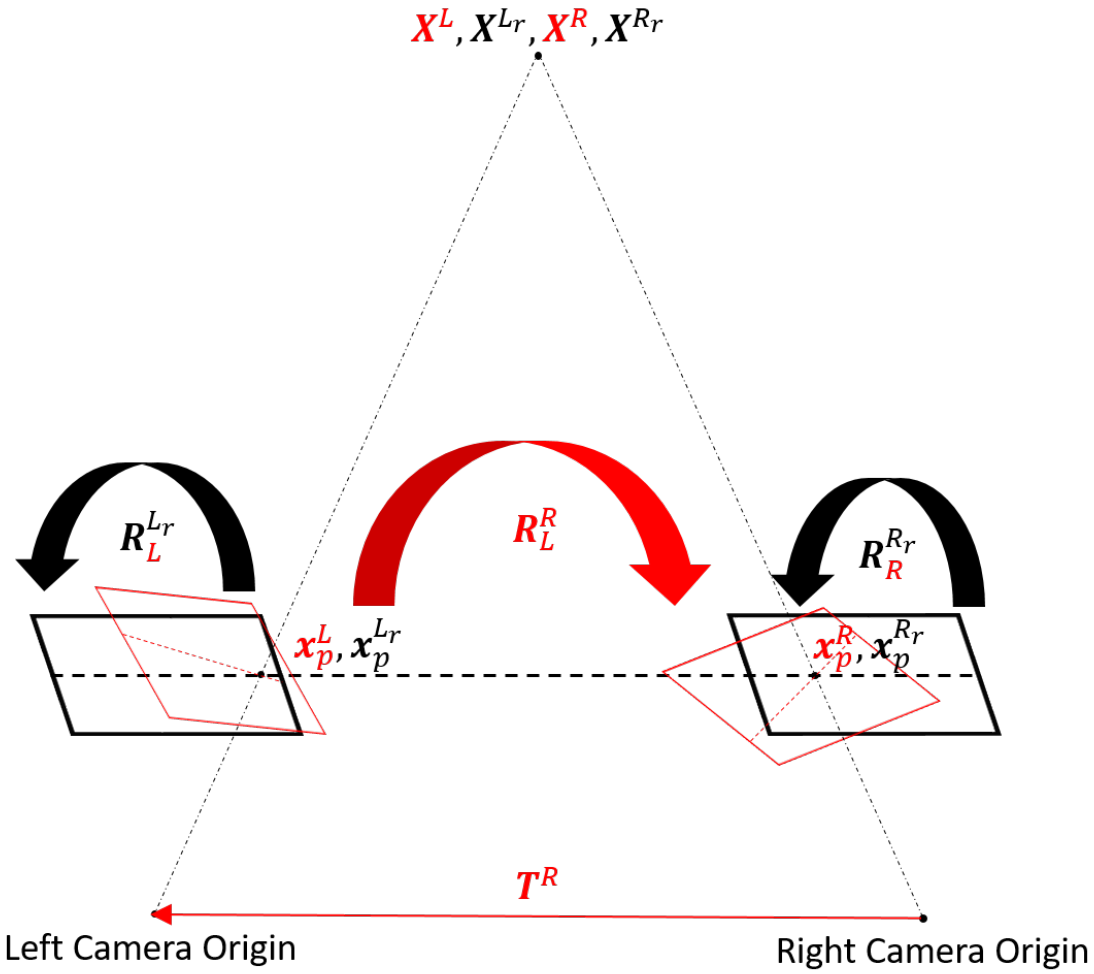


Figure 22. Depiction of Stereo Camera Image Rectification.

Besides the rectification DCMs described above, Bouguet's rectification algorithm also computes new camera calibration matrices, $\mathbf{K}_{L_r}$ and $\mathbf{K}_{R_r}$, for the left and right

rectified camera frames, respectively. An important consequence of the rectified camera calibration matrices is discussed in Section 2.3.6.6.

Once these DCMs and camera calibration matrices have been computed, the algorithm generates a pixel mapping from the raw, non-rectified images, to the rectified images. Both the rectification DCMs and the rectified camera calibration matrices are used to generate the pixel mapping. Based on this mapping, the algorithm determines what pixel in each raw image corresponds to a given pixel in the newly created rectified image. In this manner, the rectified image is computed pixel-by-pixel.

To summarize, in a non-rectified stereo image pair, common features will fall along corresponding epipolar lines, but these epipolar lines will not fall along the same rows of pixels (scan lines). By contrast, in a rectified pair of images, pixels corresponding to common features fall along the same pixel rows (horizontal scan lines) in both transformed images. This is because the corresponding epipolar lines are aligned in rectified images. In a rectified image pair, the horizontal displacement of common features in a scan line can be directly related to depth information.

2.3.6.5 Pixel Matching and Disparity Map Generation.

Once a stereo image pair has been rectified, identifying matching pixels present in the two images is greatly simplified. The relationship between common feature pixel coordinates in a rectified stereo image pair is given by [39]:

$$x^{R_r} = x^{L_r} - d(x^{L_r}, y^{L_r}),\ y^{R_r} = y^{L_r} \tag{57}$$

For a given feature, this value d can also be expressed as:

$$d = x^{L_r} - x^{R_r} \tag{58}$$

Here, the x-axis of each image plane is aligned with the horizontal and the y-axis with the vertical. The variables x^{L_r} and y^{L_r} represent the pixel coordinates of the feature of interest in the left rectified camera image while the variables x^{R_r} and y^{R_r} represent the pixel coordinates of the feature of interest in the right rectified camera image. The value $d(x^{L_r}, y^{L_r})$ is termed the disparity value. The disparity is equal to the difference in the x-location of the feature in the left rectified image relative to the x-location of the feature in the right rectified image. Once rectification has been done, one can obtain disparity values of all common features in the rectified stereo image pair and store these values as a disparity map [39].

The disparity map is the same size as each rectified image and uses the coordinate system of the left rectified image. Each pixel is assigned a value equal to the disparity observed at that pixel. Since the algorithm operates on rectified images, pixel matches must occur in the same row.

The disparity map algorithm used in this thesis is an OpenCV implementation based upon the work of Konolige [41]. This method uses local block matching, as opposed to global or semi-global block matching, as computational speed was considered desirable for both the V5 and R7 algorithms. Prioritizing speed makes the method more readily applicable to real-time implementation of AAR, relative navigation, and/or automated formation flight. For completeness and the sake of comparison, the added features that would be included in a semi-global block matching algorithm are also discussed in this section.

The algorithm has three principle steps [32], [41], [42]:

1. For the semi-global matching technique only, each image is filtered to normalize image brightness and increase texture. This is accomplished by applying a window of user-specified size over the image. Within each window, the center

pixel's intensity is normalized according to the equation:

$$I_{out} = \min[\max(I_{in} - \bar{I}, -I_{max}), I_{max}] \tag{59}$$

Where I_{out} is the output intensity of the center pixel, I_{in} is the input intensity of the center pixel, $\bar{I}$ is the mean pixel intensity within the window, and I_{max} is a user-specified positive intensity limit.

2. A sum of absolute differences (SAD) window is used to search along each horizontal epipolar line and identify matching pixels. The user specifies a minimum and maximum disparity value for the search. Below, these are termed d_0 and d_1, respectively. The algorithm uses each pixel in the left image as a starting point. For reference, say this pixel is at position (x_l, y_l) in the left image. For every disparity value, d, in the inclusive set $[d_0, d_1]$, the algorithm computes a SAD metric. The SAD metric sums the absolute differences for a window of size n-by-n, where n must be odd. The window is centered on pixel (x_l, y_l) in the left image and on pixel $(x_l - d, y_l)$ in the right image. The absolute difference of intensities between pixel pairs of the form $(x_l + i, y_l + j)$ in the left image and $(x_l + i - d, y_l + j)$ in the right image are computed, for all pixels included in the set $-m < i < m$, $-m < j < m$, where $m = (n-1)/2$. These absolute intensity differences are summed together to generate the SAD metric. Mathematically, this can be expressed as [43],:

$$\mathrm{SAD}(x_l, y_l, d) = \sum_{i=-m}^{m} \sum_{j=-m}^{m} |I_L(x_l + i, y_l + j) - I_R(x_l + i - d, y_l + j)| \tag{60}$$

In this manner the algorithm generates a SAD metric for all candidate disparity values in the set $[d_0, d_1]$. The algorithm also estimates intensity differences

at sub-pixel increments of 1/16 by fitting a curve to the intensity difference computed at each pixel. The right image pixel that minimized the SAD metric is declared as a preliminary match with the corresponding left image pixel. The preliminary disparity value is the difference between the right and left image x-coordinates of the matched pixels.

Within this step, the algorithm applies logic to enforce the geometry of the features being observed. Specifically, this is done with an order constraint which dictates that the order of image features must be preserved in the left and right images. Say the algorithm has already identified feature A as being located at pixel (x_l, y_l) in the left image and pixel $(x_l - 3, y_l)$ in the right image. Say a second feature, feature B, has been preliminarily declared. The algorithm will reject the feature B pixels as a valid match if the SAD window search has returned a matching pair at pixel $(x_l + 1, y_l)$ in the left image and pixel $(x_l - 4, y_l)$ in the right image. Doing so would mean the feature B was to the right of feature A in the left image, but to the left of feature A in the left image. Hence, the matching process ensures that feature order is preserved.

If no pixel pair is identified for a given pixel location in the left image due to rejection, then no preliminary disparity value is assigned. Disparity values for these pixels are assigned in the final step of the algorithm.

3. The algorithm applies logic to eliminate bad matches. In both block matching and semi-global block matching algorithms, a speckle filter is applied. Speckle filters seek to eliminate extreme disparity values that occur at the boundary of objects. The filter works by examining contiguous regions of preliminary disparity values. The maximum size of these regions (in pixels) is user specified and is termed the speckle window size. Regions larger in size than this value are not speckle filtered. Within a speckle window, the difference between

the minimum and maximum observed disparity value must not exceed a user-specified threshold, called the maximum disparity difference. If the threshold is exceeded, all pixels within the blob are assigned a user-specified disparity value (typically zero).

If a semi-global block matching algorithm is used, then two additional filters are applied. First, any matches are eliminated that exceed a uniqueness ratio according to the following:

$$\mathrm{R} = (I_1 - I_{\min})/I_{\min} \tag{61}$$

Where R is the computed uniqueness ratio, I_1 is the intensity difference of the potential match, and $I_{\min}$ is the minimum intensity difference of all potential matches observed in the image. If R exceeds a user defined threshold, then the match corresponding to I_1 is thrown out. Second, if the intensity difference I_1 of a potential match exceeds a user-specified threshold, the match is rejected.

Finally, for both the block matching and semi-global matching methods, any pixels for which no disparity value has been assigned as a result of pixel matching are assigned a disparity value of zero.

The images shown in Figure 23 were used to generated the disparity map shown in Figure 24 with the MATLAB® function `disparity`. These images were generated in simulation and model the three-dimensional position of C-12C aircraft. As can be seen in the figures, more distant features have lower disparity values while closer features have greater disparity values. This property of disparity values can be used to deduce depth information.



Figure 23. Left and Right Images Used to Generate the Disparity Map Shown in Figure 24.

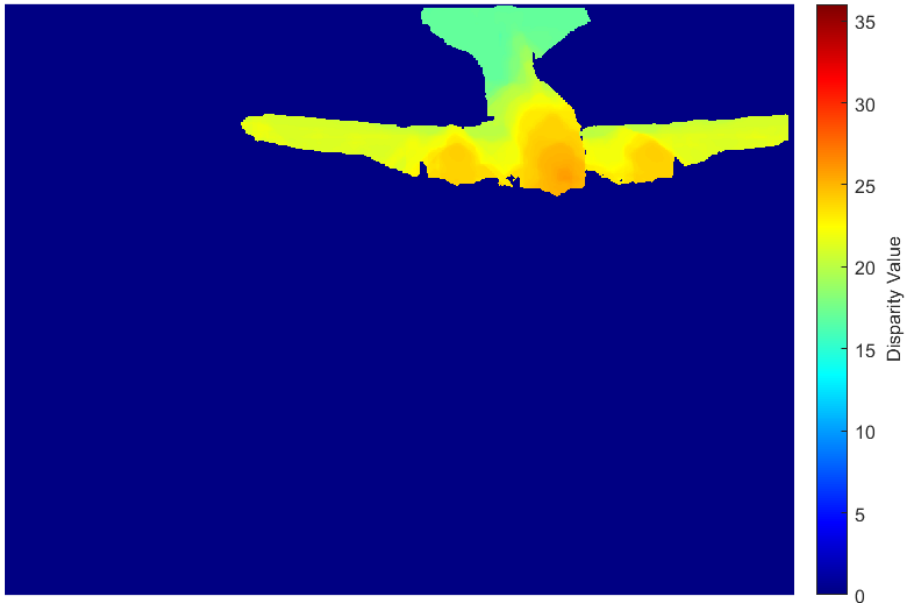


Figure 24. Disparity Map Generated from the Images in Figure 23.

2.3.6.6 Obtaining Depth Information from Disparities.

A disparity map can be reprojected into a three-dimensional point cloud. Each point in the three-dimensional point cloud represents the real-world point corresponding to every pixel in the disparity map for which there is a measured disparity value.

For points in the disparity map without a valid disparity value, no reprojection is computed.

The governing equation for reprojection is [32]:

$$\mathbf{Q}\begin{bmatrix} x_{L_r} \\ y_{L_r} \\ d \\ 1 \end{bmatrix} = \begin{bmatrix} t \\ u \\ v \\ w \end{bmatrix} \tag{62}$$

Where (x_{L_r}, y_{L_r}) is the pixel coordinate in the left rectified image of a pixel with a disparity value of d. The resulting parameters t, u, v, and w are discussed later in this section.

The matrix $\mathbf{Q}$ is the reprojection matrix. The values of the elements in $\mathbf{Q}$ are based upon the rectified camera calibration matrices discussed in Section 2.3.6.4. The $\mathbf{Q}$ matrix is parameterized as:

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 & -c_{x_{L_r}} \\ 0 & 1 & 0 & -c_{y_{L_r}} \\ 0 & 0 & 0 & f_c \\ 0 & 0 & -\frac{1}{T_x^{R_r}} & \frac{c_{x_{L_r}} - c_{x_{R_r}}}{T_x^{R_r}} \end{bmatrix} \tag{63}$$

Here, $(c_{x_{L_r}}, c_{y_{L_r}})$ is the location of the principal point in the left rectified image, f_c is the focal length in pixels of the left rectified camera, $c_{x_{R_r}}$ is the x-coordinate of the principal point in the right rectified image, and $T_x^{R_r}$ is the x-component of the translation vector $\mathbf{T}^{R_r}$ between the right and left rectified cameras. Note that due to the alignment of the each camera's x-axis in rectification, the magnitude of $T_x^{R_r}$ will be equal to the magnitude of the entire vector $\mathbf{T}^{R_r}$. This figure is equal to the baseline (i.e. the distance between the camera origins or the magnitude of $\mathbf{T}^R$) and

its sign will be negative. The sixteenth element of the matrix, $\frac{c_{x_{L_r}} - c_{x_{R_r}}}{T_x^{R_r}}$, will be equal to zero unless one deliberately utilizes a rectification scheme that does not simulate frontal parallel cameras.

This thesis utilizes a frontal parallel rectification, therefore:

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 & -c_{x_{L_r}} \\ 0 & 1 & 0 & -c_{y_{L_r}} \\ 0 & 0 & 0 & f_c \\ 0 & 0 & -\frac{1}{T_x^{R_r}} & 0 \end{bmatrix} \tag{64}$$

Applying Equation (62), yields the vector $[t, u, v, w]^T$. This vector is directly related to the three-dimensional coordinates of the rectified left camera frame according to the equation [32]:

$$\mathbf{X}^{L_r} = \begin{bmatrix} X_{L_r} \\ Y_{L_r} \\ Z_{L_r} \end{bmatrix} = \begin{bmatrix} t/w \\ u/w \\ v/w \end{bmatrix} \tag{65}$$

Expanding this relationship in terms of each element in Equation (62), the following relationship is established:

$$\mathbf{X}^{L_r} = \begin{bmatrix} X_{L_r} \\ Y_{L_r} \\ Z_{L_r} \end{bmatrix} = \begin{bmatrix} \frac{(c_{x_{L_r}} - x_{L_r})T_x^{R_r}}{d} \\ \frac{(c_{y_{L_r}} - y_{L_r})T_x^{R_r}}{d} \\ -\frac{f_c T_x^R}{d} \end{bmatrix} = \begin{bmatrix} \frac{(x_{L_r} - c_{x_{L_r}})Z_{L_r}}{f_c} \\ \frac{(y_{L_r} - c_{y_{L_r}})Z_{L_r}}{f_c} \\ -\frac{f_c T_x^R}{d} \end{bmatrix} \tag{66}$$

Recall that due to the alignment of the each camera's x-axis in rectification, the magnitude of $T_x^{R_r}$ will be equal to the baseline (i.e. the total separation of the cameras) and its sign will be negative. Additionally, for frontal parallel cameras the smallest disparity value observable is zero. Hence, Z_{L_r} will be positive by definition for frontal parallel cameras.

Since a disparity map corresponds to a digital image, the coordinates that comprise it are inherently discrete. Additionally, the set of possible disparity values any coordinate can have is a discrete set. Recall from Equation (58) that a disparity value is merely the difference, in pixels, of the pixel location of a feature in the left rectified image and the pixel location of that same feature in the right rectified image. Since the algorithm used in this thesis can use values of x_{L_r} and x_{R_r} that are non-integer, in 1/16 pixel increments, the smallest change in disparity that the algorithm can resolve is 1/16 of pixel. This is referred to as the disparity resolution of the algorithm. Based on this disparity resolution, the smallest possible change in depth that can be resolved based upon the value of Z_{L_r} given in Equation (66) is [44]:

$$\Delta Z_{L_r} = \frac{Z_{L_r}^2}{f T_x^{R_r}} \Delta d \tag{67}$$

Where Δd is the algorithm's disparity resolution and ΔZ_{L_r} is the depth resolution.

Recognizing that X_{L_r} and Y_{L_r} are proportional to Z_{L_r}:

$$\Delta X_{L_r} = \left| \frac{\left(c_{x_{L_r}} - x_{L_r}\right) \Delta Z_{L_r}}{f} \right| \tag{68}$$

$$\Delta Y_{L_r} = \left| \frac{\left(c_{y_{L_r}} - y_{L_r}\right) \Delta Z_{L_r}}{f} \right| \tag{69}$$

Where ΔX_{L_r} and ΔY_{L_r} are the x and y position resolution in the left rectified camera frame.

Hence, resolution in all left rectified camera frame dimensions worsens quadratically with the object's depth along the z-axis in the left rectified camera frame. Additionally, position resolution along the left rectified camera frame's x-axis and y-axis worsens proportionally to the feature's pixel displacement from the x and y coordinates of the disparity map's principal point. Finally, note that these values of

ΔX_{L_r}, ΔY_{L_r}, and ΔZ_{L_r} are best case values. If there is random or systematic error in the location of the principal point, the focal length, the camera baseline, or the disparity values, then the accuracy of the coordinate estimates given by Equation (66) will worsen. Hence, to obtain optimal results, intrinsic, stereo, and extrinsic camera calibrations, as well as the disparity map generation algorithm, must be as precise as possible.

2.3.6.7 Relating Points to an Arbitrary Frame.

The coordinates yielded by Equation (62) can be related to an arbitrary world frame according to a modification of equation (32):

$$\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} = \mathbf{R}_L^w \mathbf{R}_{L_r}^L \begin{bmatrix} X_{L_r} \\ Y_{L_r} \\ X_{L_r} \end{bmatrix} + \mathbf{L}^w \tag{70}$$

Here, the vector $[X_w, Y_w, Z_w]^T$ gives the coordinates of a point in an arbitrary world frame. The DCM $\mathbf{R}_L^w$ goes from the left camera frame to the chosen world frame. As discussed in Section 2.3.6.4, $\mathbf{R}_{L_r}^L$ is the DCM describing the rotation from the left rectified camera frame to the left camera frame. The vector $[X_{L_r}, Y_{L_r}, Z_{L_r}]^T$ gives the coordinates computed from Equation (62). Lastly, the vector $\mathbf{L}^w$ is the location of the left camera in the chosen arbitrary world frame.

Hence, using the output from Equation (62), one can apply Equation (70) to relate three-dimensional points observed with a stereo camera system to an arbitrary world frame.

2.3.6.8 Stereo Vision Summary.

With a pair of stereo camera images of the same scene, one can generate a three-dimensional point cloud of the observed scene. As previously discussed, this is done in the following manner:

1. **Perform a stereo camera calibration**. The stereo camera calibration will yield the camera calibration matrices of both cameras ($\mathbf{K}_L$ and $\mathbf{K}_R$), the distortion coefficients for both cameras, the DCM between the two cameras ($\mathbf{R}_L^R$), and the translation vector between the two cameras ($\mathbf{T}^R$).

2. **Perform an extrinsic camera calibration for both cameras**. The extrinsic camera calibration will yield the translation vector between each camera and the world frame (for instance, the location of the left camera in the world frame $\mathbf{C}_L^w$) as well as the DCMs between the world frame and each camera frame (for instance, $\mathbf{R}_L^w$).

3. **Capture a set of stereo camera images of the scene of interest**. If the scene is dynamic, these images must be captured as close to the same time as possible.

4. **Rectify the captured stereo image pair**. This can be done with Bouguet's rectification algorithm. This algorithm takes $\mathbf{R}_L^R$, $\mathbf{T}^R$, $\mathbf{K}_L$, $\mathbf{K}_R$, and the distortion coefficients for both cameras as inputs. The algorithm yields DCMs describing the rotation between the real-world cameras and the virtual rectified cameras ($\mathbf{R}_L^{L_r}$ and $\mathbf{R}_R^{R_r}$). This calibration algorithm will also yield a camera calibration matrix for both the rectified cameras ($\mathbf{K}_{L_r}$ and $\mathbf{K}_{R_r}$), and, importantly, the reprojection matrix $\mathbf{Q}$.

5. **Conduct pixel matching on the rectified image pair**. In other words, identify features common to both images. Since the images have been rectified, these features should be in the same pixel row in both images.

6. **Generate a disparity map from the matched pixel pairs identified in the rectified images**. The disparity map characterizes the pixel location difference between features common to both scenes.

7. **Project the disparity map into three dimensions**. This is done using the reprojection matrix, $\mathbf{Q}$. If using the methodology described in this chapter, the resulting point cloud is expressed in the left rectified camera frame.

8. **Relate the left rectified point cloud to an arbitrary world frame**. This can be done with Equation (70).

2.4 The Iterative Closest Point Algorithm

The Iterative Closest Point (ICP) algorithm can be used to identify the translation and rotation between an observed point cloud, P, and a model point cloud, X. Published by Besl and McKay [8], they proved that ICP converges monotonically to a local minimum of the mean square distance function between the two point clouds. More generally, the algorithm can also be applied to sets of line segments, parametric curves, implicit curves, triangles, parametric surfaces, and implicit surfaces. This section's discussion on the ICP algorithm summarizes Besl and McKay's paper and uses similar notation [8].

The algorithm works by applying sequential registration vectors to the observed point cloud until sufficient registration with the model point cloud is achieved. The unit quaternion describing the rotation of a point cloud is called $\vec{q}_R = [q_0, q_1, q_2, q_3]^T$. The translation vector describing the translation of the observed point cloud is called

$\vec{q}_T = [q_4, q_5, q_6]^T$. The authors therefore define the complete registration state vector as $\vec{q} = [\vec{q}_R|\vec{q}_T]$. The observed point cloud, comprised of N_p points, is termed $P = \vec{p}_i$ while the model point cloud, comprised of $N_x = N_p$ points is termed $X = \vec{x}_i$. Each observed point corresponds to the model point of the same index i. For example, the model point corresponding to $\vec{p}_i$ is $\vec{x}_i$.

The authors demonstrate that the following function, describing the mean square difference between the translated and rotated observed point cloud and the model point cloud, is minimized with the ICP algorithm:

$$f(\vec{q}) = \frac{1}{N_p} \sum_{i=1}^{N_p} \|\vec{x}_i - \mathbf{R}\vec{p}_i - \vec{q}_T\|^2 \tag{71}$$

Where $\mathbf{R}$ is the DCM corresponding to the unit quaternion $\vec{q}_R$.

The mean of the observed and model point clouds, are defined as $\vec{\mu}_p$ and $\vec{\mu}_x$, respectively:

$$\vec{\mu}_p = \frac{1}{N_p} \sum_{i=1}^{N_p} \vec{p}_i, \qquad \vec{\mu}_x = \frac{1}{N_x} \sum_{i=1}^{N_x} \vec{x}_i \tag{72}$$

The cross-covariance between the two point clouds, Σ_{px} is defined as:

$$\Sigma_{px} = \frac{1}{N_p} \sum_{i=1}^{N_p} \left[\vec{p}_i \vec{x}_i^T\right] - \vec{\mu}_p \vec{\mu}_x^T \tag{73}$$

With these definitions, the authors form the matrix $\mathbf{A} = (\Sigma_{px} - \Sigma_{px}^T)$, and from $\mathbf{A}$, the vector $\Delta = [A_{2,3}, A_{3,1}, A_{1,2}]^T$. Using the above definitions, the authors form a matrix based principally on the cross-covariance of the two point clouds:

$$Q(\Sigma_{px}) = \begin{bmatrix} \mathrm{tr}(\Sigma_{px}) & \Delta^T \\ \Delta & \Sigma_{px} + \Sigma_{px}^T - \mathrm{tr}(\Sigma_{px})\mathbf{I}_3 \end{bmatrix} \tag{74}$$

Where $\mathrm{tr}(\cdot)$ is the trace operator and $\mathbf{I}_3$ is an order three identity matrix.

From this matrix $Q(\Sigma_{px})$, the maximum eigenvalue's corresponding unit eigenvector is identified. This eigenvector is taken to be the optimal rotation quaternion, $\vec{q}_R = [q_0, q_1, q_2, q_3]^T$, with a corresponding DCM $\mathbf{R}$. The optimal translation vector, is then identified as:

$$\vec{q}_T = \vec{\mu}_x - \mathbf{R}\vec{\mu}_p \tag{75}$$

The identified translation and rotation is then applied to the entire observed point cloud, P. The authors denote this translation and rotation operation with an operator:

$$(\vec{q}, d_{ms}) = \Gamma(P, X) \tag{76}$$

Where d_{ms} is the mean square of the point matching error given by the function $f(\vec{q})$.

With these operations now defined, the ICP algorithm is developed as follows. First, each point $\vec{p}_i$ in the observed point cloud, P, is paired with a point $\vec{x}_i$ in the model point cloud, X. Prior to this pairing N_p need not be equal to N_x. This is accomplished by simply identifying the point $\vec{x}_i$ that is closest to the point $\vec{p}$ according to the equation:

$$d(\vec{p}, X) = \min_{\vec{x}_i \epsilon X} \|\vec{x}_i - \vec{p}\| \tag{77}$$

Where $d(\vec{p}, X)$ is the smallest distance between all model points $\vec{x}_i$ in X and the observed point $\vec{p}$.

Once these pairs have been identified, the point $\vec{x}_i$ that minimized the distance metric with $\vec{p}_i$ is assigned to be equal to $\vec{y}_i$. From these, the point set Y is formed. The operation is notationally summarized as:

$$Y = C(P, X) \tag{78}$$

Where $C(\cdot)$ denotes the operation to identify the entire point set Y from the original model point set X.

With the definitions of the translation and rotation operator $\Gamma(\cdot)$ and the closest point operator $C(\cdot)$ as given above, the authors summarize the ICP algorithm as follows [8]:

1. The algorithm is provided with an observed point cloud P with N_p points and a model point cloud X with N_x points.

2. The algorithm is initialized with the initial, unaltered observed point cloud, $P_0 = P$ and with the index set to $k = 0$. Iterated registration vectors are relative to the previous data set, such that the final registration vector represents the complete rotation and translation from the initial point cloud P to the identified local minimum. Next, steps 3-6 are repeated until the desired tolerance level τ is reached.

3. The closest points $Y_k = C(P_k, X)$ are computed.

4. The registration vector is computed with the translation and rotation operator, $(\vec{q_k}, d_k) = \Gamma(P_k, Y_k)$.

5. The translation and rotation are applied to the observed point cloud according to $P_{k+1} = \mathbf{R}P_k + \vec{q}_T$.

6. Repeat steps 3-5 until $d_k - d_{k+1} < \tau$.

As previously mentioned, the ICP algorithm converges monotonically to a local minimum. Additionally, the ICP algorithm yields useful translations and rotations even when the observed point cloud P matches only a subset of the model point cloud X. However, the algorithm is susceptible to large statistical outliers and noise. Additionally, the algorithm may not produce useful results if some portion of the

observed point cloud, P, is not represented in the model point cloud, X [8]. Hence, in practical applications, such as using stereo vision to deduce the relative pose of two aircraft, pre-processing of the observation point cloud is often needed.

III. Methodology

The overall objective of this thesis is to develop and analyze two approaches to vision-based relative navigation, the V5 and R7 algorithms. This chapter discusses the implementation of both algorithms. Analysis and development are conducted in the context of AAR and algorithm design is intended to have direct applicability to the KC-46 program. The V5 algorithm is a Bayesian-inference integrity monitor that yields a PMF describing the relative position of a receiver aircraft. Using this PMF the V5 algorithm returns a relative position estimate and an associated protection level. The R7 algorithm yields a relative position and relative attitude estimate in part by leveraging the ICP algorithm.

This chapter begins with a description of the frames and variables relevant to both methods. Next, the simulation environment used to analyze both methods is discussed. After these preliminaries, the V5 algorithm is discussed in detail, followed by a description of the R7 algorithm. Then, the method used to compute V5 and R7 algorithm errors is described. Finally, the

The V5 and R7 algorithms outlined in this chapter are analyzed with simulation data in Chapter IV and with flight test data in Chapter VI. Methodology pertaining only to flight test data analysis is described in Section 5.4.

3.1 Frame and Coordinate System Definitions

This thesis requires the definition of a multitude of frames and coordinate systems. Starting globally, flight test hardware used in this thesis outputs aircraft position in the Earth-Centered Earth Fixed (ECEF) frame. The ECEF frame origin is located at the center of the Earth. A right-handed Cartesian frame, the frame's z-axis points

through the Earth's true North pole, while the x-axis points through the intersection of the Equator and the Prime Meridian [45].

Additionally, flight test hardware also referenced local North-East-Down (NED) frames in attitude calculations. The origin is defined at a point of interest. For flight test data analysis in this thesis, the origin was defined to be located at the origin of the tanker aircraft body frame. A right-handed Cartesian frame, the frame's x-axis points to true North, the frame's y-axis points due East, and the z-axis points down with respect to the local vertical [46].

On the scale of the aircraft formation, seven Cartesian frames are relevant. First, the p-frame, or primary frame, is a nose-right wing-down frame centered on a reference point on the lead aircraft (i.e. the aircraft upon which the cameras are mounted). In the context of the AAR problem examined in this thesis, this would be the tanker aircraft.

Second, a re-coordinization of the p-frame, called the v-frame, is also used. The v-frame is a nose-left wing-up frame centered on the same point on the lead aircraft as the p-frame. This frame is primarily used to relate the returns from the simulation environment described in Section 3.3.

Third, the L-frame is a camera frame centered on the left camera origin. This camera frame follows the convention defined in Section 2.3.1.

Fourth, the R-frame is an analogous camera frame centered on the right camera. This camera frame also follows the convention defined in Section 2.3.1.

Fifth, the B-frame, or boom frame, is centered at the boom joint (i.e. the point on the tanker surface where the boom attaches to the aircraft hull). The frame's z-axis extends down the length of the boom. The frame's x-axis is orthogonal to this axis and extends upward. Upward in this sense means it parallels the the z-axis of the

v-frame as closely as possible while maintaining orthogonality with the z-axis of the B-frame.

Sixth, the s-frame, or secondary frame, is a nose-right wing-down frame centered on a reference point on the trailing aircraft (i.e. the aircraft which the cameras are intended to observe). In the context of AAR, this aircraft would be the receiver.

Seventh, the d-frame is a nose-left wing-up frame centered on the receiver. The d-frame is the same as the s-frame except that the directions of the y and z axes are reversed. Similar to the v-frame, the d-frame is defined to ease use of computer modeling software. The relationship between the p-frame and v-frame as well as the relationship between the s-frame and d-frame is defined by the DCM:

$$\mathbf{R}_v^p = \mathbf{R}_s^d = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}. \tag{79}$$

Figure 25 depicts four of the formation frames discussed in this section.

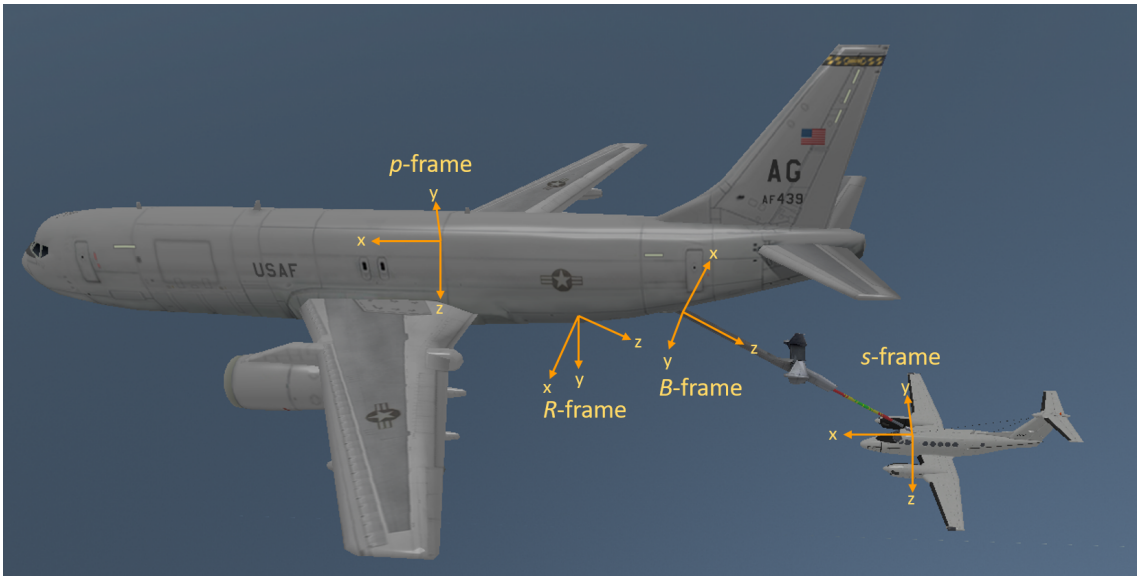


Figure 25. Relevant Formation-Level Coordinate Frames.

On the scale of the cameras, two additional frames merit discussion. These are the left and right rectified camera frames. Per the discussion in Section 2.3.6.4, the rectified camera frames are conceptual frames that enable the generation of a rectified image. The DCM from the rectified camera frame to the non-rectified camera frame completely describes the relationship between two frames. The relationship between these frames is depicted in Section 2.3.6.4 in Figure 22.

Finally, the relationship between the p-frame and the camera frames merits emphasis. The DCM and translation vectors returned from an extrinsic camera calibration completely characterize this relationship. As discussed in Section 2.3.5 and Figure 19, the parameters of interest are the DCMs between the frame of reference (in this thesis, the p-frame) and the two camera frames, as well as the translation vectors to the two camera origins expressed in the frame of reference.

3.2 Variable Definitions and Notation

Throughout this thesis, the relationship between the various frames are defined in terms of vectors and DCMs. In general, the vectors represent the location of one frame's origin in another frame and the DCMs represent the rotation between two frames. Vectors and matrices are bolded. For vectors, the superscript denotes the frame in which the vector is expressed. For DCMs, the subscript represents the initial frame and the superscript represents the ending frame. For instance, the DCM $\mathbf{R}_K^J$ would operate on a vector expressed in the K-frame and would transform it into an equivalent vector expressed in the J-frame (provided the origins of the K and J frames are collocated).

Additionally, each camera has its own set of physical properties, distortion coefficients, and camera calibration parameters. The camera such parameters correspond to is denoted with a subscript.

Column vectors are used for all three-dimensional and pixel point coordinates. Finally, point clouds are constructed as arrays wherein the i^{th} column contains a vector representing the location of the point i in the point cloud.

Table 1 lists all frames from Section 3.1 (as well as objects of interest such as the left camera) and their corresponding notational abbreviation. Using these notational conventions, Table 2 defines key variables of interest. The contents of these tables are used for the remainder of this thesis. Parameters expressed in other frames use analogous notation to the variables given in Table 2. For instance, a three-dimensional coordinate in the left rectified camera frame is expressed as $\mathbf{X}^{L_r}$.

Table 1. Frames and Their Corresponding Superscripts and Subscripts.

Frame, Coordinization, or Object	Superscript or Subscript
ECEF frame	ECEF
NED frame	NED
p-frame	p
v-frame	v
s-frame	s
d-frame	d
L-frame or Left Camera/Image	L
L_r-frame or Left Rectified Camera/Image	L_r
R-frame or Right Camera/Image	R
R_r-frame or Right Rectified Camera/Image	R_r
B-frame	B

Table 2. Variable Definitions. Vectors and Matrices are Bolded. Point Cloud Variables are Arrays Comprised of Columns of Three-Dimensional Coordinate Vectors Wherein Each Column is a Distinct Point.

Variable	Left Camera	Right Camera	p-Frame
Focal Length (mm):	F_L	F_R	N/A
Focal Length in x and y (pixels):	f_{x_L}, f_{y_L}	f_{x_R}, f_{y_R}	N/A
Principal Point (pixels):	(c_{x_L}, c_{y_L})	(c_{x_R}, c_{y_R})	N/A
Distortion Coefficients:	k_{i_L}	k_{i_L}	N/A
Camera Calibration Matrix:	$\mathbf{K}_L$	$\mathbf{K}_R$	N/A
Translation Between Cameras:	$\mathbf{T}^L$	$\mathbf{T}^R$	N/A
Rotation Between Cameras:	$\mathbf{R}_L^R$	$\mathbf{R}_R^L$	N/A
Position of Camera in p-frame:	N/A	N/A	$\mathbf{C}_L^p$, $\mathbf{C}_R^p$
Rotation from Camera to p-frame:	$\mathbf{R}_L^p$	$\mathbf{R}_L^p$	$\mathbf{R}_p^L$, $\mathbf{R}_p^R$
Rotation from Rectified Camera:	$\mathbf{R}_{L_r}^L$	$\mathbf{R}_{R_r}^R$	N/A
Pixel Coordinate:	$\mathbf{x}_p^L$	$\mathbf{x}_p^R$	N/A
Three-Dimensional Coordinate:	$\mathbf{X}^L$	$\mathbf{X}^R$	$\mathbf{X}^p$
Observed Point Cloud in Rectified Frame:	$\mathbf{O}^{L_r}$	$\mathbf{O}^{R_r}$	N/A
Observed Point Cloud:	$\mathbf{O}^L$	$\mathbf{O}^R$	$\mathbf{O}^p$
Model Point Cloud:	$\mathbf{M}^L$	$\mathbf{M}^R$	$\mathbf{M}^p$
Location of Receiver:	$\mathbf{s}^L$	$\mathbf{s}^R$	$\mathbf{s}^p$
Orientation of Receiver:	$\mathbf{R}_L^s$	$\mathbf{R}_R^s$	$\mathbf{R}_p^s$

3.3 Simulation Environment

The simulation environment used in this thesis, hereafter denoted as the three-dimensional virtual world (3DVW), was developed by Dr. Scott Nykl of AFIT. Among many other capabilities, the 3DVW enables the generation of realistic aircraft, camera, and scene representations in real-time. The geometry of the 3DVW and the objects therein can be specified by the user. For the purposes of this thesis, a particularly useful application of the 3DVW is the ability to test and implement vision algorithms on synthetically generated virtual images as well as images captured by real-world systems.

Figure 26 shows an example snapshot from the 3DVW. In the figure, one can see a tanker and aircraft that have been rendered above imagery of Wright-Patterson Air Force Base. The tanker is geometrically identical to a KC-135 and the receiving aircraft is geometrically identical to a C-12C. In the 3DVW, any aircraft for which the user has a model could be rendered as the receiver or as the tanker. The terrain has the same coordinates (including elevation) as the real world. The boom on the tanker is that of a KC-10. In this thesis, a KC-10 boom was affixed to a KC-135 body to increase relevance to the KC-46 program. The KC-46 has a boom similar to that found on the KC-10. For the purposes of gathering 3DVW simulation images, the body of the aircraft upon which the cameras are mounted is irrelevant. Hence, a KC-135 body is used merely for the purpose of aiding user visualization.

In the lower left and lower right corners of the figure are the simulated stereo images captured by the tanker's cameras. These images have the same dimensions as the stereo cameras used in the flight test portion of this thesis. Additionally, these stereo cameras are mounted on the tanker in the same configuration used in flight testing. The baseline of the stereo camera pair is 0.5 meters and the cameras are mounted at a 25° down-look angle relative to the longitudinal axis of the tanker.

This configuration was designed to mimic the configuration found on the KC-46. The 3DVW can render aircraft poses and output images captured by the 3DVW cameras in real-time.

Using the 3DVW, one can apply vision algorithms to this stereo image pair. An example result of this process is depicted on the figure. On the boom and C-12, there are multiple yellow dots. These dots represent a three-dimensional point cloud that a stereo vision processing algorithm operating on the stereo image pair output has yielded. The 3DVW enables such algorithms to be tested for both speed and accuracy. Algorithm design and analysis in this thesis focus primarily on algorithm accuracy rather than speed.



Figure 26. Example Snapshot Taken of the 3DVW.

3.4 Vision & Bayesian Inference Based Integrity Monitor (V5)

This section describes the V5 algorithm. The algorithm generates a PMF describing the relative position of an aircraft. This PMF is generated by comparing observation imagery to a reference image database with Bayesian inference. From this PMF, the V5 algorithm provides a relative position estimate and a degree of navigation integrity in the form of a protection level. This section describes the V5 algorithm in detail. First, an overview of the V5 algorithm is presented. Following the overview, this section discusses algorithm components in detail. Where relevant, this discussion includes the MATLAB® functions used in implementation. Much of the algorithm description in this section is identical to the description found in the Have Vision test information memorandum [7]. The V5 algorithm is analyzed with simulation and flight test data in Chapters IV and VI, respectively.

3.4.1 V5 Algorithm Overview.

The V5 algorithm builds upon Calhoun's Bayesian inference integrity monitor [2], [5] which is outlined in Section 2.2. Broadly speaking the V5 algorithm uses the same structure as Calhoun's integrity monitor. However, several important modifications and additions are included. For clarity, the same format that is used to outline Calhoun's Bayesian inference integrity monitor in Section 2.2.1 is used in this section. In the outline, the most substantial differences between the V5 algorithm and Calhoun's approach are *emphasized*.

1. **Likelihood Function Determination**:

   (a) **Reference Image Database Generation**: Generate a reference image database comprised of rendered images. These reference images depict the trailing aircraft at a level attitude at known, discrete points in space rela-

tive to the observing aircraft. Process these images in accordance with Step 1(c). *In this thesis the reference aircraft is the receiver and the observing aircraft is the tanker. Additionally, the tanker is assumed to be equipped with a stereo camera pair. Therefore, the reference database includes a left and right camera image for every position included in the database. The stereo camera pair is constructed to mimic the stereo camera system used on the KC-46.*

(b) **Observation Image Collection**: Collect a set of observation image pairs either in the 3DVW or in flight test. Process these images in accordance with Step 1(c). *Each distinct observation position will have two corresponding simulation images—one for each camera in the stereo image pair.* The relative position depicted in each observation image must be known. *Additionally, the stereo configuration parameters must be known.*

(c) **Image Processing**: Both reference and observation images must be processed. *The V5 algorithm processes both the left and right camera images obtained from the stereo camera configuration mounted on the observing aircraft (the tanker). Additionally, these stereo images are used to filter out any background pixels in the images, resulting, ultimately, in a blurred depiction of only the trailing aircraft in edge-space. Image processing is detailed in Section 3.4.3.*

i. **Edge Detection**: Apply a Prewitt edge detector to all images in the reference image database and to each observation image.

ii. **Disparity Map** *Using the raw images, generate a disparity map for each image pair in the reference image database and for all observation image pairs. Each disparity map is based upon the known stereo config-*

uration parameters and is generated according to the method described in Section 2.3.6.

iii. **Depth Filter** *Using the disparity map generated in Step 1(c)(ii), filter out any pixels from the edge-space images that are determined to be outside the region of interest for aircraft observation.*

iv. **Blurring**: Apply a pre-determined level of Gaussian blurring to all edge-space, depth filtered images.

(d) **Identify Best Reference-Observation Image Pair**: Identify the reference image pair generated in Step 1(a) that is nearest to representing the same geometric formation represented by each observation image pair generated in Step 1(b). For instance, assume there are only three reference image pairs. These depict the reference aircraft located at three-dimensional positions (1,2,3), (4,5,6), and (7,8,9) relative to the observing aircraft. If there is an observation image pair whose relative position is known to be (1,2,4), then the first reference image pair is the best match and would be selected for comparison to the observation image pair.

(e) **Compare Best Reference-Observation Image Pairs**: Compare the features depicted in the best reference-observation image pair. *The V5 algorithm randomly selects an equal number of edge pixels from both the observation and reference image. This results in an essentially symmetric likelihood function. This method is discussed in detail in Section 3.4.4.*

(f) **Construct the Likelihood Function**: Identify a PDF that models the feature differences between the best reference-observation image pairs. Enough reference-observation image pairs must be used to adequately represent the sample space of interest. This PDF serves as the likelihood

function used in all subsequent steps. The details of this process are discussed in Section 3.4.4.

2. **Observation Image Collection**: Collect a set of observation images either in the real-world or in simulation. Process these images in accordance with Step 1(c). *This simulation and flight test observation images analyzed in this thesis include attitude perturbations.* Now that a likelihood function has been determined, the relative position of these observation images need not be known.

3. **Prior Probability Determination**: Determine the prior probability for each pose represented in the reference image database generated in Step 1(a). This thesis explores the use of both uniform and Gaussian priors.

4. **Reference Database-Observation Image Comparison**:

   (a) **Reference Database Attenuation**: If possible, confine the search space of the reference image database to a subset of the entire reference image database generated in Step 1(a). This is necessary to reduce processing times.

   (b) **Image Comparison**: Compare the observation image pair to all reference image pairs identified in Step 4(a) according to the same metric used in Step 1(e). Optionally, only examine edge pixels that exceed a user-specified intensity threshold after image processing. Compare multiple pixels to make the ultimate results more robust.

   (c) **Compute Likelihood Scores**: Based on the results from Step 4(b) and the likelihood function determined in Step 1(f), compute a likelihood score for each reference image pair-observation image pair coupling.

5. **Bayesian Inference**: For each point in the reference image database, determine the posterior probability that the observation image depicts the formation at that pose. This is done using each computed likelihood score and each prior probability with Bayes' Law. Since reference image database is discrete, the posterior probability distribution obtained from Bayes' Law is a PMF. *In this the V5 algorithm, this process is performed separately for both left and right camera images. These results are then probabilistically combined to obtain a single PMF. The details of this process are discussed in Section 3.4.6.*

6. **Relative Position Estimate**: The relative position estimate can be taken to be the location of the reference image with the highest associated posterior probability. *The V5 algorithm also yields an alternative relative position estimate which takes into account the entire PMF. This technique is explained in Section 3.4.7.*

7. **Protection Level Determination**: Starting from the relative position estimate, geometrically move outward from that position in the reference image database, summing posterior probabilities as you move outward. Continue this process until reaching the desired integrity risk level. The distance moved away from the relative position estimate when the desired integrity risk level is reached serves as the protection level. *The V5 algorithm moves out uniformly in all directions while summing posterior probabilities captured within the bounding sphere. This results in what can best be described as a spherical protection level. This is described in detail in Section 3.4.8.*

Once the likelihood function has been determined, the V5 algorithm operates by applying Steps 2-6. These steps are depicted in block diagram form in Figure 27.

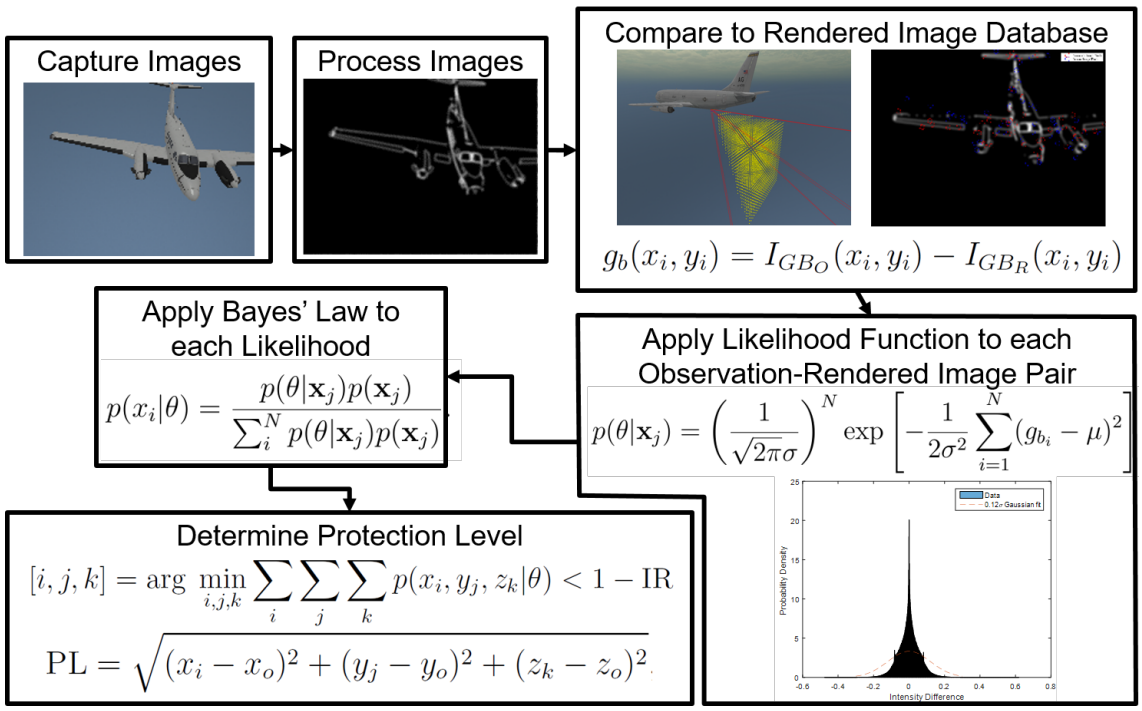


Figure 27. Block Diagram Depicting the V5 Algorithm.

3.4.2 Reference Image Database Generation.

For both flight test and simulation data analysis, the raw images used to create the reference image database were created in the 3DVW. The geometry of the database used in data analysis is described below. Other geometries could be applied with the V5 algorithm.

The database consisted of images of the receiver captured at surfaces of equal range from the center of the tanker's stereo camera pair. Within each surface, every image was at the same distance from the center of the stereo camera pair. Successive surfaces were separated by a user-specified Euclidean distance in meters. Within each surface, images were separated by a user-specified equiangular spacing in azimuth and elevation, as measured from a ray beginning at the center of the stereo camera pair and extending to the location of the last reference position generated. The azimuth

limits of each surface, extending left or right of the tanker's longitudinal axis, were user-specified. Similarly, in elevation the angular limits of each surface below the tanker's longitudinal axis were user-specified. Therefore, each surface within which reference images were generated was a portion of the surface of a sphere. As a result of this method, each rendering surface contained the same number of images. This precluded an unnecessarily dense reference image database at far ranges. Figure 28 depicts this scheme. More detail on the reasons for using this scheme is discussed in Section 4.1.4.

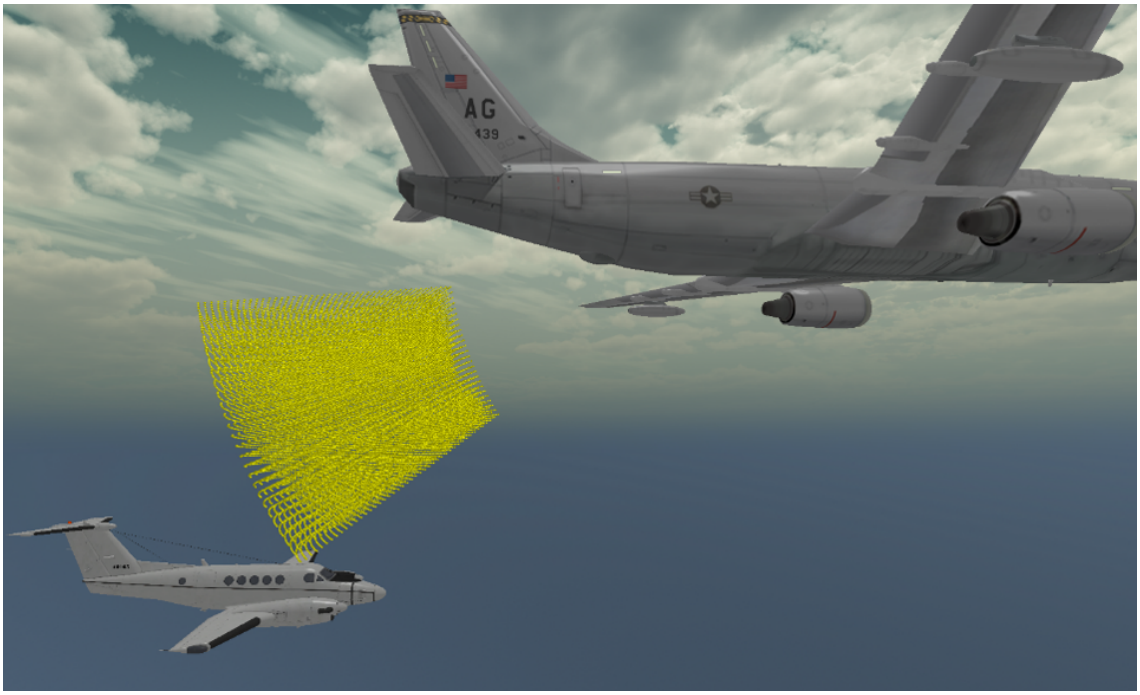


Figure 28. Depiction of the Rendered Image Database.

3.4.3 Image Processing.

Image processing for the V5 algorithm was implemented in MATLAB®. MathWorks® developed functions included as part of the Computer Vision System Toolbox were used to perform the majority of image processing. The V5 algorithm could use other image processing libraries in the future.

3.4.3.1 Prewitt Edge Detection.

Edge detection was performed with the MathWorks® developed function `edge`. This function includes the option to use a Prewitt filter in order to perform edge detection. A Prewitt filter was used in order to make the results of this thesis more directly comparable to the work of Calhoun, but other edge detection options could be explored in future work.

The function `edge` also includes the ability to use an algorithmically defined threshold for edge declaration. This option was exercised in this thesis in order to eliminate what would have been an extensive tuning process and in an attempt to improve the algorithm's robustness to environmental change. Using this method, the threshold is set to the mean of the squared intensity gradient values in the image [47].

3.4.3.2 Disparity Map Generation.

Disparity maps were generated with raw reference database image pairs and sample image pairs via the four step process described below.

1. Intrinsic camera calibration and stereo camera calibration results were obtained. In simulation, these calibration results were deterministic and could be deduced based upon the properties of the simulated images captured in the 3DVW. For real-world cameras, these results were obtained via Bouguet's MATLAB® Camera Calibration routine [34].

2. Images were rectified with the MATLAB® function `rectifyStereoImages`. Required inputs to this function include stereo camera calibration results and the raw image pairs.

3. The rectified images were converted to black and white with the MATLAB® function `rgb2gray` to facilitate further processing.

4. The black and white rectified image pairs were used to generate a disparity map with the MATLAB® function `disparity`.

3.4.3.3 Depth Filter.

As described in Section 2.3.6.6, three-dimensional coordinates can be obtained for each pixel in a disparity map if the stereo parameters of the camera pair are known. This property of stereo cameras was used to develop the V5 algorithm's depth filter.

This depth filter eliminates features in both the left and right processed images that fall outside a user-specified depth window. This process is made possible by the relationship between disparity maps and rectified images described in Section 2.3.6.5. Since a disparity map has the same coordinate system as the left rectified image, any three-dimensional point obtained from a disparity map corresponds to a particular pixel in the left and right rectified images. The pixels in the left rectified image have the same mapping as the pixels in the disparity map. Corresponding pixels in the right rectified image are determined with Equation (58).

The depth window MATLAB® implementation which was used in data analysis is described below, but an analogous process could be built with other libraries. In simulation stereo parameters were deterministic. In flight test, these parameters were obtained with a stereo camera calibration.

1. The disparity map was projected into a set of three-dimensional coordinates in the left camera frame with the MATLAB® function `reconstructScene`. Necessary function inputs include the disparity map and the stereo parameters of the camera pair.

2. The user specified two threshold depth values, a minimum and a maximum. These constituted a depth window. This depth value corresponds to the z-

component of each three-dimensional point as expressed in the left rectified camera frame.

3. Every three-dimensional point whose z-component did not fall within the depth window was identified.

4. The intensities of pixels in the blurred, edge-space, rectified images that corresponded to the three-dimensional points falling outside the depth window (identified in Step 3) were set to zero.

3.4.3.4 Gaussian Blurring.

Blurring was applied to all edge-space, depth filtered images. Gaussian blurring was performed with the MathWorks® function `imgaussfilt`. The function operates in the manner described in Section 2.2.2.3. The only arguments to the function are the input image and the σ parameter desired for use. The sole output is the blurred image. The level of Gaussian blurring for this thesis was set to a σ parameter of 5. The same level of blurring was used on simulated and flight test images. The analysis conducted to determine this blurring level is discussed in Section 4.1.4.

3.4.3.5 Image Processing Block Diagram.

The overall image processing method is depicted in Figure 29. For this thesis, pixels that were algorithmically determined to be between 0 and 100 meters from the stereo camera pair center were retained; all others were eliminated. As Figure 29 shows, without the depth filter, it would be essentially impossible to distinguish a trailing aircraft in the edge-space image.

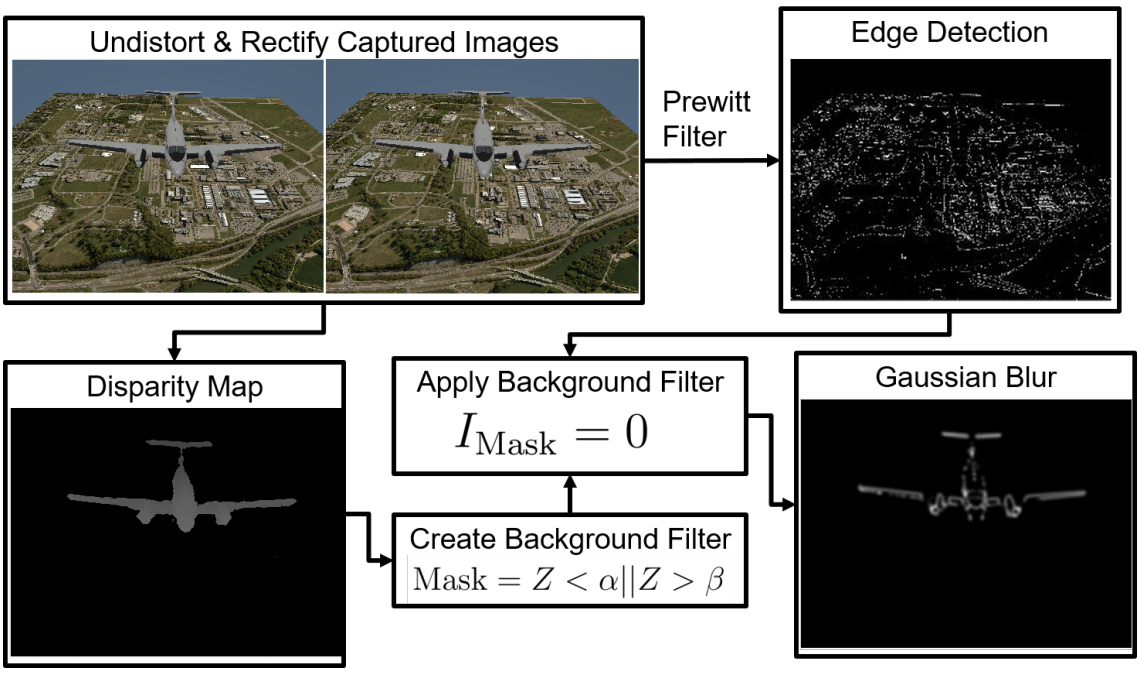


Figure 29. Depiction of the Image Processing Method Used in this Thesis.

3.4.4 Likelihood Function Determination.

Likelihood function determination with the V5 algorithm is a tuning intensive process and must be completed prior to application of the algorithm. As described above, the reference image whose relative position most closely represents the observation image is identified. Next, n edge pixels from the processed reference image (processed images depict blurred edge pixels of the aircraft) and n pixels from the simulation-generated processed observation image are randomly selected. Figure 30 depicts this random sampling for an example reference-observation image pair.

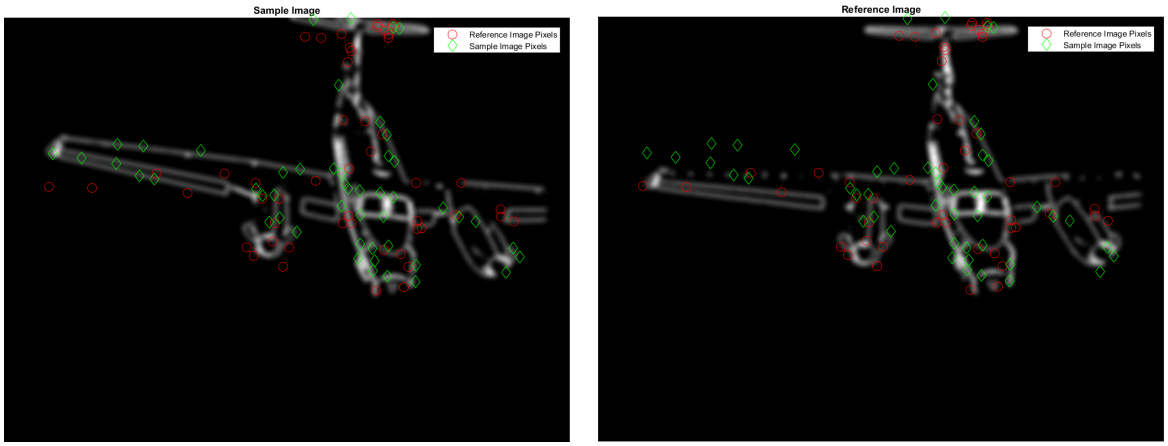


Figure 30. Random Pixel Sampling Example Shown For an Observation Image and the Reference Image that is Closest to Depicting the Same Relative Position. Both Images Were Generated in the 3DVW and Have Been Transformed to Edge Space and Been Blurred with a 5σ Gaussian Blur.

The intensity values at these randomly selected pixels are compared to one another using the same method used by Calhoun in [2]:

$$\text{IntDiff}_i = \text{IntRef}(x_i, y_i) - \text{IntObs}(x_i, y_i) \tag{80}$$

Where (x_i, y_i) are the coordinates of the randomly selected pixel, $\text{IntRef}(x_i, y_i)$ is the intensity of the randomly selected pixel in the reference image, $\text{IntObs}(x_i, y_i)$ is the intensity of the randomly selected pixel in the observation image, and IntDiff_i is the intensity difference present at random pixel i. These results are then used to find a probability distribution that best fits the intensity difference distribution while still yielding acceptable V5 performance. Specific results for this thesis are discussed in Section 4.1.5. The same likelihood function was used on both simulation and flight test data.

3.4.5 Likelihood Score Computation.

In [2] and as described in Section 2.2.6, Calhoun assumed that every random intensity difference was independent from all other observed intensity differences. The V5 algorithm makes the same assumption. Additionally, to facilitate processing the V5 algorithm computes likelihood scores in the logarithmic domain as did Calhoun in [2]. Leveraging the independence assumption, a likelihood score is computed via the following process for any given reference-observation image pair. During V5 algorithm implementation, this process iterates over numerous reference-observation image pairs in an attempt to identify the reference image that most closely represents the relative position captured in the observation image.

1. A total of n edge pixels are randomly sampled from the reference image and n edge pixels are randomly sampled from the observation image. Edge pixels are pixels that have an intensity value that exceeds a user-specified threshold after image processing.

2. The intensity difference between the images at each pixel is computed per Equation (80).

3. The likelihood of observing the computed intensity difference at each pixel is calculated using the likelihood function obtained via the method described in Section 3.4.4. Again, this is the same approach as that used by Calhoun in [2]. For a Gaussian likelihood function:

$$p(\text{IntDiff}_i|\mathbf{x}_j) = \left(\frac{1}{\sqrt{2\pi}\sigma}\right) \exp\left[-\frac{1}{2\sigma^2}(\text{IntDiff}_i - \mu)^2\right], \tag{81}$$

   Where σ is the standard deviation of the distribution, IntDiff_i is the pixel intensity difference computed per Equation (80), μ is the mean of the normal

distribution, and $p(\text{IntDiff}_i|\mathbf{x}_j)$ is the likelihood of observing the pixel intensity difference IntDiff_i given that the reference image is the best match for the observation image.

4. In the linear domain and under the independence assumption, these likelihoods could be combined via multiplication. However, performing this combination in the logarithmic domain enables the use of addition. Again, this is a similar approach to that used by Calhoun in [2] as discussed in Section 2.2.6. For a Gaussian likelihood function, the total log-likelihood score for a given reference-observation image pair is computed as:

$$\log(p(\theta|\mathbf{x}_j)) = N\log(\frac{1}{\sqrt{2\pi}\sigma}) - \sum_{i=1}^{N}\frac{\text{IntDiff}_i - \mu^2}{2\sigma^2}, \tag{82}$$

Where $N = 2n$ is the total number of pixels examined from the reference-observation image pair, and $p(\theta|\mathbf{x}_j)$ is the likelihood of observing the entire set of pixel intensity differences if the reference image is the best match for the observation image. All other parameters are identical to those used in Equation (81).

3.4.6 Probability Mass Function Computation.

Once likelihood scores have been obtained for all reference-observation image pairs, Bayesian inference can be used to obtain a PMF describing the probability that the aircraft is at any relative position depicted in the reference image database. This approach is similar to that developed by Calhoun in [2] as described in Section 2.2.7.

Bayes' Law, for a discrete set of mutually exclusive and mutually exhaustive events, $x_1, x_2, ..., x_N$, is presented in Equation (16) as:

$$p(\mathbf{x}_j|\theta) = \frac{p(\theta|\mathbf{x}_j)p(\mathbf{x}_j)}{\sum_i^N p(\theta|\mathbf{x}_j)p(\mathbf{x}_j)} \tag{83}$$

In the context of this thesis, these "events" are the condition that the aircraft is located at the relative position $\mathbf{x}_j$ contained within the reference image database.

With respect to Equation (83), logarithms are also useful in computing the numerator, $p(\theta|\mathbf{x}_j)p(\mathbf{x}_j)$. Here, $p(\mathbf{x}_j)$ is the prior probability that the aircraft is located at the relative position $\mathbf{x}_j$. Taking the logarithm of this product results in the computationally more tractable summation:

$$\log(p(\theta|\mathbf{x}_j)) + \log(p(\mathbf{x}_j)) \tag{84}$$

In order to obtain the PMF and to compute a protection level, the probabilities must be transformed back into linear space per the equation:

$$p(\theta|\mathbf{x}_j)p(\mathbf{x}_j) = \exp\left[\log(p(\theta|\mathbf{x}_j)) + \log(p(\mathbf{x}_j))\right] \tag{85}$$

Finally, these resulting probabilities are normalized according to Bayes' Law, yielding a PMF describing the variable of interest, x, i.e. the relative pose of the formation:

$$p(x_i|\theta) = \frac{p(\theta|\mathbf{x}_j)p(\mathbf{x}_j)}{\sum_i^N p(\theta|\mathbf{x}_j)p(\mathbf{x}_j)} \tag{86}$$

The net result is a PMF describing the posterior probability that the aircraft is located at all relative positions, x_i, represented by the reference images that were examined. Since the reference image database is discrete, so too is the probability space output from the application of Bayes' Law. Hence, the ultimate result from

the V5 algorithm is a PMF that assumes the relative pose of the tanker-receiver formation can only occupy a discrete set of positions. It is critical to understand that x_i is inherently a position depicted in the reference image database. As a result, the V5 algorithm implicitly assumes that the true aircraft state is represented somewhere in the discrete database. As noted by Calhoun in [2], this makes the algorithm executable but also inherently limits algorithm accuracy based upon the spacing between adjacent reference images in the database.

When using images from two cameras, a left and a right, two PMFs are obtained. These PMFs are combined by giving equal weighting to both distributions per the equation:

$$p(\mathbf{x_i}|\theta)_{\text{net}} = 0.5p(\mathbf{x_i}|\theta)_L + 0.5p(\mathbf{x_i}|\theta)_R \tag{87}$$

Where $p(\mathbf{x_i}|\theta)_{\text{net}}$ is the PMF giving equal weight to the left and right camera solutions, $p(\mathbf{x_i}|\theta)_L$ is the PMF obtained from the left camera reference-observation image comparisons, and $p(\mathbf{x_i}|\theta)_R$ is the PMF obtained from the right camera reference-observation image comparisons.

3.4.7 Relative Position Estimation.

Two different relative position estimators can be used with the V5 algorithm. The first is a maximum likelihood estimator that identifies the mode of the PMF returned by the algorithm. This was the method developed by Calhoun in [2] and [5]. This method uses the PMF found with Equation (87), to identify the reference image location that best represents the observation image. This determination is made by simply identifying which reference image position has the highest probability mass:

$$[i] = \arg \max_i p(\mathbf{x_i}|\theta)_{\text{net}} \tag{88}$$

The relative position depicted by these reference images is the most likely location of the aircraft. The maximum likelihood estimator can only return relative position estimates that are present in the rendered image database.

The second estimator is called the composite position estimate in this thesis. Rather than identifying the mode of the PMF, this estimator identifies the mean. The composite position estimate makes use of the entire PMF obtained in Equation (87) by weighting each discrete relative position contained in the PMF by its associated probability mass:

$$\mu_{\mathbf{x_i}} = p(\mathbf{x_i}|\theta)_{\text{net}} \cdot \mathbf{x_i}. \tag{89}$$

The composite position estimate can return relative position estimates that are not present in the reference image database. Additionally, the composite position estimate better accounts for the fact that the PMF returned by Equation (87) is not necessarily unimodal

3.4.8 Protection Level Computation.

Equation (87) is also used to compute a spherical protection level. This is done by summing probability mass outward from the most likely aircraft position until the desired integrity risk threshold is reached. This outward summing action is done in a radial sense. This same method was used when using either relative position estimation technique. For example, suppose the V5 algorithm run with an integrity risk level of 0.1. If the probability mass of all points falling within one meter of the best match sums to 0.9, then the V5 algorithm would return a protection level of one meter. This means the algorithm is asserting there is a 0.1 probability that algorithm error exceeds one meter in the spherical sense.

Mathematically, this process is similar to Calhoun's in [5] as described in Section 2.2.8, and can be represented as:

$$[i] = \arg \min_{i} \sum_{i} p(\mathbf{x}_i|\theta) < 1 - \text{IRL} \tag{90}$$

$$\text{PL} = \sqrt{(x_i - x_e)^2 + (y_i - y_e)^2 + (z_i - z_e)^2} \tag{91}$$

Here, (x_e, y_e, z_e) is the relative position estimate, IRL is the integrity risk level, the PL is the protection level. The set of relative positions depicted in the reference images examined, $\mathbf{x}_i$, have been sorted by distance from what the V5 algorithm's relative position estimate. In practice, this was done with the MATLAB® function `sort`. Hence, the relative position depicted in index $i + 1$ is an equal or greater distance form the relative position estimate than the relative position depicted in index i. As a result, one can simply sum upward in index values until the desired integrity risk level is reached. Correspondingly, the position (x_i, y_i, z_i) is the closest position to (x_e, y_e, z_e) for which the required integrity risk level was met. As a result, the spherical protection level is simply the Euclidean distance between this point in the reference image database and the location of the relative position estimate.

3.5 Relative Pose Estimation with Computer Vision and ICP (R7)

The R7 algorithm leverages computer vision algorithms and the ICP algorithm to return a relative position and a relative attitude estimate. As with the V5 algorithm, the algorithm was designed and analyzed in the context of AAR, but could be applied to close formation relative navigation more generally. This section outlines the R7 algorithm and the implementation used in this thesis. A similar description can be found in the Have Vision test information memorandum [7].

3.5.1 R7 Algorithm Overview.

The R7 algorithm operates on a pair of images captured by a stereo camera configuration. Three high-level steps describe the basic operation of the R7 algorithm:

1. The first step is observation point cloud generation. A series of computer vision algorithms transform the observed stereo images into an observation point cloud. This process is detailed in Section 3.5.3. In this thesis, this process was implemented with OpenCV.

2. Next, the observation point cloud is filtered to remove noise and outliers. Filtering includes a boom filter, a median filter, denoising, and downsampling. This process is detailed in Section 3.5.4. This thesis implemented this step in MATLAB®.

3. Lastly, the ICP algorithm is used to compare the observation point cloud to a model point cloud. This thesis implemented this step in MATLAB®. Section 3.5.5 details this process. The return from the ICP algorithm provides the relative position and relative attitude measurements. The R7 algorithm also includes an optional modification wherein the relative attitude of the two aircraft is provided to the ICP algorithm.

Within all three high-level steps, multiple sub-steps are taken. The sub-steps are detailed in the sections referenced above. While not explicitly part of the R7 algorithm, the model point cloud must be developed and pre-processed prior to algorithm operation. Model point cloud development and pre-processing is discussed in Section 3.5.2. Figure 31 depicts the R7 algorithm in block diagram form.

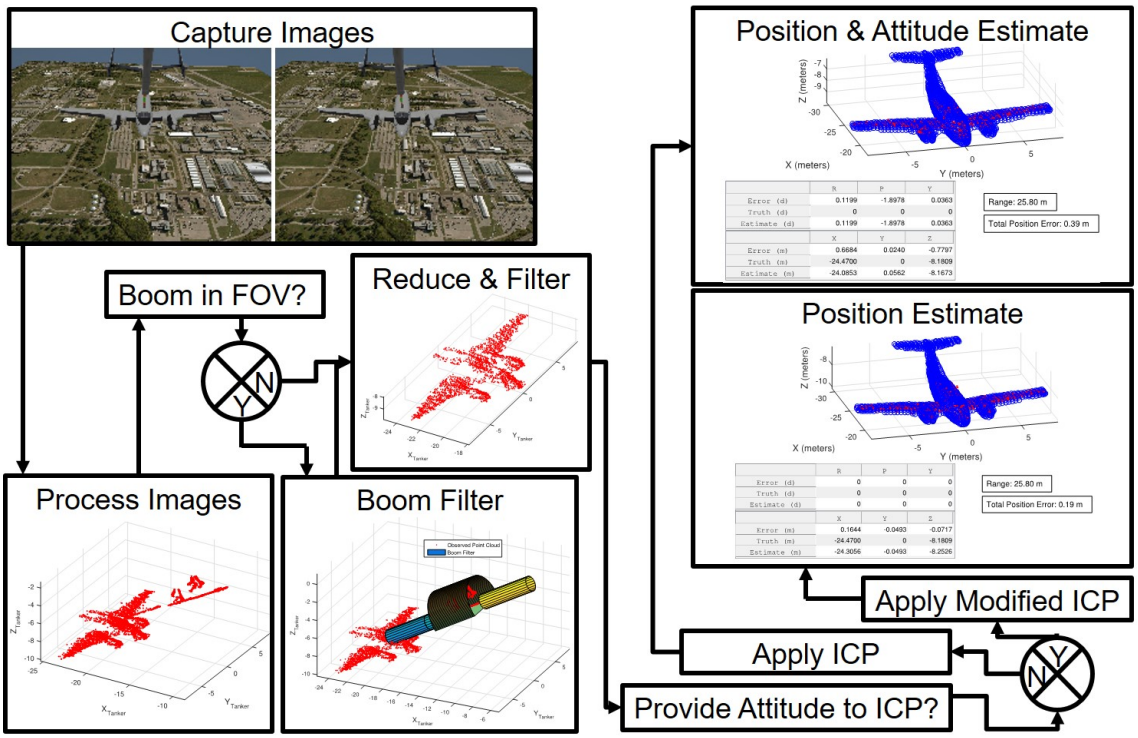


Figure 31. Block Diagram of the R7 Algorithm.

3.5.2 Model Point Cloud Development.

The model point cloud used in this thesis was based on a C-12 computer model obtained from TurboSquid™. This model was edited within the 3DVW by Dr. Scott Nykl of AFIT to ensure it matched actual C-12 dimensions provided by the USAF Test Pilot School. This model was used to render all C-12 imagery processed in simulation.

This computer model was used to develop the model point cloud. This was done in the following manner:

1. The computer model file, a .obj file type, was opened in Blender™ (Blender™ is a free, open-source computer modeling program).

2. In Blender™, extra vertices were added to the C-12 model in locations of sparse density, such as the wings. Excess vertices were removed from the C-12 model in locations of high density, such as the fuselage.

3. The modified .obj file was saved to disk and then opened with a text editing program. Within the text editor, all parameters not associated with vertices were eliminated.

4. The edited text file, containing only object vertex coordinates, was read into a MATLAB® array and used to construct a MATLAB® point cloud object.

5. The resulting MATLAB® point cloud object was downsampled using the same `pcdownsample` function discussed in Section 3.5.4.4.

6. The resulting point cloud object was saved to file as a .MAT file type so that it could be loaded prior to running the R7 algorithm. This point cloud is the model point cloud, $\mathbf{M}^v$, used in the ICP step of the R7 algorithm.

The model point cloud used on simulated data is shown in Figure 32. As can be seen in the figure, this point cloud models all parts of the C-12C, even those not likely to be visible to the cameras.

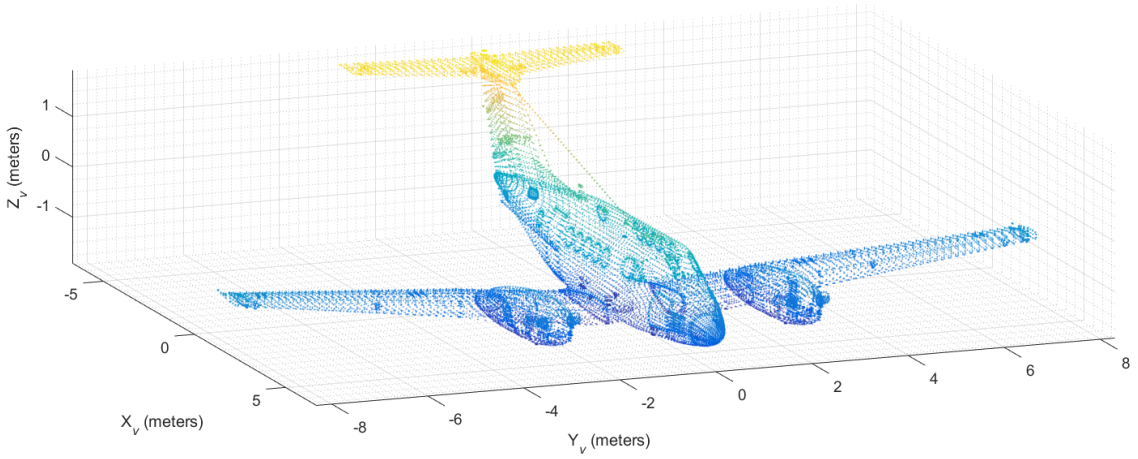


Figure 32. Model Point Cloud Used with Simulation Images.

3.5.3 Point Cloud Generation with OpenCV.

Section 2.3.6.8 summarizes three-dimensional point cloud generation from stereo image pairs. This thesis primarily implements that methodology with a few caveats. One caveat pertains to camera calibrations in simulation, and two caveats apply to camera calibrations in flight test.

With respect to simulation data, the 3DVW cameras were assumed to be ideal and distortion free. Hence, for simulation data analysis all distortion coefficients were zero for both 3DVW cameras, and the camera calibration matrix for both cameras was known perfectly. Additionally, the precise location and orientation of both cameras was known perfectly with respect to the p-frame and to one another. Hence, no camera calibrations of any kind were required to generate and process 3DVW simulation imagery.

With respect to flight test data, two key modifications were made to the calibration steps outlined in Section 2.3.6.8. First, prior to flight tests, intrinsic camera calibrations were performed on both cameras using Bouguet's MATLAB® camera calibration software. Additionally, a stereo camera calibration was performed using

Bouguet's MATLAB® camera calibration software. The results from the intrinsic calibrations (i.e. distortion coefficients, k_{i_L} and k_{i_R}, and camera calibration matrices for both cameras, $\mathbf{K}_L$ and $\mathbf{K}_R$) were used to seed and validate the stereo camera calibration results.

Second, the extrinsic camera calibrations performed on both cameras were combined in order to obtain an alternative estimate for the stereo camera calibration results (i.e. alternative estimates of $\mathbf{R}_L^R$ and $\mathbf{T}^R$). Ultimately, analysis was conducted to confirm that these various calibration methods yielded consistent results. This comparison can be found in the Have Vision test information memorandum [7].

With these modifications in mind, the steps outlined in Section 2.3.6.8 constitute the approach used to generate a three-dimensional point cloud from stereo imagery. The OpenCV implementation is very similar to that used by Parsons in [6]. In the implementation used in this thesis, the observation point cloud was returned in the v-frame. Notationally, it can be expressed as $\mathbf{O}^v$. The v-frame was chosen to allow for utilization of the processing and visualization capabilities of the 3DVW. The same coordinate system was used for the model point cloud, $\mathbf{M}^v$.

In Table 2 the reprojection steps from Section 2.3.6.8 are listed along with the approach used to implement them in this thesis. Also included are the pertinent input and output variables for the function or algorithm used. These variables correspond to those defined in Table 2. Many of these functions require the user to specify a set of tuning parameters. Due to time constraints, an extensive tuning process was not conducted in this thesis. Hence, some improvements in algorithm performance could be gained with a greater emphasis on tuning. Sections 3.5.3.1 through 3.5.3.7 discuss the steps and corresponding functions in greater detail.

Table 3. R7 Implementation with OpenCV and MATLAB®.

Step	OpenCV or Other Implementation	Principal Inputs	Principal Outputs
1. Stereo Camera Calibration	Performed with Bouguet's MATLAB® implementation.	$\mathbf{K}_L$, $\mathbf{K}_R$, k_{i_L}, k_{i_R}, Raw Images	$\mathbf{K}_L$, $\mathbf{K}_R$, k_{i_L}, k_{i_R}, $\mathbf{R}_L^R$, $\mathbf{T}^R$
2. Extrinsic Camera Calibration	Performed with direct measurement.	Vector measurements. See Section 5.4.1	$\mathbf{R}_L^p$, $\mathbf{R}_R^p$
3. Image Capture	N/A	N/A	N/A
4. Image Rectification	`StereoRectify()`, `InitUndistortRectifyMap()`, `Remap()`	$\mathbf{K}_L$, $\mathbf{K}_R$, k_{i_L}, k_{i_R}, $\mathbf{R}_L^R$, $\mathbf{T}^R$	$\mathbf{R}_L^{L_r}$, $\mathbf{R}_R^{R_r}$, $\mathbf{Q}$, Rectified Images
5. Pixel Matching	Completed in Step 6.	Completed in Step 6.	Completed in Step 6.
6. Disparity Map Generation	`StereoBM()`, `filterSpeckles`	Rectified Images, Tuning Parameters	Disparity Map
7. Projection into 3D	`reprojectImageTo3D()`	$\mathbf{Q}$, Disparity Map	$\mathbf{O}^{L_r}$
8. Relate to v-frame	Coded with C++ in 3DVW.	$\mathbf{O}^{L_r}$, $\mathbf{R}_{L_r}^L$, $\mathbf{R}_L^v$, $\mathbf{C}_L^v$	$\mathbf{O}^v$

3.5.3.1 Stereo Camera Calibration.

As discussed in Section 2.3.6.3, stereo camera calibrations are primarily used to obtain estimates of $\mathbf{R}_L^R$ and $\mathbf{T}^R$. As previously discussed, this calibration was only performed when using real-world data since calibration parameters were known perfectly in the 3DVW simulation runs. When examining the flight test data, intrinsic camera calibrations were also performed on both cameras prior to the stereo camera calibration. This enabled seeding of the stereo camera calibration algorithm with estimates of the camera calibrations matrices and distortion coefficients.

After the stereo camera calibration was completed, that algorithm's returns for $\mathbf{K}_L$, $\mathbf{K}_R$, k_{i_L}, and k_{i_R} were compared to the inputs provided by the intrinsic calibrations. The returns were found to be consistent, as detailed in the Have Vision test information memorandum [7].

3.5.3.2 Extrinsic Camera Calibration.

Section 2.3.5 describes how extrinsic camera calibrations are used to relate camera frames to an arbitrary world frame. In this thesis, the world frame of interest is the p-frame, and the quantities of interest are $\mathbf{R}_L^p$, $\mathbf{R}_R^p$, $\mathbf{C}_L^p$, and $\mathbf{C}_R^p$. Figure 19 depicts these DCMs and vectors.

For extrinsic camera calibrations, direct measurements were used during flight testing. These measurements returned DCM estimates for $\mathbf{R}_p^L$ and $\mathbf{R}_p^R$ and vector estimates of $\mathbf{C}_L^p$ and $\mathbf{C}_R^p$. The USAF Test Pilot School special instrumentation team performed these measurements using a FaroArm® system. The FaroArm® was capable of providing vector measurements at the sub-millimeter level and relative orientation estimates at the sub-milliradian level. More detail on this process is found in Sections 5.3.1.1, 5.4.1, and 5.4.4.

Since both cameras were extrinsically calibrated, the results were used to validate the stereo camera calibration results. The two DCMs, $\mathbf{R}_p^L$ and $\mathbf{R}_p^R$, were used to obtain stereo camera calibration estimates according to the equation:

$$\mathbf{R}_L^R = \mathbf{R}_p^R \mathbf{R}_L^p \tag{92}$$

For this DCM, the extrinsic and stereo camera calibration results were found to be consistent, as detailed in the Have Vision test information memorandum [7].

The two vectors obtained from direct measurement, $\mathbf{C}_L^p$ and $\mathbf{C}_R^p$, were used to validate the stereo camera calibration estimate of $\mathbf{T}^R$ per the equation:

$$\mathbf{T}^R = \mathbf{R}_p^R(\mathbf{C}_L^p - \mathbf{C}_R^p) \tag{93}$$

Again, both estimates of $\mathbf{T}^R$ were consistent, as detailed in the Have Vision test information memorandum [7]. Table 4 summarizes how intrinsic, stereo, and extrinsic camera calibrations can be used to validate one another.

Table 4. Overview of Camera Calibration Routines. Items Marked with a † Are Quantities That Must Be Provided as Estimates. Items Marked with a * Are Optional Outputs. Deducible Outputs Are Not Directly Returned but Can Be Estimated from the Outputs.

Camera Calibration	Required Inputs	Outputs	Deducible Outputs	Can Validate
Intrinsic	None	$\mathbf{K}_L$, $\mathbf{K}_R$, k_{i_L}, k_{i_R}	None	Stereo Extrinsic
Stereo	$\mathbf{K}_L^\dagger$, $\mathbf{K}_R^\dagger$, $k_{i_L}^\dagger$, $k_{i_R}^\dagger$	$\mathbf{K}_L^*$, $\mathbf{K}_R^*$, $k_{i_L}^*$, $k_{i_R}^*$, $\mathbf{R}_L^R$, $\mathbf{T}^R$	None	Intrinsic Extrinsic
Extrinsic (Direct Measurement)	None	$\mathbf{C}_L^p$, $\mathbf{C}_R^p$, $\mathbf{R}_p^L$, $\mathbf{R}_p^R$	$\mathbf{T}^R$, $\mathbf{R}_L^R$	Stereo

3.5.3.3 Image Capture.

In the 3DVW, stereo images were generated to be consistent with distortion free cameras of the same focal length and image size as those captured in flight test. In flight test, stereo images were captured by a pair of Prosilica GT1290C models with a Kowa 4.4 millimeter fixed focus lens and a Gigabit Ethernet (GigE) interface produced by Allied Vision®. These cameras operated in the 400-to-700 nanometer spectral range with a 55.5° horizontal field of view and a 43.0° vertical field of view. The cameras produced images at 30 Hz; image sizes were 1280-by-960 pixels. Section 5.2 discusses the cameras and all other flight test hardware in greater detail.

3.5.3.4 Image Rectification.

The stereo image rectification implementation followed the process outlined in Section 2.3.6.4. Three OpenCV functions were used to execute the process. These

functions are described in detail by Bradski and Kaehler [32] and in OpenCV's online documentation [42]. The first, `StereoRectify()`, accepts as inputs the variables $\mathbf{K}_L$, $\mathbf{K}_R$, k_{i_L}, k_{i_R}, $\mathbf{R}_L^R$, and $\mathbf{T}^R$. Applying the process described in Section 2.3.6.4, the function returns $\mathbf{R}_L^{L_r}$, $\mathbf{R}_R^{R_r}$, and $\mathbf{Q}$.

The second, `InitUndistortRectifyMap()`, uses the `StereoRectify()` DCM outputs to generate a mapping from raw images to rectified images. This mapping takes the form of four arrays—`map1x`, `map1y`, `map2x`, and `map2y`. The arrays `map1x` and `map1y` describe the x and y pixel mapping, respectively, for the raw left camera image. The arrays `map2x` and `map2y` describe the analogous mapping for the raw right camera image. Third, the function `Remap()` accepts each mapping array pair along with a raw image from the corresponding camera and returns the rectified image.

3.5.3.5 Pixel Matching and Disparity Map Generation.

Section 2.3.6.5 describes the method used to match pixels and generate a disparity map from a rectified image pair. This thesis implemented the block matching method described therein with the OpenCV functions `StereoBM()` and `filterSpeckles()`. A semi-global block matching approach was briefly examined, but ran approximately 20 times more slowly than the block matching implementation. In the future, with greater computational power, a semi-global block matching technique could reach viable speeds for real-time implementation.

The first function, `StereoBM()`, is initialized with a user-specified number of disparities parameter and a SAD window size. Throughout this thesis, the disparities were set to vary between $[0, 64]$, inclusive, and the SAD window size was set to 9-by-9. These parameters were not extensively tuned, but were determined to yield acceptable performance.

The second function, `filterSpeckles()`, accepts a maximum speckle size argument, a maximum disparity difference argument, and a disparity value to assign any eliminated speckles. The first two parameters are as described in Section 2.3.6.5 and were set to 144 and 4, respectively. The third parameter was set to zero. As a result, any pixels contained within an eliminated speckle were assigned a disparity value of zero. Again, these parameters were not tuned extensively.

3.5.3.6 Projection into 3D.

Section 2.3.6.6 describes how a disparity map can be transformed into a three-dimensional point cloud in the left rectified camera frame with the matrix $\mathbf{Q}$. The OpenCV function `reprojectImageTo3D()` was used to perform this task. The only required inputs to the function are listed in Table 3—no tuning parameters are required. The output is an three-dimensional point cloud expressed in the left rectified camera frame, $\mathbf{O}^{L_r}$.

3.5.3.7 Relate to v-frame.

Finally, Section 2.3.6.7 describes how points in the left rectified camera frame can be re-defined into an arbitrary frame. With real world data, this process relies upon the DCM between the left rectified camera and the left camera frames as well as the returns from the left camera extrinsic camera calibration. Based on Equation 70, the point cloud returned by `reprojectImageTo3D()` can be transformed into the v-frame according to,

$$\mathbf{O}^{v} = \mathbf{R}^{v}_{L}\mathbf{R}^{L}_{L_r}\mathbf{O}^{L_r} + \mathbf{C}^{v}_{L}. \tag{94}$$

Here, $\mathbf{R}_L^v = \mathbf{R}_p^v \mathbf{R}_L^p$ and $\mathbf{C}_L^v = \mathbf{R}_p^v \mathbf{C}_L^p$, where:

$$\mathbf{R}_p^v = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{95}$$

In this manner, the observed point cloud expressed in the v-frame was obtained using OpenCV functions. Next, this point cloud was filtered and then used to deduce the relative pose of the two aircraft with the ICP algorithm.

3.5.4 Observed Point Cloud Filtering.

As discussed in Section 2.4, the ICP algorithm is sensitive to noise and outliers. More specifically, the ICP algorithm will converge to a local minimum under the condition that the observed point cloud contains only points that correspond to the model point cloud. Any points derived from objects other than the object of interest will cause errors in the ICP algorithm's rigid transformation estimate. Conversely, the model point cloud need not necessarily contain points corresponding to all parts of the observed point cloud provided that the ICP algorithm is seeded with an appropriate initial transformation.

Based on these considerations, it is essential to eliminate noise and outliers from the observed point cloud prior to matching it with the model point cloud with the ICP algorithm. As a result, the R7 algorithm implements three distinct filtering methods to eliminate nuisance points, outliers, and noise. First, a boom filter is applied in cases in which a tanker boom is in the field of view. Second, a median filter is applied to eliminate gross outlier clusters. Third, a MathWorks® developed denoising algorithm is applied to eliminate any remaining outliers. After these three filters have been

applied, the point cloud size is reduced to enable quicker implementation of ICP with a MathWorks® developed downsampling algorithm.

3.5.4.1 Boom Filtering.

In the AR context examined in this thesis, the tanker boom will typically be in the field of view of both cameras. As a result, the boom will be captured in the disparity map and ultimately in the reprojected point cloud with the R7 algorithm. The ICP algorithm will not converge properly if the observed point cloud contains points not found in the model point cloud [8]. Hence, in order to obtain an accurate relative pose estimate from the ICP algorithm, a method to remove the boom-generated points from the observed point cloud is required, referred to henceforth as a boom filter.

The boom filter discussed in this section achieved good results with 3DVW simulation data. However, further improvements upon it could be made and filter design did not account for data sampling rates or the sensors used on USAF tanker booms. The design of the boom filter used in this thesis is intended to provide a useful starting point for future research efforts.

USAF tankers can output information on boom extension, boom azimuth, and boom elevation. The definitions of these three parameters can be defined in any arbitrarily selected Cartesian frame affixed to the tanker. This thesis used the following definitions of these three parameters, which can easily be related to the output available from USAF tanker boom systems.

Boom extension was defined as the total length of the boom from the point at which it is affixed to the airframe body to the boom nozzle. Boom elevation and boom azimuth were defined with Euler angles. The Euler angles were initialized from the v-frame (a nose-left wing-up right-handed, Cartesian frame affixed to the tanker). Following the 3-2-1 convention, a rotation about the v-frame's y-axis followed by a

rotation about the resulting frame's x-axis will align the z-axis of the final frame with the shaft of the tanker boom. Using this convention, the boom elevation was defined as the required rotation about the v-frame's y-axis and the boom azimuth was defined the required rotation about the resultant x-axis.

These rotations result in the B-frame described in Section 3.1. The origin of the B-frame was defined to be located at the point where the tanker boom affixes to the body the tanker.

For example, a "neutral" boom orientation on a KC-10 or KC-135, the boom is located 30° below the longitudinal axis of the tanker (the x-axis of the v-frame). Based on the definition of boom elevation above, this is a boom elevation of 240°. Any left or right angular movement of the boom from this position would constitute a change in the boom azimuth from zero. A move toward the right wing of the tanker would be a positive azimuth and a move toward the left wing would be a negative azimuth. The relationship between the v-frame and the B-frame is depicted in Figure 33.

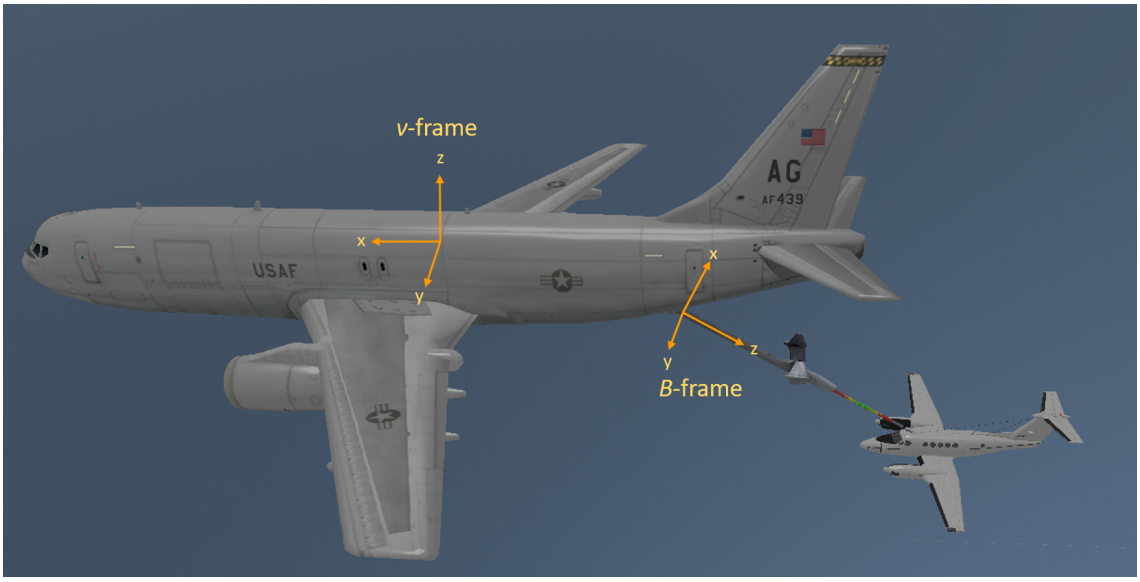


Figure 33. Depiction of the v-Frame and the B-Frame. Following the 3-2-1 Convention, Boom Elevation is the Required Rotation About the v-Frame's y-Axis and Boom Azimuth is the Required Rotation About the Resultant x-Axis to Align the Two Frames.

The DCM describing the rotation between these two frames, was specified in terms of these definitions of boom elevation, θ, and boom azimuth, ϕ:

$$\mathbf{R}_v^B = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ \sin\theta\sin\phi & \cos\phi & \cos\theta\sin\phi \\ \sin\theta\cos\phi & -\sin\phi & \cos\theta\cos\phi \end{bmatrix} \tag{96}$$

In order to filter out the boom in simulation, the observed point cloud was transformed into the B-frame per the equation:

$$\mathbf{O}^B = \mathbf{R}_p^B(\mathbf{O}^v - \mathbf{B}_{\text{MAT}}^v) \tag{97}$$

Where $\mathbf{B}_{\text{MAT}}^v$ is a matrix of the same size as $\mathbf{O}^v$ wherein every column is the location of the B-frame origin in the v-frame (i.e. the vector $\mathbf{B}^v$). Equation (97) was used to

express the observed point cloud in the B-frame. As a result, every column in $\mathbf{O}^B$ was a Cartesian point coordinate of the form (X_B, Y_B, Z_B).

In order to apply the designed boom filter, these Cartesian coordinates were partially expressed as cylindrical coordinates according to:

$$L_B = Z_B, \tag{98}$$

$$r_B = \sqrt{X_B^2 + Y_B^2}. \tag{99}$$

Here, L_B is the cylindrical coordinate form of the Cartesian B-frame's z-axis. The two coordinates are equivalent. Hence, there is no need to compute L_B. The coordinate r_B is the radius component of the cylindrical coordinate system. It describes how far out a point is from cylindrical coordinate system's L_B-axis. The third cylindrical component, the angular deflection, was not required. This resulted in computational savings since computing the angular component requires knowledge of the point's x-component in the Cartesian frame, the radius component in the cylindrical frame, and an arcsin operation.

For every point in $\mathbf{O}^B$, the x-component and the z-component were stored in a vectors of the same length as $\mathbf{O}^B$ called $\mathbf{X}_B$ and $\mathbf{L}_B$, respectively. Additionally, for every point in $\mathbf{O}^B$ a value for r_B was computed. This value was stored in a vector of the same length $\mathbf{O}^B$ termed $\mathbf{r}_B$. These three vectors were used to apply the boom filter.

The boom itself can be thought of as consisting of four components, as shown in Figure 34. This components are most easily defined with respect to the z-axis of the B-frame. The first segment extends from the point at which the boom connects to the tanker (where $Z_B = 0$) to a point near the boom wings (where $Z_B = L_1$). The second portion consists of a section encompassing the boom wings. It begins where $Z_B = L_1$

and ends where $Z_B = L_2$. The third section begins at the end of the second section (where $Z_B = L_2$) and extends to the point where the extendable portion of the boom emerges from the boom housing (where $Z_B = L_3$). The final portion consists of the extendable portion of the boom. It begins where $Z_B = L_3$ and terminates at the end of the boom (where $Z_B = L_4$). Each of these components can also be thought of as having a radius, R_i, as shown in the figure. Finally, point cloud points resultant form the boom wings only manifest above a certain value of X_B. In Figure 34, this value is X_{min}.

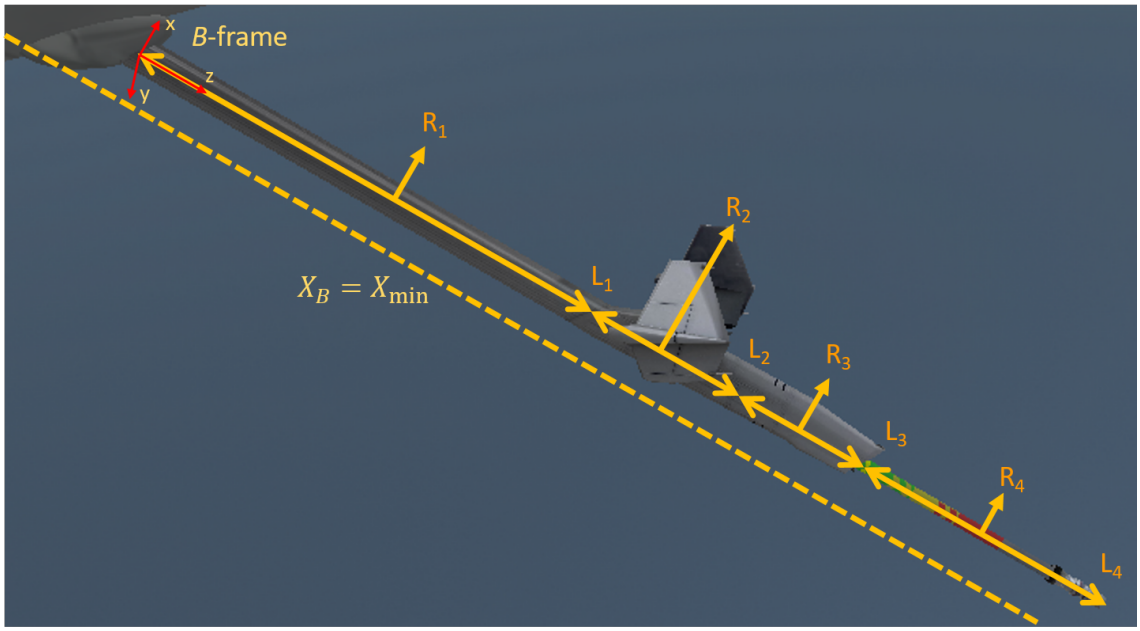


Figure 34. Depiction of the Tanker Boom and the Parameters Defining the Boom Filter.

Based on these definitions, the boom filter was implemented as follows. A point was removed from the point cloud if any of the following four conditions were met:

1. $L_B \leq L_1$ AND $r_B \leq R_1$
2. $L_1 \leq L_B \leq L_2$ AND $r_B \leq R_2$ AND $X_{\text{min}} \leq X_B$
3. $L_2 \leq L_B \leq L_3$ AND $r_B \leq R_3$

4. $L_3 \leq L_B \leq L_4$ AND $r_B \leq R_4$

Where L_B, r_B, and X_B are the elements corresponding to the point $\mathbf{X}^B$ in the vectors $\mathbf{L}_B$, $\mathbf{r}_B$, and $\mathbf{X}_B$, respectively. Table 5 shows the values that were used for the boom filter during simulation in this thesis.

Table 5. Boom Filter Parameterization. For Simulation Data, L_4 Was Set to Equal the Total Length of the Boom, as Rendered, in Meters. At Full Extension, $L_4 = 18$ meters.

L_1	L_2	L_3	L_4	R_1	R_2	R_3	R_4	$X_{\min}$
6 m	5 m	2 m	Boom length	0.8 m	2.75 m	0.8 m	0.8m	-0.5 m

These settings achieved what the author considered to be acceptable results, but further tuning could lead to improvement. Alternate implementations could also result in improved performance.

3.5.4.2 Median Filtering.

In simulation the boom filter typically eliminated almost all points corresponding to the tanker boom. However, at times outlier point clusters, arising from the boom or poor pixel matching, remained. Additionally, outlier point clusters also arose from background features in simulation. These clusters did not correspond to any actual object at that distance, but would arise when the stereo camera pair observed highly patterned objects in the far distance. An example of this is shown in Figure 35.

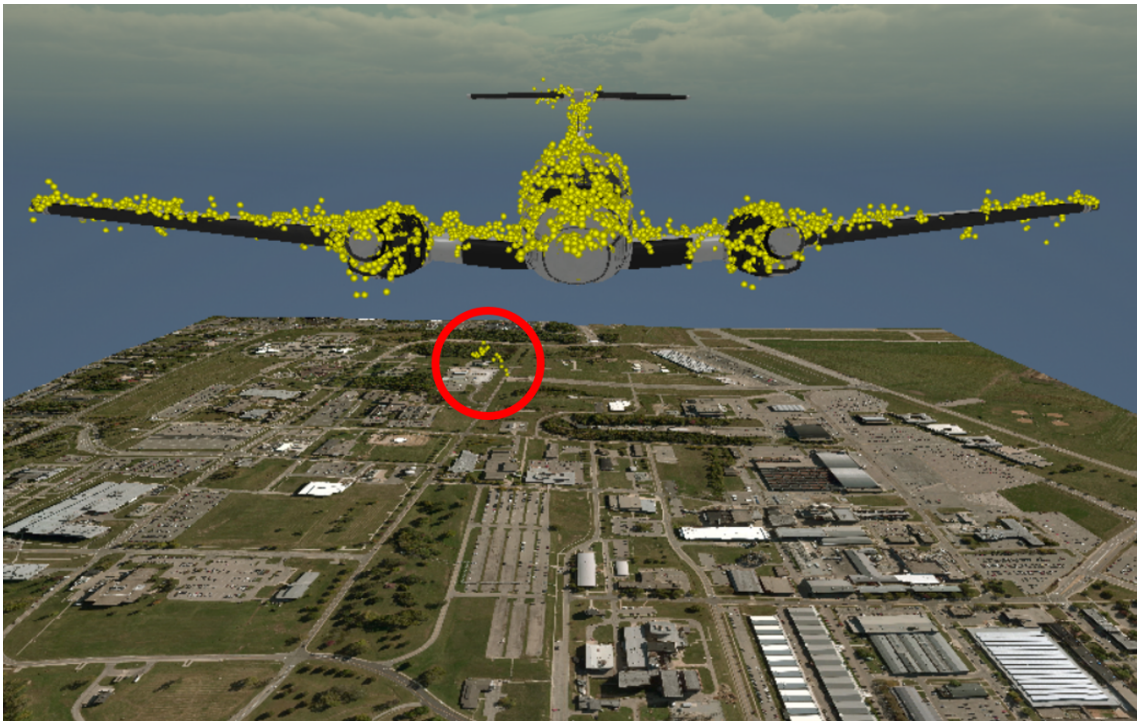


Figure 35. Example of a Nuisance Point Cluster Manifesting After Model Point Cloud Generation.

The median filter was designed specifically to eliminate such nuisance clusters. As implemented, it filtered out any points that were farther than 20 meters plus the Euclidean norm of the ΔX_{L_r}, ΔY_{L_r}, and ΔZ_{L_r} from the location of the point cloud's median point. The values of ΔX_{L_r}, ΔY_{L_r}, and ΔZ_{L_r} were computed at the location of the point cloud's median point. The equations needed to compute these three discrete resolution values are given in Equations (67), (68), and (69).

In order to accomplish the filtering process, the observed point cloud, $\mathbf{O}^v$ was converted back to the L_r-frame per the equation:

$$\mathbf{O}^{L_r} = \mathbf{R}_L^{L_r}\mathbf{R}_p^L(\mathbf{R}_v^p\mathbf{O}^v - \mathbf{C}_{L,\text{MAT}}^p) \tag{100}$$

Where $\mathbf{C}_{L,\text{MAT}}^p$ is an array of the same dimensions as $\mathbf{O}^v$. Every column in $\mathbf{C}_{L,\text{MAT}}^p$ is the vector $\mathbf{C}_L^p$, the location of the left camera frame origin expressed in the p-frame.

Next, the disparity value, d, corresponding to the median point was computed from Equation (66) as were the image pixel values corresponding to the median point. These pixels coordinates are denoted as x_{L_R} and y_{L_R} in Equation (66). These pixel coordinates and disparity values were then used to compute the values of ΔX_{L_r}, ΔY_{L_r}, and ΔZ_{L_r} at the median point.

The design of the median filter was heuristic and not considered by the author to be the ideal solution to such nuisance point clusters. Two important aspects of the filter merit emphasis. First, it operates under the assumption that the median point in the point cloud relates to the aircraft. Second, due to the nature of the filter's logic, nuisance point clusters that fall within 20 meters plus the Euclidean norm of the position resolutions will not be filtered.

In the future, this filter could be improved with any number of modifications. Filter operation may be improved if the filter was based on a position estimate of the receiving aircraft from a Kalman filter as opposed to basing operation on the median point. Additionally, a host of other algorithms, such as K-nearest neighbors implementations and clustering algorithms used in multi-target tracking, could merit exploration as alternative implementations.

3.5.4.3 Point Cloud Denoising.

After filtering, the point cloud was denoised with a MathWorks® developed algorithm. Within MATLAB®, the algorithm is called with the function `pcdenoise`. Using this function, the algorithm can be configured in a number of ways. The configuration used in this thesis performs the following:

1. Computes the mean distance of every point to a user-defined number of nearest neighbors, k, in the point cloud.

2. The mean and standard deviation of this mean distance value is calculated for the point cloud population.

3. If an individual point's mean distance is greater than the population's mean distance plus a user-defined number of standard deviations, n, then that point is eliminated from the point cloud.

This thesis used values of $k = 4$ and $n = 1$. Again, an extensive tuning process was not undertaken.

3.5.4.4 Point Cloud Downsampling.

After filtering and denoising, the point cloud was downsampled to enable faster operation of the ICP algorithm. A MathWorks® developed algorithm was used for downsampling via the MATLAB® function `pcdownsample`. A number of different algorithm implementations are available with this function. The configuration used in this thesis performs the following:

1. The function computes a bounding box for the entire point cloud.

2. The bounding box is divided into cubes of user-defined dimensionality with side length s.

3. Points falling within each cube are averaged to return a single point whose location is the mean location of all points within that cube.

This thesis used a value of $s = 0.3$ meters. Again, an extensive tuning process was not undertaken.

3.5.5 ICP Implementation.

This section describes the implementation of the ICP algorithm used in this thesis. First, an explanation of how the ICP algorithm returns the desired relative position

and attitude measurements is given. Second, the method used to position and rotate the model point cloud is described. Third, the implementation used in MATLAB® is described. Fourth, the method used to transform the ICP algorithm's outputs into a desired reference frame is described. Fifth, the method used to identify an initial transformation for the ICP algorithm is described. Lastly, the attitude fed version of the ICP algorithm is developed.

3.5.5.1 Obtaining Measurements from ICP.

In the context of the R7 algorithm, the ICP algorithm works by translating and rotating the observation point cloud to achieve an acceptable match with the model point cloud. Both the model and observation point clouds represent the body of the target aircraft. However, the position and orientation of the model point cloud must be carefully defined in order to achieve useful measurements from the ICP algorithm. The method used to achieve this in an AR context is described below.

In AR a particular point of interest on or within the tanker aircraft is the point from which relative position is measured. This point can be referred to as the origin of the frame of interest. Likewise, a particular point of interest on or within the receiver aircraft is the point to which relative position is measured. This point is referred to as the receiver feature of interest. The points in the model point cloud correspond to the receiver. To achieve the desired measurements from the ICP algorithm, the points in the model point cloud must be translated such that its representation of the receiver feature of interest is positioned at the origin of the frame of interest. Additionally the axis system of the model point cloud must be aligned with the frame of interest. The frame of interest is typically the body frame of the tanker aircraft. However the frame of interest could also be defined to be any frame that has a known geometry with respect to the tanker aircraft's body frame.

The ICP algorithm will return a translation vector and DCM that align the observation point cloud with the model point cloud. If the position and orientation of the model point cloud is conditioned as described above, then the ICP algorithm will return the desired relative position and relative attitude measurements.

3.5.5.2 Model Point Cloud Positioning.

The 3DVW was used for processing of all simulation and flight test observation images. As a result, the model point cloud origin was positioned so as to be co-located with the origin of the v-frame. When using flight test data, the origin of the v-frame was defined to be collocated with the origin of the truth data measurement system (Section 5.4.6.2 explains this in greater detail). Additionally, the axis system of the model point cloud was defined to be a nose-left wing-up, right-handed, Cartesian coordinate system. This is the same convention used with the v-frame and the d-frame. Prior to execution of the ICP algorithm, the model point cloud's axis system was aligned with the axis system of the v-frame. Hence, the model point cloud can be referred to as $\mathbf{M}^v$.

Similarly, since the 3DVW was used for observation point cloud generation, the observation point cloud was expressed in the v-frame. Recall that the equivalent to the v-frame for the receiver is called the d-frame. Hence, the relative attitude depicted by the observation point cloud is a representation of the relative orientation of the d-frame with respect to the v-frame.

Since this pose was chosen for the model point cloud and since observation point clouds were expressed in the v-frame, the translation vector and DCM returned by the ICP algorithm represented the relative position, $\mathbf{s}^v$, and attitude, $\mathbf{R}_v^d$, of the trailing aircraft's d-frame with respect to the lead aircraft's v-frame. In practice, the nose-right wing-down frames p and s, or the left camera frame, the L-frame, are of

greater interest than the nose-left wing-up v and d-frames. Results were converted to the frames of interest via the appropriate DCMs and vectors, yielding the results of interest, such as $\mathbf{s}^p$ and $\mathbf{R}_p^s$.

3.5.5.3 Implementation in MATLAB®.

The MATLAB® function `pcregrigid` was used to implement the ICP algorithm. The function includes the ability to set several parameters dictating algorithm operation.

First, the `pcregrigid` function enables the user to specify an inlier ratio parameter with a value between 0 and 1. The inlier ratio determines the percentage of points in the observed point cloud that will be used during ICP. Two things occur if the inlier ratio is set to a fraction, such as 0.8. First, the Euclidean distance between all points and their nearest neighbor is computed. Second, for an inlier ratio of 0.8, only points corresponding to the smallest 80% of nearest neighbor distances are retained for that iteration of the ICP algorithm. In this manner points far from the rest of the point cloud are not considered. If the parameter is set to 1, then all points are used at every iteration; if it were set to 0, then ICP would not be executed. For data in this thesis, an inlier ratio of 1 was used.

Second, the user can specify a maximum number of iterations to allow the algorithm to run. This parameter was set to 20 for the data in this thesis. More iterations can yield more accurate results at a cost of greater computational time. Additionally, the benefit of an additional iteration diminishes as the number of allowed iterations increases.

Third, the user can specify a maximum translation and rotation change tolerance. The algorithm will stop iterating if both the translation and rotation tolerance values are reached in three consecutive iterations before the maximum number of

iterations is reached. This enables faster algorithm operation. The translation tolerance parameter relates the Euclidean distance between successive overall translation vector estimates. The rotation change tolerance relates the total difference in angular change, in radians, between successive rotation vector estimates. In this thesis, a value of 1 centimeter was used for the translation change tolerance and a value of 0.001 radians was used for the rotation change tolerance.

Fourth, the user can specify an initial affine transformation with which to seed the algorithm. This affine transformation is passed as an affine transformation matrix containing a DCM and translation vector. The method of determining an initial transformation is discussed in detail in Section 3.5.5.5.

3.5.5.4 Transforming ICP Outputs.

Since the model and observed point clouds used in MATLAB® are expressed in the v-frame ($\mathbf{M}^v$ and $\mathbf{O}^v$, respectively), the ICP portion of the R7 algorithm will return relative position and attitude estimates in relation to the v-frame and d-frame (in the form of the vector $\mathbf{s}^v$ and the DCM $\mathbf{R}_v^d$). Additionally, when using simulated data, truth data is available relative to the d-frame for the relative attitude DCM (in the form of the matrix $\mathbf{R}_v^d$).

These outputs can be converted to quantities relative to the p-frame and s-frame with the equations:

$$\mathbf{s}^p = \mathbf{R}_v^p \mathbf{s}^v \tag{101}$$

$$\mathbf{R}_p^s = \mathbf{R}_d^s \mathbf{R}_v^d \mathbf{R}_p^v \tag{102}$$

More generally, the relative pose estimates returned by R7 can be transformed into any set of arbitrary frames with linear algebra.

3.5.5.5 Initial Transformation for ICP.

As discussed in Section 2.4, an appropriate initial transformation is important to aiding convergence of the ICP algorithm. Ideally, the initial transformation would be based upon some combination of data contained in the observed point cloud and the output from a Kalman filter. Alternatively, a feature matching algorithm could be used to help identify the origin of the s-frame in the imagery, and that point could be used as an initial translation vector. Since the methods in this thesis did not include the operation of a relative navigation Kalman filter and did not utilize any feature matching algorithm, only data from the observed point cloud was available for use.

For the initial transformation vector, the location of the median point in the point cloud was used. This value was selected for two reasons. First, the median was selected instead of the mean since the median is far less sensitive to outliers. Second, it was assumed that the median point would correspond to a point somewhere near the center of the observed aircraft.

For the DCM, the identity matrix was chosen, unless the attitude fed version of ICP was being examined. This value was chosen because the receiving aircraft should be expected to be somewhere relatively close to level flight during AR operations.

As discussed above, the initial translation vector and DCM must be combined into an affine transformation matrix in order to be passed to the MATLAB® `pcregrigid` function. An understanding of how `pcregrigid` utilizes this transformation matrix is essential to proper seeding of it. The transformation matrix operates in the following fashion:

$$\left[\left(\mathbf{p}^B\right)^T,\ 1\right] = \left[\left(\mathbf{p}^A\right)^T,\ 1\right] \begin{bmatrix} \mathbf{R}^A_{B,3\text{x}3} & 0_{3\text{x}1} \\ -\left(\mathbf{R}^B_A \mathbf{T}^A\right)^T & 1 \end{bmatrix}. \tag{103}$$

The four-by-four matrix on the right hand side is the affine transformation matrix used in `pcregrigid`. Here, $\mathbf{p}^B$ and $\mathbf{p}^A$ represent the point $\mathbf{p}$ expressed in the B-frame

and A-frame, respectively. The matrix $\mathbf{R}_A^B$ is a DCM that goes from the A-frame to the B-frame, and $\mathbf{T}^A$ is the location of the origin of the B-frame in the A-frame. The net effect of Equation (103) is to transform the point $\mathbf{p}^A$ into the B-frame.

Therefore, if one had perfect knowledge of the the translation and rotation between the observed and model point clouds, then perfect initial transformation matrix, $\mathbf{F}$, for the variables of interest in this thesis would be:

$$\mathbf{F} = \begin{bmatrix} \mathbf{R}_d^v & 0_{3\text{x}1} \\ -\left(\mathbf{R}_v^d \mathbf{d}^v\right)^T & 1 \end{bmatrix}. \tag{104}$$

Where the vector $\mathbf{d}^v$ gives the location of the origin of the d-frame in the v-frame. The result of the operation $-\left(\mathbf{R}_v^d \mathbf{d}^v\right)$ is therefore to return the location of the origin of the v-frame in the d-frame, $\mathbf{v}^d$.

Recall that the d-frame is a nose-left wing-up frame centered on the receiver and the v-frame is a nose-left wing-up frame centered on the tanker. As a result, the vectors $\mathbf{d}^v$ and $\mathbf{s}^v$ are identical. Therefore, if one has an estimate for $\mathbf{s}^p$, say from a Kalman filter, then one can easily obtain an estimate of $\mathbf{d}^v$ via the operation $\mathbf{s}^v = \mathbf{R}_p^v \mathbf{s}^p$. As described above, for this thesis, the median point cloud point served as an estimate of $\mathbf{s}^v$.

Likewise, the DCM $\mathbf{R}_v^d$ can easily be related to the DCM $\mathbf{R}_p^s$ that would be output from a Kalman filter or that could be obtained directly from inspection of IMU measurements. This is done via the simple relationship $\mathbf{R}_v^d = \mathbf{R}_s^d \mathbf{R}_p^s \mathbf{R}_v^p$. As described above, for this thesis, the identity matrix was used as an initial estimate for $\mathbf{R}_v^d$ unless the attitude fed version of the ICP algorithm was being used. The attitude fed version of the ICP algorithm is described in the next section.

3.5.5.6 Attitude Fed ICP.

When providing the ICP algorithm with a highly accurate relative attitude estimate, it is possible to confine the ICP iterations to only estimate a translation vector from iteration to iteration. The rotation estimate can be left fixed at the initial estimate. This reduces the required computations to two types. First, the nearest neighbor model point for each observation point must be identified. Second, an incremental shift given by the optimal translation vector in Equation (75) must be computed. In this equation, $\mathbf{R}$ is identity.

The result is a large savings in computational cost. No cross-covariance between point clouds need be computed, nor does the $\mathbf{Q}(\mathbf{\Sigma}_{px})$ matrix described in Equation (74) need to be formed, nor do eigenvalues and eigenvectors of this $\mathbf{Q}(\mathbf{\Sigma}_{px})$ matrix need to be computed.

In practice, the position estimates obtained via this alteration to ICP proved to be as accurate or better than those obtained via full implementation of the ICP algorithm.

3.6 Error Computations

For tests of the V5 and R7 algorithms conducted with simulated imagery in the 3DVW, perfect knowledge of the true relative position, $\mathbf{s}^p$ and relative orientation of the tanker-receiver formation, $\mathbf{R}_p^s$, were available. When using flight test data, precise estimates of the relative position and attitude of the two aircraft were available from the truth data system. The truth data system's estimates were assumed to be perfect for error computations.

Relative position estimate errors were computed as:

$$\mathbf{p}_{\text{error}} = \mathbf{p}_{\text{estimate}} - \mathbf{p}_{\text{truth}} \tag{105}$$

Where $\mathbf{p}$ is a 3-by-1 position vector in the frame of interest, $\mathbf{p}_{\text{estimate}}$ is the position estimate returned from V5 or R7, and $\mathbf{p}_{\text{truth}}$ is the truth data value taken either from the 3DVW or the truth data system.

To compute relative attitude estimate errors, the DCM $\mathbf{R}_p^s$ was converted to a relative roll, pitch, and yaw in accordance with the 3-2-1 convention. Then, the relative errors were computed using the same method as that for positions shown in Equation (105).

3.7 Tests of Statistical Significance

On simulation and flight test data, MATLAB® functions were used to execute tests of statistical significance. A significance level of 5% was considered to be statistically significant. These tests were used to analyze the means and variances of relative position and attitude estimate errors. The results of these tests were not taken to strictly quantify performance achievable in the various algorithm configurations, but were used to support qualitative conclusions, especially regarding comparisons.

The tests and their corresponding MATLAB® functions are described below. Each of these tests assumes that the sample data are independent draws from a normal distribution [48], [49]. Hence, to facilitate qualitative analysis, it was assumed the V5 and R7 estimator errors met this condition. This is a typical assumption used on measurements applied in a Kalman filter, and these measurements are intended to ultimately be applied in a Kalman filter. However, tests of normality were not applied to error data in this thesis, but should be before the measurements of either algorithm are applied in a filter. Also, spherical errors inherently will not be normally distributed. However, these tests were still applied when comparing spherical errors to facilitate qualitative conclusions.

3.7.1 One-Sample and Paired *t*-tests.

The t-test statistic is:

$$t = \frac{\bar{x} - \mu}{s/\sqrt{n}} \tag{106}$$

Here, the sample mean is $\bar{x}$, the population mean is μ, the sample standard deviation is s, and the sample size is n. The test has $n - 1$ degrees of freedom [48].

The MATLAB® function `ttest` was used to execute one-sample and paired t-tests. One sample t-tests were used to obtain the 95% confidence intervals of mean algorithm errors. In this case, $\bar{x}$ was the average error and s the standard deviation of the errors.

Paired t-tests were used to compare estimated population means when the two data sets being compared were drawn from the same sample set of images. In the paired t-tests, $\bar{x}$ was the mean of the difference of the two errors being compared and s was the standard deviation of those differences.

3.7.2 Two-Sample *t*-tests.

The two-sample t-test statistic is:

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} \tag{107}$$

Where the sample means are $\bar{x}_i$, the sample standard deviations are s_i, and the sample sizes are n_i. The test has $n_1 + n_2 - 2$ degrees of freedom [48].

When observation images were not drawn from the same sample set, the function `ttest2` was used to execute two-sample t-tests. These tests were executed without an assumption of equal variances.

3.7.3 Two-Sample *F*-tests.

The two-sample F-test statistic is:

$$F = \frac{s_1^2}{s_2^2} \tag{108}$$

Where the sample standard deviations are s_i. The numerator has $n_1 - 1$ degrees of freedom and the denominator has $n_2 - 1$ degrees of freedom where n_i is the sample size [49].

The MATLAB® function `vartest2` was used to execute two sample F-tests. Two-sample F-tests were used to compare the population variances of the two samples.

IV. Simulation Data and Analysis

Simulation data for the V5 and R7 algorithms are presented and analyzed in this chapter. First, V5 simulation results are presented. The V5 discussion includes generation of reference and observation images, determination of Gaussian blur level, likelihood function determination, and pixel intensity threshold determination. Then, V5 simulation results are presented making use of different relative position estimators, likelihood functions, prior probability distributions, and pixel intensity threshold levels. Overall, it is shown that an overbounded likelihood function is required for V5 implementation and that a composite position estimator outperforms a maximum likelihood estimator.

Second, R7 simulation results are presented. The R7 discussion begins with simulation results with no tanker boom present in the image. For the no boom case, results when providing the R7 algorithm with the relative attitude of the two aircraft are contrasted with results when allowing R7 to estimate the relative attitude. Results show that using the attitude fed version of the R7 algorithm reduces mean position errors, but may increase position error variances. Next, results with a boom in the camera fields of view are presented. The discussion highlights the efficacy of the boom filter and contrasts performance with the no boom case. It is shown that a boom in the field of view, even with an effective boom filter, modestly increases R7 estimate errors.

Third, R7 and V5 simulation results are compared. This comparison was made using the same random images used in the V5 simulation results presented earlier in the chapter. This comparison shows that the R7 algorithm had lower overall mean errors and error variances, but was more prone to depth bias than the V5 algorithm.

4.1 V5 Data and Analysis, Simulation

This section details the analysis of the V5 algorithm conducted in simulation. Sections 4.1.1 through 4.1.3 describe the images used in algorithm analysis. Section 4.1.4 describes how the level of Gaussian blurring was determined. Section 4.1.5 describes how the likelihood function was identified. Section 4.1.6 describes how a suitable pixel intensity threshold was determined. Finally, leveraging the findings of these preceding sections, Section 4.1.7 details the most significant V5 simulation results.

4.1.1 Reference Image Database Generation.

The reference image database was created in the 3DVW as discussed in Section 3.4. The database consisted of images of the C-12C receiver in front of a blue background captured at ranges varying between 22 and 30 meters from the center of the stereo camera pair. Successive surfaces were separated by 0.5 meters of Euclidean distance. Within each plane, images were separated by an equi-angular spacing of 1° in azimuth and elevation, as measured from a ray beginning at the center of the stereo camera pair and extending to the location of the last point rendered position generated. In azimuth, each surface extended to ±10° left or right of the tanker's longitudinal axis. Azimuth was defined as the angular position left or right of the tanker's longitudinal axis in the tanker xy-plane. In elevation, each surface extended from 15° to 35° below the tanker's longitudinal axis. Elevation was defined as the angular position below the tanker's longitudinal axis in the tanker xz-plane.

This simulation space was chosen to mimic the relative positions actually observed during flight test. Props were not included in the C-12C model. Figure 36 shows an example reference image pair.



Figure 36. From Left to Right, Examples of a 3DVW Left Camera Reference Image and Right Camera Reference Image.

4.1.2 Reference Image Database Attenuation.

When comparing observation images to the reference image database, the search space was confined to a 6 meter-by-6 meter-by-6 meter box. This search space was centered on the relative position in the reference image database closest to the relative position depicted in the observation image. Hence, instead of comparing the entire reference database to a given observation image, only those reference images falling within this 6 meter-by-6 meter-by-6 meter box were used for comparison. This was the same scheme used by Calhoun in [2] and [5]. From a practical standpoint, this bounding was necessary to reduce V5 processing times to durations low enough to enable data analysis. A similar bounding scheme could only be used in a fielded system if the V5 algorithm had high confidence in the likely location of the aircraft before running a given iteration. Flight test results presented in Chapter VI indicate such a scheme is feasible. Figure 37 shows how this bounding scheme operates for relative positions near the center and boundary of the reference image database.

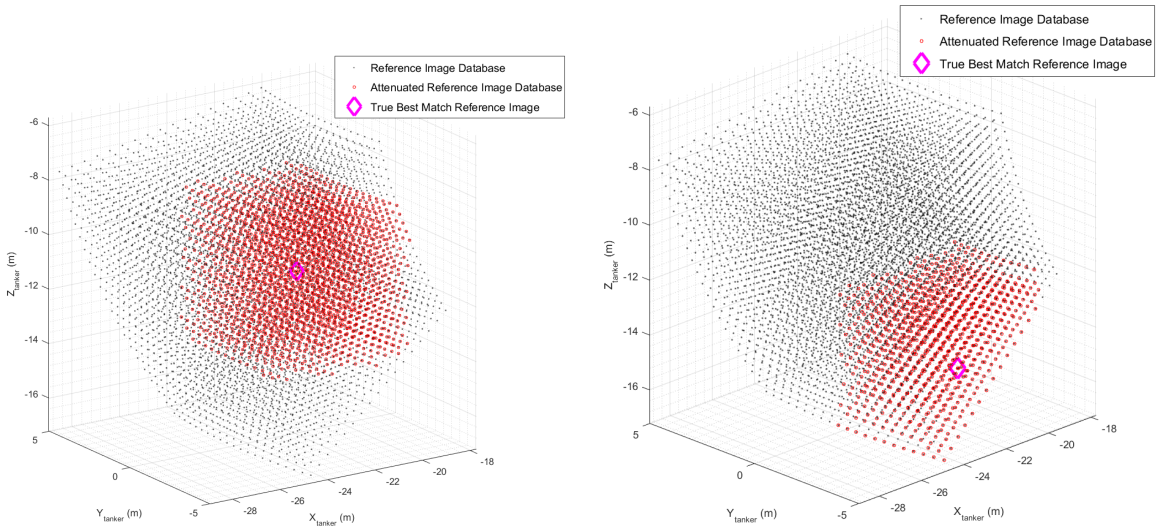


Figure 37. Examples of a the Attenuated Reference Image Database for a Position Near the Center of the Database and for a Position Near the Boundary of the Database.

4.1.3 Observation Image Generation.

Simulated observation images were generated in the 3DVW at an altitude of 1,500 meters above a depiction of Area B at Wright-Patterson Air Force Base. The images depicted 250 randomly generated C-12C receiver positions contained within the reference image database. Props were not included in the C-12C model. Additionally, the attitude of the receiver at each of these positions was randomly generated. The relative roll of the receiver was randomly selected from a uniform distribution varying between $\pm 7°$. The relative pitch of the receiver was randomly selected from a uniform distribution varying between $\pm 4°$. The relative yaw of the receiver was randomly selected from a uniform distribution varying between $\pm 1°$. Figure 38 shows an example observation image pair.

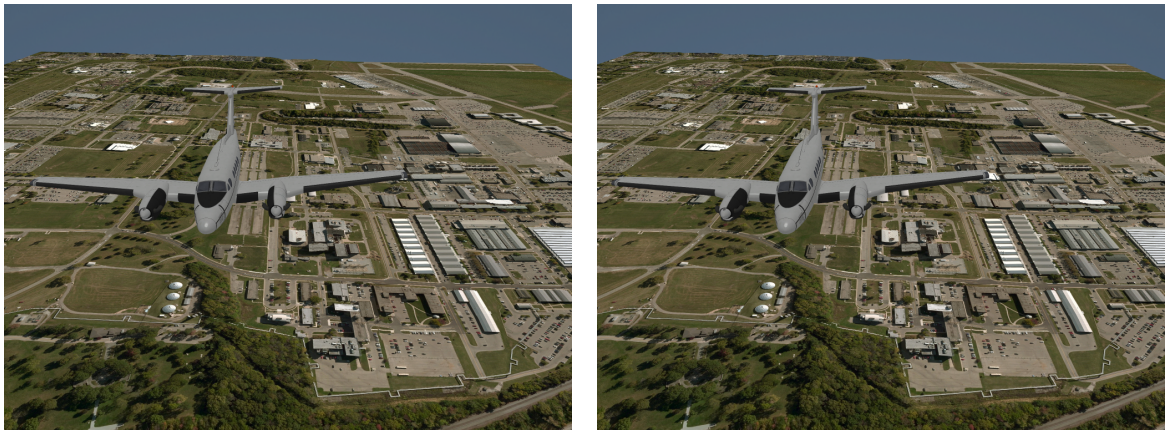


Figure 38. From Left to Right, Examples of a 3DVW Left Camera Observation Image and Right Camera Observation Image.

4.1.4 Gaussian Blur Level Determination.

In order to determine an appropriate level of Gaussian blurring, 150 randomly generated observation images were created in the 3DVW and compared to their best matching image from the reference database. In this context, the best matching reference image is the reference image that most closely depicts the same relative position depicted in the observation image. For the purposes of this analysis, no random attitude perturbations were applied to the observation images and no background features were included. Only a level aircraft and a flat blue background were in each image. A Prewitt edge detector was then applied to every observation-reference image pair, but no other form of processing was applied. Next, the distance from every edge pixel in the observation image to the nearest edge pixel in the best matching reference image was measured. This process was performed for images generated by both the left and right cameras.

Figure 39 shows a histogram plot of the results for every edge pixel in all 150 images analyzed. Figure 40 depicts the same data as a cumulative distribution function (CDF). As can be seen in both figures, the vast majority of edges in the observation

images have a nearest neighboring edge pixel in the appropriate reference image within 10 pixels of distance. However, some nearest neighbors are separated by as many as 86 pixels.

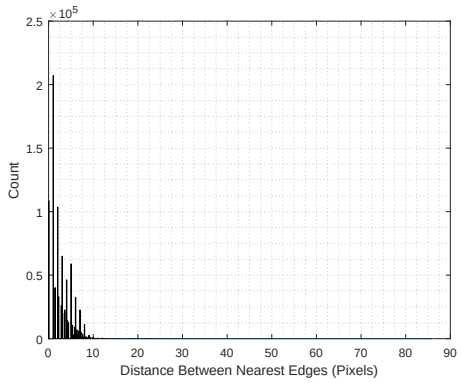


Figure 39. Histogram of Pixel Distance from all Edges in all Observation Images to the Nearest Edge Pixel in the Best Match from the Rendered Database. Data Depicted as Counts in Each Bin.

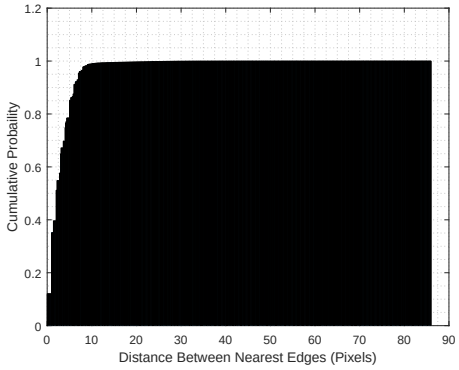


Figure 40. Histogram of Pixel Distance from all Edges in all Observation Images to the Nearest Edge Pixel in the Best Match from the Rendered Database. Data Depicted as a CDF at Each Bin.

In order to determine an appropriate level of Gaussian blurring, the mean and median of these edge pixel distances were computed. The overall mean was approximately 2.88 pixels and the overall median was 2 pixels. Additionally, the mean and median of every observation image examined was plotted versus two quantities. First, these parameters were plotted versus the separation, in meters, between the position of the aircraft in the observation image and the position of the aircraft in its best matching reference image. Second, these parameters were plotted versus the range, in meters, from the observation aircraft to the center of the stereo camera pair capturing the images.

These results are shown in Figure 41. As can be seen in the figure, the mean separation did not exceed 6 pixels while the median separation did not exceed 5 pixels. For this reason, and based upon the satisfactory results obtained in doing so, the σ parameter for the Gaussian blurring operation was set to 5.

A brief analysis of greater and lesser σ parameters was executed and the results with a the parameter set to 5 was deemed the most acceptable. Tuning was not the focus of this thesis, so a more extensive tuning process could be carried out which could marginally enhance the results obtained from this algorithm.

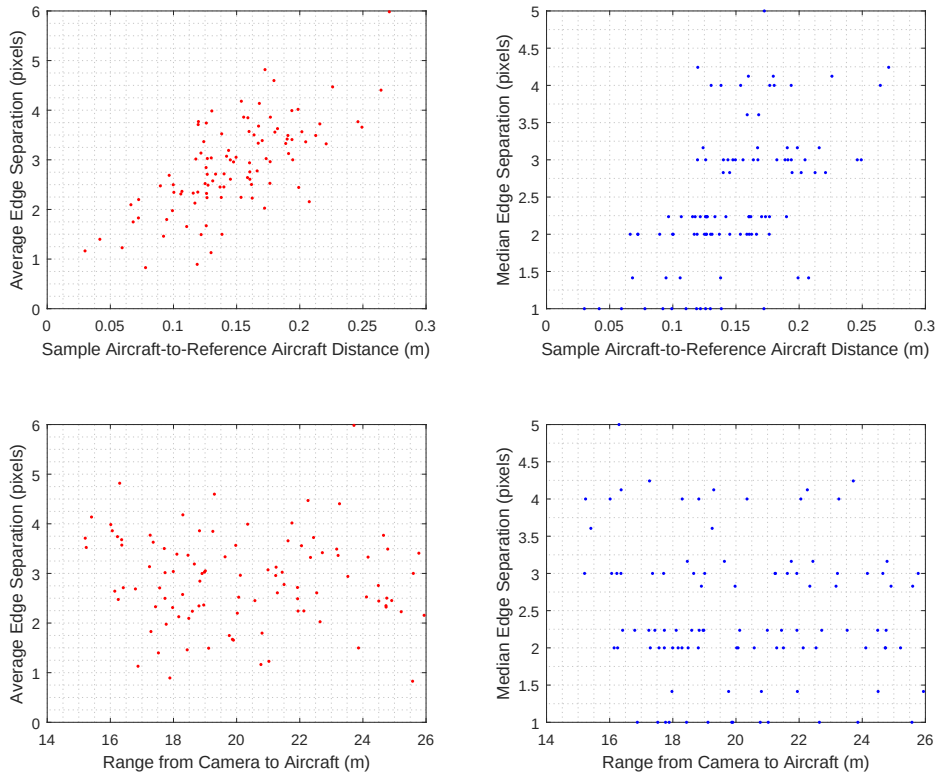


Figure 41. Plots Depicting Mean or Median Edge Separation Distance of all Edges in all Observation Images to the Nearest Edge Pixel in the Best Match from the Reference Image Database. Data was generated when Using a "Spherical" Reference Image Database.

Additionally, the option of using a square rendered database was also examined during the Gaussian blur level determination. Notice in Figure 41, that there is a fairly strong correlation between sample aircraft-to-reference aircraft distance and the average edge separation. As one would expect, the farther the observed aircraft is from its best match in the rendered database, the greater the average and median edge

separation. However, no such correlation is readily apparent with respect to range from the camera. An analogous plot to Figure 41 is shown in Figure 42 and was generated by comparing the random imagery to a "square" database of the sort used by Calhoun in [2] and [5]. Note that when using a "square" type database there is a strong correlation between range from the cameras and average edge separation. This would imply that different levels of blurring would be needed based upon aircraft range from the camera pair. Hence, the author deemed it preferable to use a "spherical" type reference image database as described in Section 3.4.

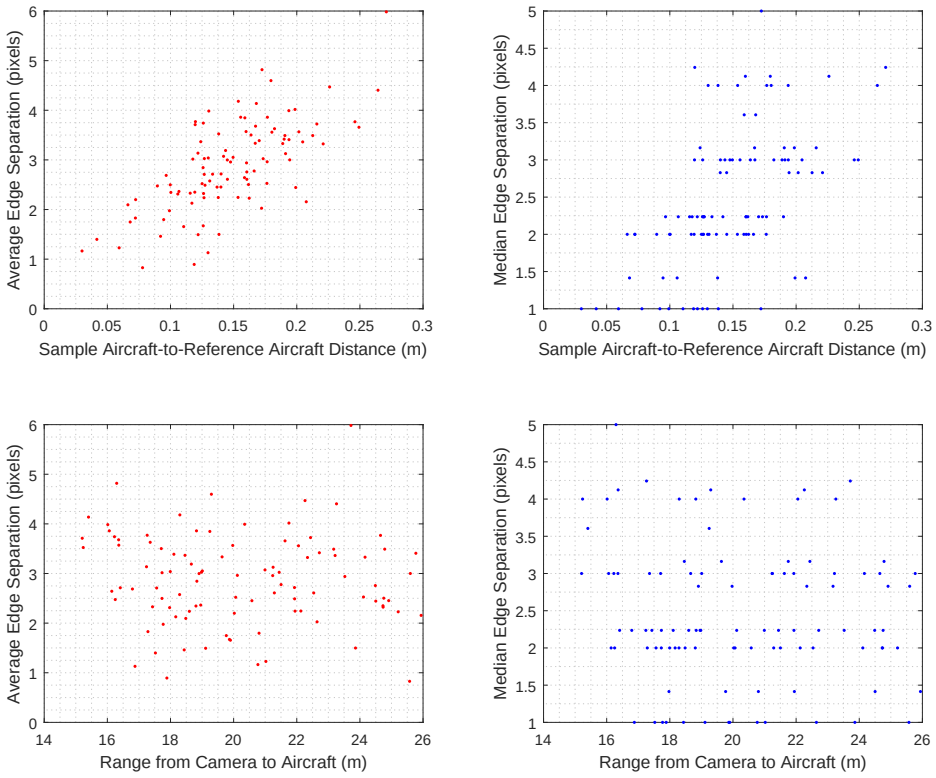


Figure 42. Plots Depicting Mean or Median Edge Separation Distance of all Edges in all Observation Images to the Nearest Edge Pixel in the Best Match from the Rendered Database. Data was generated when Using a "Square" Reference Image Database.

4.1.5 Likelihood Function Determination.

The likelihood function identified in this section was used in both simulation and flight test data analysis. To identify candidate likelihood functions, a total of 150 observation-sample image pairs were generated in the 3DVW. Next, $n = 500$ pixels from each reference image and from each observation image were randomly selected. Intensity differences were computed for the entire ensemble of all randomly selected pixels from all sample image pairs. Intensity differences were computed according to Equation (80). These pixel intensity differences were used to create a pixel intensity difference distribution.

Next, the MATLAB® function `fitdist` was used to identify the best fit for this pixel intensity difference data for a Gaussian distribution. This best fit Gaussian distribution was determined to have a mean of zero and a standard deviation of 0.057. Additionally, several parameterizations of the Cauchy and Laplace distributions were examined. Figure 43 shows a plot of example fits superimposed on a histogram depicting the observed intensity differences for the ensemble of all samples. The histogram has been normalized to model a PDF.

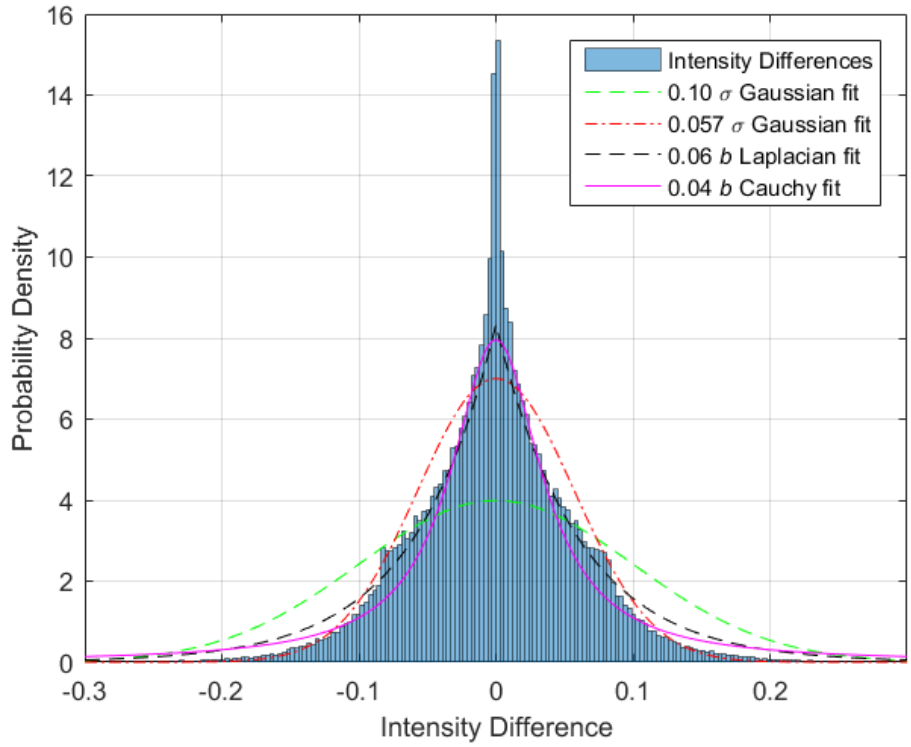


Figure 43. Candidate Fits for a Likelihood Function. The Histogram Depicts the Intensity Differences for the Entire Ensemble of Observation-Reference Image Pairs.

However, as observed by Calhoun in [2], using the best fit Gaussian distribution as a likelihood function produced an unacceptable number of integrity violations when applied in the V5 algorithm. This same phenomenon was observed when using parameterizations of the Cauchy distribution and Laplace distribution as a likelihood function, even though these fits tightly contoured the intensity difference histogram.

The simulation results presented in Figures 44 through 47 and Table 7 show this phenomenon. The following likelihood function parameterizations were used to generate the data in the figures and the table:

- The overbounded Gaussian distribution had a mean of 0 and a standard deviation of 0.1. Results are shown in Figure 44.

- The best fit Gaussian distribution had a mean of 0 and a standard deviation of 0.057. Results are shown in Figure 45.
- The overbounded Cauchy distribution had a location parameter of 0 and a scale parameter of 0.06. Results are shown in Figure 46.
- The overbounded Cauchy distribution had a location parameter of 0 and a scale parameter of 0.06. Results are shown in Figure 47.

These simulations were run on a set of 250 observation images. The same configuration of the R7 algorithm was used in all four cases. This configuration is shown in Table 6. In the table, the mean of the Gaussian prior probability distribution, $\boldsymbol{\mu}_i$, is defined to be the true simulated position of the aircraft, $\mathbf{x}_i$, and the covariance matrix is $\boldsymbol{\Sigma}$. Additionally, the results shown made use of the composite position estimator. As will be shown in Section 4.1.7, these simulation conditions result in superior R7 performance and hence gave the best chance of acceptable performance when making the use of a given likelihood function form.

Table 6. Configuration of the V5 Algorithm Used to Compare Likelihood Functions.

Prior Probability Distribution	**Pixel Intensity Threshold**	**Relative Position Estimator**	**Integrity Risk Level**
Gaussian $\boldsymbol{\mu}_i = \mathbf{x}_i$ $\boldsymbol{\Sigma} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ m^2	0.05	Composite	0.05

Despite these ideal conditions, each of these likelihood functions except for the overbounded Gaussian resulted in a significant number of integrity violations. On average, only 12.5 integrity violations should be observed on 250 sample images when using an integrity risk level of 0.05. Additionally, as can be seen in the table, all three of these likelihood forms resulted in greater mean spherical error and greater spherical error variance than when using the overbounded Gaussian likelihood function. Results from paired sample t-tests showed that using each "tight" form of the likelihood function resulted in greater mean spherical error than use of the overbounded Gaussian likelihood function at a confidence level $\gg 99\%$. Additionally, results from two-sample F-tests showed that using each "tight" form of the likelihood function resulted in greater spherical error variance than use of the overbounded Gaussian likelihood function at a confidence level $\gg 99\%$.

Table 7. Candidate Likelihood Function Spherical Errors and Integrity Violations for a 0.05 Integrity Risk Level.

	Spherical Error Mean (m)	**Spherical Error Standard Deviation (m)**	**Integrity Violations**
Overbounded Gaussian	0.539	0.335	6
Best Fit Gaussian	0.863	0.537	40
Overbounded Cauchy	0.725	0.448	24
Overbounded Laplacian	0.770	0.471	31

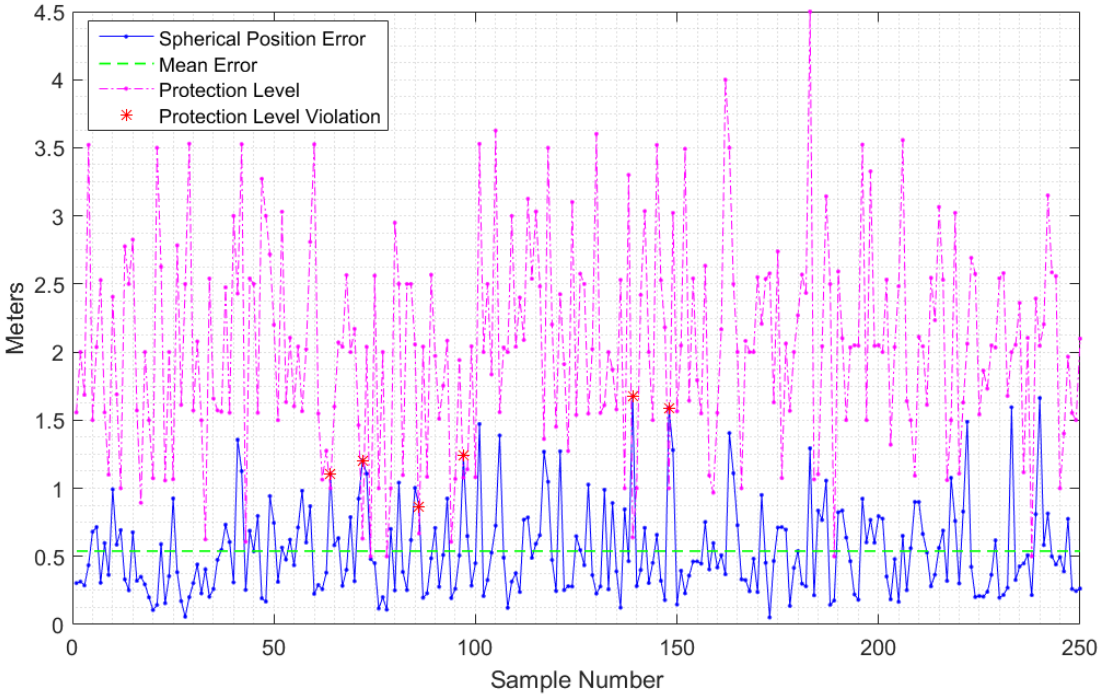


Figure 44. Simulation Results with Overbounded Gaussian Likelihood Function.

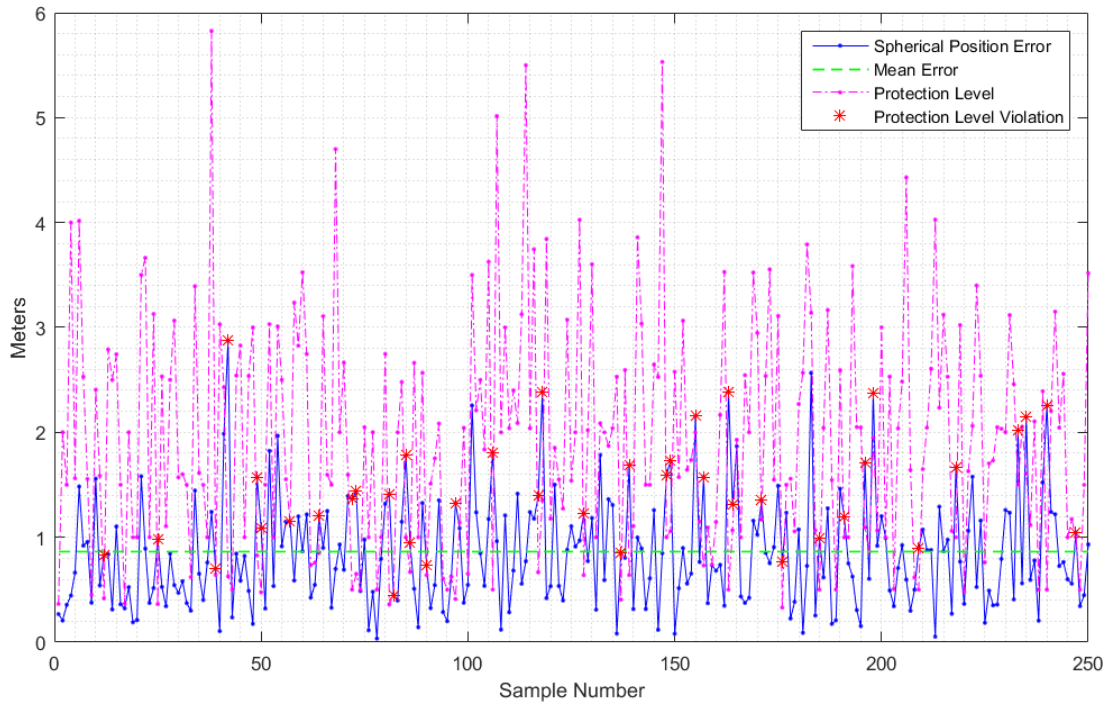


Figure 45. Simulation Results with Best Fit Gaussian Likelihood Function.

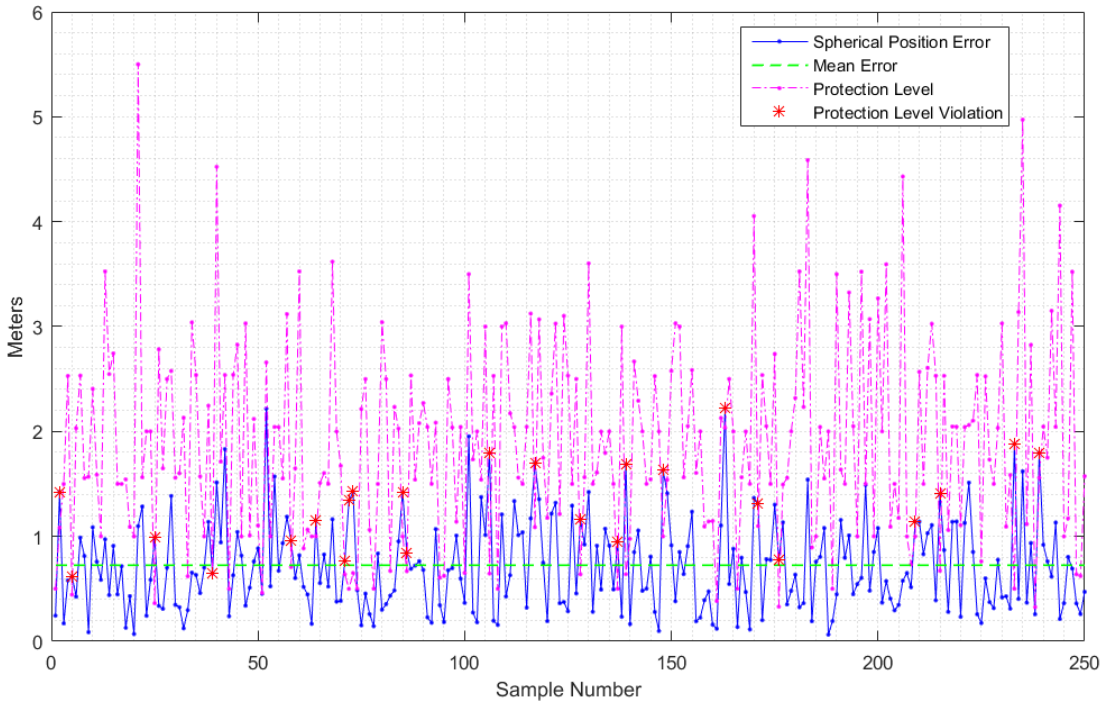


Figure 46. Simulation Results with Overbounded Cauchy Likelihood Function.

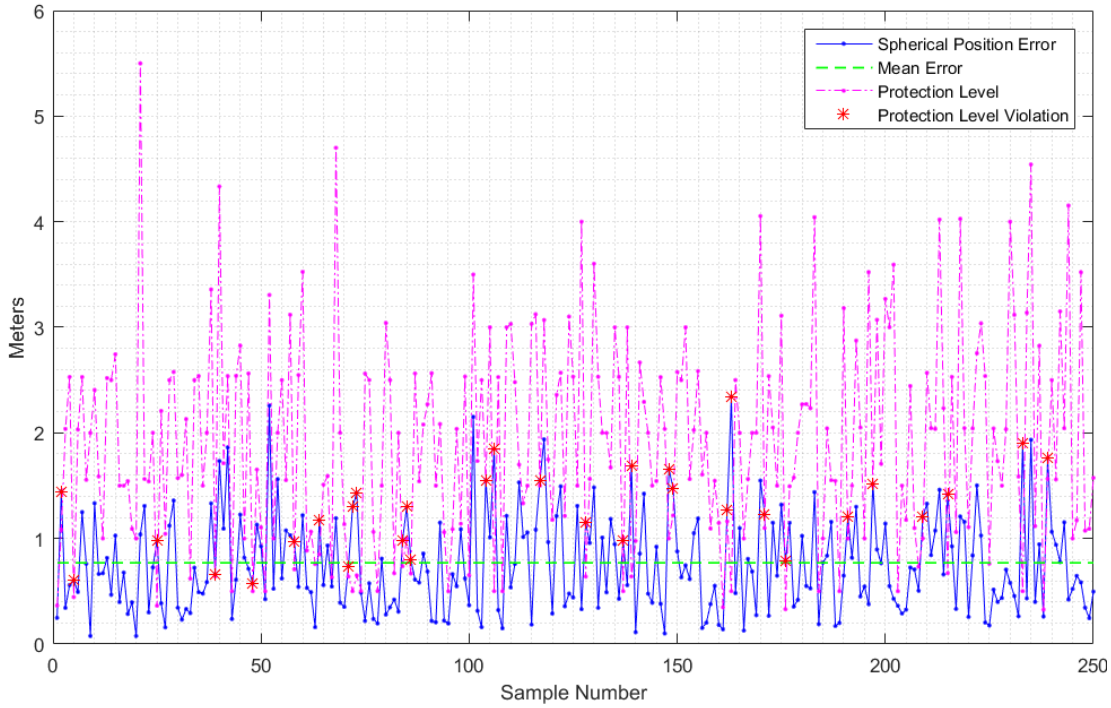


Figure 47. Simulation Results with Overbounded Laplacian Likelihood Function.

An examination of the pixel intensity difference distribution on the level of single reference-observation image pairs elucidates why tight distributional fits perform poorly as likelihood functions. The intensity difference distribution of a single reference-observation image pair does not necessarily match the intensity difference distribution of the ensemble of all reference-observation image pairs. Figure 48 depicts this phenomenon. In the figure, while Sample B closes matches the distribution of the ensemble, Sample A has much more probability density concentrated in areas further from zero. A tight fit on the ensemble distribution would fail to adequately model the distribution exhibited in Sample A. Hence, the diversity of sample distributions explains why an overbounded distribution is required for a likelihood function in the V5 algorithm.

Ultimately, an overbounded Gaussian distribution yielded acceptable performance with simulation and flight test data. This was the same result obtained by Calhoun in [2]. However, an extensive tuning process was not undertaken to identify an ideal likelihood function. An ideal likelihood function for all sets of environmental conditions and all types of formation dynamics likely does not exist. Were the V5 algorithm to be implemented in a real-world system, the chosen likelihood function or set of likelihood functions would need to be generalizable and not be over-tuned. As the data in Figure 48 shows, the high variability in sample pixel intensity difference distributions must be taken into account.

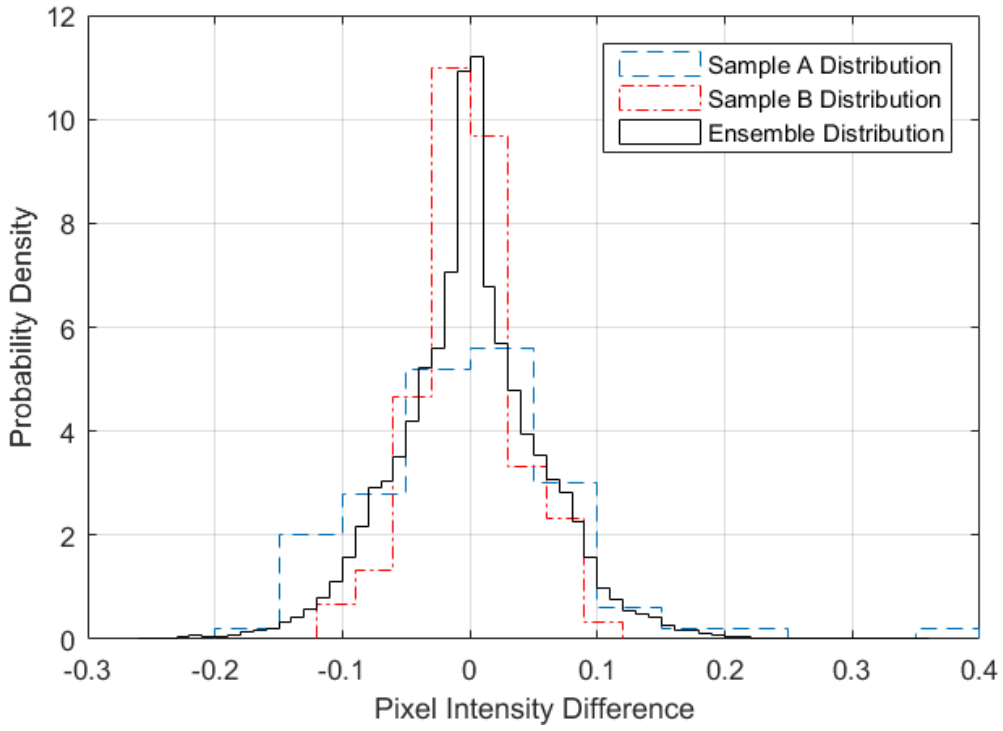


Figure 48. Pixel Intensity Difference Distributions for Two Sample Reference-Observation Image Pairs and an Ensemble of 150 Reference-Observation Image Pairs. A Total of 100 Pixels Were Randomly Sampled from Each Pair.

4.1.6 Pixel Intensity Threshold Determination.

As discussed in Section 3.4.5, the V5 algorithm can be applied with a user-specified pixel intensity threshold. When exercising this option, the V5 algorithm only compares pixels in the reference and observation images that exceed the threshold intensity value. As a result, only relatively strong edges are used for comparison. To determine a candidate threshold value, 150 processed observation images were examined in simulation. All pixels with an intensity value greater than zero were used to generate a cumulative distribution function (CDF) of pixel intensities with the MATLAB® function `histogram`. The results are depicted in Figure 49.

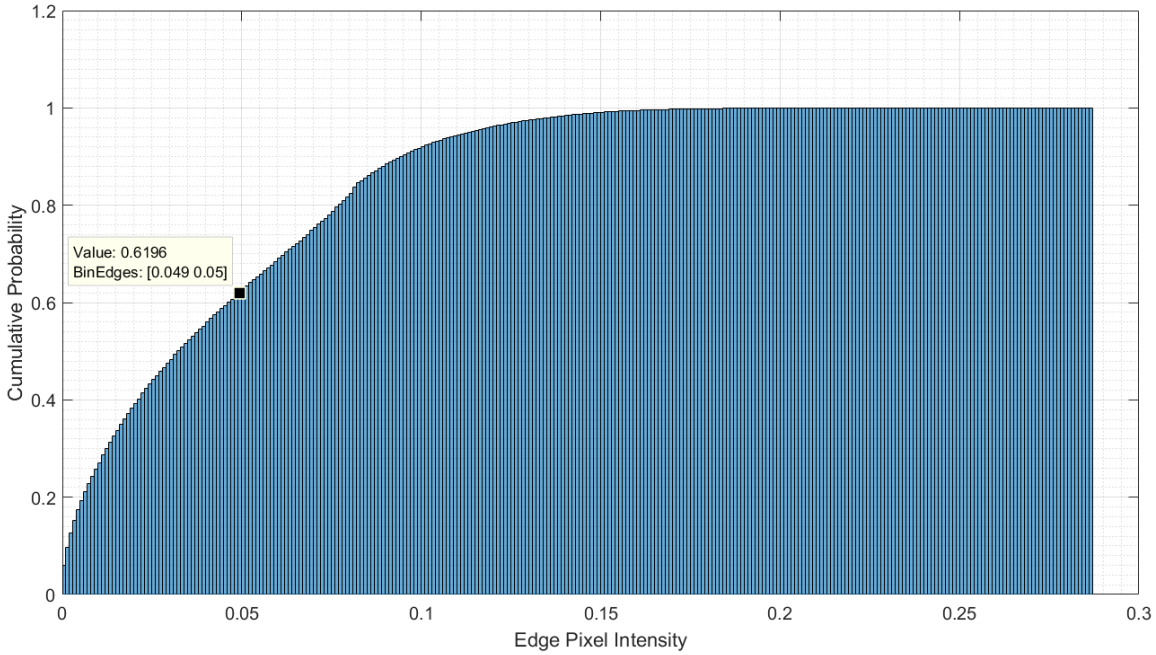


Figure 49. CDF of Pixel Intensities for 150 Processed Observation Images.

As shown in the figure, the strongest edge pixels had intensity values as great as 0.28 while the weakest edge pixels had intensity values only slightly greater than 0. A threshold intensity value of 0.05 was selected for analysis of flight test and simulation data since approximately 38% of pixels in this data had intensity values equal to or greater than this value. This determination was somewhat arbitrary, but ultimately proved to yield superior results to lower threshold values. Precise determination of an ideal threshold value would be tuning intensive and would likely yield a result particularly suited to the data set under examination.

4.1.7 V5 Simulation Results.

The likelihood function identified in Section 4.1.5 and the pixel intensity threshold of 0.05 that was identified in Section 4.1.6 were used to perform a series of simulations with the V5 algorithm. These simulations compared the estimate errors resulting from various algorithm configurations. Component position errors were computed in the

left camera frame for reasons outlined in Section 4.1.7.4. Simulation results revealed seven primary findings:

1. The composite position estimator performed better than the maximum likelihood estimator. This result is detailed in Section 4.1.7.1.

2. As expected, higher integrity risk levels resulted in smaller magnitude protection levels. This result is detailed in Section 4.1.7.2.

3. Using a non-zero pixel intensity threshold value led to lower errors and protection levels. This result is detailed in Section 4.1.7.3.

4. Using a non-zero pixel intensity threshold value resulted in PMF concentration along the camera system's line-of-sight. This result is detailed in Section 4.1.7.4.

5. Combining the posterior probability estimates from the left and right cameras led to better performance than when using a single camera. This result is detailed in Section 4.1.7.5.

6. Informed prior probability distributions can lead to better protection level estimation than uniform prior probability distributions. This result is detailed in Section 4.1.7.6.

7. As implemented, the V5 algorithm requires significant processing time to operate on a single observation-reference image pair. This result is detailed in Section 4.1.7.7.

Based on these results, the configuration specified in Table 8 was selected as the baseline configuration for analysis of flight test data in Chapter VI. A uniform prior was selected as a baseline because the informative priors used in simulation depended upon knowledge of truth data. Additionally, this configuration was used to compare V5 and R7 simulation performance in Section 4.3.

Table 8. Baseline Configuration of the V5 Algorithm Identified for Use on Flight Test Data.

Prior Probability Distribution	Likelihood Function	Pixel Intensity Threshold	Relative Position Estimator	Integrity Risk Level
Uniform	Gaussian $\mu = 0$ $\sigma = 0.1$	0.05	Composite	0.05

4.1.7.1 Relative Position Estimator Comparison.

The performance of the maximum likelihood estimator defined in Equation (88) was compared against the performance of the composite position estimator defined in Equation (89). As discussed in Section 3.4.7, the maximum likelihood estimator identifies the mode of the PMF returned from the V5 algorithm while the composite position estimator identifies the mean of the PMF. This comparison was made via use of uniform and Gaussian prior probability distributions, various likelihood functions, and threshold intensity values of 0 and 0.05. A total of 250 observation images were used in each case. For all V5 algorithm configurations examined, the composite position estimator was on average more accurate than the maximum likelihood estimator.

Results using the V5 configuration shown in Table 9 are presented in this section. These results were typical of those found with other V5 configurations. The uniform prior probability distribution assigned an equal probability mass to all relative positions contained within the attenuated reference image database described in Section 4.1.2. A non-uniform prior probability distribution would have skewed the results returned from both estimators toward the prior probability distribution. Hence, a

uniform prior probability distribution was considered to be ideal for comparing relative position estimator performance.

Table 9. Configuration of the V5 Algorithm Used for Position Estimator Comparison.

Prior Probability Distribution	**Likelihood Function**	**Pixel Intensity Threshold**	**Integrity Risk Level**
Uniform	Gaussian $\mu = 0$ $\sigma = 0.1$	0	0.05

Figure 50 plots the spherical error of the maximum likelihood estimator relative position solution for the simulation. Figure 51 depicts the same information with respect to the composite position estimator. As can be seen in the figures, the composite position estimator was much more accurate overall than the maximum likelihood estimator. When using the maximum likelihood estimator, a total of four protection level violations were observed while none were observed when using the composite position estimator.

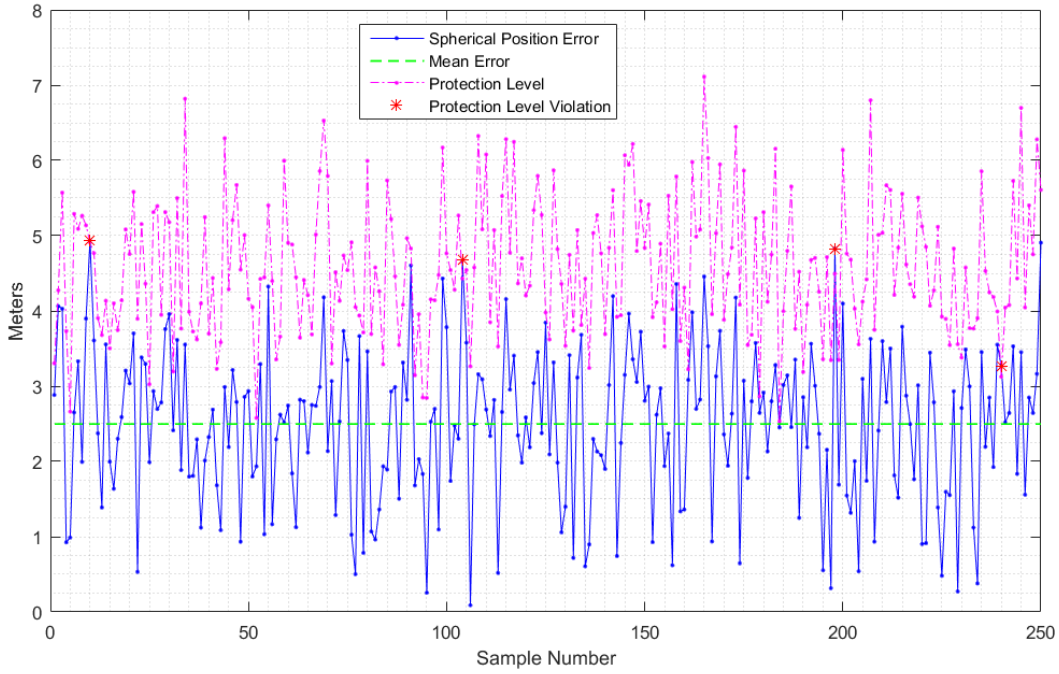


Figure 50. Spherical Error for the V5 Algorithm Using the Maximum Likelihood Estimator.

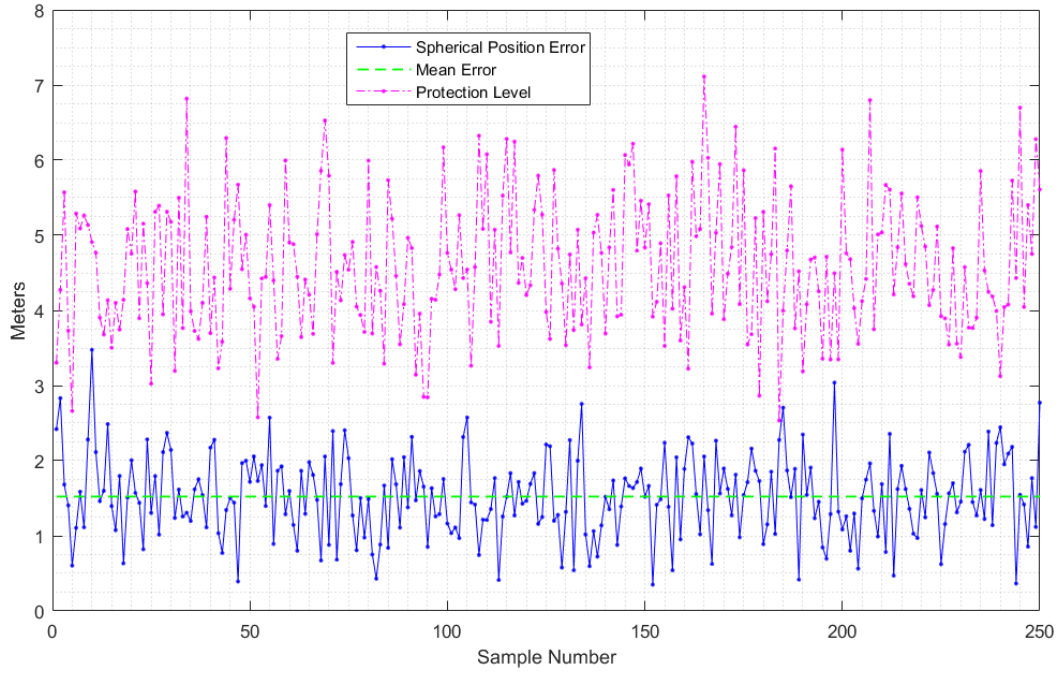


Figure 51. Spherical Error for the V5 Algorithm Using the Composite Position Estimator.

As shown in Table 10, the mean spherical error of the composite position estimator was 39% lower than that of the maximum likelihood estimator. A paired t-test revealed that the mean spherical errors had a statistically significant difference at a confidence level $\gg 99.9\%$. The 95% confidence interval for this difference was [0.871 m, 1.078 m] greater error in the maximum likelihood estimator. Additionally, the standard deviation of the spherical error of the composite position estimator was 47% lower. A two-sample F-test revealed that the variance of the maximum likelihood estimator mean spherical error was greater than that of the composite position estimator at a confidence level $\gg 99.9\%$. The 95% confidence interval for the true ratio of the variances (maximum likelihood estimator to composite position estimator) was $[2.771, 4.559]$.

Likewise, the mean error and the standard deviation of the error were lower in each axis of the camera frame for the composite position estimator. However, the differences in means were not found to be statistically significant using one-sample t-tests. Conversely, the difference in variances along each position error component were found to be statistically significant at confidence levels $\gg 99.9\%$ with two-sample F-tests. Again, these significantly lower errors manifested when using other prior probability distributions, likelihood functions, and threshold values.

Table 10. Maximum Likelihood Estimator and Composite Position Estimator Errors, Left Camera Frame.

	Maximum Likelihood Estimator Mean	Maximum Likelihood Estimator Standard Deviation	Composite Position Estimator Mean	Composite Position Estimator Standard Deviation
Spherical Error (m)	2.498	1.041	1.523	0.552
x Position Error (m)	0.071	1.733	0.048	1.136
y Position Error (m)	0.507	1.132	0.495	0.540
z Position Error (m)	-0.440	1.614	-0.3521	0.8242

An analysis of the PMF returned by the V5 algorithm reveals why the composite position estimator typically performs better. Figure 52 depicts the PMF returned from sample number seven of the observation images. As can be seen in the figure, the distribution is multi-modal. Since the maximum likelihood estimator identifies the mode of the PMF, the multi-modal properties of the PMF subject it to higher errors. As a result, the composite position estimator does a superior job of accurately estimating the receiver position than the maximum likelihood estimator. Additionally, there is significant distributional scatter in all axes of both the tanker frame and the camera frame. Figure 53 depicts this scatter from side and top-down views. The distributional spread directly influences the computed protection levels. The amount of scatter relates to the threshold value of zero run with this simulation. The effect of the threshold value on the PMF distribution will be analyzed further in Section 4.1.7.4.

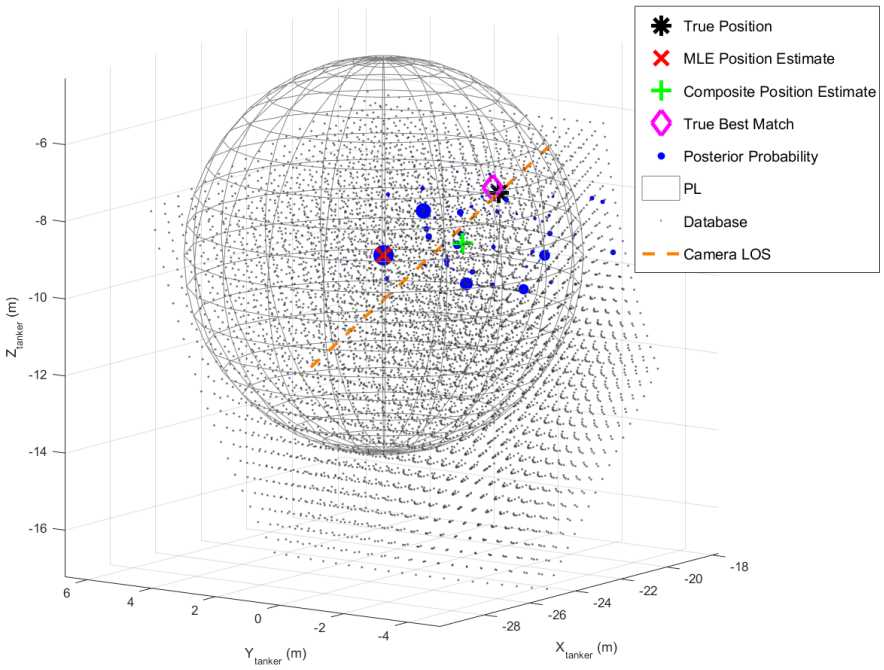


Figure 52. PMF Distribution Computed with a Pixel Intensity Threshold of Zero.

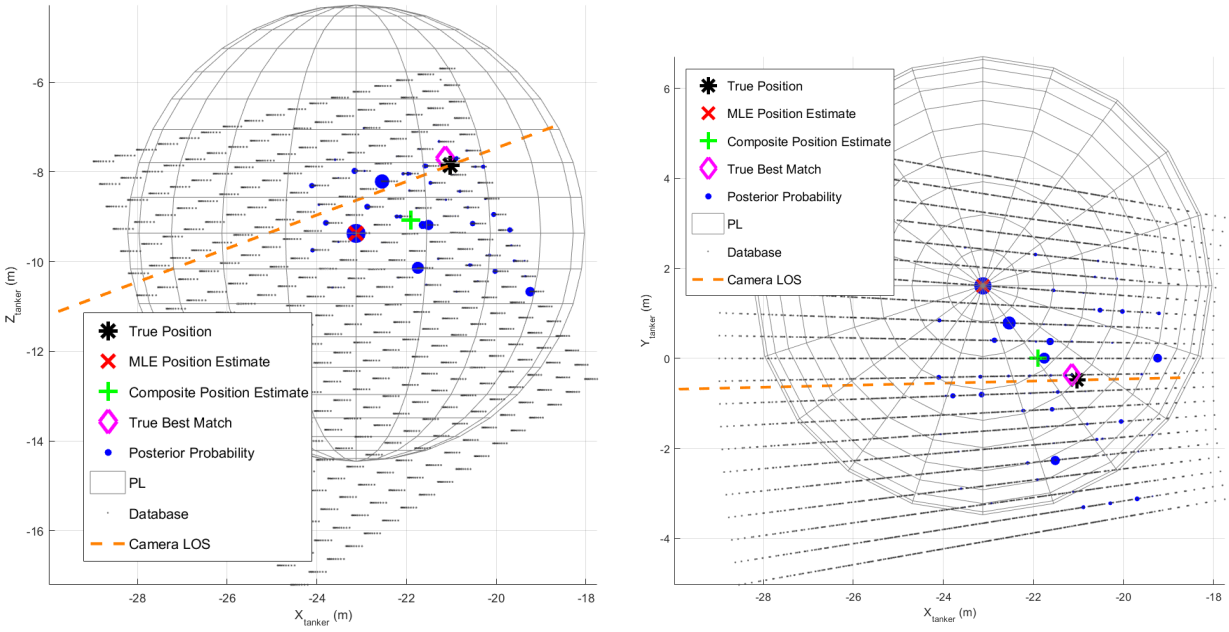


Figure 53. PMF Distribution Computed with a Pixel Intensity Threshold of Zero, Side and Top-Down Views.

4.1.7.2 Integrity Risk Level Effect.

As noted by Calhoun in [2], smaller integrity risk levels should lead to larger protection levels. During simulation this result was observed, confirming proper operation of the V5 algorithm's protection level computation. The results presented in this section used the V5 configurations given in Table 11. The presented results were typical of the results with other prior probability distributions, likelihood functions, and threshold intensity values.

Table 11. Configurations of the V5 Algorithm Used for Integrity Risk Level Analysis.

Prior Probability Distribution	Likelihood Function	Pixel Intensity Threshold	Relative Position Estimator	Integrity Risk Level
Uniform	Gaussian $\mu = 0$ $\sigma = 0.1$	0	Composite	0.05 and 10^{-6}

Figure 54 depicts the protection level returned with integrity risk levels of 0.05 and 10^{-6}. As can be seen in the figure, the protection level for the 10^{-6} integrity risk level mostly contours with that of the 0.05 integrity risk level, but is substantially larger. For an integrity risk level of 0.05, the mean protection level was 4.56 meters with a standard deviation of 0.91 meters. For an integrity risk level of 10^{-6}, the mean protection level was 6.18 meters with a standard deviation of 0.99 meters. A two-sample F-test was applied to the two protection level cases with the MATLAB® function `vartest2`. These results showed that the variance of these two protection levels did not have a statistically significant difference. This indicates that protection level fluctuations of similar magnitude would be expected when using integrity risk levels of 0.05 or greater.

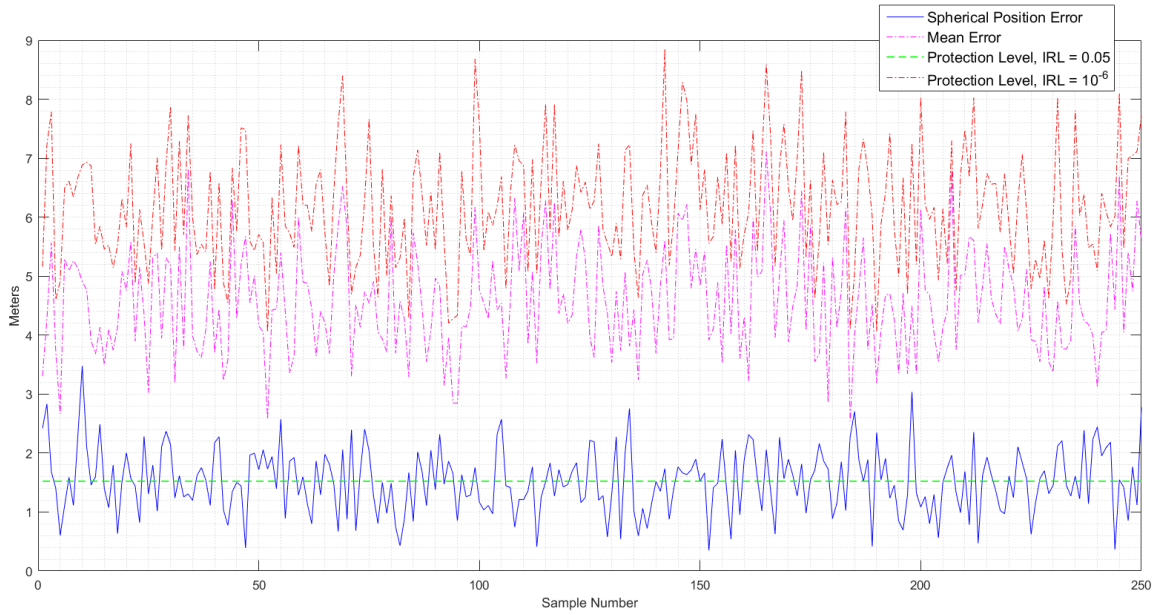


Figure 54. Protection Level Comparison with Different Integrity Risk Levels.

4.1.7.3 Thresholding Comparison.

The results presented in this section used the V5 configurations given in Table 12. These two test cases utilized the same set of randomly generated observation images, but utilized different edge pixel samples drawn from the reference and observation images. Despite the fact that two different pixel sets were used, the results strongly suggested that utilizing a non-zero threshold value yields more accurate V5 results. It is important to understand that a pixel intensity threshold of zero does not imply that pixels with an intensity of zero will be sampled. With a zero threshold setting, a pixel must be just greater than zero to be sampled.

Table 12. Configurations of the V5 Algorithm Used for Pixel Intensity Threshold Analysis.

Prior Probability Distribution	Likelihood Function	Pixel Intensity Threshold	Relative Position Estimator	Integrity Risk Level
Uniform	Gaussian $\mu = 0$ $\sigma = 0.1$	0 and 0.05	Composite	0.05

Figure 55 depicts the spherical error and protection level with a pixel intensity threshold of 0. Figure 56 depicts the spherical error and protection level with a pixel intensity threshold of 0.05. As can be seen in the two figures, typical spherical errors and protection levels were much lower in the 0.05 pixel intensity threshold case. Additionally, the 0.05 pixel intensity threshold case yielded 11 instances of protection level violations, while the 0 pixel intensity threshold case yielded no protection level violations. At an integrity risk level of 0.05, an average of 12.5 integrity risk violations would be expected from 250 random samples. Observing a number significantly less than 12.5 indicates that the PMF assigned to great a probability mass to positions far from the estimated position. Hence, the higher number of protection violations in the 0.05 threshold case indicates superior performance.

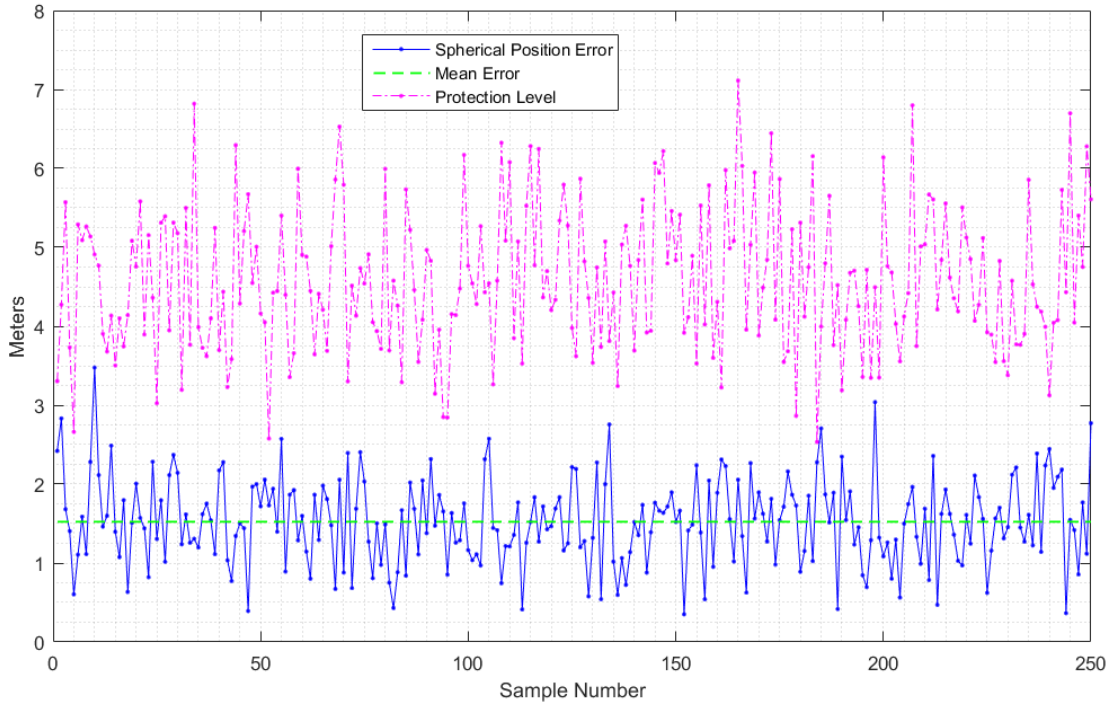


Figure 55. Spherical Error When Using a Pixel Intensity Threshold of 0.

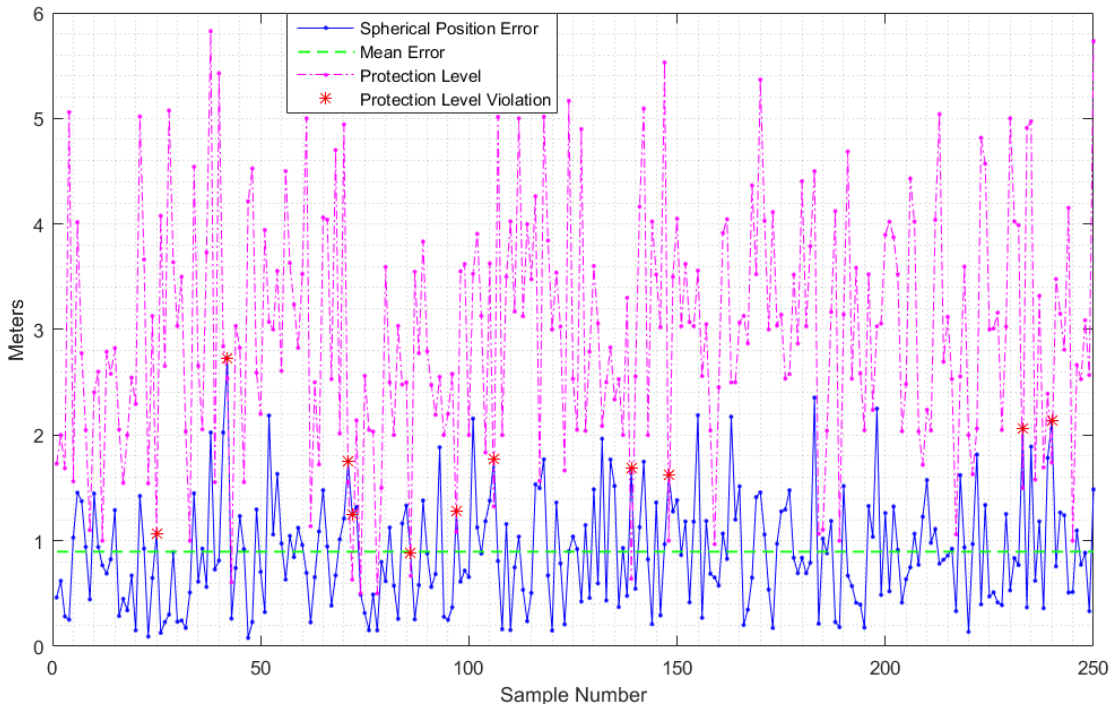


Figure 56. Spherical Error When Using Using a Pixel Intensity Threshold of 0.05.

Table 13 compares the accuracy of the relative position estimate and the magnitude of the protection levels returned in the two different threshold cases. As can be seen in the table, mean position component errors were much closer to zero and the mean spherical error was 42% less in the 0.05 threshold case. A two-sample t-test, without an assumption of equal variances, was used to test if the differences observed in the mean errors were statistically significant. A two-sample t-test was used because each threshold case sampled different sets of edge pixels. The difference in mean spherical error was found to be statistically significant at a confidence level $\gg 99\%$. The 95% confidence interval for this difference was $[0.532\text{m}, 0.722\text{m}]$ greater error in the 0 pixel intensity threshold case.

With respect to each position error component, only the mean error of the y-component was found to have a statistically significant difference. This conclusion was reached at a confidence level $\gg 99\%$. The 95% confidence interval for the y-component mean error difference was $[0.416\text{m}, 0.557\text{m}]$ greater error in the 0 pixel intensity threshold case. The z-component mean error difference was found to have a p-value of 0.0501, only just failing the chosen threshold for statistical significance.

Additionally, the standard deviations of the errors were generally lower in the 0.05 threshold case. The sole exception was along the z-axis in the left camera frame. Results from a two-sample F-test revealed that the difference in spherical error variances was not statistically significant. However, all position error component variances had statistically significant differences.

In the x and y-components, the differences in these variances were significant at a confidence level $\gg 99\%$. For the x and y-dimensions, the 95% confidence intervals for the ratio of the 0 threshold error variance to the 0.05 threshold error variance were [7.187, 11.826] and [7.499, 12.339], respectively. In these components, the 0 intensity threshold case was found to have significantly greater variance. In the z-component,

the 0.05 intensity threshold case was found to have modestly greater error variance at the 95% confidence level. In the z-dimension, the 95% confidence interval for the ratio of the 0 threshold error variance to the 0.05 threshold error variance was [0.606, 0.997]. Hence using a pixel intensity threshold of 0.05 was found to significantly reduce error variance in the left camera frame x and y-dimensions while only modestly increasing error variance in the z-dimension.

Moreover, the average protection level was 35% smaller in the 0.05 threshold case, but had a larger standard deviation. Using a two-sample t-test, the difference in means was statistically significant at a confidence level $\gg 99\%$. The 95% confidence level for this difference was [0.907m, 1.801m] greater mean protection levels in the 0 threshold case. Additionally, using a two-sample F-test the difference in protection level variances was statistically significant at the 99% confidence level, with more variance in the 0.05 intensity threshold case. The 95% confidence interval for the ratio of the 0 threshold protection level variance to the 0.05 threshold protection level variance was [0.496, 0.815]. Overall, as discussed above, given the number of integrity risk level violations observed in the two cases, the PMF and resultant protection level estimated in the 0.05 threshold case was likely superior to that estimated in the 0 threshold case.

Overall, using a pixel intensity threshold value of 0.05 significantly improved estimation of the receiver relative position, the PMF describing that relative position, and the protection level returned from the PMF. As a result, this threshold value was selected for use on analysis of flight test data.

Table 13. Threshold Comparison, Composite Position Estimator Errors, Protection Levels, Left Camera Frame.

	Mean, Threshold = 0	Standard Deviation, Threshold = 0	Mean, Threshold = 0.05	Standard Deviation, Threshold = 0.05
Spherical Error (m)	1.523	0.552	0.896	0.527
x Position Error (m)	0.048	1.136	0.004	0.374
y Position Error (m)	0.495	0.540	0.008	0.174
z Position Error (m)	-0.3521	0.8242	-0.197	0.935
Protection Level (m)	4.575	0.907	2.956	1.138

4.1.7.4 Consequences of Thresholding.

As noted in Section 4.1.7.3, using a threshold of 0.05 increased the standard deviation z-position estimate error in the left camera frame. Additionally, as can be seen in Table 13, the standard deviation of the error in the z-component was greater than the standard deviations along the x and y-components.

Figure 57 depicts the PMF distribution resulting from sample number two in the 0.05 intensity threshold case. Figure 58 depicts the same sample distribution from side and top-down views. Qualitatively, the PMF form from this sample was typical of the other samples generated in the 0.05 intensity threshold case. As can be seen in the figures, the probability mass of the PMF was concentrated along the camera line-of-sight. The camera line-of-sight in this case is defined as a ray drawn from the origin of the left camera to the true position of the receiver. Based on the position

of the boom envelope and hence the reference image database, this line-of-sight most closely corresponds to the z-axis of the camera frame. Hence, PMF mass is also mostly concentrated along the z-axis of the left camera frame in the figure.

Contrastingly, the PMF for the 0 intensity threshold case depicted in Figures 52 and 53 manifested no such concentration along the camera line-of-sight. The fact that probability mass tends to concentrate along the camera line-of-sight in the non-zero threshold case explains why the bulk of PMF uncertainty occurs along the z-axis of the camera frame.

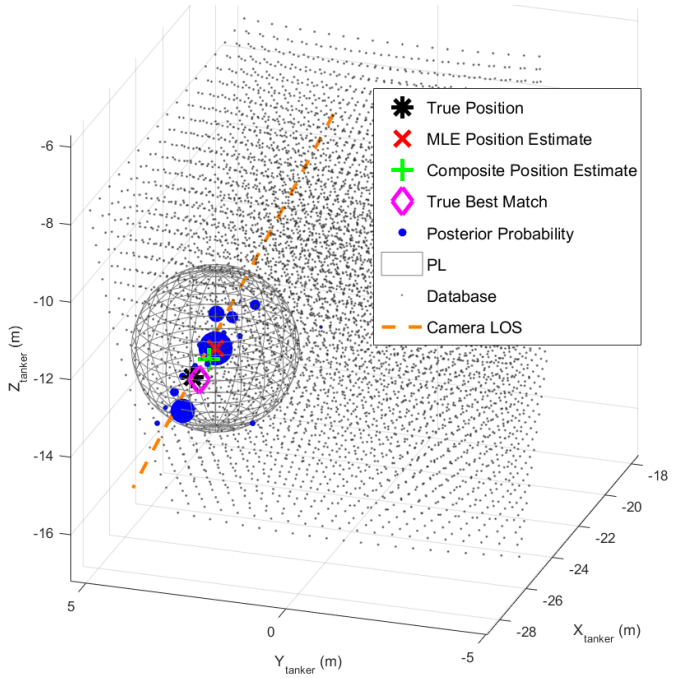


Figure 57. PMF Distribution Computed with a Pixel Intensity Threshold of 0.05.

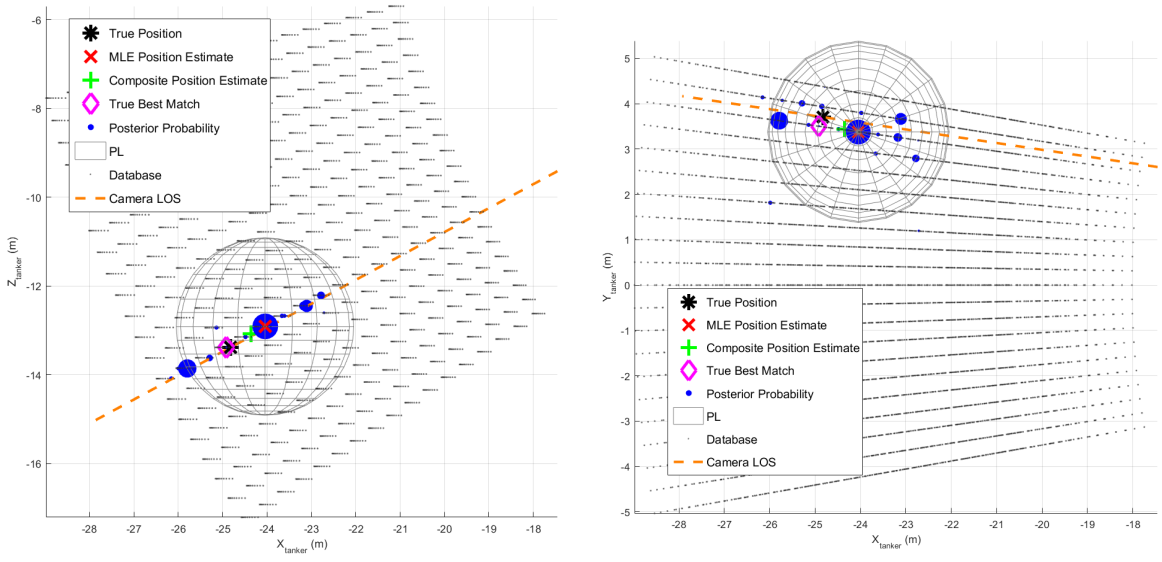


Figure 58. PMF Distribution Computed with a Pixel Intensity Threshold of 0.05, Side and Top-Down Views.

From the perspective of the tanker frame (the p-frame), the camera line-of-sight falls most strongly along the x-axis. Figure 59 depicts the 0.05 intensity threshold simulation errors in the tanker frame. As can be seen in the figure, the bulk of the error is concentrated in the x-axis. Indeed, in eight cases protection level violations could be solely attributed to error in the x-component of the tanker frame relative position estimate.

Figure 60 depicts the same errors in the left camera frame. As can be seen in the figure, the bulk of the error is concentrated in the z-axis. In ten cases protection level violations could be solely attributed to error in the z-component of the left camera frame relative position estimate. Based on the geometry of the rendered databases used in simulation and flight test, the camera line-of-sight projected more strongly onto the camera frame z-axis than onto the tanker frame x-axis. As a result, V5 relative position errors are plotted with respect to the left camera frame for the remainder of this thesis.

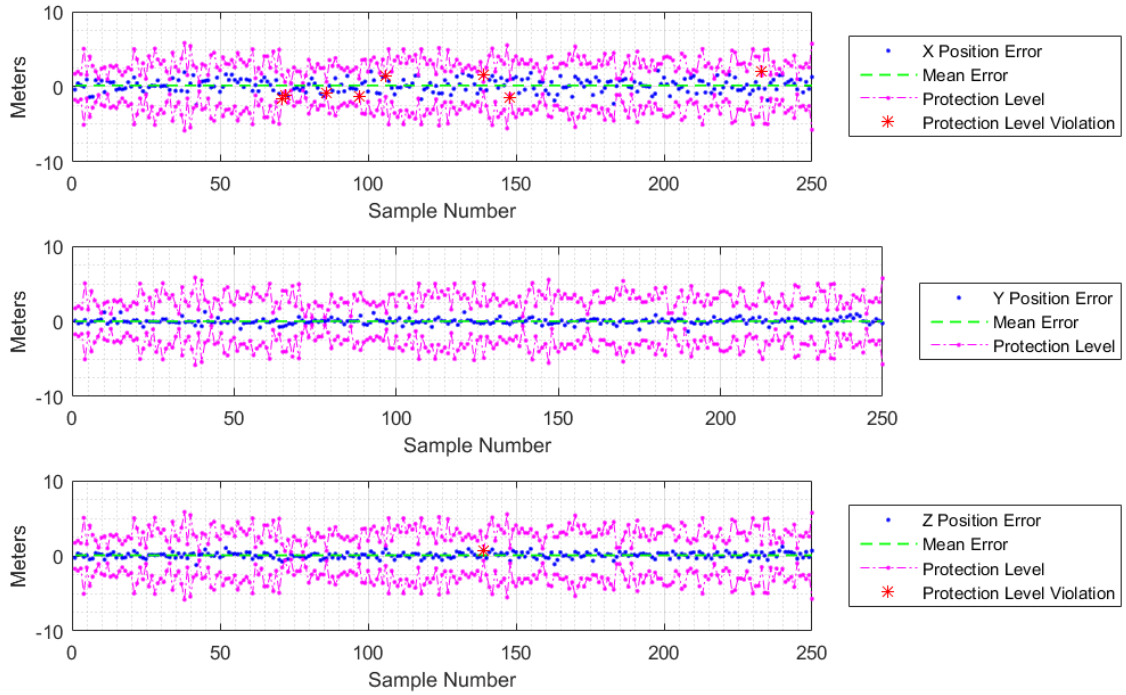


Figure 59. Composite Position Estimator Errors, Tanker Frame.

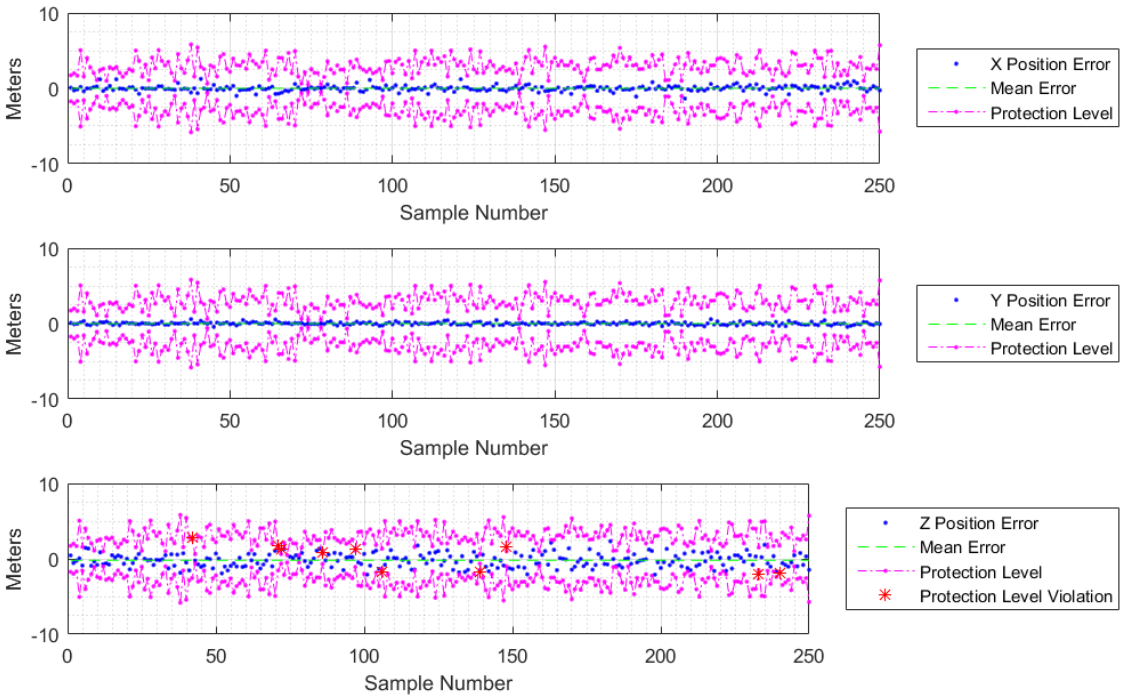


Figure 60. Composite Position Estimator Errors, Left Camera Frame.

Figure 61 reveals one additional property associated with non-zero thresholds. For receiver positions closer to the stereo camera system, the algorithm tended to overestimate the z-position in the left camera frame. In other words, the algorithm tended to estimate a position farther away than the actual receiver position. Contrastingly, for receiver positions farther from the stereo camera system, the algorithm tended to underestimate the z-position in the left camera frame. In other words, the algorithm tended to estimate a position closer than the actual receiver position.

This property manifests because the reference image database is finite and is bounded at specific left camera frame z-ranges. Since the PMF is concentrated along the camera line-of-sight, more probability mass will be concentrated aft of the actual receiver position when the receiver is close to the front of the reference database. The converse is true for receiver positions close to the back of the reference database. This phenomenon could be overcome with use of a larger reference image database.

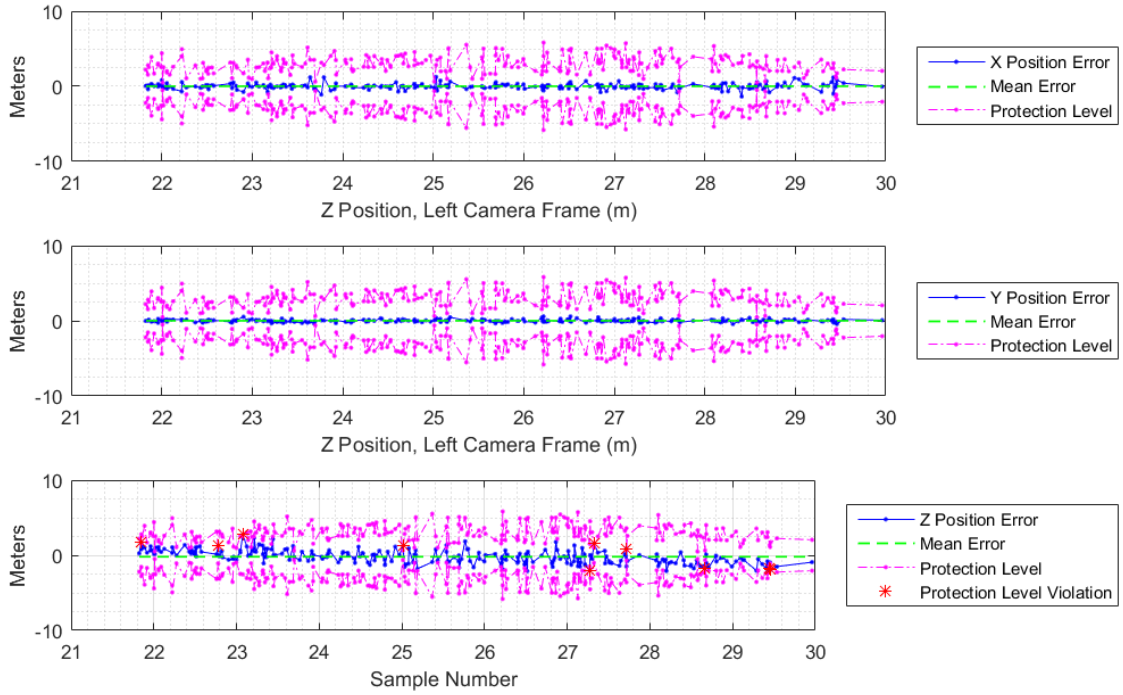


Figure 61. Composite Position Estimator Errors Sorted by z-Position, Left Camera Frame.

4.1.7.5 Single Camera Versus Stereo Camera Results.

In the V5 algorithm, as described in Section 3.4.6, the left and right camera images are used to generate two separate PMFs. These are referred to as the left camera PMF and the right camera PMF. These PMFs are then combined by giving equal weighting to each as shown in Equation (87). This resulting PMF is called the combined PMF. The results in this section show that use of the combined PMF results in better algorithm performance than using a single camera PMF. The V5 algorithm configuration specified in Table 14 was used for this comparison. A total of 250 simulation observation images were used.

Table 14. Configuration of the V5 Algorithm Used to Compare Single Camera PMF and Combined Camera PMF Performance.

Prior Probability Distribution	Likelihood Function	Pixel Intensity Threshold	Relative Position Estimator	Integrity Risk Level
Uniform	Gaussian $\mu = 0$ $\sigma = 0.1$	0.05	Composite	0.05

Table 15 shows the results from the left camera PMF, the right camera PMF, and the combined PMF for this scenario. Figures 62 through 64 plot the spherical errors, protection levels, and integrity risk violations from each PMF. The superior performance observed in the combined PMF was typical of other configurations.

As the data show, the combined PMF resulted in lower errors and error variances in almost all cases. A paired t-test showed that the spherical error reduction achieved in the combined PMF case was statistically significant at a confidence level $\gg 99\%$. Additionally, a two-sample F-test showed that the reduction in spherical

error variance achieved in the combined PMF case was statistically significant at the 98% confidence level.

The combined PMF resulted in a statistically significant increase in the mean protection level. However, most critically, both the left and right camera PMFs resulted in unacceptable numbers of integrity risk violations. For a set of 250 samples and an integrity risk level of 0.05, an average of 12.5 integrity risk violations would be expected. Both the left and camera PMFs grossly exceeded this number. Hence, the results show that the combined PMF did a superior job of estimating an appropriate protection level.

Overall, these results show that the combined PMF produces superior results in terms of both estimate errors and protection levels to the single camera PMFs. Consequently, the combined PMF was used for analysis of flight test data.

Table 15. Comparison of V5 Algorithm Errors When Using a the Left Camera PMF, the Right Camera PMF, and the Combined PMF.

	Left Camera		Right Camera		Combined	
	Mean	**Standard Deviation**	**Mean**	**Standard Deviation**	**Mean**	**Standard Deviation**
Spherical Error (m)	1.041	0.661	0.992	0.619	0.896	0.527
X Position Error (m)	0.028	0.479	-0.020	0.469	0.004	0.374
Y Position Error (m)	0.023	0.194	-0.007	0.193	0.008	0.174
Z Position Error (m)	-0.264	1.089	-0.131	1.047	-0.197	0.935
Protection Level (m)	2.496	1.282	2.689	1.255	2.956	1.138
Integrity Risk Violations	35		25		11	

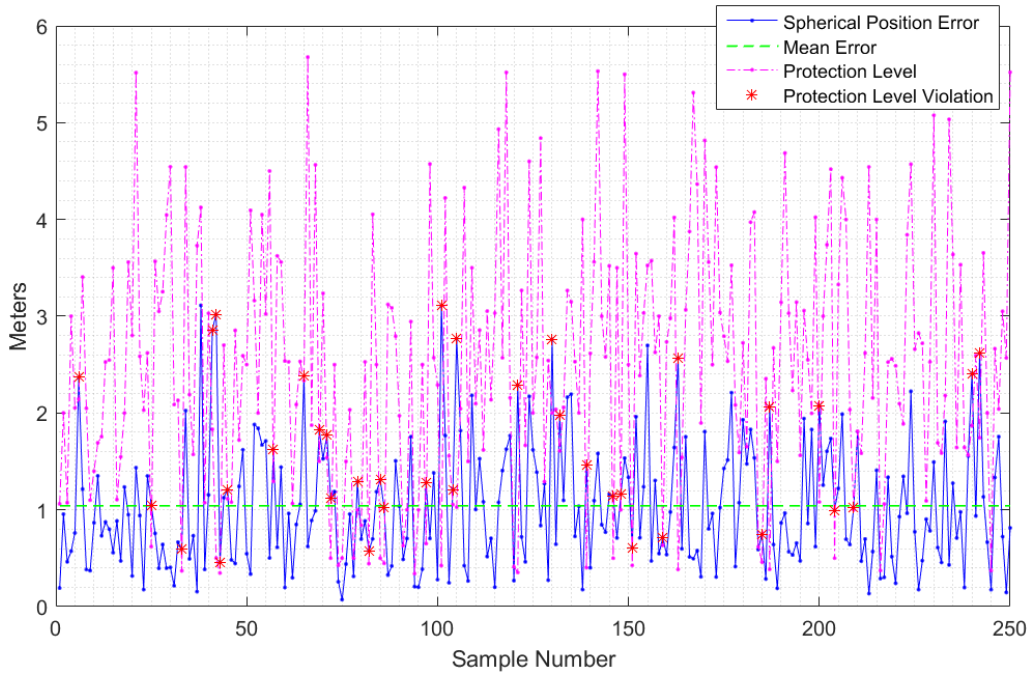


Figure 62. Spherical Error When Using Only the Left Camera PMF.

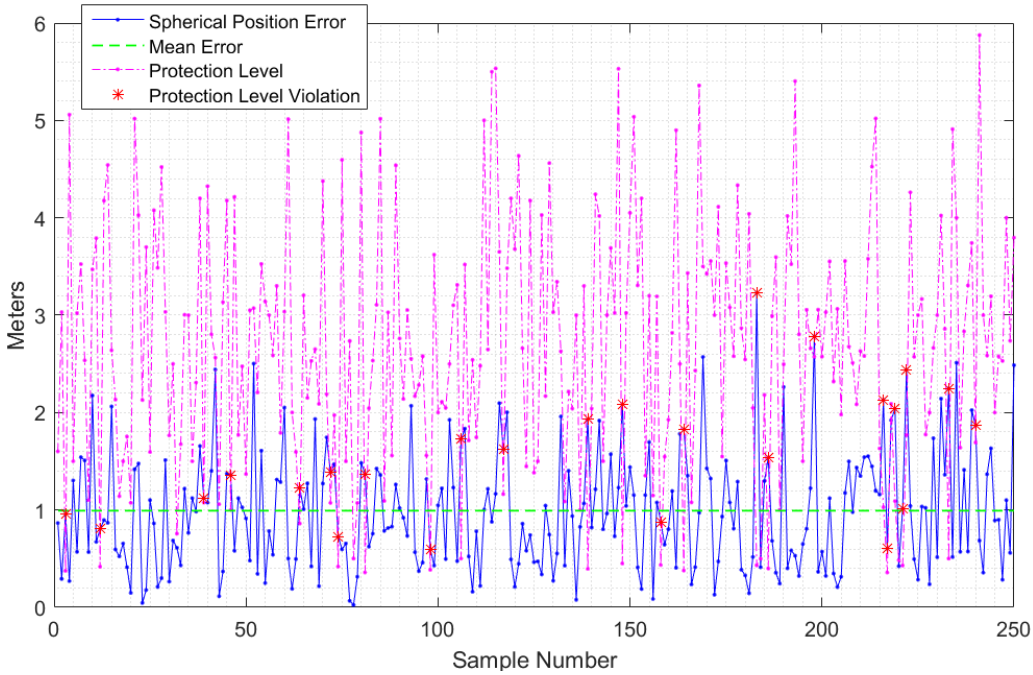


Figure 63. Spherical Error When Using Only the Right Camera PMF.

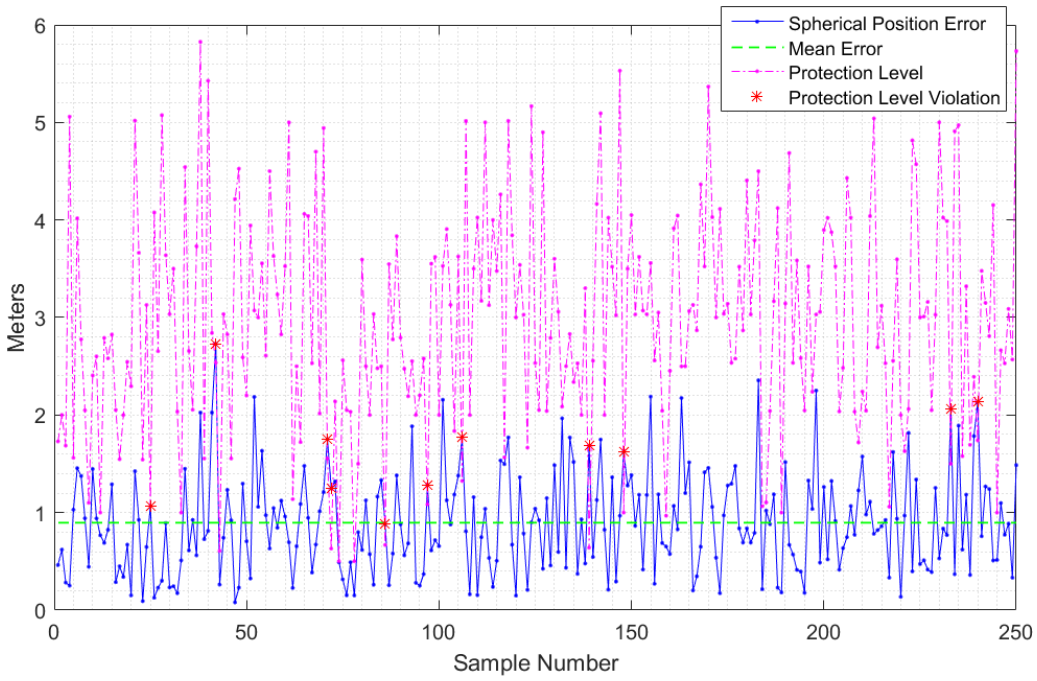


Figure 64. Spherical Error When Combining Camera PMFs.

4.1.7.6 Prior Probability Distribution Comparison.

The V5 algorithm configuration specified in Table 16 was used to examine the effect of different prior probability distributions on performance.

Table 16. Configurations of the V5 Algorithm Used for Prior Probability Distribution Analysis.

Prior Probability Distribution	Likelihood Function	Pixel Intensity Threshold	Relative Position Estimator	Integrity Risk Level
Various	Gaussian $\mu = 0$ $\sigma = 0.1$	0.05	Composite	0.05

Three prior probability distributions were examined. The first was a non-informative uniform prior. The uniform prior probability distribution assigned an equal proba-

bility mass to all relative positions contained within the attenuated reference image database as described in Section 4.1.2.

The second was an informative prior called Gaussian distribution A. This distribution was a three-dimensional Gaussian distribution with a mean centered precisely on the true relative position of the receiver. The covariance matrix of this distribution was set to:

$$\mathbf{C} = \begin{bmatrix} 4 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 4 \end{bmatrix} \text{m}^2 \tag{109}$$

The third was an informative prior called Gaussian distribution B. This distribution was a three-dimensional Gaussian distribution with a covariance matrix of $\mathbf{C}$ and a mean centered on the true relative position of the receiver plus an amount of random error. The random error was drawn from a three-dimensional Gaussian distribution with a mean of $[0,\ 0,\ 0]^T$ and a covariance matrix equal to $\mathbf{C}$.

Figure 65 depicts the spherical errors and protection levels when using a uniform prior probability distribution, Figure 66 depicts the spherical errors and protection levels when using a Gaussian distribution A as the prior, and Figure 67 depicts the spherical errors and protection levels when using a Gaussian distribution B as the prior. Table 17 shows the mean relative position estimate errors and standard deviations as well as the mean protection levels and standard deviations returned when using each of the prior probability distributions.

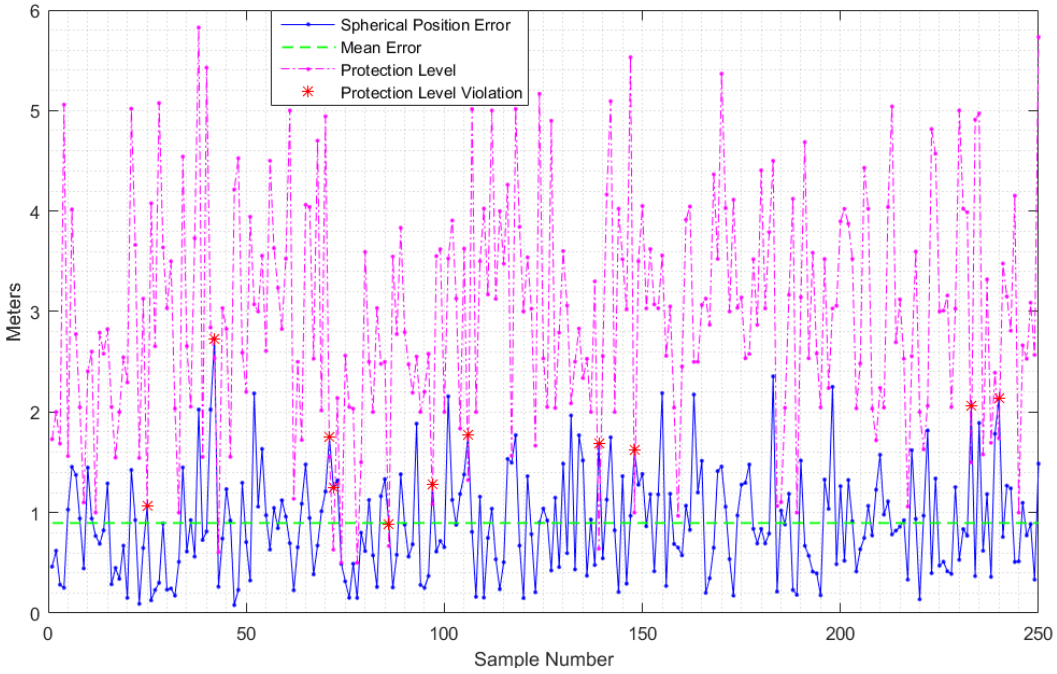


Figure 65. Spherical Error When Using a Uniform Prior Probability Distribution.

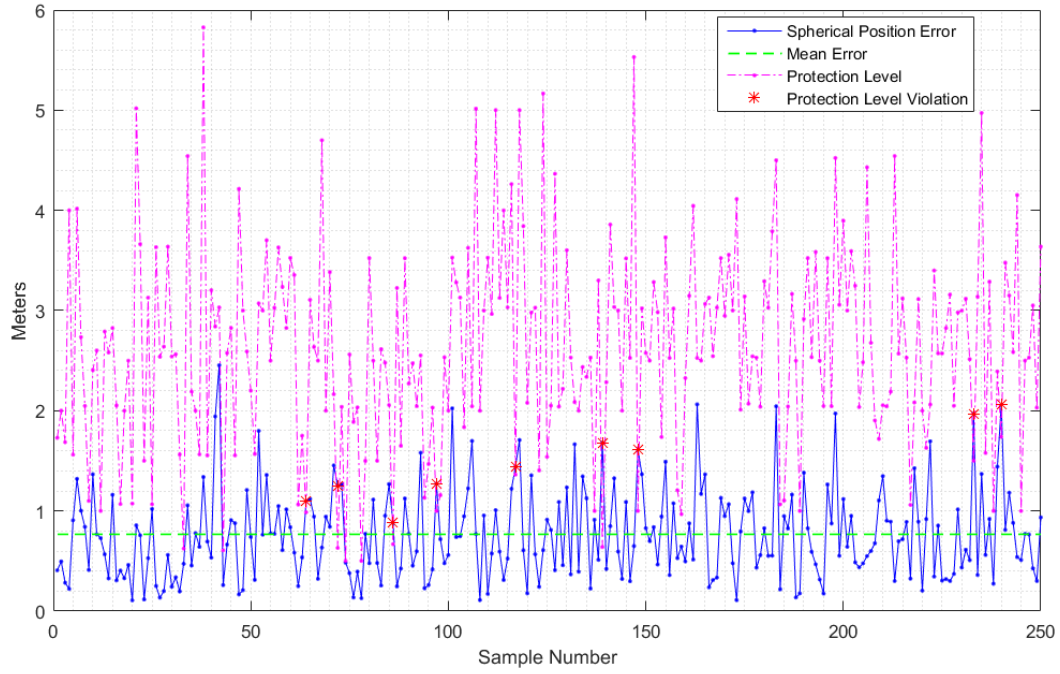


Figure 66. Spherical Error When Using Gaussian Prior Probability Distribution A.

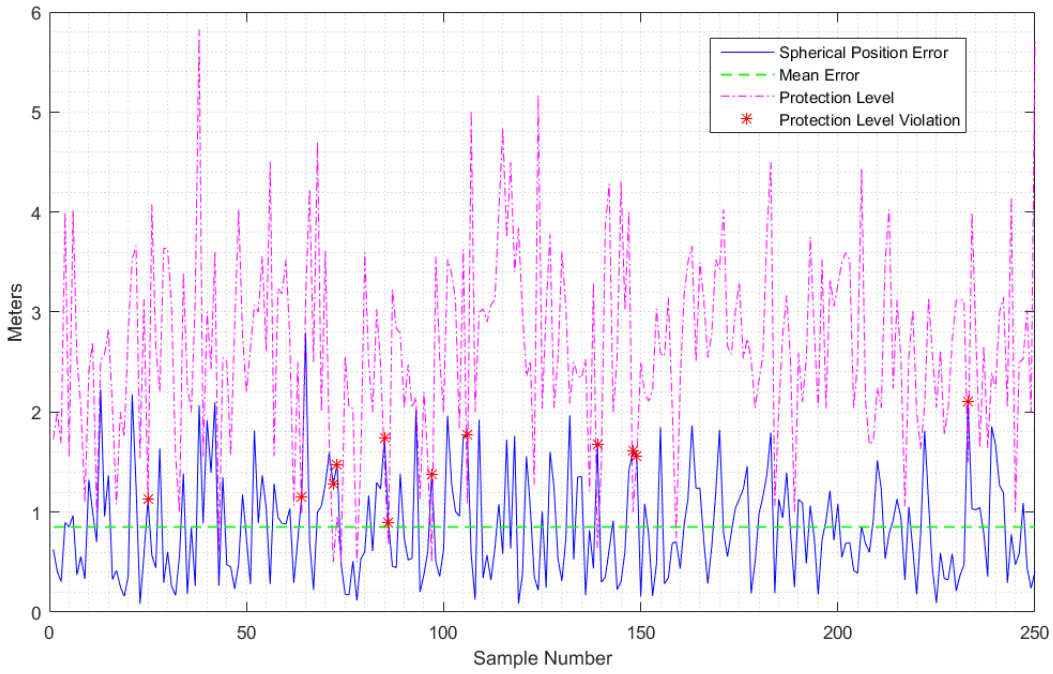


Figure 67. Spherical Error When Using Gaussian Prior Probability Distribution B.

As would be expected, Gaussian distribution A resulted in the lowest mean errors and the lowest error variances since it weighted the PMF toward the true position of the aircraft. Using a one-sample t-test, the reduction in mean errors when using Gaussian distribution A instead of a uniform distribution as a prior was statistically significant at a confidence level of more than 99% in all dimensions except for the x-component. The reduction in mean spherical error was statistically significant at a confidence level $\gg 99\%$. Additionally, according to the results of two-sample F-tests, using Gaussian distribution A as a prior led to lower variances of statistical significance in spherical error, the x-component of the error, and the z-component of the error than the uniform prior case.

Using Gaussian distribution B as a prior also resulted in lower average errors than the uniform prior case. However, only the difference in mean y-component errors was found to be statistically significant at the 95% confidence level with a paired

t-test. Based on two-sample F-tests, using Gaussian distribution B as a prior did not result in statistically significant differences in variances from the uniform prior case. The biggest advantage of using Gaussian distribution B as a prior was found in the protection level computation as will be discussed below.

With respect to protection levels, both forms of the Gaussian prior led to lower average protection levels than the uniform prior case. Using a one sample t-test, these reductions were found to be statistically significant at confidence levels $\gg 99\%$. Eleven protection level violations were observed when using a uniform prior probability distribution, nine were observed when using Gaussian distribution A as the prior, and twelve were observed when using Gaussian distribution B as the prior. Since an integrity risk level of 0.05 was utilized in these experiments, an average of 12.5 protection level violations would be expected in each case. Hence, each form of the prior probability distribution yielded an acceptable number of protection level violations.

Again, the mean spherical error in the Gaussian distribution B prior case was not found to have a statistically significant difference from the mean spherical error with the uniform prior case. However, it reduced the average protection level by 11% without radically increasing the number of protection level violations. Additionally, based on the results of a two-sample F-test, the Gaussian B prior case was found to have a lower protection level variance than the uniform prior case at a 95% confidence level. Hence, simulation results indicate that the use of an informative prior, including one with some amount of random error, may marginally improve algorithm position estimate performance and will likely lead to smaller, less variable, but still accurate protection levels.

Previously, Calhoun's work [2] showed that a uniform prior outperformed a Gaussian prior of the form of Gaussian distribution B. However, he utilized a much tighter form of the covariance matrix, $\mathbf{C}$, with elements equal to 0.25 m^2 rather than 4 m^2.

Simulations run with the same Gaussian distribution used by Calhoun resulted in similarly poor performance. When using that distribution, mean spherical error was 1.861 meters and actually exceeded the mean protection level of 1.523 meters. This resulted in 140 protection level violations out of 250 total samples. However, the results with Gaussian distribution B indicate that the use of an informative prior can improve R7 performance if the magnitude of the covariance matrix is not set too low.

Table 17. Prior Probability Distribution Comparison, Composite Position Estimator Errors, Left Camera Frame, Pixel Intensity Threshold of 0.05.

	Mean, Gaussian Prior	Std Dev, Gaussian Prior	Mean, Gaussian Prior With Error	Std Dev, Gaussian Prior With Error	Mean, Uniform Prior	Std Dev, Uniform Prior
Spherical Error (m)	0.767	0.457	0.851	0.530	0.896	0.527
x Position Error (m)	0.008	0.316	0.008	0.376	0.004	0.374
y Position Error (m)	0.002	0.163	-0.002	0.171	0.008	0.174
z Position Error (m)	-0.151	0.806	-0.131	0.905	-0.197	0.935
Protection Level (m)	2.619	1.011	2.624	0.974	2.956	1.138

4.1.7.7 V5 Simulation Processing Times.

No effort was made to optimize processing times in this thesis, but algorithm processing times were examined to assess how easily each algorithm could be implemented in a real-time system. Simulation observation images were processed in MATLAB® using the method described in Section 3.4.3. The total processing time for the 250 simulation observation images was 101.2 seconds, an average of 0.405 seconds per

image. Performing the image processing in a program other than MATLAB® could reduce image processing times.

Simulation images were processed prior to execution of the remainder of the V5 algorithm. In other words, pre-processed images were provided to the algorithm. Figure 68 shows the processing time required to execute the R7 algorithm on each of the 250 pre-processed images. The mean run time was 53.8 seconds. Hence, a mean run time of approximately 54.2 seconds would be expected if images were not pre-processed. These long run times represent a significant challenge to V5 implementation in a real-time system as such a system may need to process tens or hundreds of images each second, rather than one image every minute.

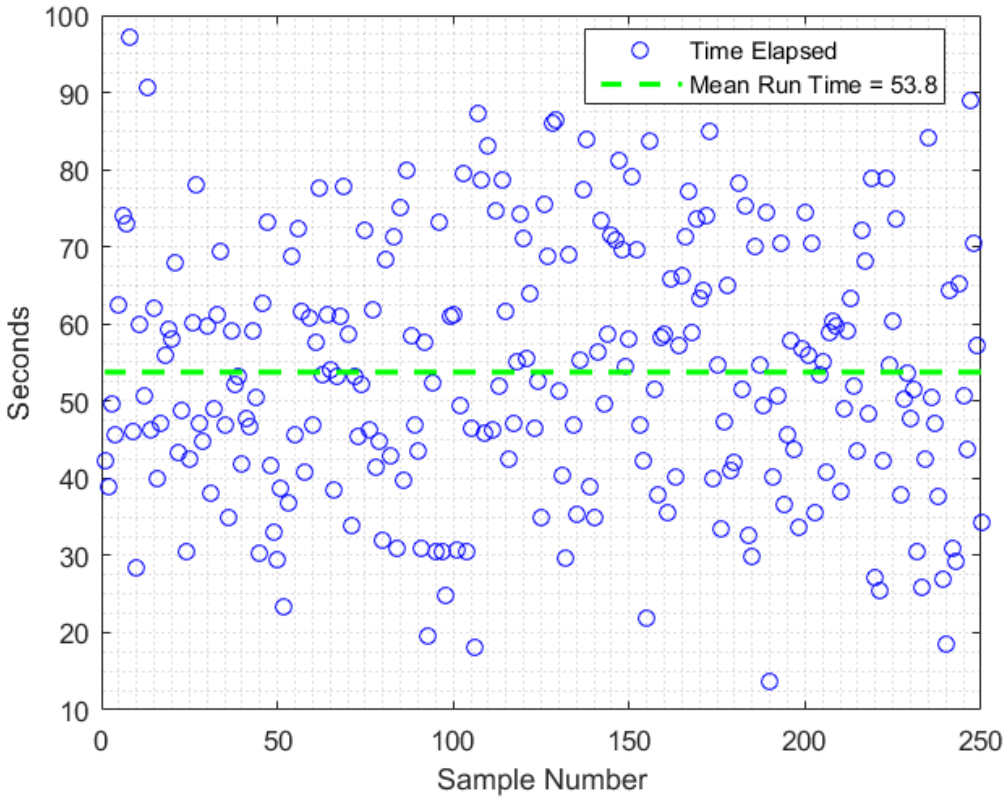


Figure 68. Time to Execute the V5 Algorithm on Pre-Processed Observation Images.

4.2 R7 Data and Analysis, Simulation

This section details the analysis of the R7 algorithm conducted in simulation. The R7 algorithm is less tuning intensive than the V5 algorithm. As a result, this section is comprised of two parts. First, Section 4.2.1 describes the observation image sets used in simulation. Second, Section 4.2.2 details the most significant R7 simulation results.

4.2.1 Observation Image Generation.

Two observation image sets were created in the 3DVW for analysis of the R7 algorithm—one with the tanker boom in the camera fields of view and one without a tanker boom present. Each set placed the simulated cameras at an altitude of 3,000 meters above a depiction of Wright-Patterson Air Force Base Area B. The images depicted 6,000 randomly generated C-12 receiver positions. Props were not included in the C-12 model. The positions were randomly sampled from a geometric space specified using the same methodology used with V5 simulations (see Section 4.1.3).

Receiver range was randomly selected from a uniform distribution spanning 20 to 250 meters. Receiver azimuth was randomly selected from a uniform distribution spanning -15° to 15°. As with the V5 database, azimuth was defined as the angular position left or right of the tanker's longitudinal axis in the tanker xy-plane. Receiver elevation was randomly selected from a uniform distribution spanning 5° to 35° below the tanker's longitudinal axis. As with the V5 database, elevation was defined as the angular position below the tanker's longitudinal axis in the tanker xz-plane. This simulation space was chosen to cover relative positions likely to be encountered during AR operations. Figure 69 depicts the positions contained within a randomly generated set of 6,000 positions.

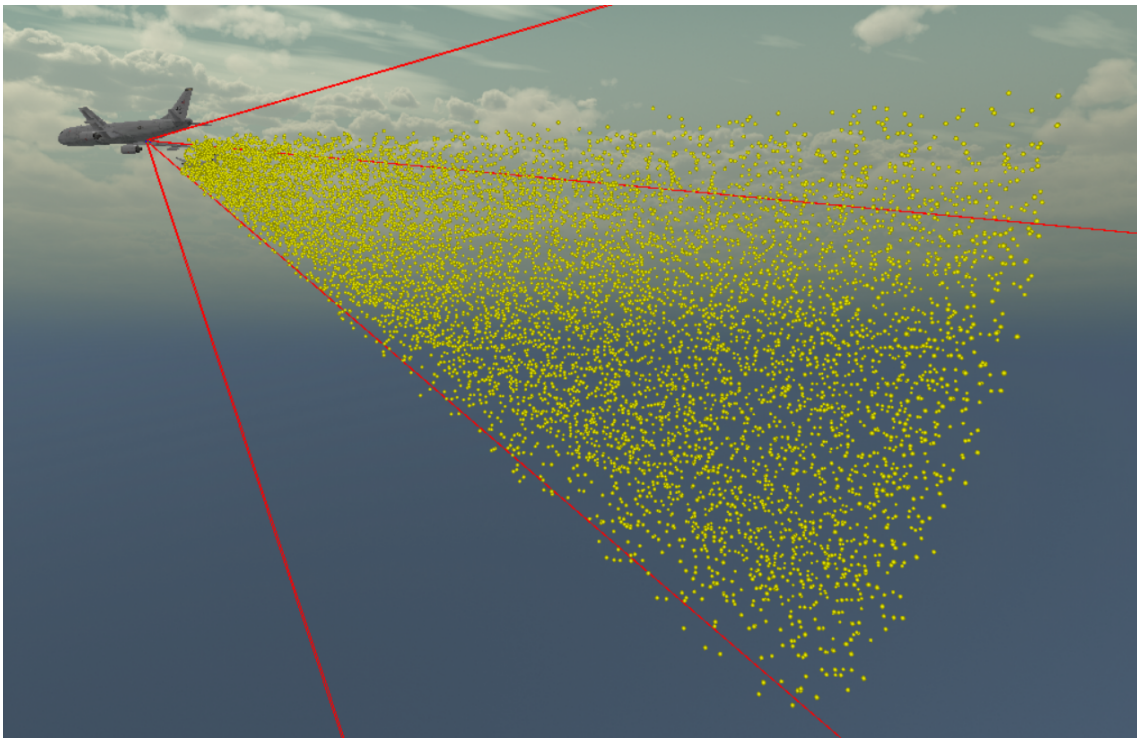


Figure 69. Depiction of 6,000 Randomly Generated Positions in Yellow and the Camera Fields of View in Red.

The attitude of the C-12 at each position was randomly generated with the same scheme used in V5 analysis. The relative roll of the receiver was randomly selected from a uniform distribution varying between $\pm 7°$. The relative pitch of the receiver was randomly selected from a uniform distribution varying between $\pm 4°$. The relative yaw of the receiver was randomly selected from a uniform distribution varying between $\pm 1°$.

4.2.2 R7 Simulation Results.

The R7 algorithm was applied to five different cases in simulation. Table 18 summarizes these cases. Cases in which the boom was not in the fields of view utilized the observation image set created without a boom. Cases in which the attitude was provided to the R7 algorithm, used the attitude fed version of the ICP algorithm

described in Section 3.5.5.6. For Case 5, the R7 algorithm was applied to the same data set used in V5 simulation analysis.

Table 18. R7 Simulation Cases.

	Case 1	**Case 2**	**Case 3**	**Case 4**	**Case 5**
Boom in Fields of View?	No	No	Yes	Yes	No
Provide Attitude?	No	Yes	No	Yes	No

This scheme enabled the comparison of algorithm performance with and without attitude feeding and with and without the tanker boom in the field of view. The simulation results revealed five primary findings:

1. Relative position error growth was most strongly correlated with the z-axis of the camera frames, that is, the depth of the observed receiver.

2. Error distributions were mostly stable within 40 meters of depth. Errors within this depth did not show depth correlation.

3. The attitude fed version of the R7 algorithm reduced average position estimate errors.

4. The presence of a boom in the camera fields of view increased R7 errors and error variances. This resulted manifested despite the implementation of an effective boom filter.

5. Using the attitude fed version of the ICP algorithm reduced R7 processing times.

The following conventions were used to generate the plots and data analysis in this section. Since relative position error growth was most strongly correlated with the

receiver's z-position in the camera frames, algorithm position errors were computed in the left camera frame and plotted against the receiver's true left camera frame z-position. Attitude errors were expressed in the tanker body frame, but were also plotted against the receiver's true left camera frame z-position. Additionally, in the figures, mean errors and standard deviations were computed at two meter intervals of depth. Each of these intervals contained, on average, approximately 50.8 samples. This method was used to help identify correlation between error growth and increasing depth. Charts in which markers indicate "No valid measurements" mean that fewer than three points were obtained in the point cloud for that observation. This small number of points precludes execution of the ICP algorithm. Finally, a depth of 40 meters was chosen for more in-depth analysis, since the bulk of contact and pre-contact operations are conducted within approximately this range.

4.2.2.1 Case 1: No Boom in Field of View, No Attitude Feeding.

In simulation, the R7 algorithm was first applied to the 6,000 observation images without the boom in the fields of view and without the true relative formation attitude being provided to the algorithm. As described in Section 3.5, this form of R7 provides both a relative position and relative attitude measurement.

Figures 70 and 71 depict the spherical error of each observation image plotted against the receiver's z-position in the left camera frame. As can be seen in the figures, both the mean and standard deviation of the error increased strongly with increasing depth from the camera. Figure 71 shows that this effect was essentially negligible within 40 meters of depth. Table 19 presents the mean and standard deviations of the spherical position errors observed within 40 meters of depth.

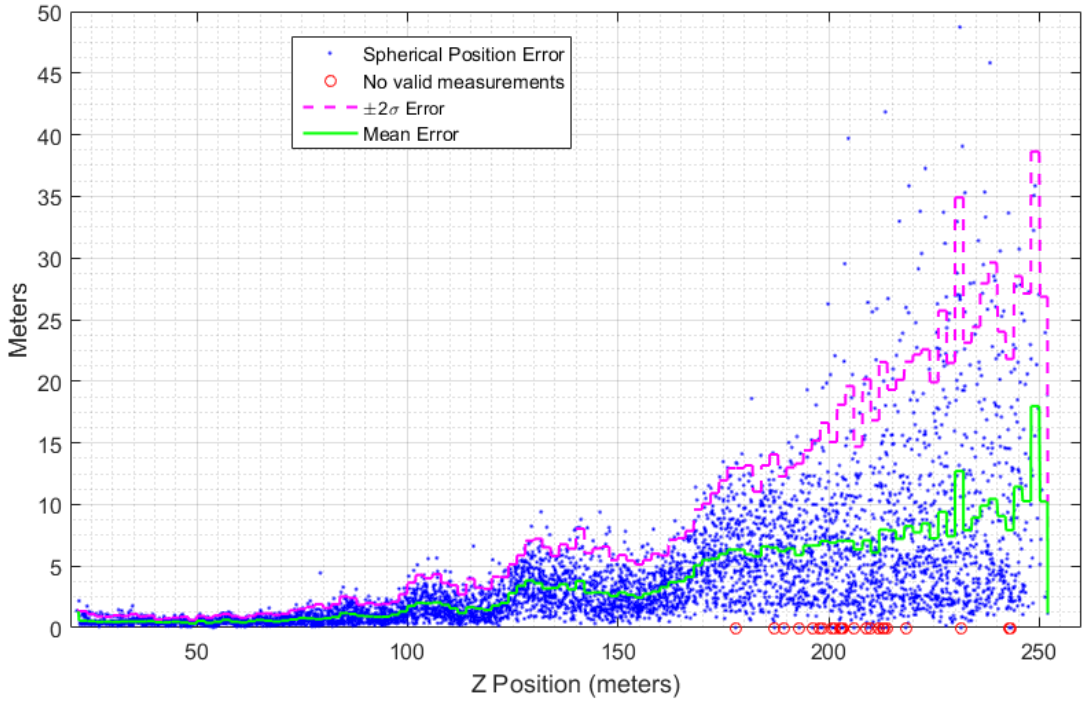


Figure 70. Spherical Position Errors, Case 1.

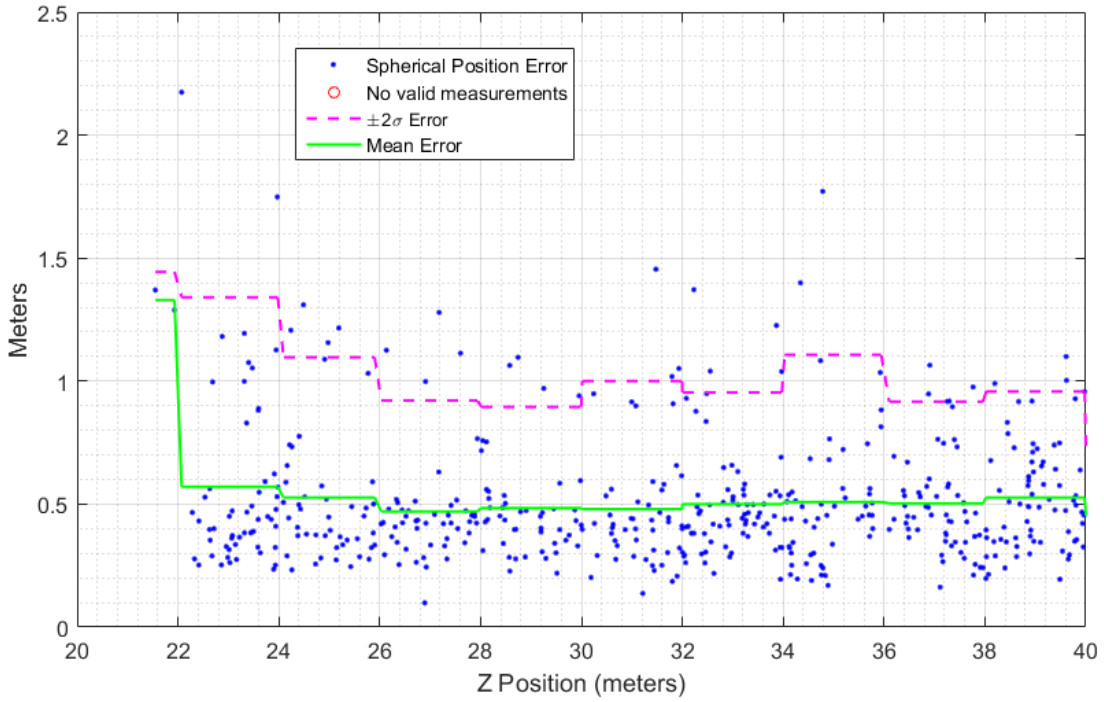


Figure 71. Spherical Position Errors, Case 1, Zoom In.

Figures 72 and 73 plot the relative position error in each axis of the left camera frame against the receiver's left camera frame z-position. As can be seen in the figures, error growth was most strongly correlated with the z-component of the relative position estimate. Additionally, the z-component mean becomes increasingly biased with depth. The error distributions were fairly stable within 40 meters of depth and did not show depth correlation. However, within 40 meters of depth the z-axis presented a consistent bias of approximately -0.48 meters. This bias was negative indicating that the R7 algorithm was underestimating the depth of the receiver. Similarly, the y-axis also presented a slight bias of approximately 0.16 meters. As shown in Section 2.3.6.6, based on how x and y-positions are computed, the y-axis bias could be attributed to the presence of the z-axis bias. Table 19 presents the mean and standard deviations of the position errors observed within 40 meters of depth.

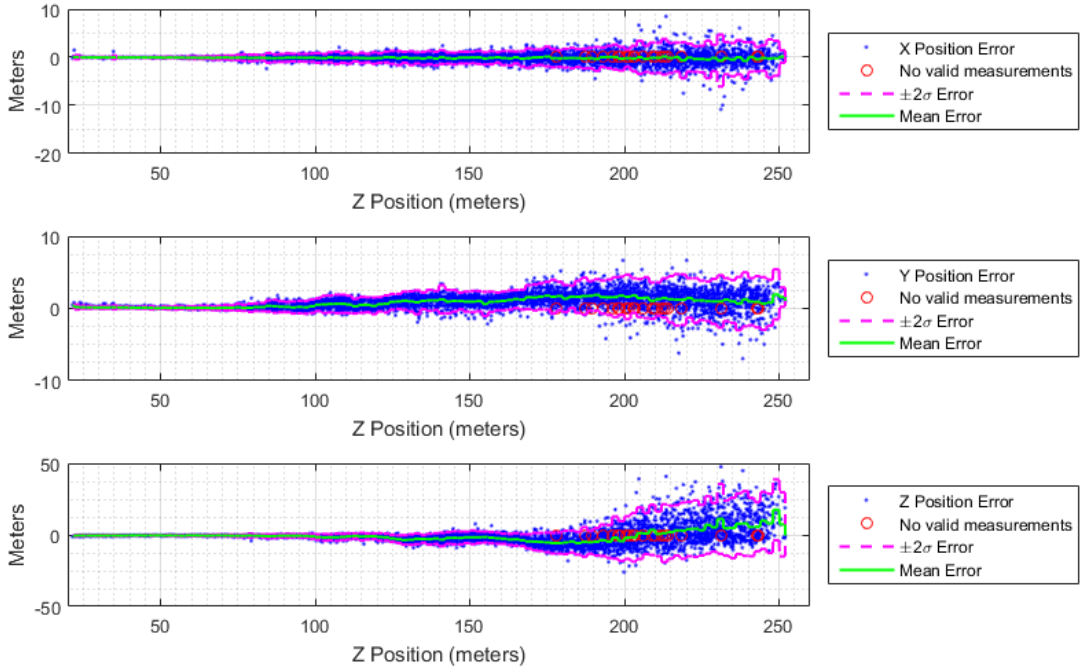


Figure 72. Position Errors, Case 1, Left Camera Frame.

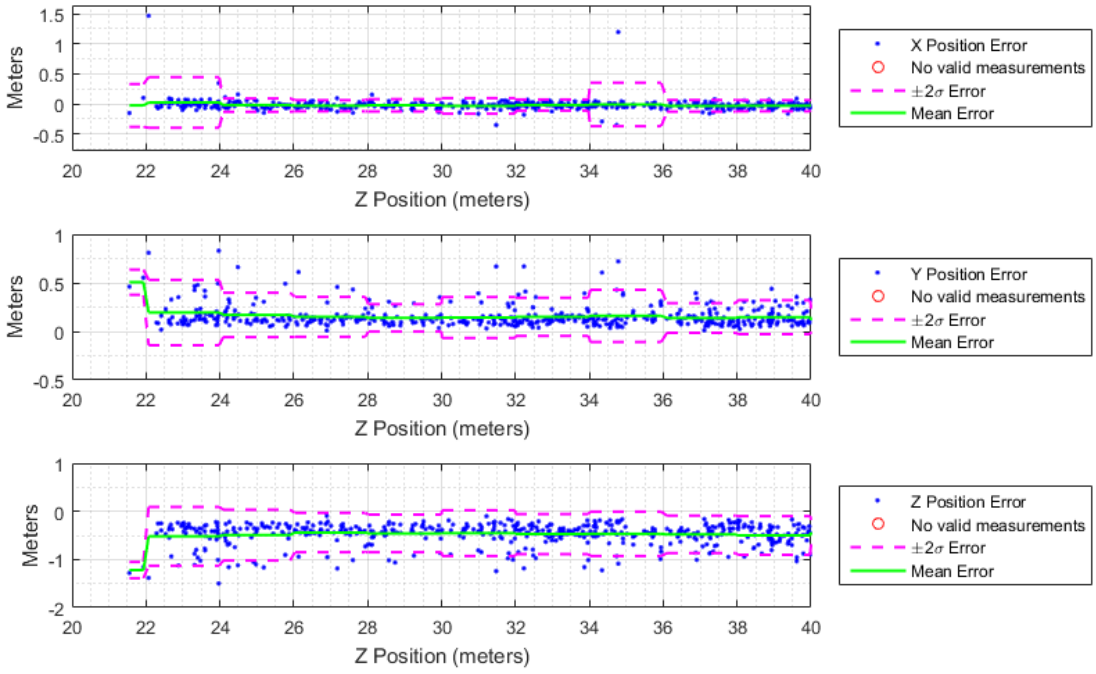


Figure 73. Position Errors, Case 1, Left Camera Frame, Zoom In.

As was shown in Section 2.3.6.6, the z-position in the camera frame is computed as:

$$Z_{L_r} = -\frac{f_c T_x^R}{d} \tag{110}$$

Where f_c is the focal length in pixels, T_x^R is the baseline between the stereo camera pair, and d is the observed disparity. Hence, the z-axis bias could be due to errors in the focal length, camera baseline, or disparity computations. However, since the camera focal lengths and baselines were known perfectly in simulation, these are unlikely causes. Additionally, errors in the model point cloud could contribute to bias, but this is also unlikely because the model point cloud was derived from the same model used to depict the simulated C-12C receiver in the 3DVW. A more likely culprit is disparity.

As was shown in Section 2.3.6.6, the depth resolution of each point in the observation point cloud is limited by the disparity resolution of the underlying algorithm:

$$\Delta Z_{L_r} = \frac{Z_{L_r}^2}{f T_x^{R_r}} \Delta d \tag{111}$$

Where Δd is the disparity resolution and ΔZ_{L_r} is the commensurate depth resolution. The R7 algorithm used a disparity resolution of $\frac{1}{16}$. Additionally, the focal length used in simulation was 962.9 pixels and the stereo baseline was 0.5 meters. Therefore, at a ranges of 22 and 40 meters, a depth resolution of 0.063 meters and 0.208 meters would be expected, respectively. This resolution limit could partially explain the observed bias.

Further analysis of R7 disparity computations, revealed that the algorithm errors tended to get worse further from integer disparity values. A simulation was run which generated 5,000 receiver images at z-ranges between 22 and 250 meters. The x and y position of the receiver in the left camera frame were forced to be 0.25 meters and 0 meters, respectively, in each instantiation. This meant that the receiver fell precisely between the optical axes of the left and right cameras at each position. Figure 74 plots the resultant spherical position errors from the R7 algorithm. In the figure, the disparity values depicted were computed for a point falling exactly at the receiver position, which in simulation was essentially the geometric center of the C-12C. As can be seen in the figure, the spherical error had a pronounced tendency to grow in magnitude at ranges corresponding to non-integer disparity values. This phenomenon became more drastic with increased depth. As can be seen in Figure 75, this tendency was less pronounced with respect to z-position errors, but was still present. Although this does not directly explain the observed depth bias, it does indicate that the disparity computations, particularly at the sub-pixel level, are

imperfect. Hence, these observations suggest that errors in disparity computations are likely a primary source of algorithm error.

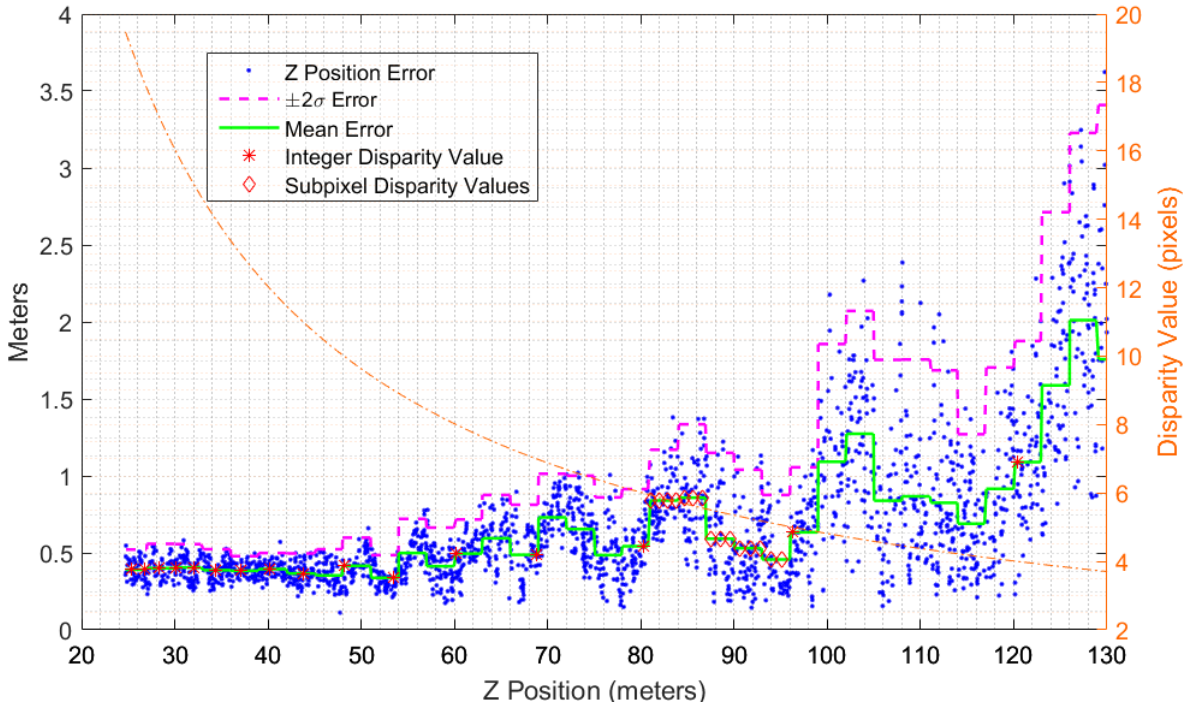


Figure 74. Spherical Position Error and Disparity Values.

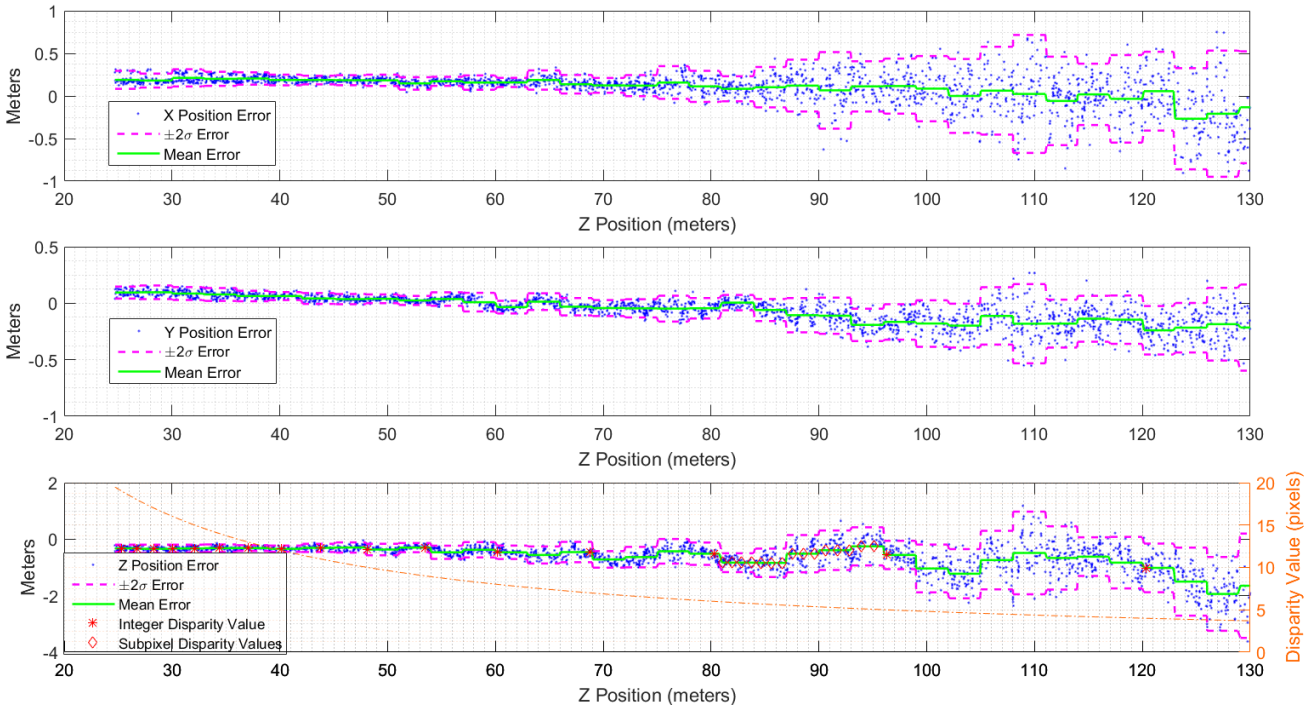


Figure 75. Position Errors, Left Camera Frame, and Disparity Values.

With respect to attitude estimation, the R7 algorithm proved to be a relatively poor estimator as can be seen in Figures 76 and 77. These attitude errors are expressed in the tanker frame. Table 19 presents the mean and standard deviations of the attitude errors observed within 40 meters of depth. Although the errors were very close to having zero means within 40 meters, the standard deviations of these errors were relatively large. Roll and pitch error standard deviations were notably higher than that of the yaw error. Additionally, a strong bias in the pitch axis manifests at farther ranges. By contrast, the roll and yaw errors tended to stay closer to zero mean even at farther ranges. These high attitude error variances could possibly be overcome by bounding magnitude of the rotations that R7 allows the ICP algorithm to apply to the observed point cloud. Doing so, however, would not enable to algorithm to observe actual unusual attitude cases.

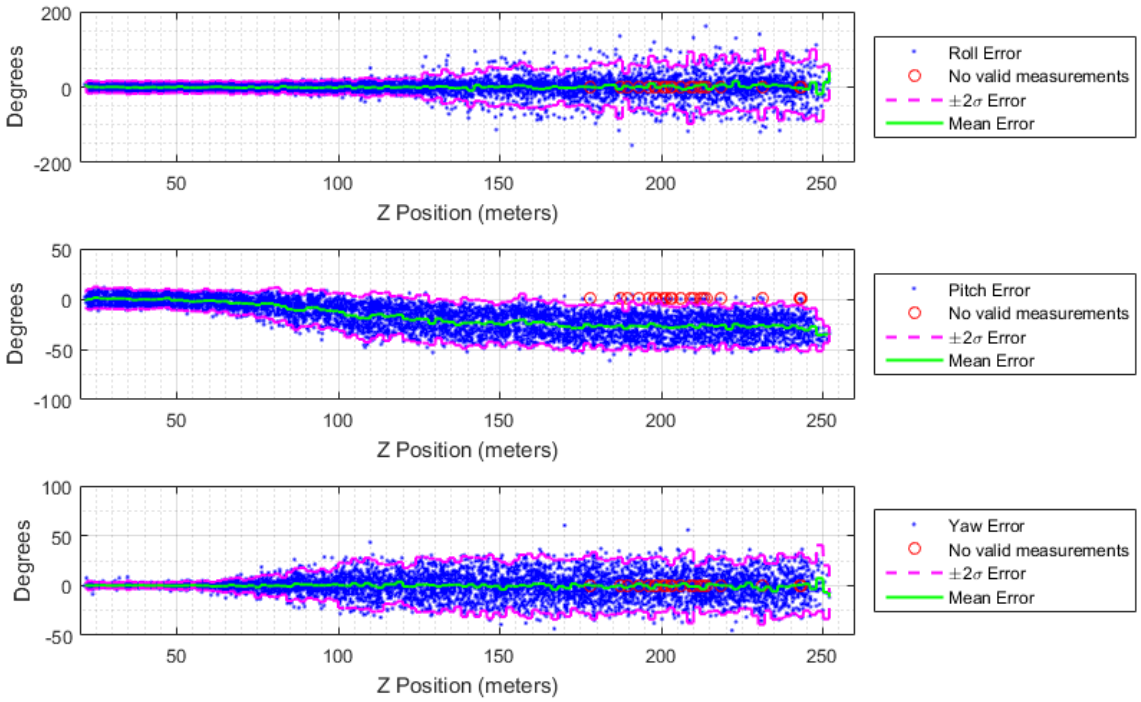


Figure 76. Attitude Errors, Case 1, Tanker Frame.

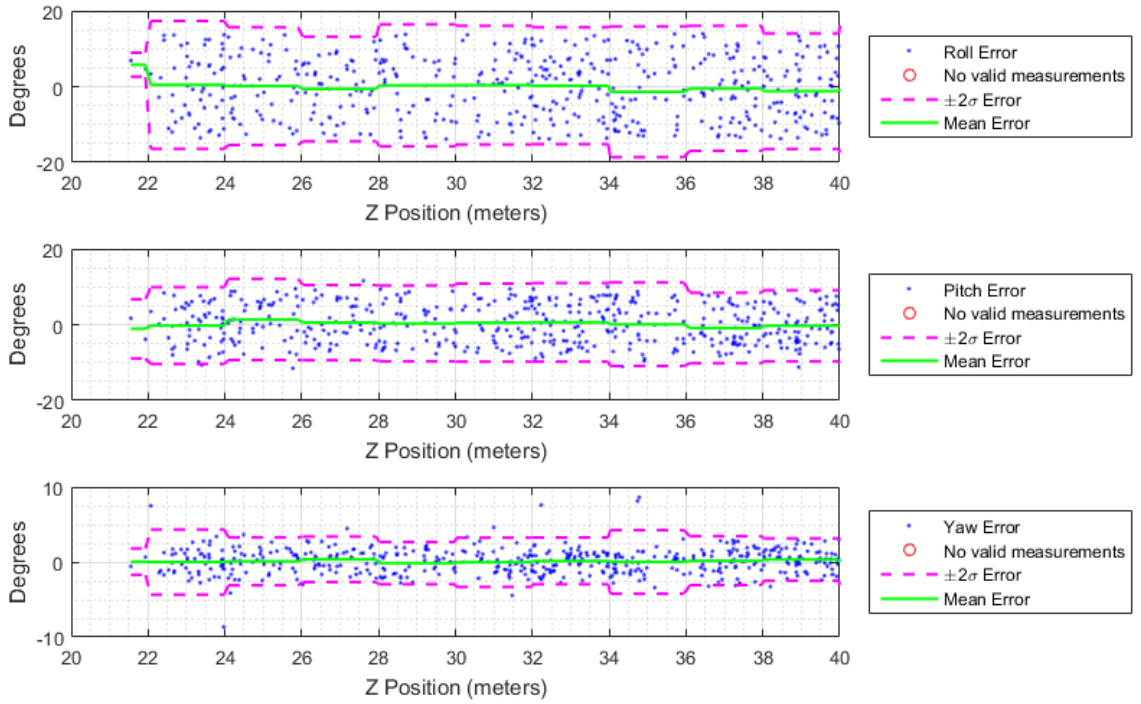


Figure 77. Attitude Errors, Case 1, Tanker Frame, Zoom In.

Table 19. Position Errors in the Left Camera Frame and Attitude Errors in the Tanker Frame for Samples at Less than 40 Meters of Depth, Case 1.

	Mean	Standard Deviation
Spherical Position Error (m)	0.512	0.264
x Position Error (m)	-0.020	0.103
y Position Error (m)	0.156	0.112
z Position Error (m)	-0.480	0.232
Roll Error (deg)	-0.248	7.879
Pitch Error (deg)	0.195	5.045
Yaw Error (deg)	0.117	1.676

4.2.2.2 Case 2: No Boom in Field of View, Attitude Feeding.

For Case 2, the R7 algorithm was applied to the 6,000 observation images without the boom in the field of view and with the true relative formation attitude being provided to the algorithm. Section 3.5.5.6 describes how ICP was modified to enable attitude provisioning. In this form, the R7 algorithm provides only a relative position measurement.

Figures 78 and 79 depict the spherical error of each observation image plotted against the receiver's z-position in the left camera frame. As can be seen in the figures, both the mean and standard deviation of the error increased strongly with increasing depth from the camera. The figures show that this effect was essentially negligible within 40 meters of depth. Figures 80 and 81 depict the position error components in the left camera frame. At first glance, results were very similar to Case 1, with z-position bias growth with depth. Table 20 presents the mean and standard deviations of the position errors observed within 40 meters of depth for Case 2 alongside those found in Case 1.

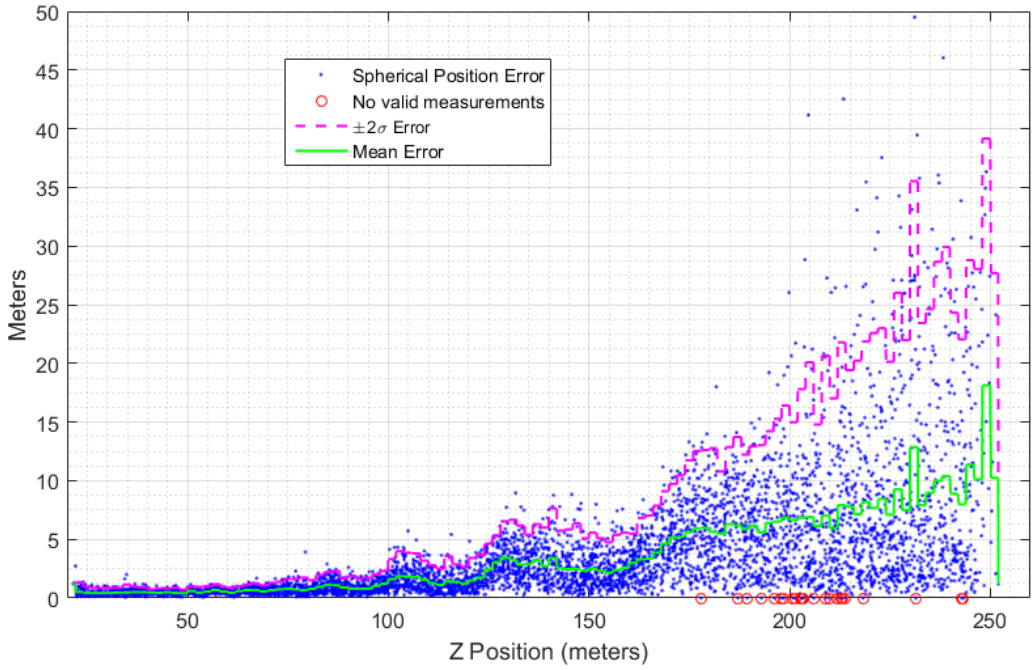


Figure 78. Spherical Position Errors, Case 2.

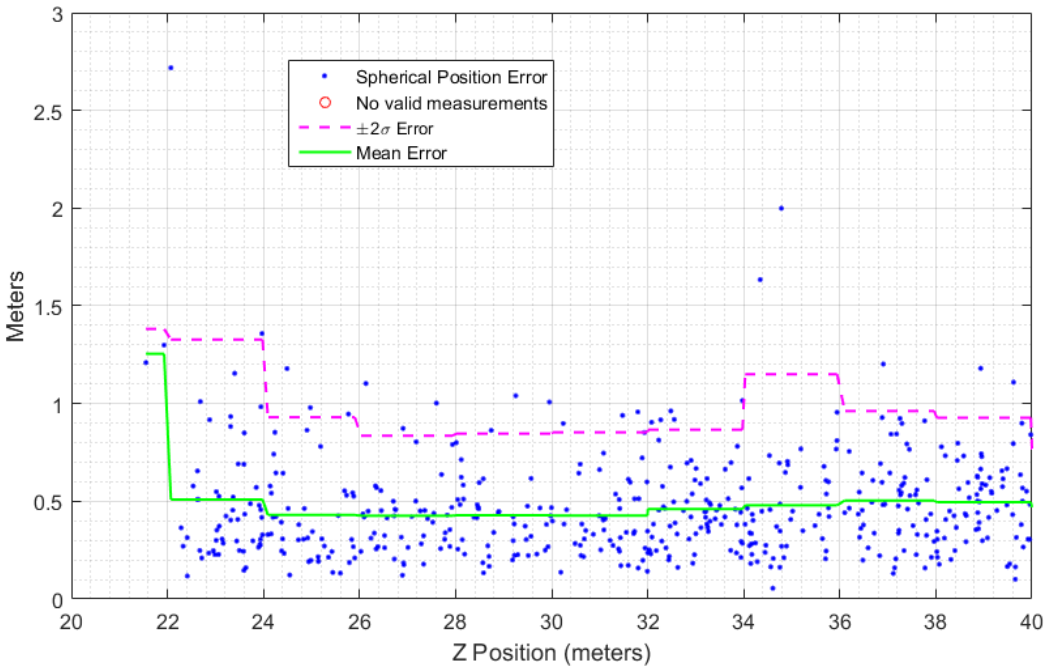


Figure 79. Spherical Position Errors, Case 2, Zoom In.

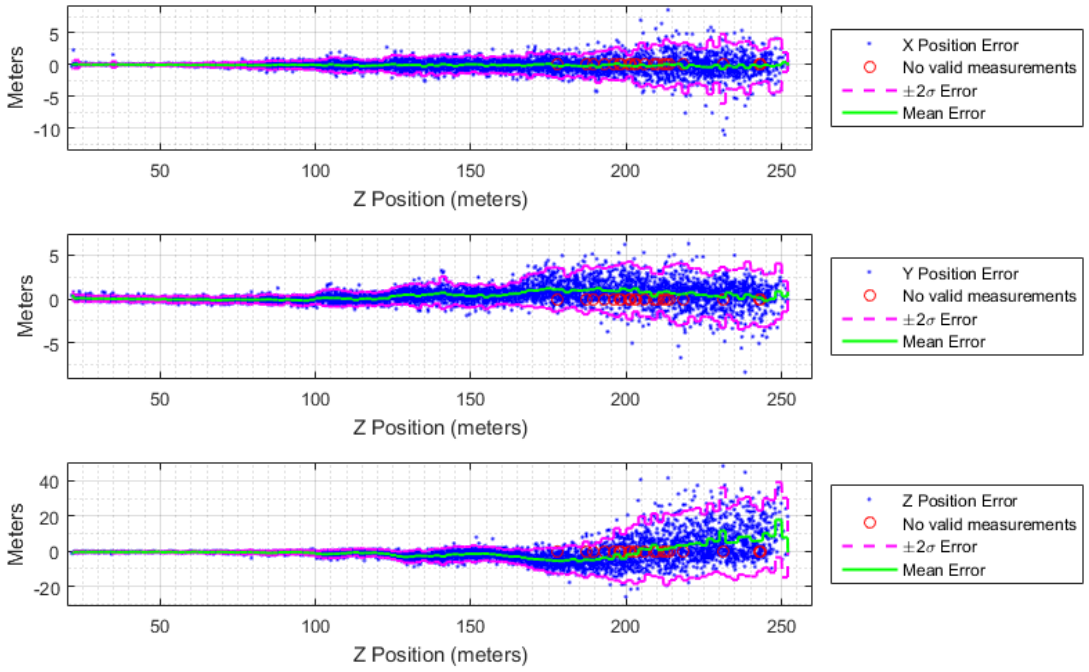


Figure 80. Position Errors, Case 2, Left Camera Frame.

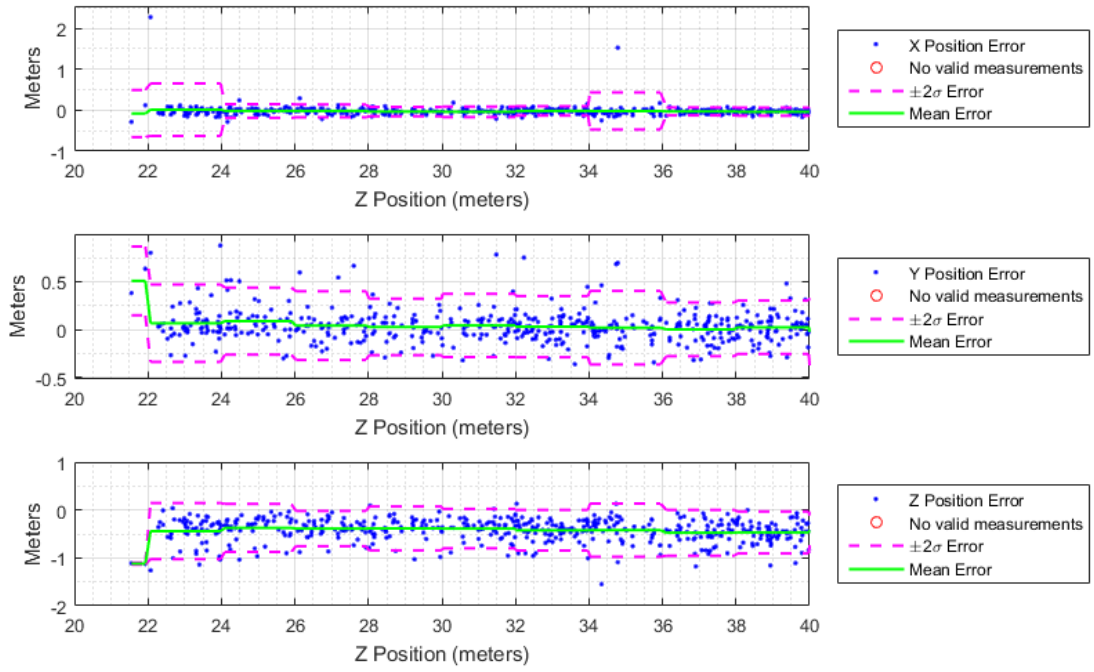


Figure 81. Position Errors, Case 2, Left Camera Frame, Zoom In.

Figures 82 and 83 plot the spherical position errors from Case 1 and Case 2 together. As can be seen in the figures and in Table 20, providing an attitude solution to the R7 algorithm reduced position errors on average, including at close ranges. Paired t-tests were performed to test whether the attitude fed version of the R7 algorithm significantly reduced errors. These tests were applied to the errors from Cases 1 and 2 for observations collected at depths of 40 meters or less.

In terms of mean spherical error, the attitude fed version of ICP (Case 2) was found to modestly reduce mean error at a statistically significant level. The 95% confidence interval for this reduction spanned 0.033 to 0.055 meters. The x-position error means were not found to be significantly different. In terms of mean y-position error, the attitude fed version of ICP was found to reduce mean error at a statistical significance level of $\gg 99\%$. The 95% confidence interval for this reduction spanned 0.105 to 0.129 meters. In other words, the non-attitude fed version (Case 1) had roughly 11 centimeters more positive y-position estimate bias. In terms of mean z-position error, the attitude fed version of ICP was found to modestly reduce mean error at a statistical significance level of $\gg 99\%$. The 95% confidence interval for this reduction spanned 0.068 to 0.042 meters. In other words, the non-attitude fed version (Case 1) had roughly 5 centimeters more negative z-position estimate bias.

The fact that the most improvement was seen in the y-axis component of the estimate may be correlated with the fact that the R7 algorithm is worst at estimating pitch. In the case in which the R7 algorithm is estimating both attitude and position, it may be mistakenly applying a pitch rotation instead of a translation in y.

Paired F-tests were applied to the variances of the position errors observed in Cases 1 and 2. Interestingly, these tests revealed that Case 2 had significantly more error variance in the x and y-position errors at confidence levels $\gg 99\%$. Spherical error variance and x-position error variance differences were not statistically signif-

icant. Hence, simulation results suggested that although the attitude fed version of R7 may be less biased, it may also be subject to greater error variances in some dimensions. However, as is discussed in Chapter VI, flight tests results indicated the opposite result. If greater variance were to manifest with real-world tests, then future researchers will have to balance the costs of this possibly greater noise with the benefits of having possibly less bias.

Finally, the attitude fed version of R7 was found to reduce processing times. It took a total of 82.7 seconds to process all the observations for Case 1. It took a total of 78.4 seconds to process all the observations for Case 2. This is a 5.2% reduction in processing time. Although this is only a single example, results from other simulations conducted during research consistently showed that utilizing the attitude fed version of ICP decreased processing times.

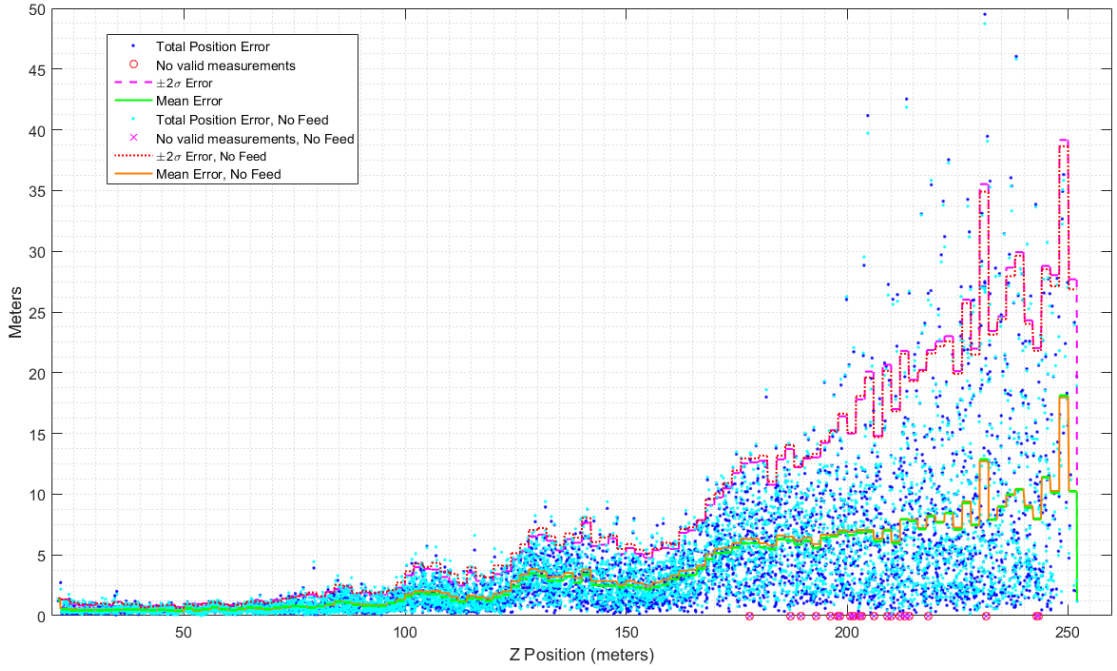


Figure 82. Position Errors, Cases 1 and 2, Left Camera Frame.

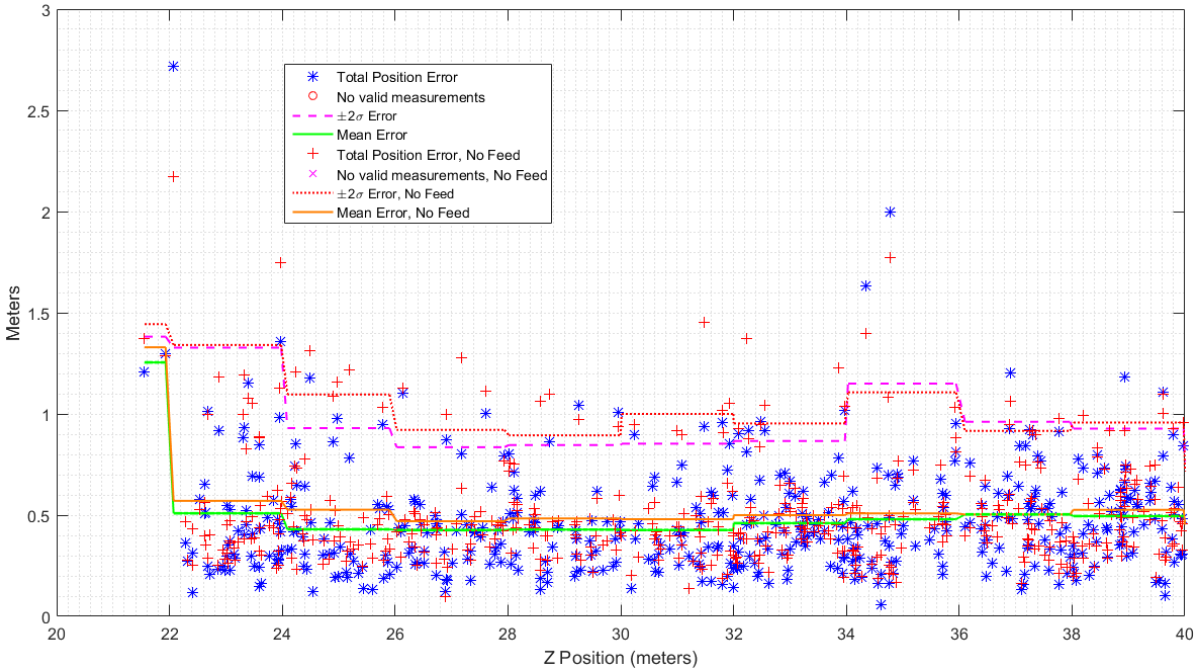


Figure 83. Position Errors, Cases 1 and 2, Left Camera Frame, Zoom In.

Table 20. Position Errors in the Left Camera Frame for Samples at Less than 40 Meters of Depth, Cases 1 and 2.

	Case 1		Case 2	
	Mean	Standard Deviation	Mean	Standard Deviation
Spherical Error (m)	0.512	0.264	0.468	0.263
X Position Error (m)	-0.020	0.103	-0.023	0.140
Y Position Error (m)	0.156	0.112	0.039	0.169
Z Position Error (m)	-0.480	0.232	-0.425	0.240

4.2.2.3 Boom Filtering.

As discussed in Section 2.4, the ICP algorithm will not converge properly if the observed point cloud contains points not found in the model point cloud [8]. Hence, boom filtering, as outlined in 3.5.4.1 is critical to R7 implementation with a boom in the field of view. With a boom filter, the algorithm can overcome many of the negative effects of boom occlusion. Figure 84 depicts the results of the R7 algorithm with and without a boom filter being applied. With the boom filter applied, the point cloud points corresponding to the boom are successfully eliminated and the R7 solution converges nicely to the observed point cloud. Spherical position error in this case was 0.05 meters and attitude errors were each less than one degree. With the boom filter not applied, the R7 solution is skewed toward the point cloud points that correspond to the boom. Spherical position error in this case was 2.5 meters and attitude errors were substantial.

It may seem logical to conclude that the biasing effect of the boom would be reduced at greater ranges. However, Figure 85 shows that this is not true. This figure plots the results of applying R7 to the same data set used in Cases 3 and 4 (images with a boom in the fields of view) but without a boom filter. As the receiver gets farther in range from the tanker, the number of points corresponding to the boom quickly begin to outnumber or reach parity with the number of points corresponding to the receiver. At far ranges, it is evident that the algorithm was estimating the boom position as the position of the aircraft. At close ranges this phenomenon can still occur and substantial errors were still present. The lower portion of the figure elucidates this fact. This effect is aggravated in cases in which the boom obscures much of the receiver. Hence, the biasing effect of the boom is substantial and highly unpredictable. These results show that a boom filter is essential for proper operation of the R7 algorithm or any algorithm leveraging the ICP algorithm in the AR context.

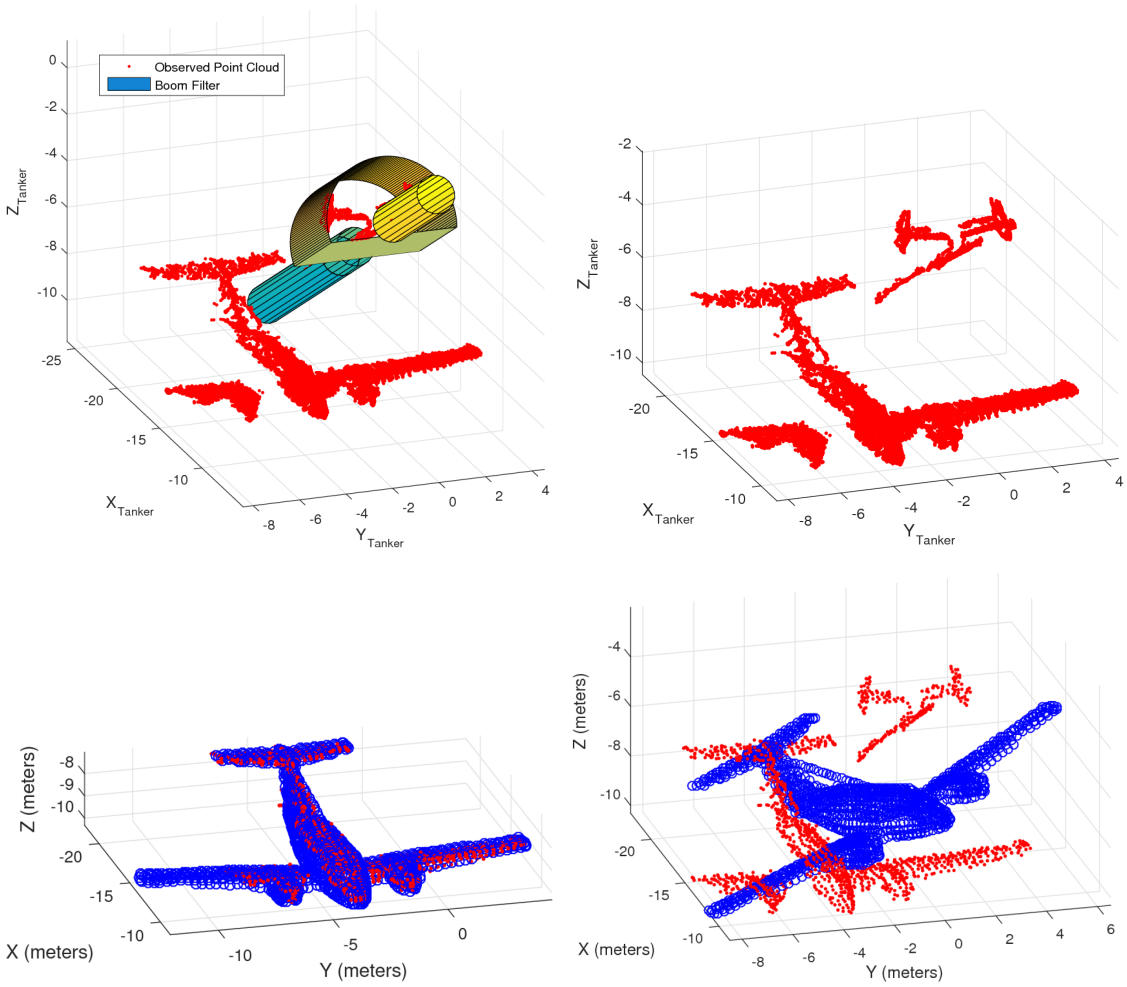


Figure 84. Effect of Boom Filtering, Observed Point Cloud Shown in Red, R7 Solution Shown in Blue.

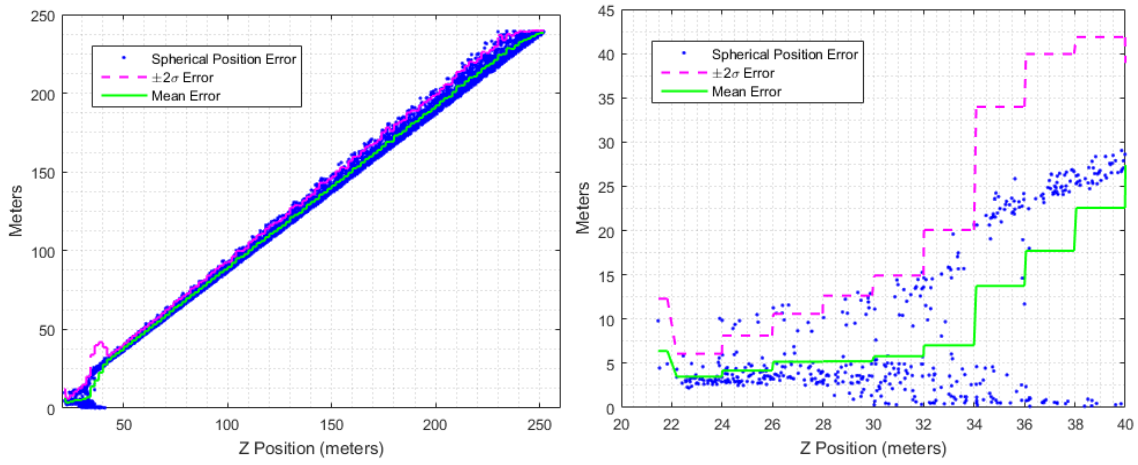


Figure 85. Spherical Position Errors, Boom in Field-of-View, No Boom Filter Applied.

4.2.2.4 Case 3: Boom in Field of View, No Attitude Feeding.

For Case 3, the R7 algorithm was applied to the 6,000 observation images with the boom in the fields of view and without the true relative formation attitude being provided to the algorithm. In this form, the R7 algorithm provides both relative position and attitude measurements.

Figures 86 and 87 depict the spherical error of each observation image plotted against the receiver's z-position in the left camera frame. Table 21 presents the mean and standard deviations of the spherical position errors observed within 40 meters of depth for Cases 1 and 3. As can be seen in the figures, both the mean and standard deviation of the error increased strongly with increasing depth from the camera as with the other cases. The figures show that this effect was essentially negligible within 40 meters of depth. However, there were many more instantiations in which the R7 algorithm lacked a sufficient number of points to provide measurements than in Cases 1 and 2. In Cases 1 and 2, only 25 such results were returned, 0.42% of all observations. In Case 3, 173 such results were returned, 2.88% of all observations. Hence, the obscuring effect of the boom substantially increased the chances that

the R7 algorithm was unable to provide a measurement. In practice, this could be mitigated with receiver flight paths that deliberately avoid the obscured portion of the fields of view.

Additionally, as can be seen in the figures, there were more outliers in Case 3 than in the Cases 1 and 2. There was a particularly strong outlier at just over 100 meters of depth. Figures 88 and 89 plot the spherical errors resultant from Cases 1 and 3 together, showing the greater errors, variance, outliers, and missed measurements present in Case 3. As discussed in the previous section, the obscuring effect of the boom drove these changes.

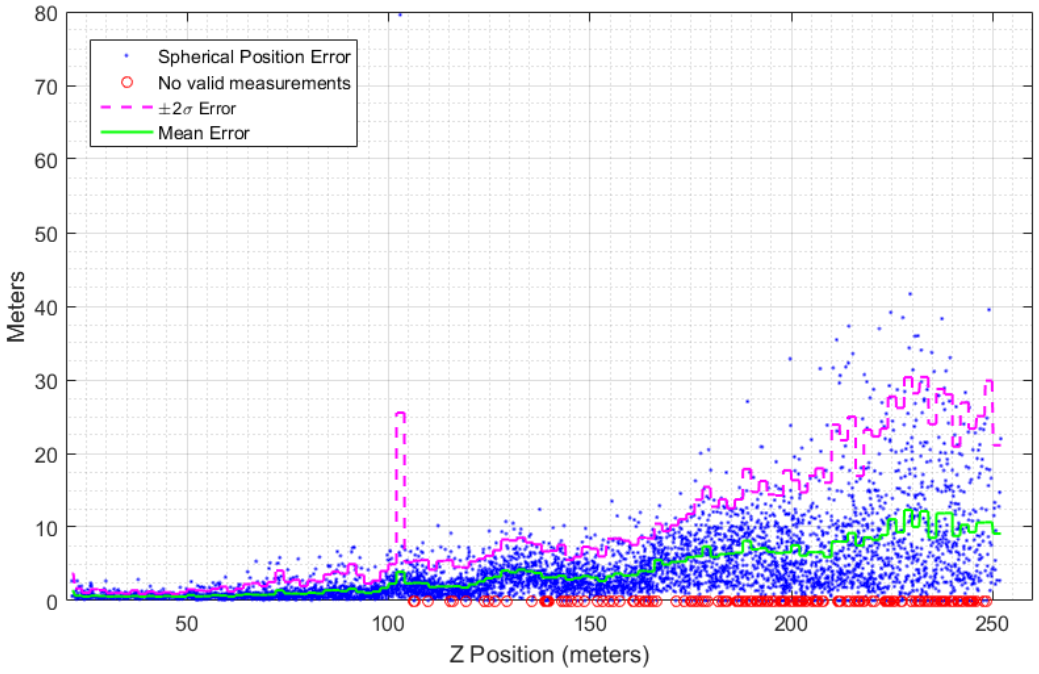


Figure 86. Spherical Position Errors, Case 3.

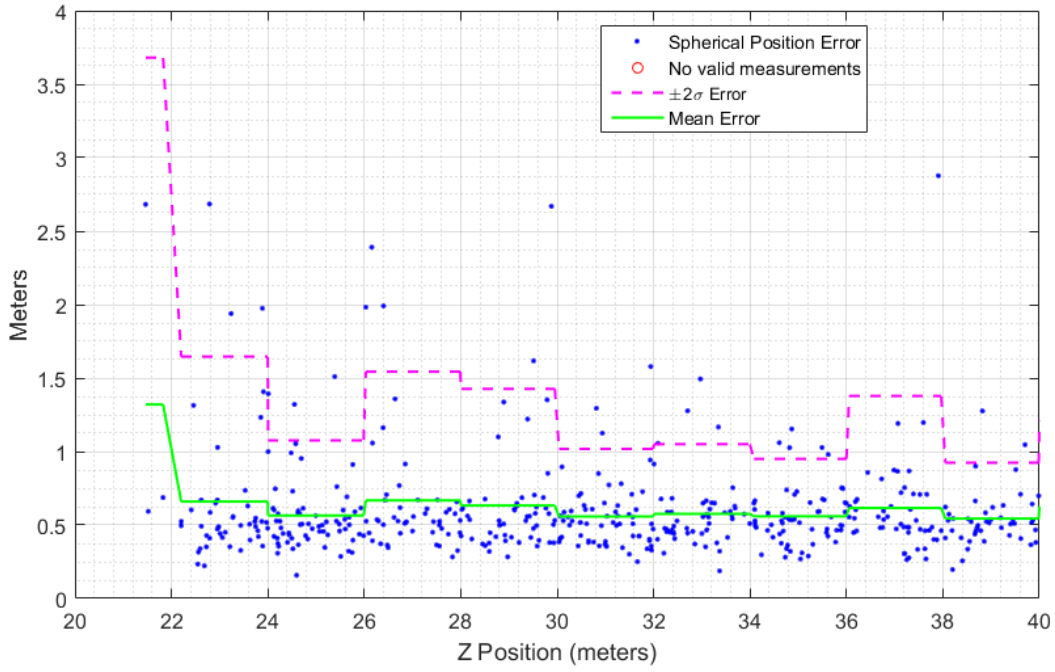


Figure 87. Spherical Position Errors, Case 3, Zoom In.

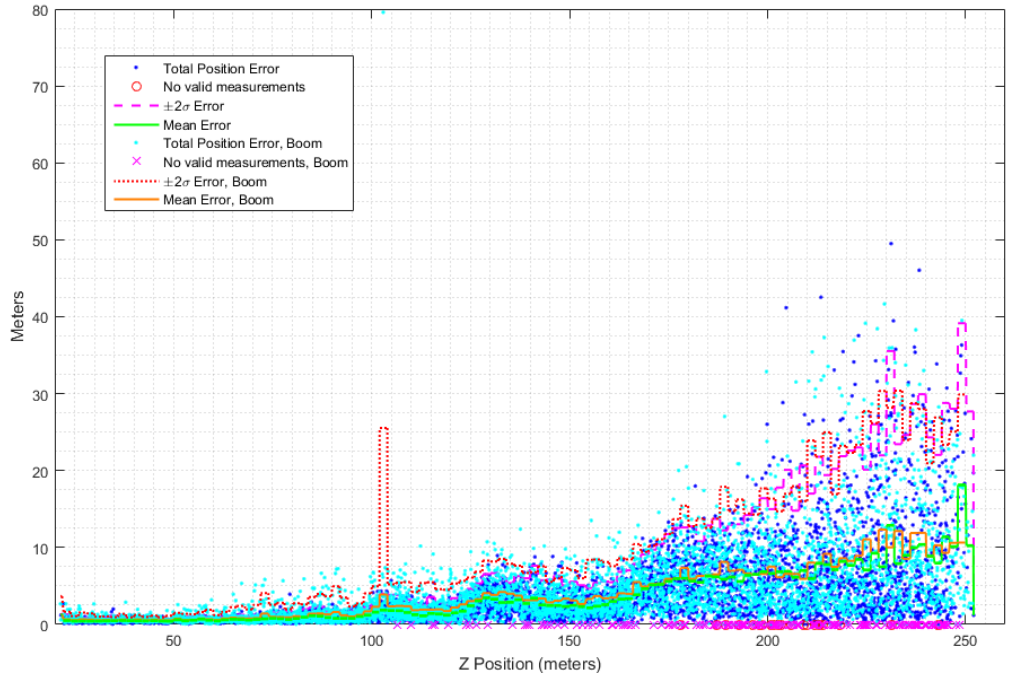


Figure 88. Spherical Position Errors, Cases 1 and 3.

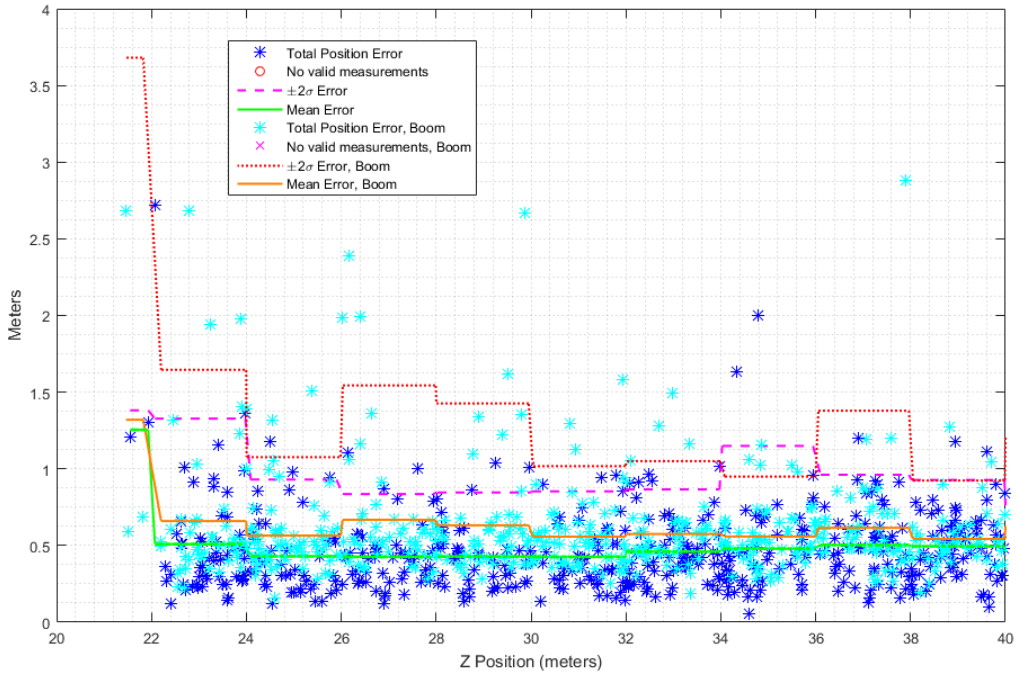


Figure 89. Spherical Position Errors, Cases 1 and 3, Zoom In.

Figure 90 shows the left and right camera images that correspond to the extreme outlier. In the right camera image, the C-12C receiver is almost completely obscured by the boom wings. With such large obscuration, the pixel matching component of the R7 algorithm was unable to correctly match receiver pixels, even in this idealized simulation environment.

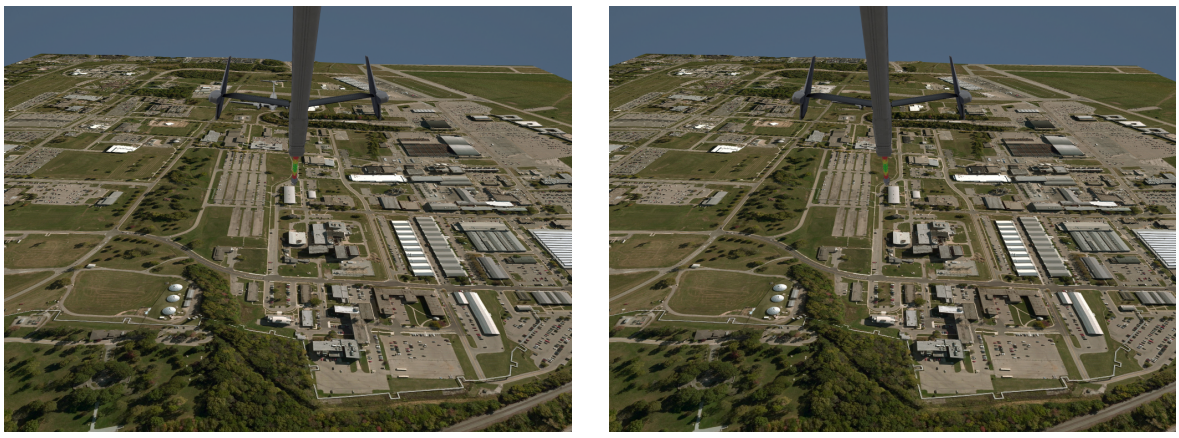


Figure 90. Simulation Images Captured on the Extreme Outlier.

Figure 91 shows the resultant point cloud and the boom filter. The "tanker frame" referenced in the figure is the v-frame. As can be seen in the figure, the point cloud that remained after filtering corresponds neither to the true position of the aircraft, which was at a range of over 100 meters from the origin of the tanker frame, nor to the boom. The remaining points were quite likely the consequence of incorrect pixel matching. For instance, it is possible that the algorithm matched pixels corresponding to the aircraft in the left image with pixels corresponding to the boom in the right image. Regardless of the precise cause of the mismatch, this outlier exemplifies the problems that a boom in the fields of view can present, even when using an effective boom filter.

This outlier could have been eliminated by changing the configuration of the R7 algorithm, such as setting a higher threshold for the minimum number of points in the filtered point cloud. As executed, the point cloud count threshold was three. If the threshold had been set to a greater number, such as twenty, then a "no valid measurements" result when have been returned and no estimate would have been provided.

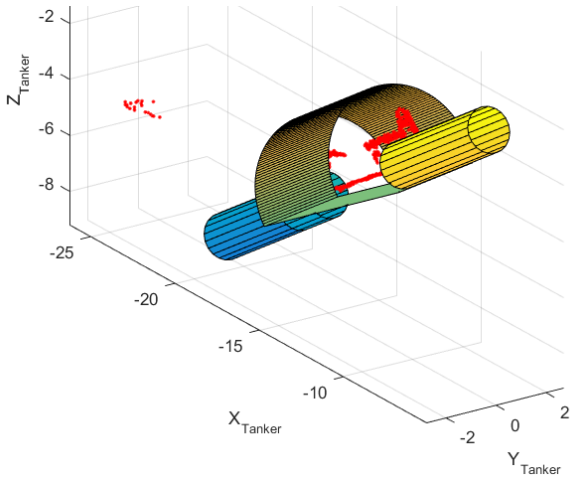


Figure 91. Boom Filtering Applied to the Outlier Point Cloud.

Figures 92 and 93 depict the position error components in the left camera frame. Table 21 presents the mean and standard deviations of the left camera frame component position errors for Cases 1 and 3. Results were qualitatively similar to Cases 1 and 2, with notable z-position bias growth with depth, but contained more outliers, and more cases in which no measurements were returned. Errors were once again fairly stable within 40 meters of depth.

With respect to attitude errors, results were qualitatively similar to Cases 1 and 2, as shown in Figures 94 and 95. Table 21 presents the mean and standard deviations of the tanker frame attitude errors for Cases 1 and 3. From the figure, a strong pitch bias manifested with depth. Additionally, standard deviations were high throughout the depth ranges, including within 40 meters. Roll and pitch error standard deviations were notably higher than that of the yaw error. Mean errors were fairly close to zero within 40 meters of depth.

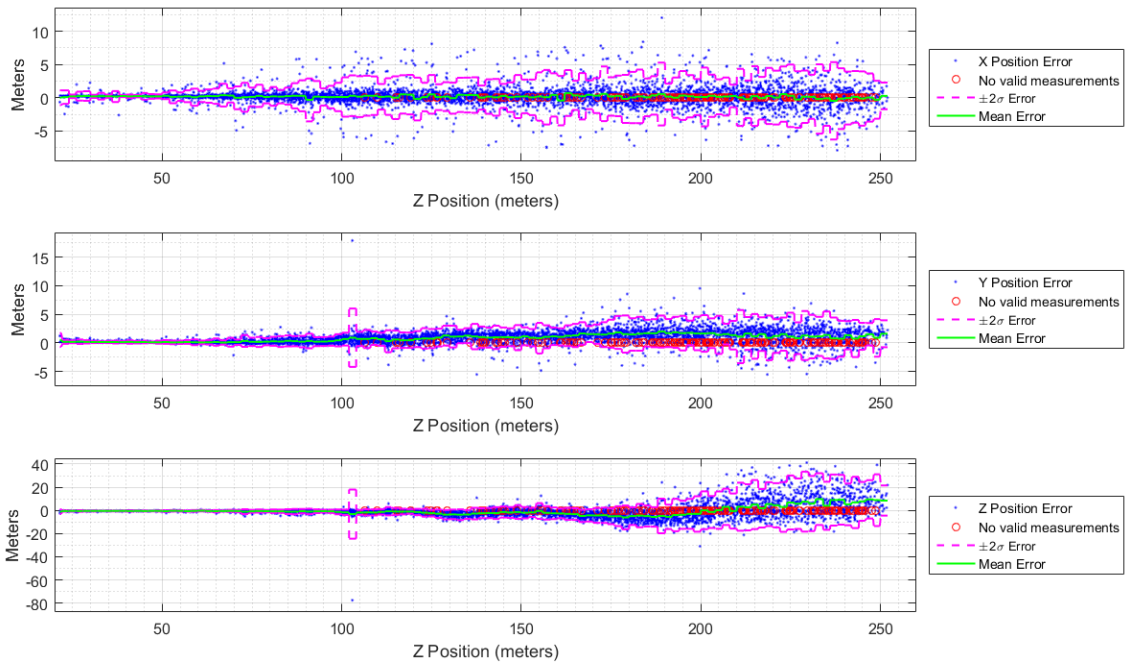


Figure 92. Position Errors, Case 3, Left Camera Frame.

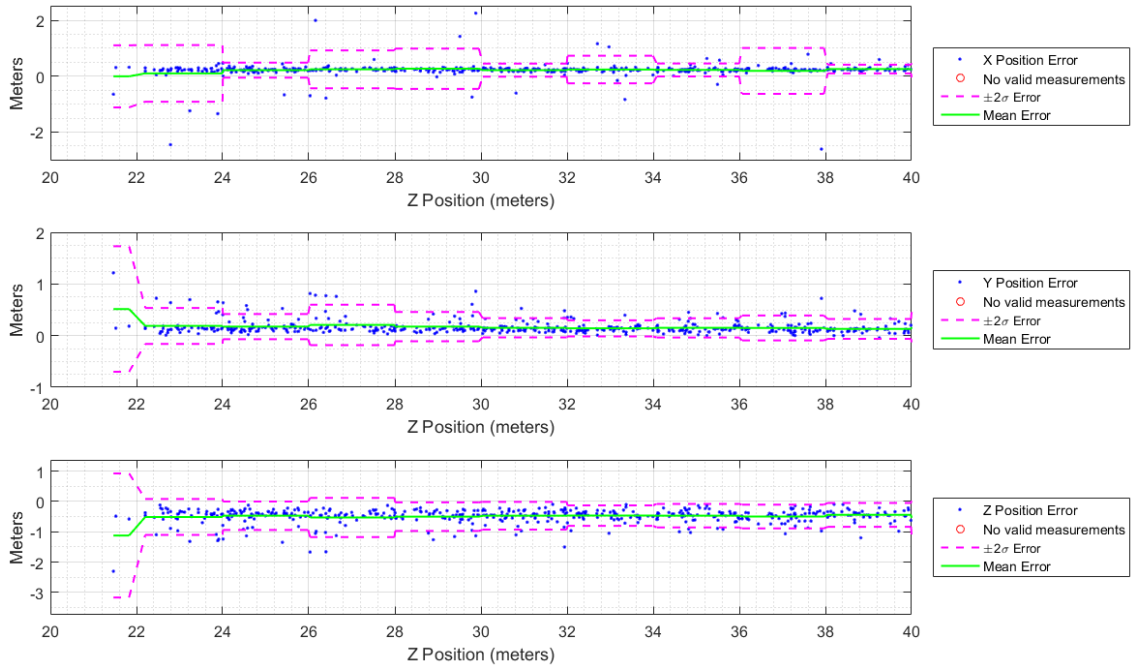


Figure 93. Position Errors, Case 3, Left Camera Frame, Zoom In.

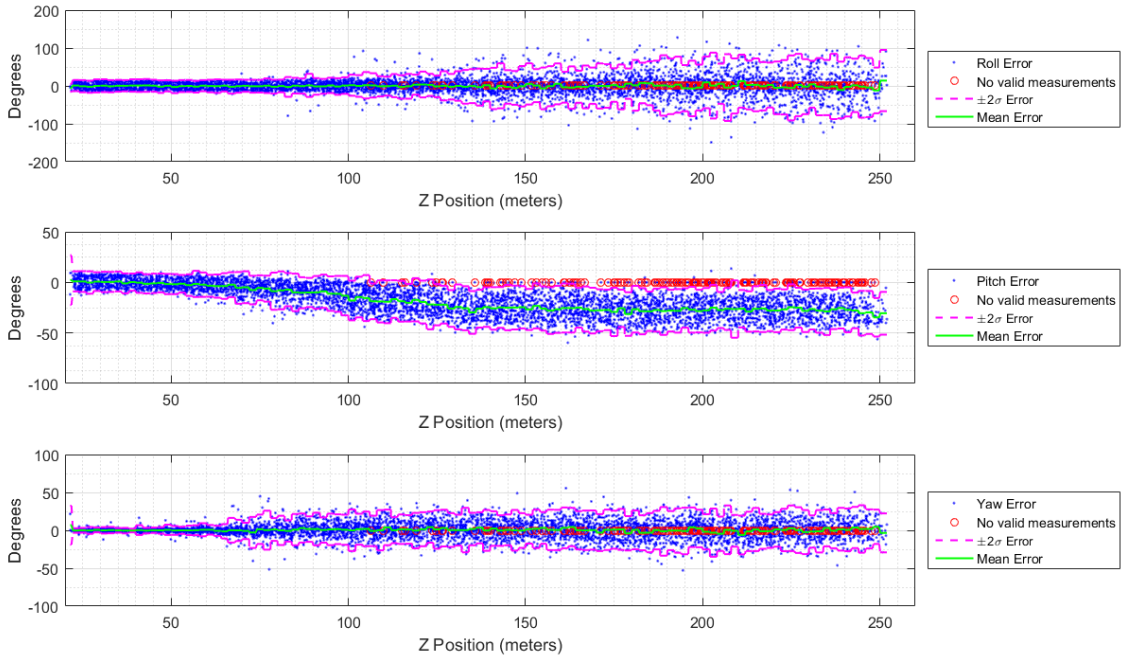


Figure 94. Attitude Errors, Case 3, Tanker Frame.

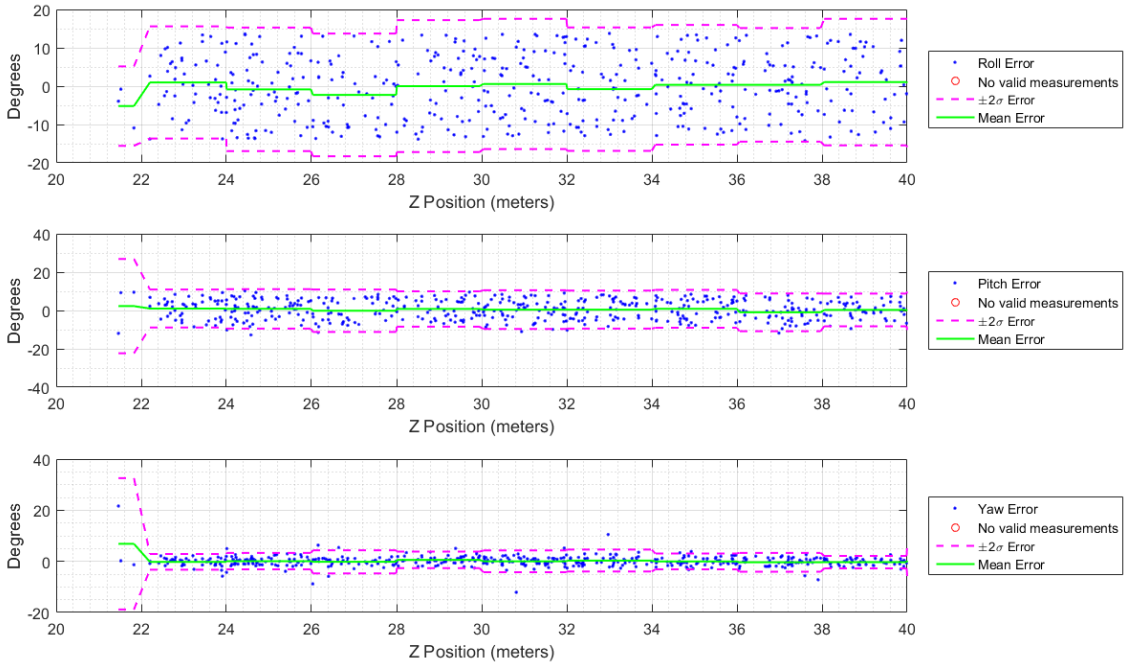


Figure 95. Attitude Errors, Case 3, Tanker Frame, Zoom In.

Two sample t-tests concluded that the Case 3 had significantly more spherical error and x-position error than Case 1 at confidence levels $\gg 99\%$. The 95% confidence intervals for the differences between the two cases were [0.125m, 0.049m] and [0.265m, 0.211m] for the spherical and x-position errors, respectively. Despite the fact that the mean of the x-position error in Case 1 was slightly negative, the results of this test indicate that Case 3 had a substantial positive x bias that was not present in Case 1. With respect to position error variances, two sample F-test results concluded that Case 3 had significantly greater spherical error variance, x-position error variance, and y-position error variance at confidence levels $\gg 99\%$. Interestingly, z-position error variances were not significantly different.

With respect to attitude errors, only the variance of the yaw estimates were found to be significantly different, with greater variance present in Case 3 than in Case 1. No significant difference was found between means or the variances of other attitude

components. These results indicate that the presence of a boom will increase the error and the uncertainty of R7 measurements even when using an effective boom filter.

Table 21. Position Errors in the Left Camera Frame and Attitude Errors in the Tanker Frame for Samples at Less than 40 Meters of Depth, Cases 1 and 3.

	Case 1		Case 3	
	Mean	**Standard Deviation**	**Mean**	**Standard Deviation**
Spherical Error (m)	0.512	0.264	0.599	0.336
X Position Error (m)	-0.020	0.103	0.218	0.286
Y Position Error (m)	0.156	0.112	0.164	0.136
Z Position Error (m)	-0.480	0.232	-0.485	0.246
Roll Error (deg)	-0.248	7.879	-0.102	8.010
Pitch Error (deg)	0.195	5.045	0.443	4.995
Yaw Error (deg)	0.117	1.676	0.040	2.057

4.2.2.5 Case 4: Boom in Field of View, Attitude Feeding.

For Case 4, the R7 algorithm was applied to the 6,000 observation images with the boom in the fields of view and with the true relative formation attitude being provided to the algorithm. In this form, the R7 algorithm provides only relative position measurements.

Figures 96 and 97 depict the spherical error of each observation image plotted against the receiver's z-position in the left camera frame. Table 22 presents the mean and standard deviations of the spherical position error for Cases 3 and 4. As can be seen in the figures, results were qualitatively very similar to Case 3 as both the mean and standard deviation of the error increased strongly with increasing depth from

the camera. Again, this effect was essentially negligible within 40 meters of depth. As with Case 3, there were 173 instantiations in which the R7 algorithm lacked a sufficient number of points to provide measurements and there were substantially more outliers than in Cases 1 and 2. The strong outlier observed in Case 3 remained when providing R7 with the true relative attitude of the receiver.

Similarly, Figures 98 and 99 show that component position errors in the left camera frame were qualitatively similar to Case 3. Table 22 presents the mean and standard deviations of the left camera frame component position errors for Cases 3 and 4.

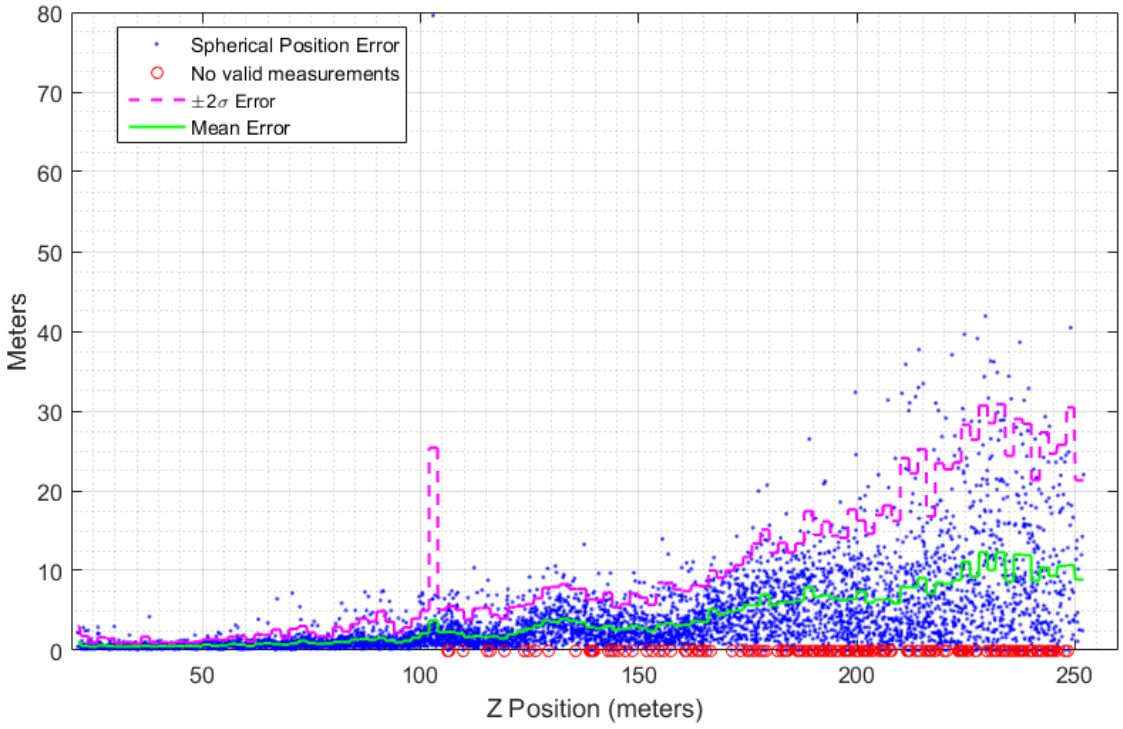


Figure 96. Spherical Position Errors, Case 4.

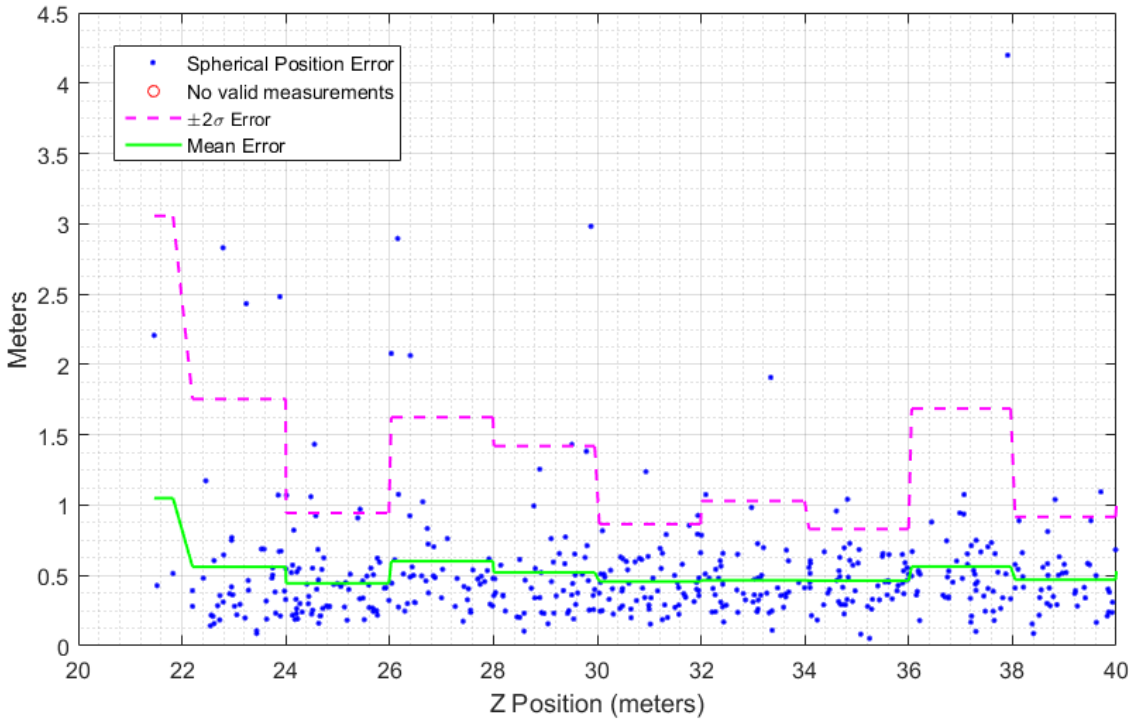


Figure 97. Spherical Position Errors, Case 4, Zoom In.

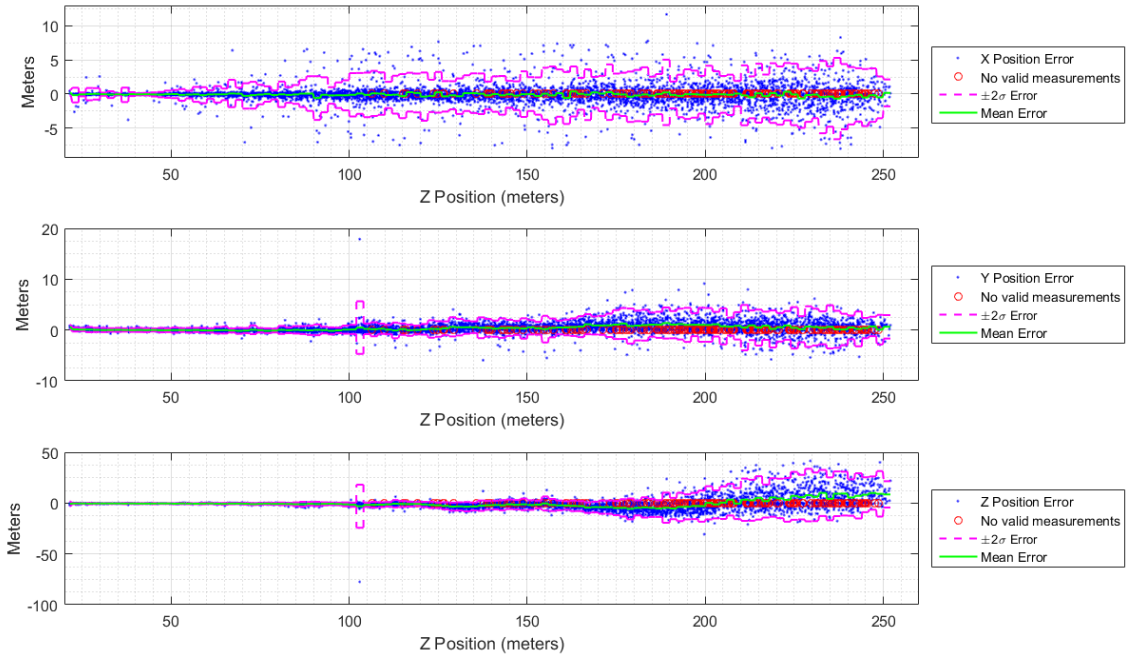


Figure 98. Position Errors, Case 4, Left Camera Frame.

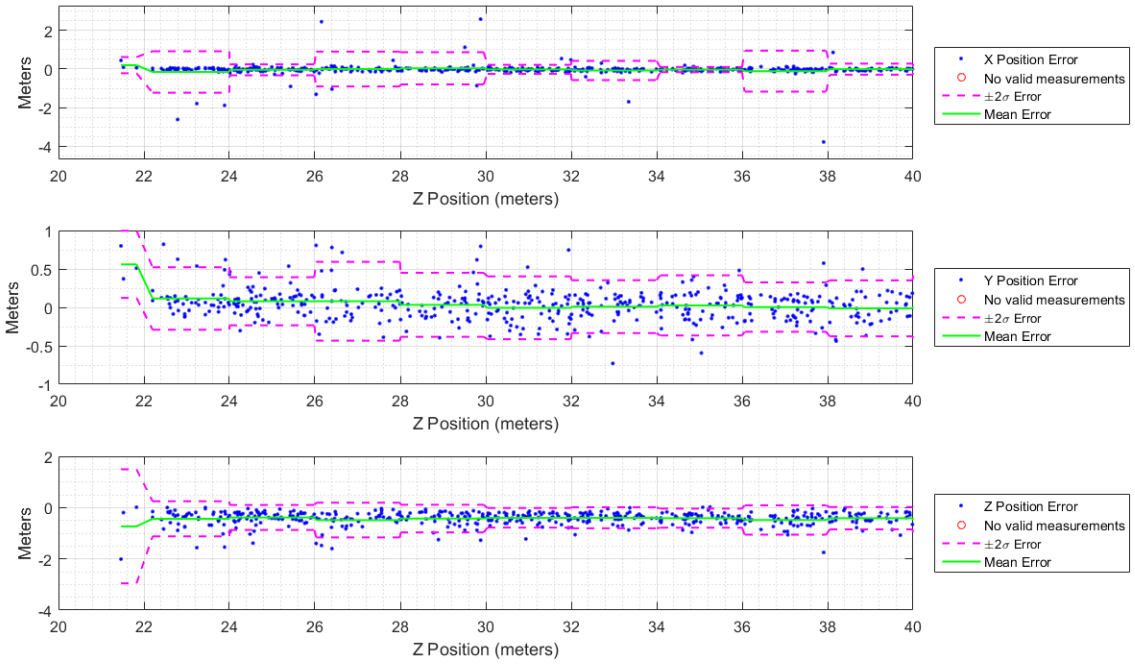


Figure 99. Position Errors, Case 4, Left Camera Frame, Zoom In.

Statistical comparisons between the results of Cases 3 and 4 revealed similar results to the the statistical comparisons conducted between Cases 1 and 2. Again, paired t-tests were used to compare means and two-sample F-tests were used to compare variances. In Case 3, when no attitude information was provided to R7, mean spherical error was greater, and mean x, y, and z-position errors were more strongly biased at confidence levels $\gg 99\%$. The 95% confidence intervals for the differences between the means were [0.080m, 0.110m], [0.247m, 0.274m], [0.111m, 0.131m], and [0.073m, 0.045m], respectively. Although the mean error in the x-position for Case 4 was slightly negative, all results indicate that Case 3 had stronger bias (positive in the x and y dimensions and negative in the z dimension) than Case 4.

Again, interestingly, with respect to variances, Case 3 was found to have less spherical error, x-position error, and y-position error variance than Case 4 at confidence levels greater than 99%. However, flight test data analyzed in Chapter VI showed the opposite result.

In terms of processing time, it took a total of 81.1 seconds to process all 6,000 observations in Case 3 and 77.5 seconds to do so in Case 4, a reduction of 4.3%. This result was consistent with the processing time reduction found when comparing Cases 1 and 2.

Table 22. Position Errors in the Left Camera Frame, Cases 3 and 4.

	Case 3		Case 4	
	Mean	**Standard Deviation**	**Mean**	**Standard Deviation**
Spherical Error (m)	0.599	0.336	0.503	0.389
X Position Error (m)	0.218	0.286	-0.042	0.331
Y Position Error (m)	0.164	0.136	0.039	0.201
Z Position Error (m)	-0.485	0.246	-0.427	0.265

4.2.2.6 Case 5: V5 Data Set.

For Case 5, the same observation set used to generate V5 results was also processed with the R7 algorithm. Specifically, R7 was applied to the 250 images processed by V5 with a pixel intensity threshold of 0.05. For this case, the R7 algorithm estimated both relative position and attitude and no boom was in the fields of view.

Figure 100 depicts the resultant spherical error. Within these close ranges, spanning 22 to 30 meters, the error of the R7 algorithm was shown to be fairly stable in Cases 1 through 4. During data analysis, plotting this data against depth revealed no growth trend over this small of a spread of ranges. Hence, the data in this section are plotted against sample number. As can be seen in the figure, the mean spherical error was very similar to that observed in Case 1 at ranges less than 40 meters. The mean and standard deviation of the spherical errors are tabulated in Table 23.

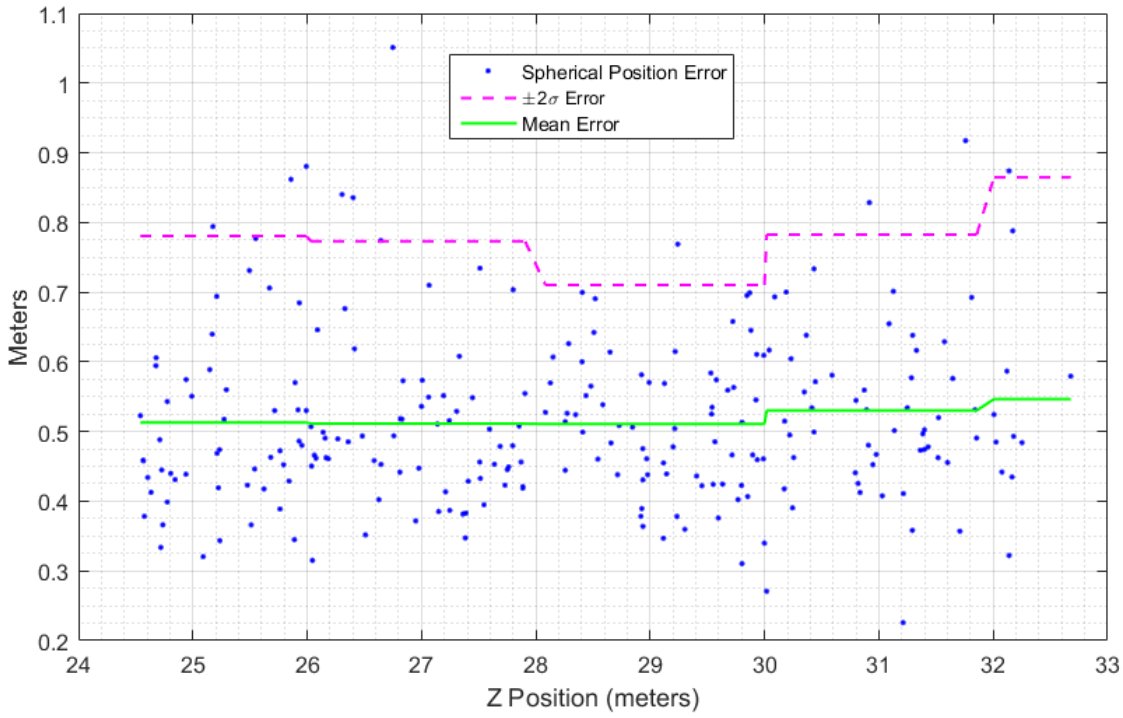


Figure 100. Spherical Error, R7 Applied to the V5 Simulation Data.

Figure 101 depicts the resultant position error components in the left camera frame. A significant z-axis bias and a slight y-axis bias were present in the data, as was seen with Case 1. The variance of the error was relatively tight in all each dimension, without a substantial number of outliers. Position error means and standard deviations are tabulated in Table 23.

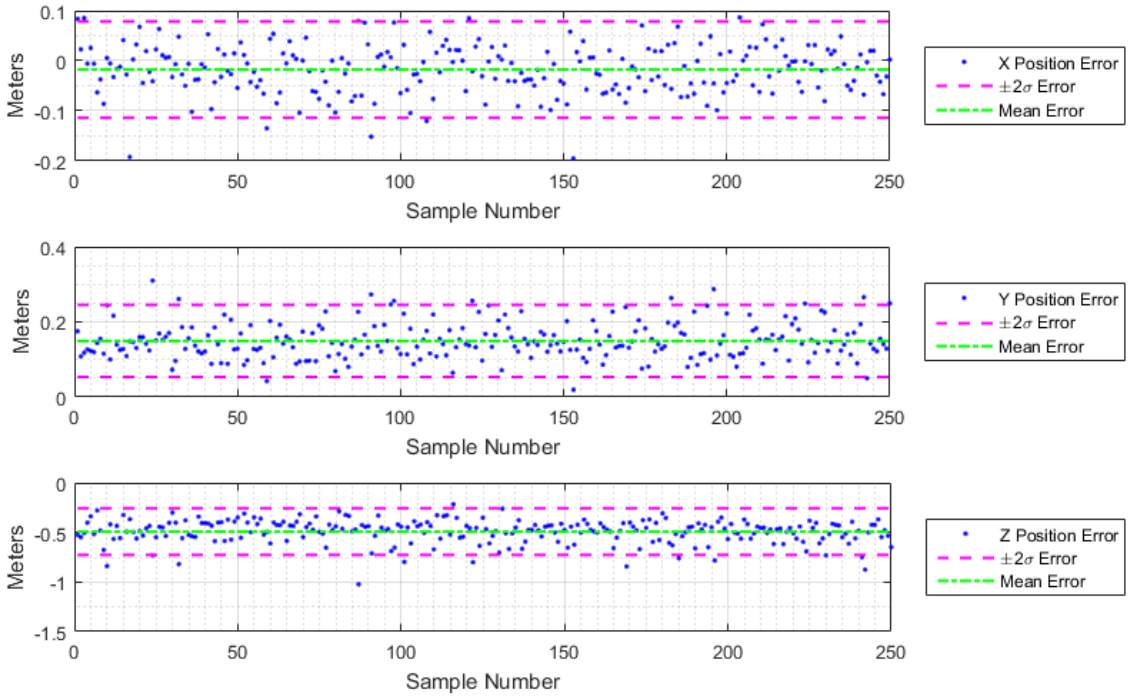


Figure 101. Position Errors, Left Camera Frame, R7 Applied to the V5 Simulation Data.

Figure 102 depicts the resultant attitude error components in the tanker frame. Attitude error means and standard deviations are tabulated in Table 23. Again, R7 attitude measurements proved to be approximately zero mean with relatively high magnitude standard deviations. Roll and pitch error standard deviations were notably higher than that of the yaw error as was observed in the other cases.

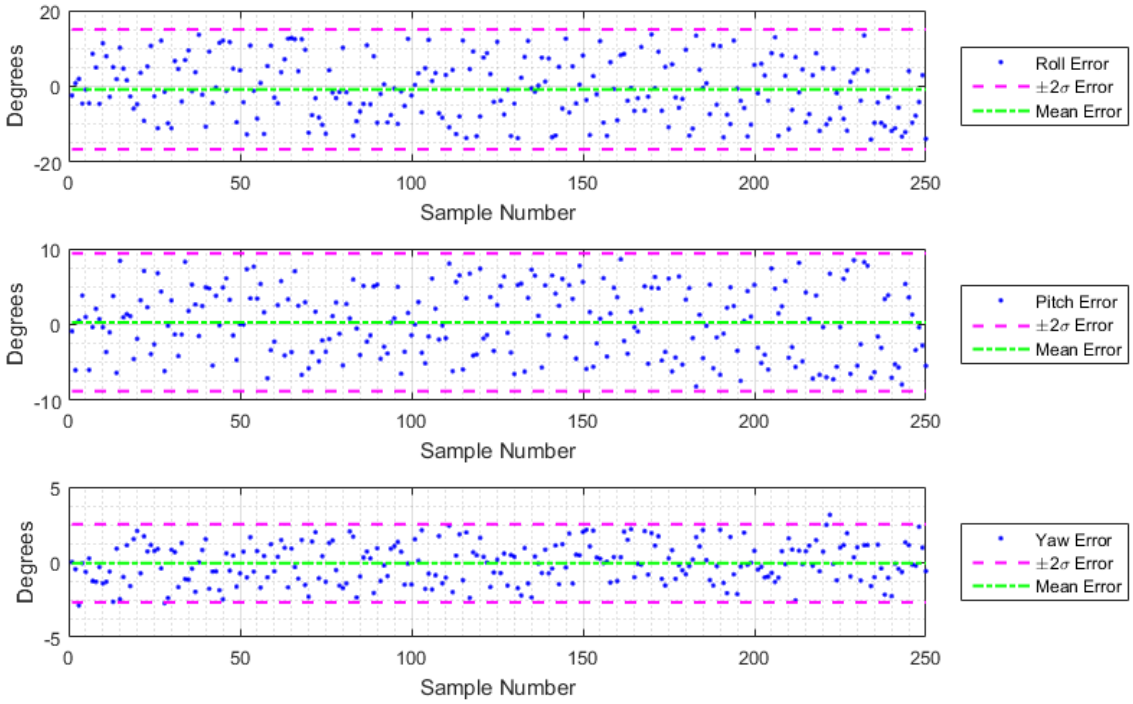


Figure 102. Attitude Errors, Tanker Frame, R7 Applied to the V5 Simulation Data.

Table 23. Position Errors in the Left Camera Frame, Attitude Errors in the Tanker Frame, R7 Applied to V5 Data Set.

	Mean	Standard Deviation
Spherical Position Error (m)	0.517	0.123
x Position Error (m)	-0.018	0.048
y Position Error (m)	0.149	0.048
z Position Error (m)	-0.491	0.118
Roll Error (deg)	-0.796	7.907
Pitch Error (deg)	0.292	4.539
Yaw Error (deg)	-0.074	1.300

4.2.2.7 R7 Simulation Processing Times.

No effort was made to optimize processing times in this thesis, but algorithm processing times were examined to assess how easily each algorithm could be implemented in a real-time system. For the R7 algorithm, image processing of 250 simulation image pairs in the 3DVW took approximately 60.18 seconds, an average of 0.241 seconds per image. The point clouds returned from the 3DVW were then processed with MATLAB®. In total, it took 7.412 seconds to process all 250 point clouds, an average of approximately 0.030 seconds per image. Image processing times represented easily the most time-intensive portion of the algorithm. Overall, the R7 algorithm required relatively little processing time in simulation.

4.3 R7 and V5 Comparison

The results from R7 simulation Case 5 were compared to the V5 results obtained with the configuration specified in Table 24. The exact same set of 250 observation image pairs was used to generate the V5 and R7 results.

Table 24. Configuration of the V5 Algorithm Used for Simulation Comparison with the R7 Algorithm.

Prior Probability Distribution	Likelihood Function	Pixel Intensity Threshold	Relative Position Estimator	Integrity Risk Level
Uniform	Gaussian $\mu = 0$ $\sigma = 0.1$	0.05	Composite	0.05

Figure 103 plots the spherical error present in the two algorithms for this data set. As can be seen in the figure, the R7 error had both a lower mean and less variance than the V5 error. A comparison of the V5 and R7 data in Table 25 shows

that in terms of spherical error V5 had both a larger mean and a higher variance. A paired t-test and two-sample F-test revealed that these differences were statistically significant at a confidence level $\gg 99\%$. The 95% confidence interval for the difference in mean spherical error was [0.312m, 0.446m] more spherical error when using V5. The 95% confidence interval for the ratio of the V5 to R7 spherical error variance was [14.3, 23.4].

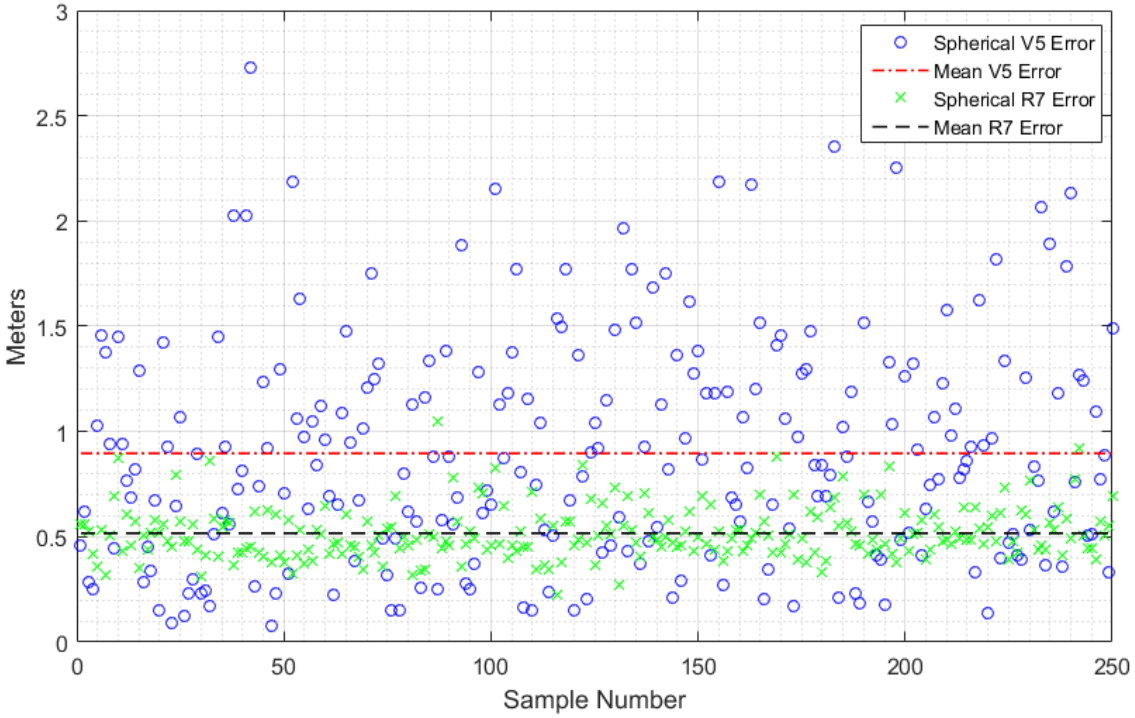


Figure 103. Comparison of R7 and V5 Spherical Errors.

However, a comparison of the V5 and R7 data in Table 25, also reveals that R7 was a more biased estimator in the y and z-dimensions than V5. Paired t-tests revealed that R7 had more bias in these dimensions at confidence levels $\gg 99\%$. The 95% confidence intervals for the difference in means were [0.164m, 0.119m] and [0.178m, 0.410m] in the y and z-dimensions, respectively. When using the attitude fed version of the R7 algorithm, the differences in the mean y and z-component errors were not statistically significant.

Two-sample F-tests revealed that the V5 algorithm had higher levels of error variance in all dimensions at confidence levels $\gg 99\%$. The 95% confidence intervals for the ratios of the V5 error variance to R7 error variance were [47.2, 77.7], [10.3, 16.9], [48.9, 80.4] in the x, y, and z-dimensions, respectively. Overall, this comparison indicates that spherical error for the R7 algorithm is likely lower with substantially less error variance than V5 spherical error. Additionally, the results indicate that R7 estimates may be more prone to bias in certain dimensions, especially depth, but have substantially less error variance even in more biased dimensions. With respect to estimate errors, the substantially smaller error variances in the R7 algorithm are the most significant result. The large variance differences help to explain why the R7 algorithm had significantly less mean spherical error than the V5 algorithm.

Table 25. Comparison of V5 and R7 Position Errors, Left Camera Frame.

	V5		**R7**	
	Mean	**Standard Deviation**	**Mean**	**Standard Deviation**
Spherical Error (m)	0.896	0.527	0.517	0.123
X Position Error (m)	0.004	0.374	-0.018	0.048
Y Position Error (m)	0.008	0.174	0.149	0.048
Z Position Error (m)	-0.197	0.935	-0.491	0.118

Average image processing time for the V5 and R7 algorithms were approximately 0.40 seconds and 0.24 seconds, respectively. The image processing routines were similar, but utilized different software. Hence, the difference in achievable image processing times is likely negligible. Both these image processing times were inflated by the need to save data to file that would not be required in a real-time implementation.

Figure 104 compares the time required to run each algorithm on pre-processed imagery. Image processing times are not included in the plot. No effort to optimize speed was attempted in this research effort. However, as can be seen in the figure, as currently implemented the R7 algorithm is three orders of magnitude faster than the V5 algorithm when operating on pre-processed imagery. These long V5 processing times represent a significant challenge to use in a real-time system.

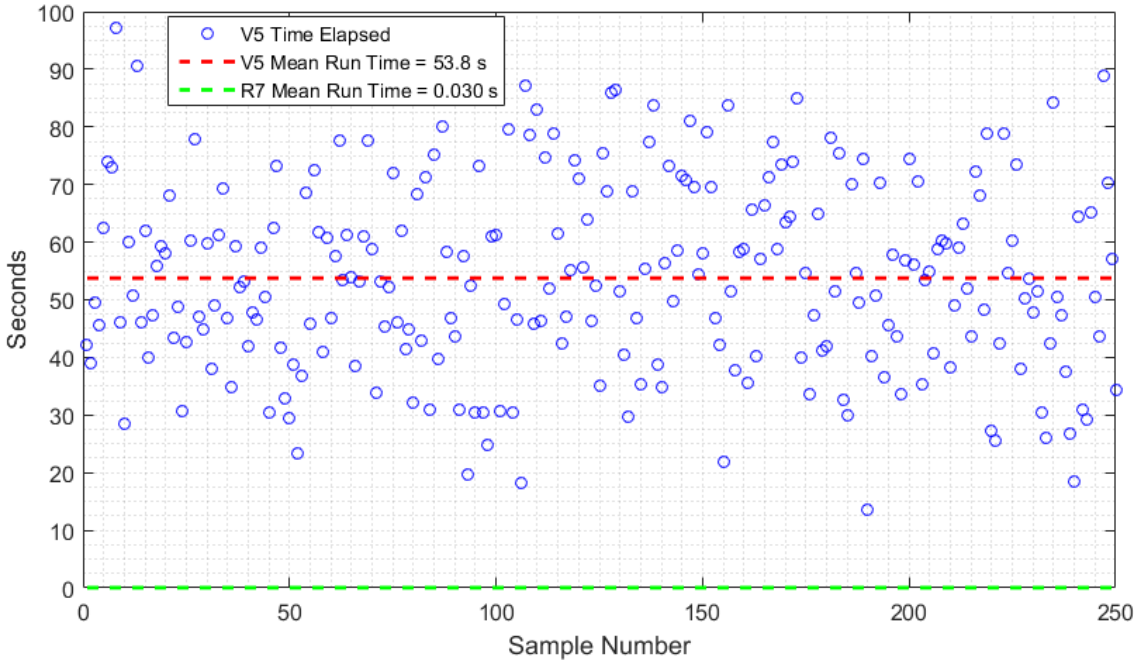


Figure 104. Comparison of R7 and V5 Processing Times.

V. Flight Test Hardware and Methodology

This chapter describes the flight test hardware and methodology used to acquire data for this thesis. First, Section 5.1 summarizes the flight test effort. Second, Section 5.2 describes the flight test equipment. Third, Section 5.3 describes flight test data collection methodology. Finally, Section 5.4 describes data reduction and analysis techniques used specifically on flight test data.

Flight test data was collected as part of the Have Vision test management project at the USAF Test Pilot School. This chapter reproduces much of the information presented by Stuart (the author of this thesis) et al in the test information memorandum prepared after completion of the Have Vision test program [7]. More details on hardware components, aircraft modifications (to include system specification sheets and design drawings), and flight test methodology can be found in the Have Vision test information memorandum.

5.1 Have Vision Test Management Project Summary

Testing was conducted from 5 to 14 September 2017 in the Restricted Area 2508 (R-2508) complex and surrounding airspace. Testing consisted of ground data collection and six flight test missions totaling 11 sorties and 17.3 flying hours. Four of the missions were flown with two C-12Cs, tail numbers 76-00161 and 76-00158, flying in formation with tail number 76-00161 flying as the pseudo-tanker and tail number 76-00158 flying as the pseudo-receiver. One of the missions used a T-38C, tail number 65-10403, as a pseudo-receiver. The sixth mission was used for troubleshooting systems issues and did not collect images of a pseudo-receiver.

Testing was accomplished by mounting two stereo camera pairs on the C-12C pseudo-tanker. The first camera pair captured images in the visible spectrum. This

stereo pair is referred to as the EO stereo camera pair in this thesis. The second camera pair captured images in the long-wave infrared (LWIR) spectrum. This stereo pair is referred to as the LWIR stereo camera pair in this thesis. These stereo camera pairs were used to capture images of the pseudo-receiver. The two aircraft flew in formation positions typical of those used in AR at altitudes ranging from approximately 10,000 to 15,000 feet above ground level (AGL), while at airspeeds between 160 to 220 knots indicated airspeed (KIAS). GPS-Aided, Lite (GLITE) time, space, and position information (TSPI) systems captured position and attitude truth data on both aircraft.

Unfortunately, technical difficulties with the LWIR cameras significantly constrained the collection of LWIR image pairs, but hundreds of thousands of EO image pairs were successfully captured. Overall, 415,800 image pairs, totaling 4.25 terabytes of data, were collected to support research efforts in this thesis and in the future. These images feature both C-12C and T-38C pseudo-receivers over various backgrounds and at various altitudes. Each flight data set included a set of checkerboard images captured during pre-flight ground operations. Additionally, truth data, captured from the GLITE TSPI systems, was recorded during each formation test mission. During post-processing, 221 EO image pairs of a C-12C in the contact position over a mountainous background were used to analyze both the V5 and R7 algorithms. The results from this analysis are presented in Chapter VI.

5.2 Flight Test Equipment

The flight test vision system included two stereo camera pairs—a LWIR stereo camera pair and an EO stereo camera pair, each mounted on a fabricated lateral support on the belly of the C-12C pseudo-tanker. Figures 105 and 106 show the camera system as installed on the C-12C pseudo-tanker. As in simulation, the cameras

within a stereo pair were separated by approximately 0.5 meters and were oriented such that they looked aft of the aircraft at an approximate 25° down look angle. This configuration was intended to be representative of the camera configuration used on the KC-46, with the cameras mounted approximately two feet forward of the hypothetical boom attachment point, as illustrated in Figure 107. The two sensor pairs (EO and LWIR) covered the same spectrum as the stereo cameras found on the KC-46, offered additional redundancy, and were intended to enable analysis of the V5 and R7 algorithms in two different spectra.



Figure 105. Have Vision Camera Configuration, View from Aft of Aircraft.

The external camera mount developed for the C-12C pseudo-tanker was connected to an orifice in the aircraft's external structure previously used for other external modifications. The orifice was located approximately below the copilot's seat, 18 inches right of centerline. Structural loads and aerodynamic analysis identified two possible problems resulting from the external modification: excessive static loads on the mount at the aircraft's maximum allowable airspeed, and objectionable stall characteristics. Ultimately, pseudo-tanker airspeed was restricted to 230 KIAS below 10,000 ft pressure altitude (PA), with standard airspeed limits (260 KIAS) applying above 10,000 ft PA. Additionally, power-off stall characteristics were evaluated to



Figure 106. Have Vision Camera Configuration, View from Side of Aircraft.

identify a safe landing configuration. As a result, a no flap configuration was used for all pseudo-tanker landings.

5.2.1 LWIR Cameras.

The LWIR cameras were Atom™ 1024G models with a 16.4 millimeter fixed focus lens and a Gigabit Ethernet (GigE) interface produced by Sofradir®. These cameras operated in the 8-to-14 micron spectral range and produced 1294 x 768 pixel images at 30 Hertz. The LWIR cameras had a 56.0° HFOV and 43.5° VFOV. Figure 108 shows the cameras installed on the pseudo-tanker.

5.2.2 EO Cameras.

The EO cameras were Prosilica GT1290C models with a Kowa 4.4 millimeter fixed focus lens and a GigE interface produced by Allied Vision®. These cameras operated in the 400-to-700 nanometer spectral range with a 55.5° HFOV and a 43.0°

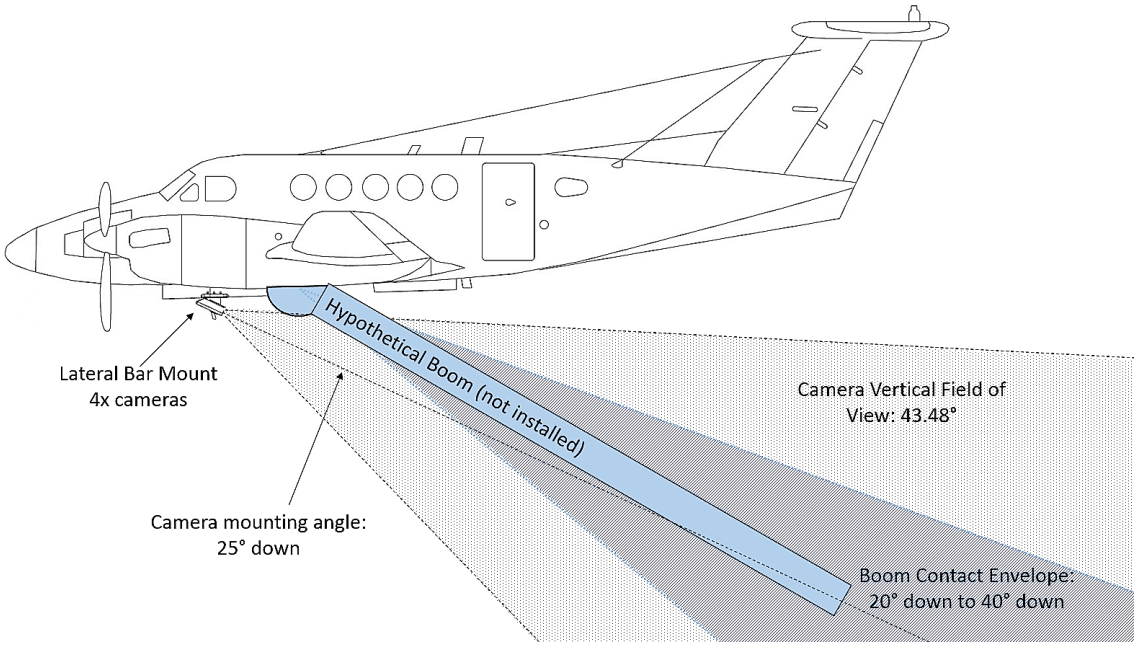


Figure 107. Stereo Camera Illustration and Field of View on Pseudo-Tanker Aircraft.

VFOV. This range of operation is essentially the visible spectrum. The cameras produced images at 30 Hertz; image sizes were 1280-by-960 pixels. Figure 108 shows the cameras installed on the pseudo-tanker.

5.2.3 Data Acquisition Computer.

The shutters on the Atom™ LWIR and Prosilica EO cameras were triggered at a rate of 30 Hertz by a data acquisition computer via an intermediate triggering board. The data acquisition computer, including applicable software, was built by Dr. Scott Nykl of AFIT specifically for use in this flight test. The data acquisition computer had an Intel® Xeon® dual central processing unit with 32 gigabytes of random access memory, an NVIDIA Titan X graphical processing unit, with a 4 terabyte peripheral component interconnect express solid state drive.

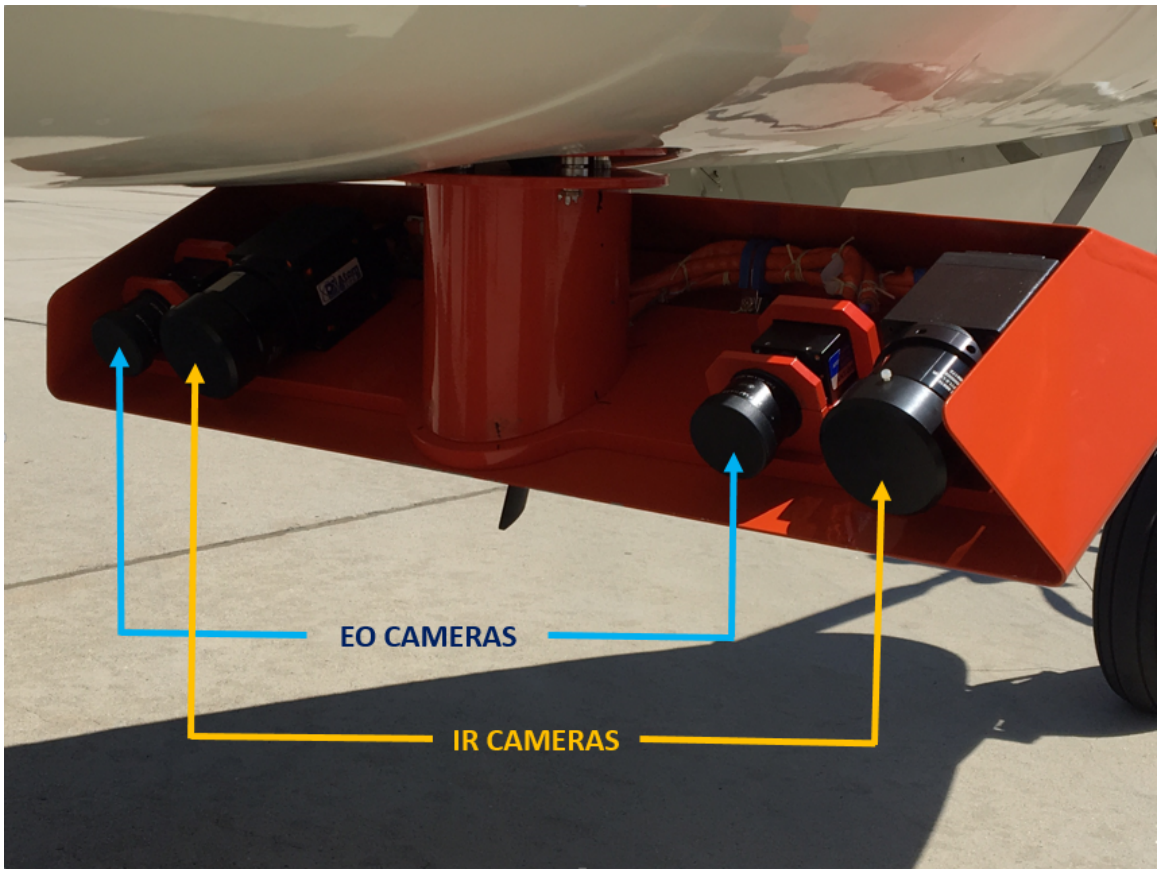


Figure 108. Close Up of the Have Vision Stereo Camera System.

5.2.4 Geodetics Geo-RelNav® System.

Truth data systems were necessary to determine the error in position and/or attitude estimates returned by the V5 and R7 algorithms. Two sources of truth data were installed to ensure redundancy. First, a Geodetics Geo-RelNav® system was installed on both C-12C aircraft. The system was designed to operate via a communications link that connected a GPS-INS installed on the pseudo-receiver to a GPS-INS installed on the pseudo-tanker. Fusing the data from the two GPS-INS systems in a Kalman filter, the Geo-RelNav® system was intended to provide a time stamped measurement of the relative position and attitude of the pseudo-receiver in

the pseudo-tanker body frame. However, this system did not achieve connectivity during flight test and no useful data was captured from it.

5.2.5 GLITE TSPI System.

As a redundant source of truth data, a GLITE TSPI system was installed on the C-12C pseudo-tanker and each pseudo-receiver aircraft. The specific configuration flown on the C-12Cs was configuration C2B. This system returned absolute position and attitude measurements from each aircraft. Position accuracy was 1.5 feet (approximately 0.46 meters) circular error probable (CEP). Based on accuracy specifications, 95% of all system errors should fall within this CEP; hence this figure could also be reported as CEP95. Attitude CEP95 uncertainties were 0.10° in roll, pitch, and yaw attitude. All measurements were time-stamped with GPS system time.

A different GLITE TSPI variant, configuration C1, was flown on the T-38C pseudo-receiver. This variant had the same position accuracy specifications as the C2B configuration, but did not include attitude measurements. Collection of attitude data occurred separately using a previously installed data acquisition system (DAS) with an accuracy of 0.05° in roll, pitch, and yaw. All DAS measurements were time-stamped with GPS system time.

For all GLITE TSPI systems, installation and single-ship data reduction were performed by 412th Range Squadron personnel at Edwards Air Force Base, California. DAS data was reduced by USAF Test Pilot School personnel. The method used to obtain differential truth data with these absolute measurements is described in Section 5.4.2.

5.2.6 Test Aircraft.

Flight testing required flying two aircraft in relative positions similar to those used during AR. As described above, the pseudo-tanker aircraft was a uniquely-modified C-12C (tail number 76-00158). The pseudo-receiver aircraft was either a C-12C (tail number 76-00161) or T-38C (tail number 64-10403).

The C-12C is a light transport aircraft primarily used operationally for executive transport. The aircraft was flown by two pilots and, in the test configuration, had seats for four additional occupants. Both C-12Cs used during the test incorporated the enhanced performance improved capability (EPIC) modification. Twin turboprop engines are mounted on the wings and, with the EPIC modification, feature four-bladed, swept propellers.

For one sortie, a T-38C Talon was used as the pseudo-receiver aircraft instead of the C-12C. The T-38C is a two-place twin-turbojet supersonic trainer. The fuselage is an area-rule (Coke bottle) shape with moderately swept-back wings and empennage. Data from the T-38 flight is not analyzed in this thesis.

5.2.7 Have Vision System Block Diagram.

The hardware and software components of the Have Vision system are depicted in block diagram form in Figure 109. After the data acquisition computer triggered the cameras, the cameras captured images and sent these images back to the computer for storage. The test conductor (TC) on board the aircraft accessed the data acquisition computer and toggled between recording and non-recording camera modes. Additionally, the TC had real-time access to images from all four cameras. After images reached the computer, the computer applied a time stamp to each image. The time stamp was referenced to the GPS clock. These images were then fed to the R7 and V5 algorithms post-flight. Additional inputs to the R7 and V5 algorithms included

the camera calibration results obtained from using checkerboard images. The outputs from the R7 and V5 algorithms were compared to the truth data captured from the GLITE TSPI systems in order to estimate algorithm errors.

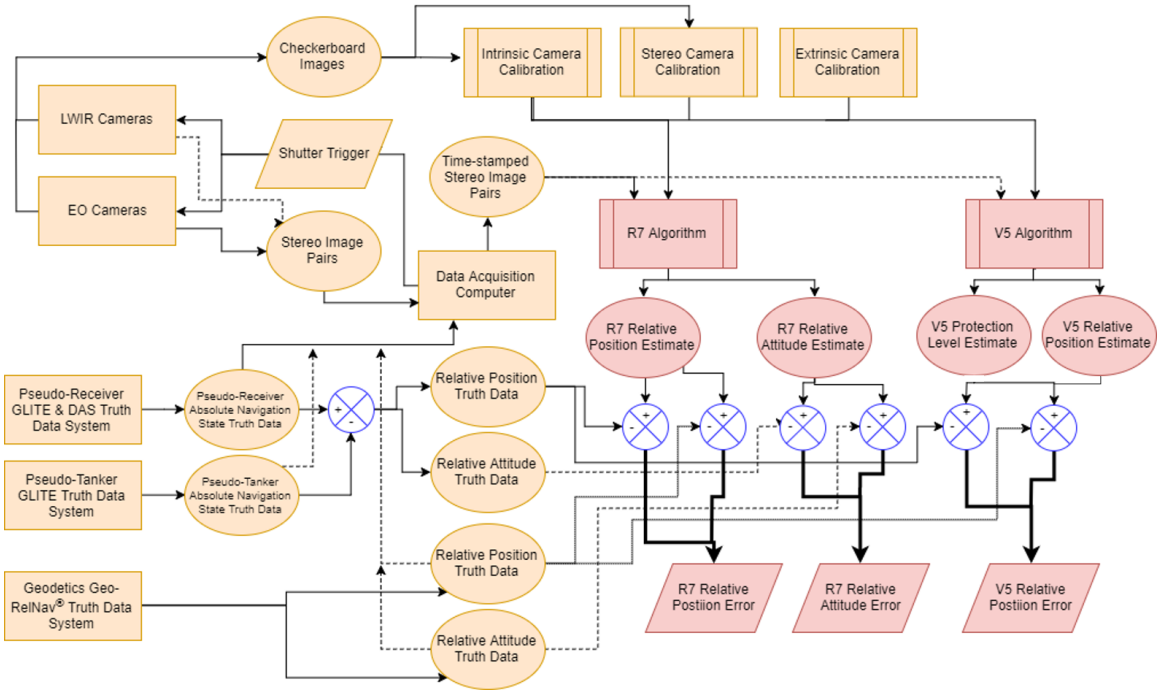


Figure 109. Block Diagram of Flight Test Hardware and Data Processing Flow.

5.2.8 Hardware Limitations and Constraints.

An insufficient quantity of LWIR image pairs were collected for algorithm characterization in the LWIR spectrum. One of the LWIR cameras became unresponsive in flight and did not accept trigger pulses throughout the test effort. Analysis revealed this was most likely due to faulty wiring. Interestingly, both LWIR cameras demonstrated proper operation on the ground including while on engine power and during taxi. During these ground operations, the cameras were successfully pulsed at 30 Hertz and time-stamped images were collected. However, the dropouts encountered during flight were not recoverable, and the symptoms remained following

in-flight restarts of the data acquisition computer. Due to these issues, only approximately 300 LWIR image pairs were collected with a pseudo-receiver aircraft in the camera fields of view. Unfortunately, these image pairs were mostly captured with the pseudo-receiver in the pre-contact position, outside the range of interest for algorithm analysis in this thesis. Hence, data analysis for this effort was restricted to the image pairs collected in the visible spectrum.

Due to unknown causes, the Geodetics Geo-RelNav® system did not establish connectivity between test aircraft during execution. Possible causes include incorrect antenna installation, faulty wiring, or other hardware defects. As a result, the system did not provide relative position and attitude truth data for analysis. Resource constraints dictated that troubleshooting efforts were focused on addressing the issues with the LWIR cameras, rather than the connectivity issue with the Geo-RelNav® system. The GLITE TSPI systems installed in the C-12C and T-38C aircraft performed satisfactorily throughout the test effort, and provided all necessary parameters. However, advertised uncertainties in GLITE TSPI position data (0.46 meters) exceeded the expected uncertainties provided by the Geo-RelNav® system (less than 0.15 meters).

5.3 Flight Test Data Collection Methodology

C-12C tail number 76-00158 was flown as the pseudo-tanker. As outlined in Section 5.2, this aircraft was modified with a data collection computer, EO and LWIR stereo camera pairs, and a C2B GLITE TSPI system utilized for truth data collection. Intrinsic, stereo, and extrinsic camera calibrations were completed in a laboratory setting prior to the first test mission. This aircraft was flown in formation with a pseudo-receiver aircraft.

During flight tests, the pseudo-receiver aircraft was either another C-12C aircraft (tail number 76-00161) modified with the C2B GLITE TSPI system, or a T-38C (tail number 65-10403) equipped with a C1 GLITE TSPI system and a DAS. The formation was flown through various formation positions to emulate operational AR maneuvers. These maneuvers included closure from trail to contact, level and turning flight in contact, movement to observation positions, and practice breakaways. The stereo camera pairs were used to collect imagery of the pseudo-receiver aircraft at these various relative positions over various background environments at various altitudes.

During post-processing, a subset of the collected EO imagery was fed into the R7 and V5 algorithms. The algorithm position and/or attitude measurements were compared to collected position/attitude truth data to characterize algorithm performance. Due to hardware limitations and constraints outlined above, only data from the EO camera pair was available for analysis. The chosen EO imagery excerpt was taken during the fifth flight test with the C-12C pseudo-receiver in a simulated contact position. This data set captured includes three minutes and three seconds of images captured over mountainous terrain and is analyzed in Chapter VI.

5.3.1 Calibrations.

Boresighting was performed on all critical components to determine precise relative positions and orientations. These components included each GLITE system's IMU and each camera. Boresighting of the camera systems comprised the extrinsic camera calibrations. Additionally, camera calibrations were executed in order to provide the necessary inputs to both the R7 and V5 algorithms. The calibrations included an intrinsic camera calibration completed on each camera and a stereo camera calibration completed for each stereo camera pair.

5.3.1.1 Extrinsic Camera Calibrations and System Boresighting.

Boresighting was accomplished by the USAF Test Pilot School instrumentation team using a Platinum Series, six-foot, seven-axis FaroArm® system. System accuracy was considered to be 0.001 inches. Using this system, the USAF Test Pilot School instrumentation team directly measured the position of the origin of each frame of interest with respect to the origin of the GeoRelNav® IMU. This measurement was expressed in the GeoRelNav® IMU frame. Additionally, one inch long unit vectors for each frame of interest were defined using this system. The location of the tips of each of these unit vectors (x, y, and z) were also measured in the GeoRelNav® IMU frame. Frames of interest include the aircraft body frame, the GLITE TSPI system frame, the left and right LWIR camera frames, and the left and right EO camera frames.

These measurements enabled the computation of vectors and DCMs expressing the relative position and orientation of any system component with respect to another. This process is outlined in Section 5.4.1.

5.3.1.2 Intrinsic Camera Calibrations.

Intrinsic camera calibrations were performed individually on each camera to determine camera distortion parameters and camera calibration matrices as described in Section 2.3.4. This was accomplished by capturing still images of a checkerboard of precisely known dimensions and processing them through the Camera Calibration Toolbox for MATLAB® [34]. The images used for algorithm analysis were captured prior to flight test with the cameras installed on the mounting tray. Subsequently, this mounting tray was installed onto the underside of the pseudo-tanker aircraft. Both the EO and LWIR camera pairs captured images. The checkerboard, as depicted in Figure 110, was made of polished steel and white paint that contrasted in both the

visible and LWIR spectrums after brief exposure to heat (sunlight). This checkerboard measured 39 centimeters by 36 centimeters and was comprised of squares with edges 50.8 millimeters long. Checkerboard pattern dimensions were measured within 0.1 millimeters of precision.

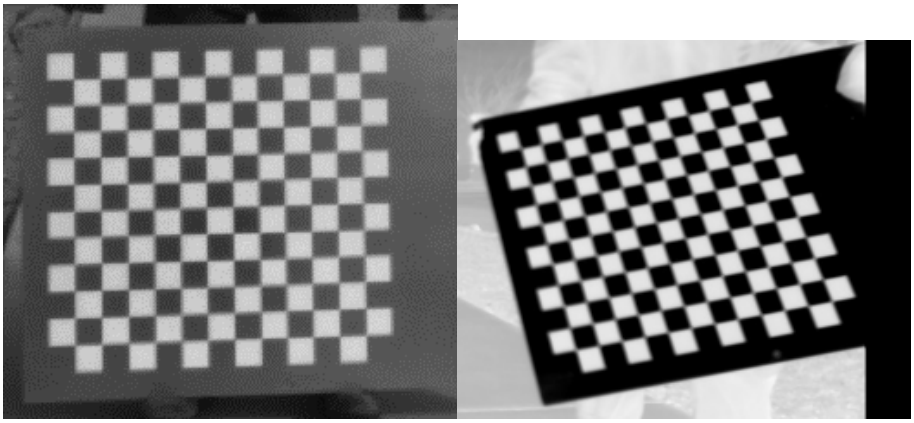


Figure 110. From Left to Right, EO and LWIR Images of the Checkerboard Used for Camera Calibrations.

The steel checkerboard reflected LWIR and EO energy from behind the cameras which caused difficulties in corner extraction during processing in MATLAB®. The software had difficulty distinguishing corners that were washed out by these reflections. Careful manipulation of the checkerboard was required to capture high contrast images by ensuring that the clutter reflected in the checkerboard was minimized. This was accomplished by gathering images that reflected areas of uniform temperature, such as the sky, as shown in Figure 111. The images in this figure were used to conduct the intrinsic and stereo camera calibrations.

Additionally, images of a smaller steel checkerboard were captured during ground operations just prior to each flight with the camera system completely installed on the pseudo-tanker aircraft. These images were retained as part of the flight test data set. The edges on the smaller checkerboard were 30.0 millimeters long. The smaller checkerboard measured 21 centimeters by 27 centimeters.

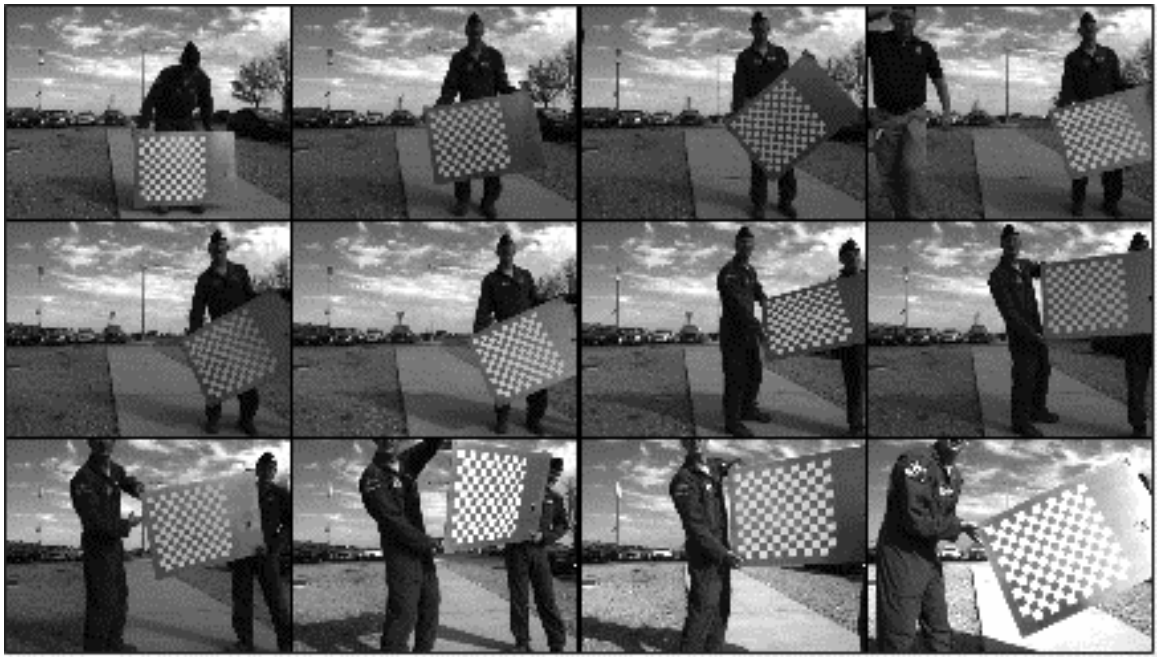


Figure 111. Images from EO Cameras Used in Intrinsic and Stereo Camera Calibrations. Images Were Down-Sampled to Create this Figure.

5.3.1.3 Stereo Camera Calibrations.

Stereo camera calibrations were performed on each camera pair (EO and LWIR) to determine relative position and orientation of the two cameras in each pair. Again, this was accomplished using Bouguet's Camera Calibration Toolbox for MATLAB® [34].

The stereo camera calibrations used the same images as those used during intrinsic calibrations. Results from the intrinsic camera calibrations were used to seed the stereo calibration algorithm with respect to distortion coefficients and camera calibration matrices. The stereo camera calibration returned new estimates for the intrinsic calibration results (distortion coefficients and camera calibration matrices) as well as estimates of the relative position and orientation of the two cameras in a given stereo pair. The intrinsic results produced by the stereo camera calibrations were compared to the initially input intrinsic calibrations for each individual camera

to ensure accuracy. Appendix A contains the results from the stereo and extrinsic camera calibrations. The Have Vision test information memorandum also contains the results for the intrinsic camera calibrations [7].

5.3.2 Simulated Aerial Refueling In-Flight Methodology.

The formation was flown in a manner representative of operational AR procedures. Formation positions for the pseudo-receiver aircraft were defined based on the camera system location on the pseudo-tanker aircraft. These relative positions attempted to mimic the geometry observed by the cameras on the KC-46 tanker aircraft. The defined positions are shown in Figure 112 and Figure 113. The red area in Figure 112 was a no-fly area due to safety considerations. Pseudo-receiver pilots maintained the proper contact window by visually aligning the leading edge of the pseudo-tanker aircrafts wing with the top of its propeller arc while also maintaining nose-tail separation between the two aircraft. Laterally, pseudo-receiver pilots centered their aircraft on the camera mount, which was offset to the right of the pseudo-tanker aircrafts centerline by roughly two feet. The view from the contact position is shown in Figure 114.

During each mission, maneuver blocks were flown to capture images of the pseudo-receiver aircraft throughout various operationally representative AR positions over various backgrounds. Two AR maneuver blocks were flown: a full AR maneuver block and an abbreviated AR maneuver block. Generally, the full AR maneuver block was utilized whenever feasible.

For the full AR maneuver block, the profile was initiated with the lead (pseudo-tanker) aircraft level at the test point altitude and airspeed (160 ± 10 KIAS with a C-12C target aircraft, 210 ± 10 KIAS with a T-38C target aircraft). The target (pseudo-receiver) aircraft was initially 500-1,000 feet below the test altitude at the

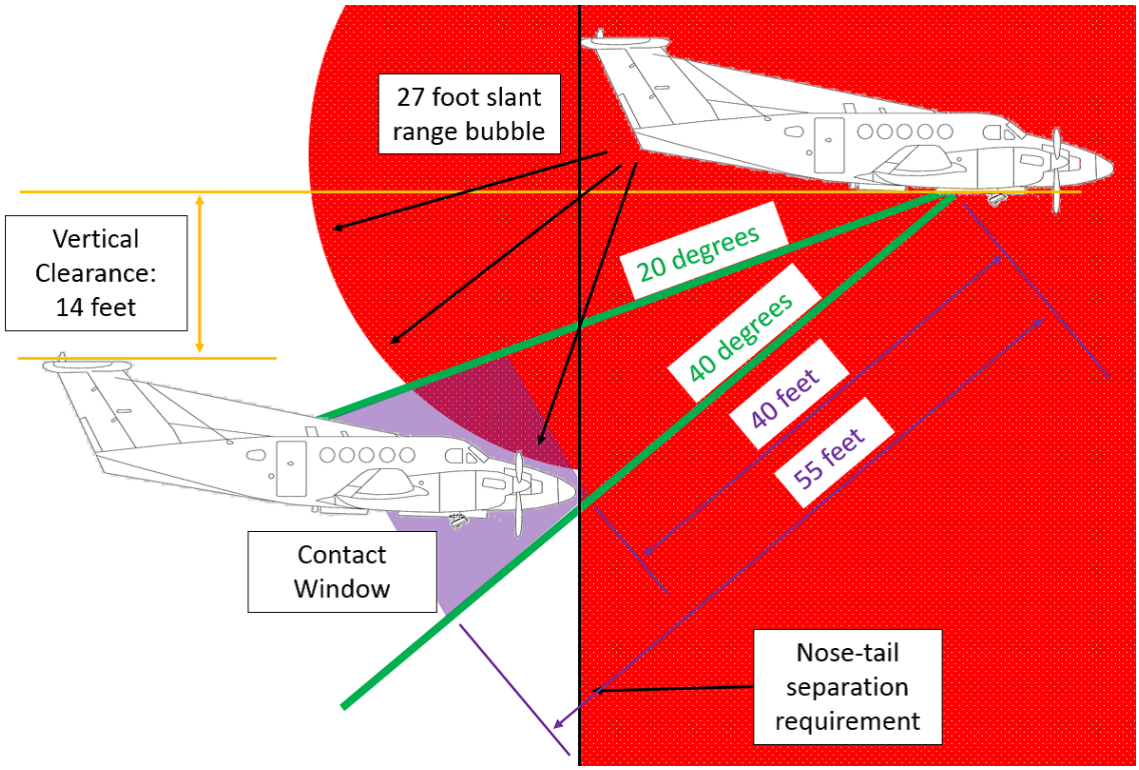


Figure 112. Side View of Contact Position Used in Flight Test.

same airspeed as the lead aircraft, at a range of 0.5-1.0 nautical miles (NM) in trail. Once both aircraft were ready, the target aircraft was cleared to the pre-contact position, closing at relative velocity of approximately 20 knots. The pre-contact position was defined as 50 feet behind the contact position, as shown in Figure 113. Once stabilized at pre-contact, the target aircraft was cleared to the contact position, closing at a rate of approximately 1 foot per second. Once in position, the target aircraft maintained the center of the contact window for at least two minutes.

Next, the formation executed a 360° turn at a 30° angle of bank while the target aircraft maintained the contact position. After both aircraft rolled wings level and stabilized, the target aircraft moved through the pre-contact position to a left or right observation position outside of camera fields of view. After moving to both

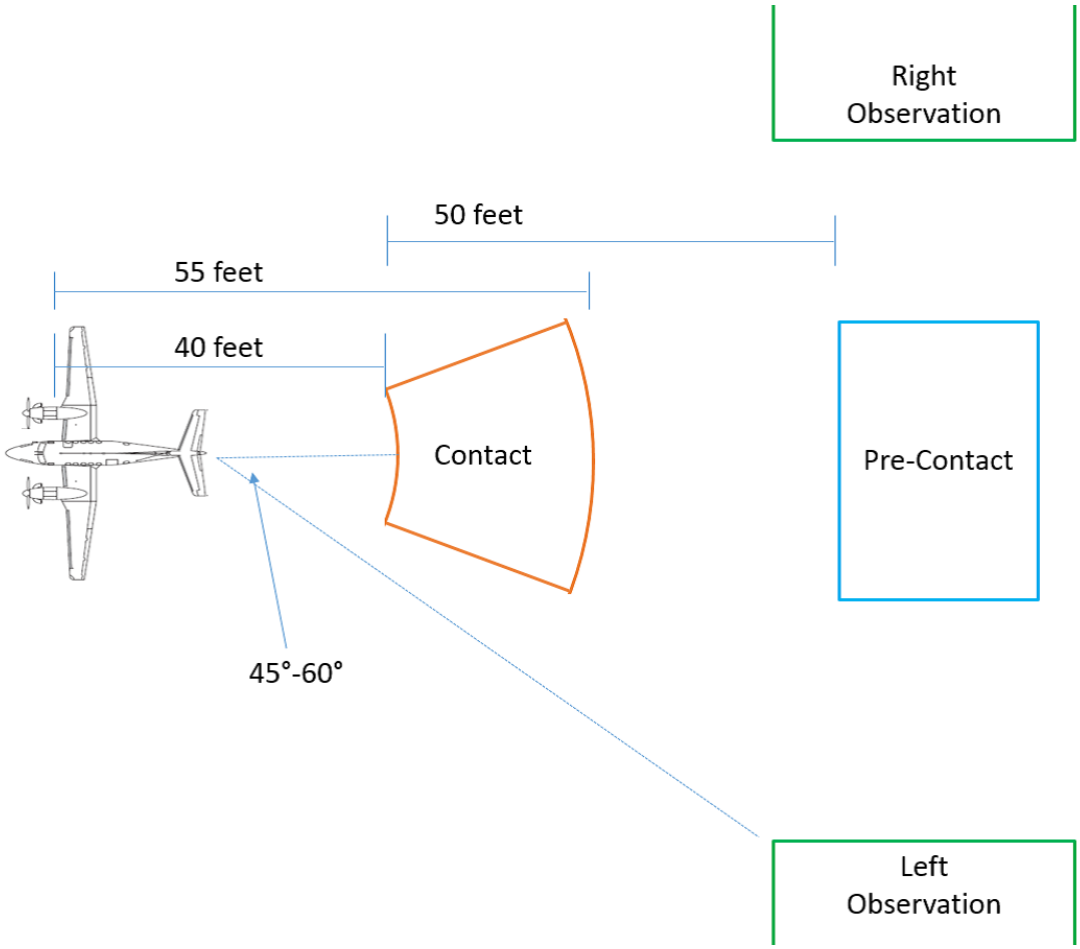


Figure 113. Formation Positions Flown During Flight Test (Not to Scale).

observation positions, the target aircraft moved back through the pre-contact position to the contact position.

Finally, from the contact position, the formation executed a simulated breakaway maneuver. During this maneuver, the lead aircraft advanced power while maintaining straight and level flight. The target aircraft retarded power and initiated a descent while maintaining visual contact with the pseudo-tanker. Once well clear, defined as outside a 500 foot bubble and in visual contact, the target aircraft terminated the maneuver. The lead aircraft maneuvered to test conditions for the next point while the target aircraft maneuvered to a position 0.5 NM in trail and 500 feet below the lead aircraft.



Figure 114. View from the Contact Position.

For the abbreviated AR maneuver block, the target aircraft closed visually to the pre-contact position. When cleared, the target aircraft closed to and maintained the contact position. The target aircraft returned to the pre-contact position when necessary to alleviate pilot fatigue.

These maneuver blocks were flown over a diverse set of backgrounds during the five formation test missions. Figure 115 depicts the maneuver areas planned to be flown over. In the figure, "D" denotes a desert background, "M" a mountainous background, "U" an urban background, and "O" an oceanic background. Of these all areas except "M2," "D2," and "U2" were utilized during flight tests. Data collected in area "M3" is analyzed in Chapter VI.

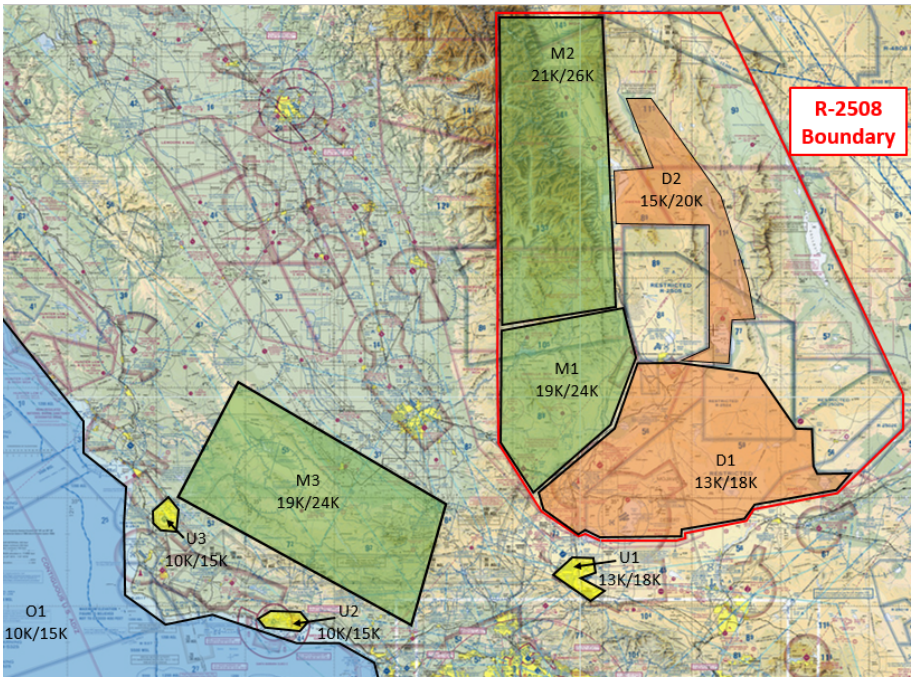


Figure 115. Background Environments Captured During Flight Test.

5.4 Flight Test Data Analysis Methodology

This section describes methodology used only on data collected in flight test. Aside from the specific methods pertaining solely to flight test data that are outlined in this section, algorithm and analysis methodology was the same as described in Chapter III.

5.4.1 Boresight Data Reduction.

As described in Section 5.3.1.1, boresight data was provided by the USAF Test Pilot School instrumentation team in the form of high precision vector measurements. These measurements were taken from the origin of the GeoRelNav® IMU installed in the C-12C pseudo-tanker and the C-12C pseudo-receiver. Two forms of measurements were taken:

1. The position of the origin of each frame of interest with respect to the origin of the GeoRelNav® IMU. This measurement was expressed in the GeoRelNav® IMU frame.

2. The position of the tips of one inch long unit vectors for each frame of interest expressed in the GeoRelNav® IMU frame. Three such measurements were made for each frame of interest—the tip of the x, y, and z unit vectors.

The following method was used to reduce this data into any given DCM or vector quantity of interest. Consider a frame of interest, A, and the GeoRelNav® frame, *Geo*. The location of the origin of the A-frame in the GeoRelNav® frame is the vector $\mathbf{A}^{Geo}$. The location of the tip of the A-frame's x, y, and z unit vectors in the *Geo*-frame are $\mathbf{A}^{x,Geo}$, $\mathbf{A}^{y,Geo}$, and $\mathbf{A}^{z,Geo}$, respectively. Hence, the unit vectors of the A-frame expressed in the *Geo*-frame are:

$$\mathbf{A}^{i,Geo} = \mathbf{A}^{x,Geo} - \mathbf{A}^{Geo} \tag{112}$$

$$\mathbf{A}^{j,Geo} = \mathbf{A}^{y,Geo} - \mathbf{A}^{Geo} \tag{113}$$

$$\mathbf{A}^{k,Geo} = \mathbf{A}^{z,Geo} - \mathbf{A}^{Geo} \tag{114}$$

Unit vectors for a frame B can be obtained in the same manner.

The resultant unit vectors for frames A and B can be used to directly compute the DCM expressing the rotation between the A-frame and B-frame:

$$\mathbf{R}_A^B = \begin{bmatrix} \mathbf{B}^{i,Geo^T} \\ \mathbf{B}^{j,Geo^T} \\ \mathbf{B}^{k,Geo^T} \end{bmatrix} \begin{bmatrix} \mathbf{A}^{i,Geo} & \mathbf{A}^{j,Geo} & \mathbf{A}^{k,Geo} \end{bmatrix} \tag{115}$$

Where a vector of the form $\mathbf{X}^p$ is a three-by-one column vector.

The translation vector from the origin of frame A to the origin of frame B can then be computed as:

$$\mathbf{B}^A = \mathbf{R}^A_{Geo}\left(\mathbf{B}^{Geo} - \mathbf{A}^{Geo}\right) \tag{116}$$

Where $\mathbf{R}^A_{Geo}$ is computed using the same methodology described in Equations (112) through (115), and $\mathbf{B}^A$ is the position of the origin of frame B in the A-frame. Note that the unit vectors for the *Geo*-frame, expressed in the *Geo*-frame would simply be $[1,0,0]^T$, $[0,1,0]^T$, and $[0,0,1]^T$. Using this methodology, the DCMs and translation vectors between all frames of interest were computed.

5.4.2 Truth Data Reduction.

The GLITE TSPI system recorded the absolute position and attitude of the both the pseudo-tanker and the pseudo-receiver. Position data was expressed in the ECEF frame. Attitude data was expressed as a roll, pitch, and yaw describing the rotation from the local NED frame to the aircraft body frame in the sense of the 3-2-1 convention. All data was time-stamped to the GPS clock. In order to serve as truth data, these absolute measurements had to be converted into measurements describing the position and orientation of the pseudo-receiver relative to the pseudo-tanker. The following method was used to obtain this result. This method utilized MATLAB® functions developed by the Autonomy and Navigation Technology (ANT) Center at AFIT. The details of each function's operation are not described in this document.

First, ECEF positions from the pseudo-receiver were converted into a local, NED position centered on the position of the pseudo-tanker. This was done with the ANT Center MATLAB® function `EcefToLocalLevel`. The resulting quantities were the position of the pseudo-receiver with respect to the pseudo-tanker expressed in the local NED frame, $\mathbf{s}^{\text{NED}}$.

Second, roll, pitch, and yaw attitude data from each aircraft were transformed into a DCM describing the rotation from the NED frame to each aircraft's body frame, $\mathbf{R}^{s}_{\text{NED}}$ and $\mathbf{R}^{p}_{\text{NED}}$. This was done with the ANT Center MATLAB® function `RpyToDcm`.

Third, using these results, the position of the pseudo-receiver in the pseudo-tanker frame was computed as:

$$\mathbf{s}^{p} = \mathbf{R}^{p}_{\text{NED}}\mathbf{s}^{\text{NED}} + \mathbf{R}^{p}_{\text{NED}}\mathbf{R}^{\text{NED}}_{s}\boldsymbol{\Delta}\mathbf{G}^{s} \tag{117}$$

Where the vector $\boldsymbol{\Delta}\mathbf{G}^{s}$ accounts for the difference in GLITE TSPI system installation location between the C-12C pseudo-receiver and the C-12C pseudo-tanker. This vector is expressed in the receiver aircraft frame and goes from where the receiver aircraft's GLITE system was actually installed to the coordinates the system would have had if it had been installed in the same location as it was in the tanker aircraft. Since the pseudo-tanker and pseudo-receiver aircraft were the same type (C-12C), the addition of this vector simplified algorithm analysis.

Fourth, the DCM expressing a rotation from the pseudo-tanker frame to the pseudo-receiver frame was computed as:

$$\mathbf{R}^{s}_{p} = \mathbf{R}^{s}_{\text{NED}}\mathbf{R}^{\text{NED}}_{p} \tag{118}$$

5.4.3 Truth Data Uncertainty.

As discussed in Section 5.2.5, the CEP95 of the GLITE TSPI system known to be 1.5 feet. The following methodology was used to convert this figure into dimensional and spherical uncertainties. Uncertainties were determined for both single data points and for mean errors for a sample of size n. Several assumptions were made in this analysis. It was assumed that the errors of the GLITE TSPI system were normally

distributed with a mean of zero. Additionally, it was assumed that the CEP95 figure described the 95% confidence interval of a single sample error in any two dimensions. These two dimensions were assumed to have equal 95% confidence intervals. Finally, it was assumed that the error distributions of each dimension were independent.

Given these assumptions and the figure for CEP95, the 95% confidence interval for a measurement in a single dimension, ϵ, can be computed as:

$$\epsilon = \pm\sqrt{\text{CEP95}^2/2} \tag{119}$$

Where ϵ describes the dimensional uncertainty of a single system taking an absolute measurement. When combining the measurements of two systems into a relative position measurement, the 95% confidence interval for the relative measurement, ϵ_{rel}, can be computed as:

$$\epsilon_{\text{rel}} = \pm\sqrt{\epsilon^2 + \epsilon^2} = \text{CEP95} \tag{120}$$

The 95% confidence interval of the spherical uncertainty, ϵ_{sphere} would therefore be:

$$\epsilon_{\text{sphere}} = \sqrt{3}\text{CEP95} \tag{121}$$

These 95% confidence intervals pertain to a single sample.

To compute the 95% confidence interval for the mean error taken from a sample size of n measurements, one must compute the standard deviation of the mean, SDOM, with the equation [50]:

$$\text{SDOM} = \frac{\sigma}{\sqrt{n}} \tag{122}$$

Where σ is the known standard deviation of the normal distribution. Based on the assumptions outlined above:

$$\text{CEP95} = 1.96\sigma \tag{123}$$

Therefore the 95% confidence interval for the mean error in the GLITE TSPI system in a sample size of n in a single dimension can be computed as [48]:

$$\left[-\frac{\text{CEP95}}{\sqrt{n}}, \frac{\text{CEP95}}{\sqrt{n}}\right] \tag{124}$$

The 95% confidence interval of the mean spherical error is a more complicated computation. This value was estimated in MATLAB® by generating three Gaussian random variables with means of zero and standard deviations of CEP95/1.96. A total of 221 random draws were taken from each of these random variables. This experiment was repeated ten million times in MATLAB®. The results were then used to generate a histogram normalized as a CDF with the MATLAB® function `histogram`. This plot is shown in Figure 116. As can be seen on the plot, this experiment indicated that 0.390 meters comprised the upper limit of the 95% confidence interval for mean spherical error.

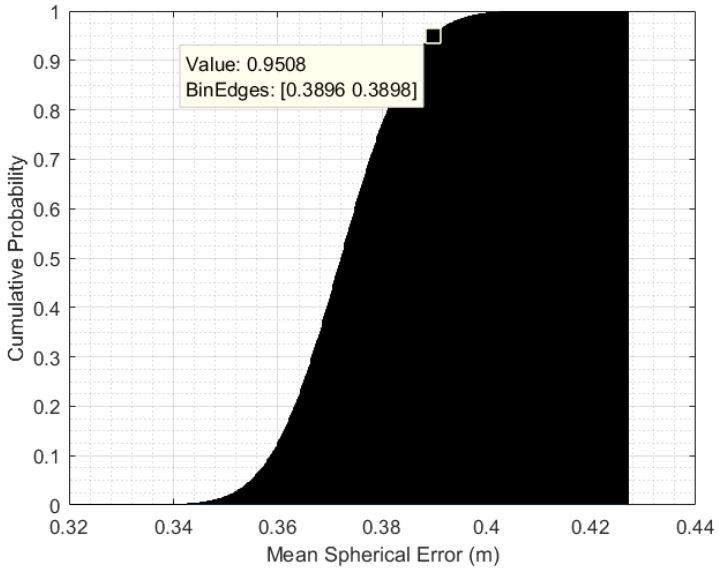


Figure 116. Experimental Results Used to Determine 95% Confidence Interval of Mean Truth Data System Spherical Error.

5.4.4 Relating Camera Data to the *p*-Frame.

The method outlined in Section 5.4.1 was used to compute DCMs and translation vectors between the EO cameras and the GLITE TSPI truth data system. The GLITE TSPI system was essentially aligned with the aircraft body frame during flight test. Hence, frame of the pseudo-tanker's GLITE TSPI system was considered to the be the p-frame for the purposes of analyzing the V5 and R7 algorithms. This frame is described in Sections 3.1 and 3.2.

5.4.5 V5 Algorithm Considerations.

5.4.5.1 Reference Image Database Generation.

In order for the V5 algorithm to operate properly, a reference image database had to be generated with simulated cameras having the same orientation with respect to the pseudo-tanker as the cameras actually used in flight test. This database was generated in the 3DVW using the results obtained from the extrinsic camera calibra-

tions. Additionally, this database was defined such that it completely contained all images that would be analyzed. The dimensions of this database were the same as those used in simulation, which are described in Section 4.1.1.

As described in Section 3.4, each image in the reference image database has a corresponding position that is known and defined. In order to correctly relate the reference images to flight test images, these positions must correspond to the positions that would be returned from the truth data system. In the 3DVW, the position of depicted in each reference image was defined to be the vector going from the tanker's left camera to the position on the C-12 receiver where a left camera would be installed were it flying as the pseudo-tanker. The "tanker" in the 3DVW is purely notional and can be considered to be a C-12C. Hence, if one considers the 3DVW tanker and receiver to be identical aircraft, this vector points from a position on the tanker to the same spot on the receiver. As a result, this is the same vector as that returned after GLITE TSPI data reduction since this vector points from a spot on the tanker to the same spot on the receiver. This is true because in the reference image database is generated with both the tanker and receiver aircraft in identical attitudes (e.g. the receiver is not yawed, pitched, or rolled with respect to the tanker body frame). Figure 117 depicts this situation.

5.4.5.2 Image Rectification and Scaling.

When working with flight test observation images, one additional challenge arose relating to image rectification. Based on the construct of the 3DVW, the simulated cameras generate ideal images. For instance, the images lack distortion, the principal point of the images is precisely centered, and the focal length of the cameras essentially matches the theoretical value. By contrast, when working with actual real-world cameras, the optics of the camera differ significantly from their idealized

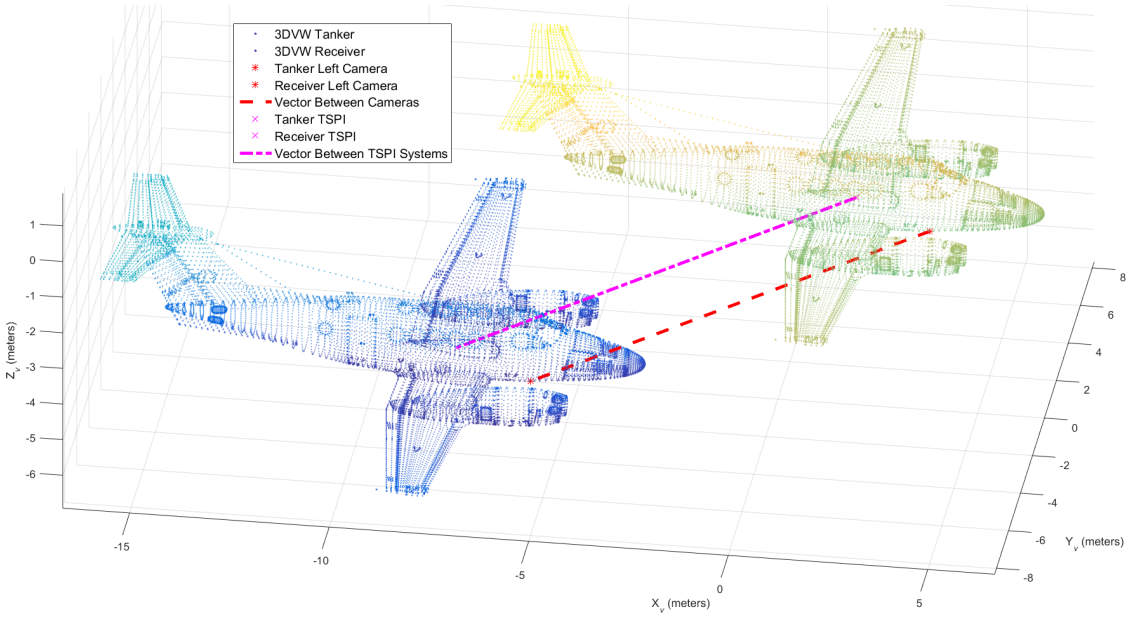


Figure 117. Methodology Used to Relate Reference Image Database to GLITE TSPI Data.

values. Hence, rectified image pairs generated from 3DVW images differ from the rectified image pairs generated from real-world cameras.

This difference manifests in the mapping describing the transformation from a non-rectified image to a rectified image. This effect of this change is most noticeable in terms of the size of the returned images. Table 26 describes how the rectified, 3DVW reference images differed from the rectified, real-world observation images in terms of size.

Table 26. Difference in Processed Image Sizes Between Reference Images and Flight Test Observation Images

Image Type	Size (pixels)
Processed 3DVW Reference Images	961 x 1283
Processed Real-World Observation Images	1000 x 1328

The V5 algorithm operates on a fundamental level by comparing the pixels located at the same coordinates in the reference and observation images. If these two image

sets are different sizes, the algorithm will not operate properly. Hence, the size difference resulting from rectification is a major challenge for the V5 algorithm.

This challenge was overcome by re-scaling the observation images to be the same size as the reference images. In order to accomplish this, the edges of the observation images were cropped such that the resultant image as the same size as the reference images, 961-by-1283 pixels. During data analysis, experimentation with the MATLAB® function `imresize` was also used to resize the observation images. This method yielded similar results. However, either method is not an ideal solution to this problem, and could likely contribute to errors in the V5 algorithm's solution as is described in Section 6.3. Ideally, one would alter the simulation environment used to generate the reference imagery such that the simulated cameras could better mimic the optics to the real-world cameras. This would result nearly identical reference and observation image rectification mappings.

5.4.6 R7 Algorithm Considerations.

5.4.6.1 Shell Model Point Cloud.

In addition to the full model point cloud used in simulation, a shell model point cloud was used for the ICP portion of the R7 algorithm on flight test data. This shell model was developed by Parsons in [6]. While this shell point cloud did not yield significantly different results in simulation in this research effort, in flight test this shell proved to be more effective as is discussed in Section 6.2. The shell model point cloud used is depicted in Figure 118. Essentially, this model point cloud eliminates points that are not likely to be visible to the cameras since they are looking down at the receiver.

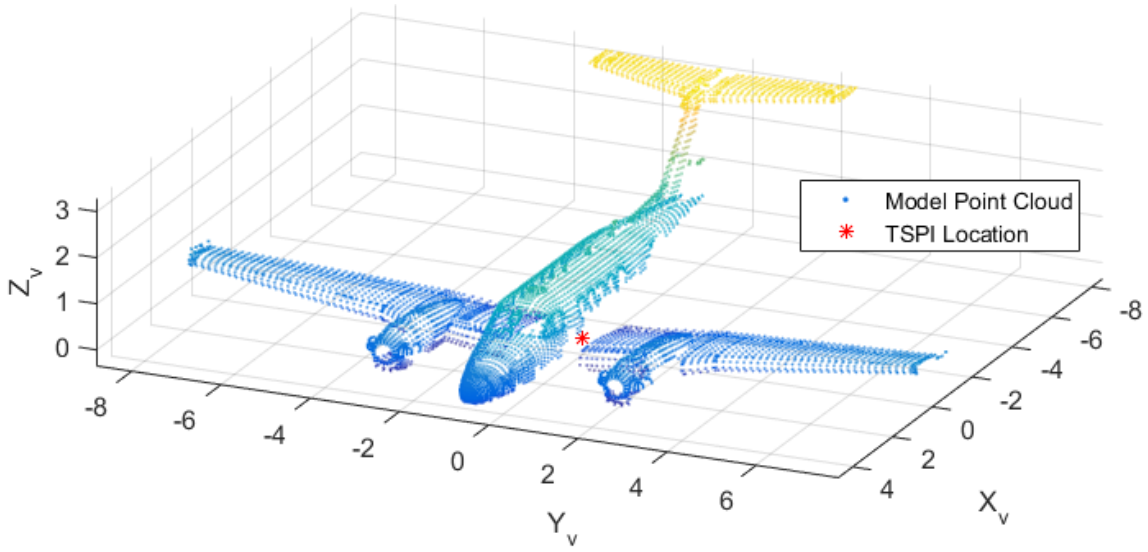


Figure 118. Shell Model Point Cloud Used with Flight Test Data.

5.4.6.2 Miscellaneous Data Processing Changes and Example.

During R7 execution on flight test data, observed point cloud points were returned in the v-frame. The origin of the v-frame and p-frame were defined to be at the location the pseudo-tanker's GLITE TSPI system. Hence, as can be seen in Figure 118, the model point cloud was translated such that its origin was positioned where the origin of the GLITE TSPI system would be located in the model point cloud.

Image pairs collected in flight test were fed into the 3DVW for processing. Using the camera calibration results obtained during ground testing, image pairs were transformed into three-dimensional point clouds expressed in the v-frame. This process made use of the same methodology used in processing simulated imagery and is described in Section 3.5. Figure 119 shows an image pair collected in flight test and the resultant point cloud generated in the 3DVW.

After conducting image processing in the 3DVW, point cloud data was then fed into MATLAB® for further processing. The MATLAB® processing performed on flight test data was identical to that used with simulation data, the only difference

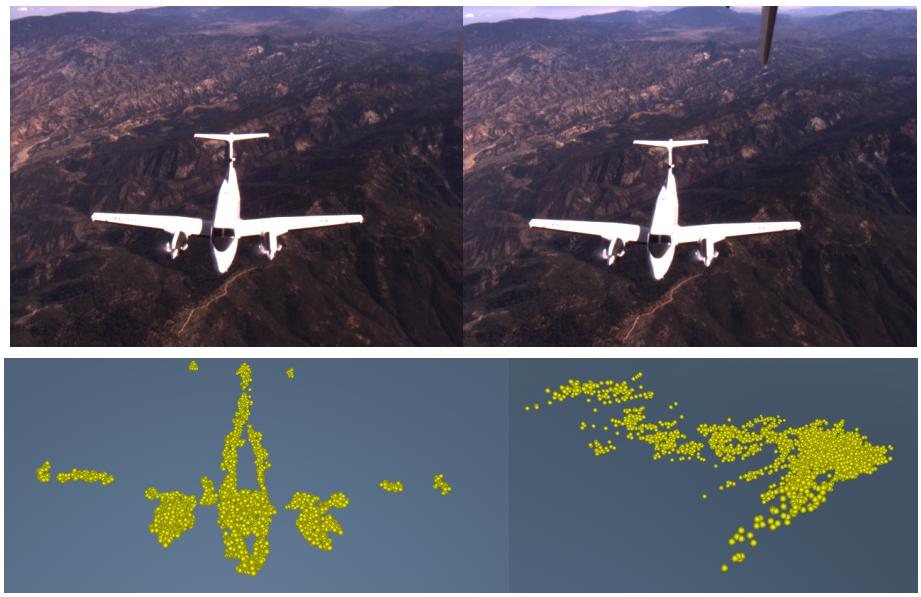


Figure 119. Image Pair and Resultant Point Cloud Generated in 3DVW Viewed from Front and Side.

being the source of truth data. As in simulation, raw point clouds were filtered and denoised using the process outlined in Section 3.5.4. Figure 120 shows an example raw point cloud and the filtered point cloud returned in MATLAB®.

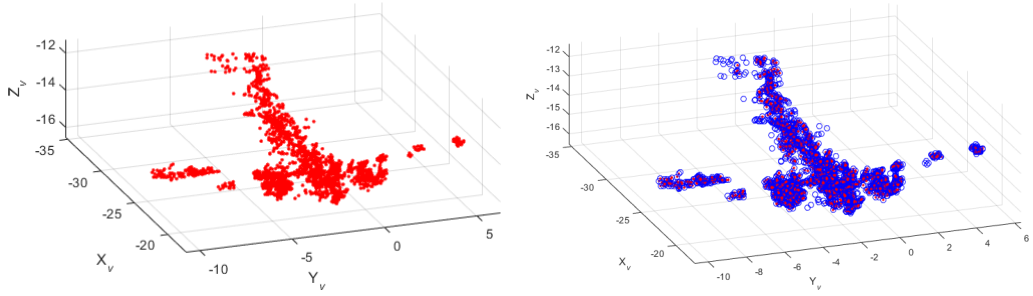


Figure 120. At Left, the Raw Point Cloud is Shown in Red. At Right, Blue Circles Depict the Raw Point Cloud and the Red Dots Depict the Results After Filtering.

Next, in MATLAB® the filtered point cloud was fed into the ICP algorithm. This was the same process used with simulated images, except for the use of the shell model point cloud. The left portion of Figure 121 shows the result of filtered point

cloud being matched to the shell model point cloud with ICP. The right portion of Figure 121 shows a visualization of the error in the R7 algorithm's solution for this example.

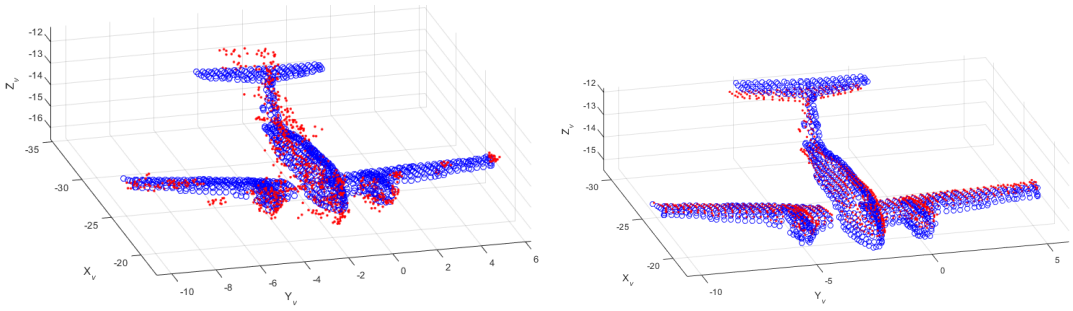


Figure 121. From Left To Right, the ICP Solution and an Error Visualization. At Left, the Red Dots Depict the Raw Point Cloud and the Blue Circles Depict the Model Point Cloud. At Right, Red Dots Depict the R7 Estimate and the Blue Circles Depict the True Solution.

5.4.7 Partial Autocorrelation Function and Data Decimation.

The partial autocorrelation function can be used to determine how much data should be decimated to attain statistically independent samples. In time series data, this process is necessary when there is a strong correlation between consecutive samples [51]. The MATLAB® function `pacf`, developed by Jorris of Lockheed-Martin, was used to apply a partial autocorrelation function to flight test data. This was done to ensure the independence of the data samples used to analyzed the R7 and V5 algorithms.

Using this function, one can determine how many consecutive samples need to be decimated from a data set to eliminate unacceptably high correlation between time-consecutive samples. The threshold used on flight test data was an R^2 value of 0.05. Based on the returns from the partial autocorrelation function on position and attitude errors, R7 error results were decimated to a subset suitable for analysis.

This decimated data set was the used for both R7 and V5 analysis. This process is detailed in Section 6.2.1.

VI. Flight Test Data and Analysis

The Have Vision test management project collected a data set of 415,800 image pairs with either a C-12C or T-38C aircraft positioned as a pseudo-receiver at 10,000 to 15,000 feet AGL over various desert, oceanic, urban, cloud, and mountainous background environments. In this chapter, both the R7 and V5 algorithms are analyzed using a subset of these images. The subset consisted of 221 independent, time-matched stereo image pairs captured from the EO cameras. This particular data set was collected with the pseudo-receiver C-12C aircraft positioned in the contact position before a mountainous background (area "M3" depicted in Figure 115) at approximately 10,000 feet AGL. This data set spanned a three minute and three second time interval.

Flight results from the Have Vision project were also presented in the program's test information memorandum [7]. However, flight test results presented in this chapter differ substantially from those presented in the test information memorandum due to algorithm developments and analysis occurring after completion of that document. Consequently, the results presented in this thesis supersede those presented in the Have Vision test information memorandum.

6.1 Truth Data Resolution

In this chapter, the 95% confidence intervals plotted for the truth data system apply to the 95% confidence interval of truth data error at a *single* data point. The 95% confidence intervals for the mean truth data system error are not plotted. The middle column of Table 27 presents the 95% confidence intervals for the truth data system with respect to single data points. The right column presents the 95% confi-

dence interval for the mean truth data system error in a sample of 221 data points. Section 5.4.3 describes how these values were estimated.

Table 27. Truth Data System Confidence Intervals, GLITE TSPI System.

	Single Data Point (m)	**Mean of 221 Data Points (m)**
One Dimension Error 95% CI	[-0.457, 0.457]	[-0.031, 0.031]
Spherical Error 95% CI	[0, 0.792]	[0, 0.390]

Any given single data point falling within the 95% truth data confidence intervals depicted on a plot may actually have been error free. However, the confidence interval for the mean error in the truth data system was much lower than the single point confidence interval. Hence, any mean algorithm errors that exceed the 95% confidence interval of the truth system mean error indicate a possible algorithm bias. The confidence intervals for attitude errors are not tabulated or plotted because the truth data attitude uncertainties at a single point were much smaller in magnitude than the attitude errors observed in the R7 algorithm.

6.2 R7 Data and Analysis, Flight Test

This section describes analysis of the R7 algorithm performed with flight test data. Section 6.2.1 describes the process used to obtain a time-independent data set suitable for analysis. Section 6.2.2 presents the results obtained when using the non-attitude fed version of the R7 algorithm. Section 6.2.3 presents the results obtained when the attitude fed version of the R7 algorithm. Section 6.2.4 details the algorithm processing

times observed when working with flight test data. Flight test data analysis revealed five primary findings:

1. Mean R7 spherical errors fell within the uncertainty of the truth data system.
2. The R7 algorithm manifested a persistent bias of several decimeters in the z-component of its position estimate in the left camera frame.
3. Attitude error variances were much smaller in magnitude than those observed in simulation at similar ranges.
4. The attitude fed version of the R7 algorithm reduced spherical errors, depth errors, and the variance of these errors at a statistically significant level compared to the non-attitude fed version. The error variance result contradicts simulation results.
5. Point cloud generation was the most time consuming component of the algorithm. Algorithm operation on the resultant point cloud required only 0.026 seconds on average.

6.2.1 Data Set Decimation.

Initially the R7 algorithm was applied to 5,501 image pairs captured between 17:50:53 coordinate universal time (UTC) and 17:53:56 UTC on 13 September 2017. These results showed a high degree of time correlation in terms of spherical position error, component position errors, and attitude position errors. Time series plots of these errors can be seen Figures 122, 124, and 125, respectively. Figure 123 depicts the spherical error plotted against three-dimensional position in the v-frame.

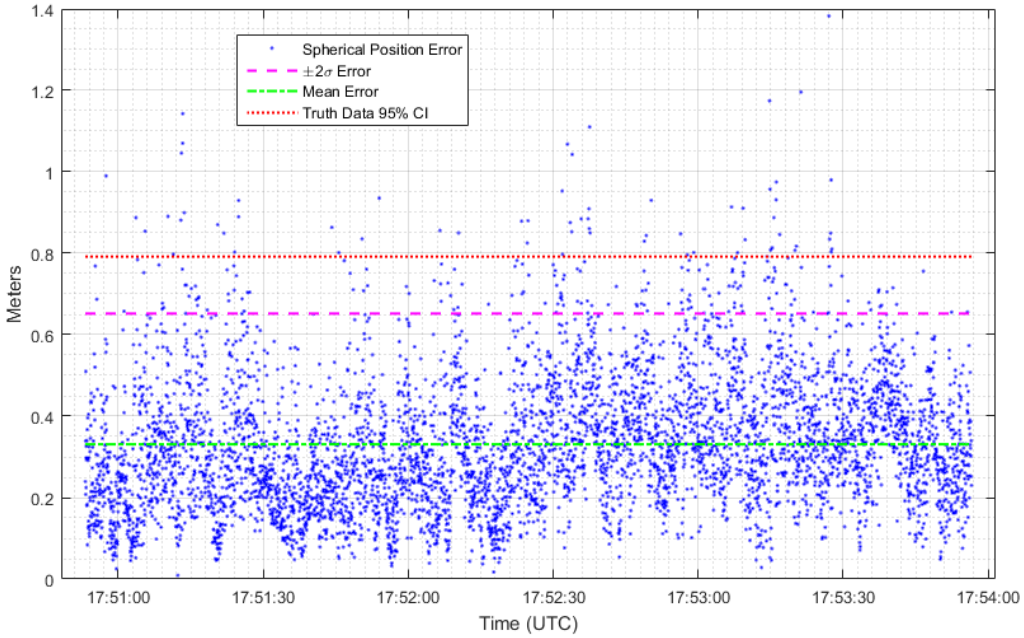


Figure 122. R7 Spherical Error in Flight Test, All Data Points, No Attitude Feeding.

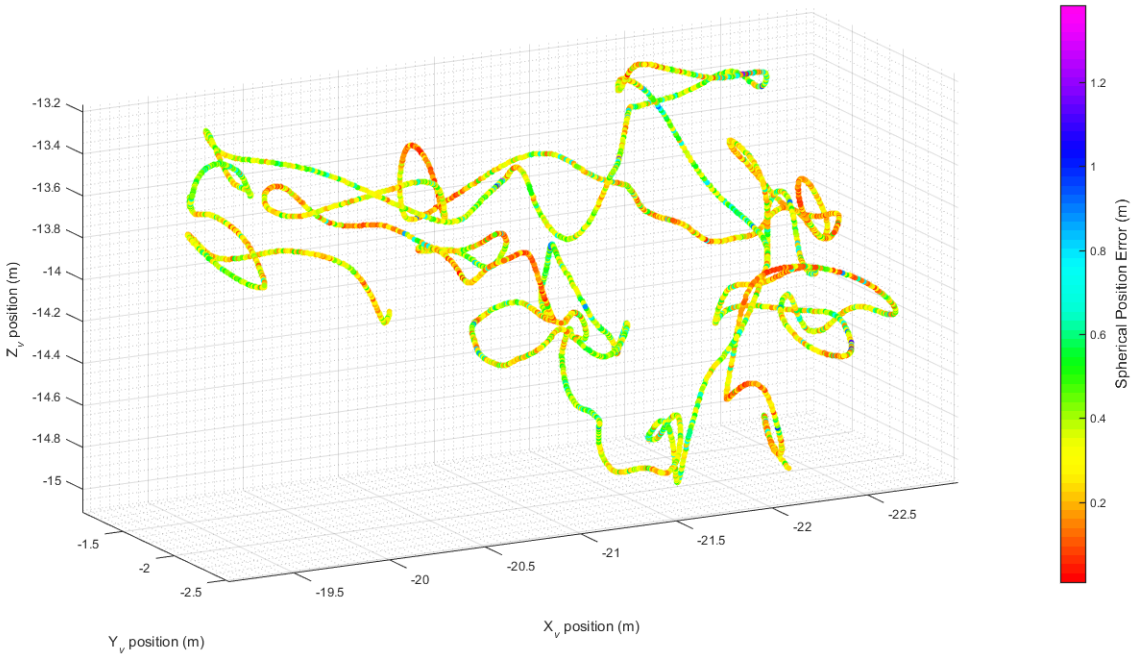


Figure 123. R7 Spherical in Flight Test, All Data Points, No Attitude Feeding.

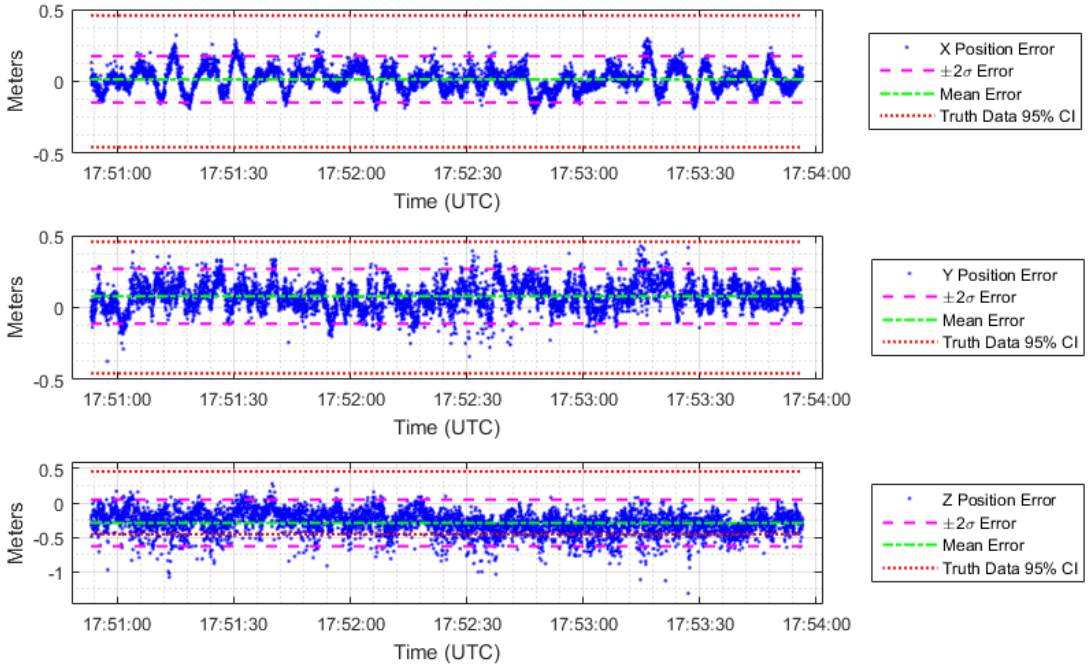


Figure 124. R7 Position Errors in Flight Test, All Data Points, No Attitude Feeding, Left Camera Frame.

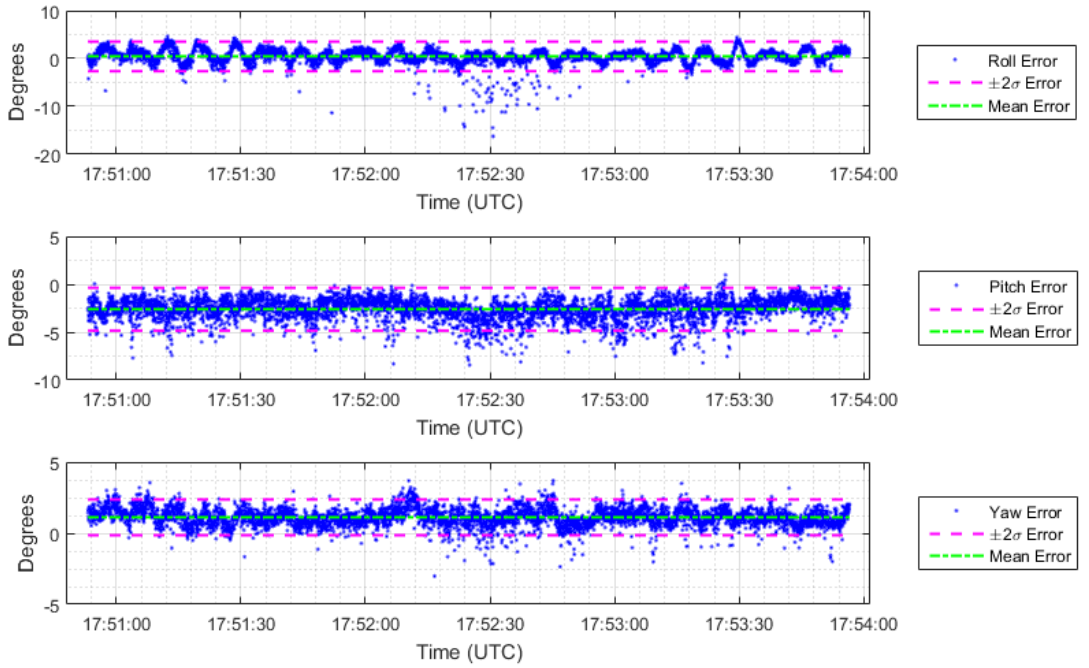


Figure 125. R7 Position Errors in Flight Test, All Data Points, No Attitude Feeding, Left Camera Frame.

The partial autocorrelation function was applied to the x, y, z, roll, pitch, and yaw errors returned from this set of 5,501 error values. The left camera frame z-position error exhibited the most persistent correlation. Figure 126 shows the return from the MATLAB® `pacf` function for this error value. PACF returns for other error components can be found in Appendix B. As can be seen in the figure, the correlation between consecutive samples does not fall below the 0.05 threshold until the 18^{th} sample. Based on this result, the data set of 5,501 image pairs was decimated to retain only every 25^{th} image. A higher value than 18 was chosen to as a conservative measure to ensure time independence and to reduce the size of the subset to be used with the V5 algorithm. This decision was based on the V5 algorithm's long processing time requirements. This resulted in a time-independent subset of 221 observation images.

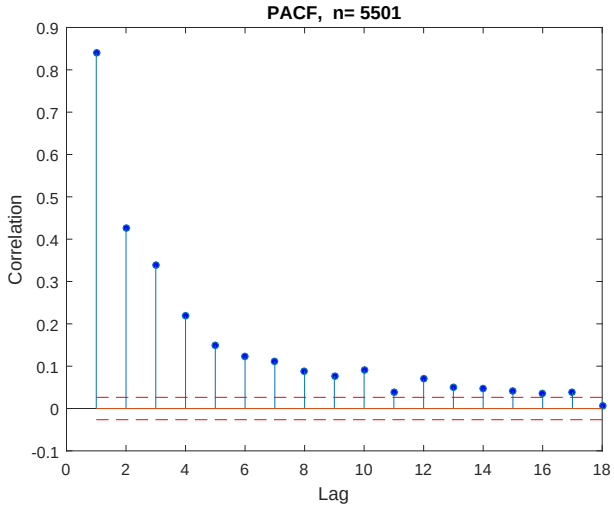


Figure 126. Partial Autocorrelation Function Results, Z Position Error, Left Camera Frame.

6.2.2 Case 1: No Attitude Feeding.

The decimated set of 221 images was first processed through the R7 algorithm configured to include attitude estimation. Spherical error results are shown in Figure

127. As can be seen in the figure, mean spherical error less than 35 centimeters and the distribution of the error was fairly consistent throughout the time series covered by this data set. All errors, except for one value fell within the single sample 95% confidence interval of the relative position computed with GLITE TSPI system data. Additionally, the mean spherical error fell within the 95% confidence interval for the mean truth data system spherical error as can be seen from Table 27. Hence, no assertion of algorithm error can be made based on examining spherical error only.

Interestingly, these errors were smaller than those observed in simulation. This phenomenon could be attributed to the fact that the receiver was essentially centered in the camera fields of view in flight test, while in simulation the receiver was deliberately placed closer to the boundaries of the fields of view. Table 28 presents the mean and standard deviation for spherical error for this case. Table 29 presents the 95% confidence interval for the mean spherical error for this case.

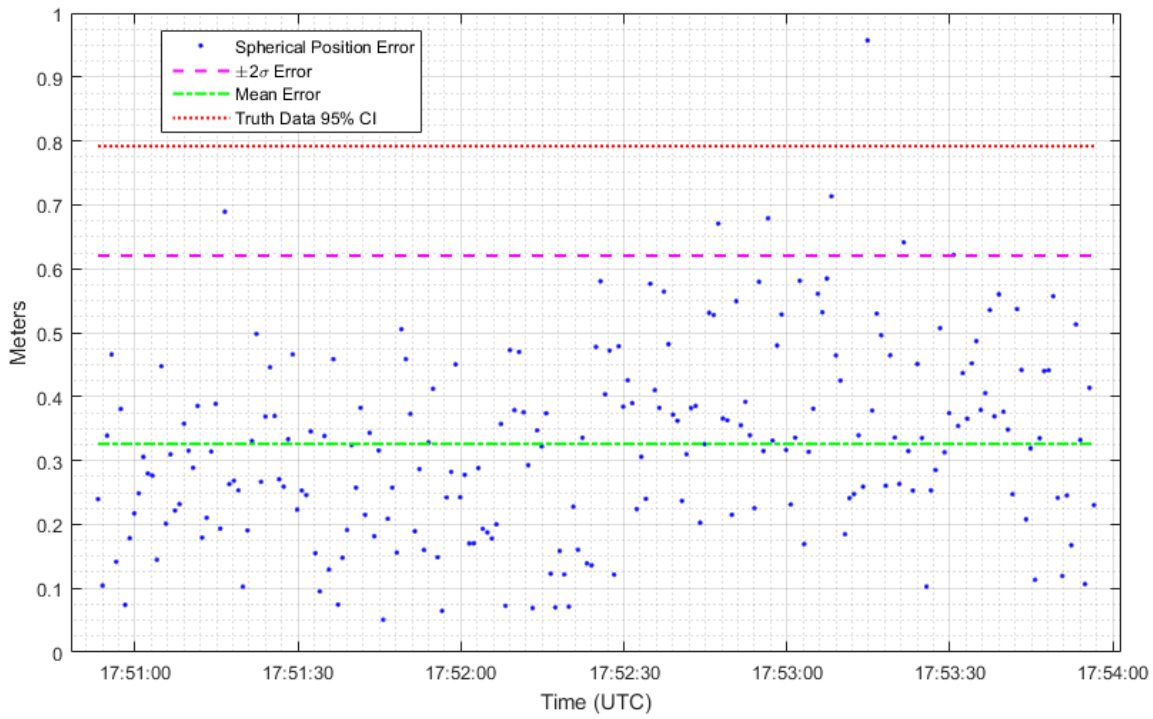


Figure 127. R7 Spherical in Flight Test, Decimated Data Set, No Attitude Feeding.

Figure 128 plots the component position errors observed in the left camera frame. As can be seen in the figure, mean x and y-position errors were very close to zero, while a bias of approximately 30 centimeters was present in the z-position error. Standard deviations in all dimensions were on the order of centimeters. Comparison with the confidence interval for the mean error of the truth data system indicates that the z-position estimate was likely biased. Additionally, small bias may exist in the y-position estimate as well. As discussed in Sections 2.3.6.6 and 4.2.2.1, any bias in the x and y-component estimates, could be attributed to errors in depth. Table 28 presents the means and standard deviations of the position errors for this case. Table 29 presents the 95% confidence intervals for the mean position errors for this case.

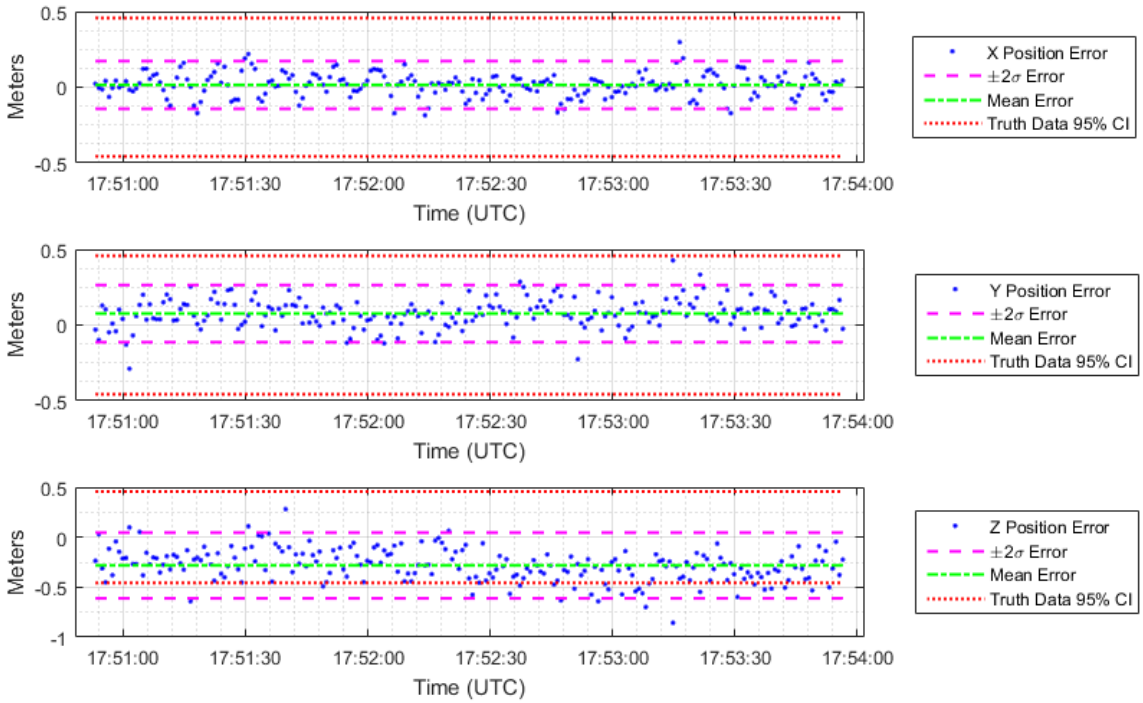


Figure 128. R7 Position Errors in Flight Test, Decimated Data Set, No Attitude Feeding, Left Camera Frame.

This bias constitutes an error in depth estimation. "Depth" in this context refers to the z-position in the left rectified camera frame, Z_{L_r}. As was discussed in Section 2.3.6.6 and Section 4.2.2.1, this depth value and the corresponding resolution are

computed as:

$$Z_{L_r} = -\frac{f_x T_x^{R_r}}{d} \tag{125}$$

$$\Delta Z_{L_r} = \frac{Z_{L_r}^2}{f_x T_x^{R_r}} \Delta d \tag{126}$$

Where f_x is the focal length in pixels, $T_x^{R_r}$ is the x distance between the stereo cameras in the right rectified camera frame, d is the computed disparity value, and Δd is the disparity resolution. An error in any f_x, $T_x^{R_r}$, or d will lead to errors in depth estimation.

This speaks to the criticality of camera calibrations. A 1% error in f_x or $T_x^{R_r}$, will lead to a persistent 1% error in depth if disparity values are perfectly computed. For the camera configuration used in flight test, 1% error in f_x would be approximately 10 pixels while a 1% error in T_x^R would be approximately 5 millimeters. With respect to the focal length obtained in camera calibrations, the 3σ uncertainty for focal length estimates was less than 7.2 pixels in all cases. Based on this result and the high precision with which the baseline was measured, errors f_x and $T_x^{R_r}$ likely do not solely account for the observed depth bias.

At a depth of 30 meters, with a disparity resolution of $\frac{1}{16}$ of a pixel, and based on the camera calibration results presented in Appendix A, the depth resolution for any given point in the observed point cloud should have been approximately 0.095 meters. Hence, depth resolution could account for a substantial portion of the bias.

Figure 129 depicts the R7 and truth data system computed depths and the ratios of these depths for the non-decimated data set of 5,501 samples. As can be seen in the figure, R7 persistently underestimated the depth by approximately 1%. Based on the discussion above, the portion of the bias not attributable to depth resolution could be attributed to small errors in f_x and T_x^R or to persistent over-estimation of disparity values as was discussed in Section 4.2.2.

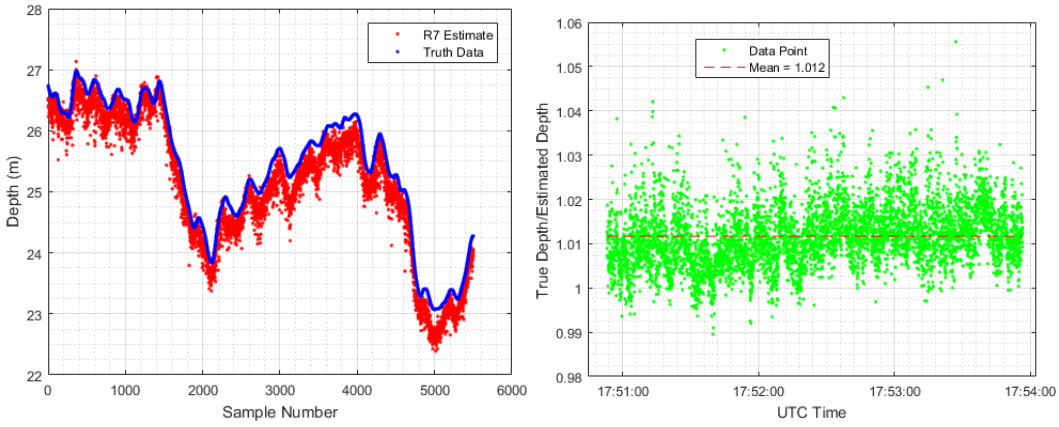


Figure 129. At Left, Left Camera Frame Depths for the Non-Decimated Data Set. At Right, the Ratio of True Depth to Estimated Depth for the Non-Decimated Data Set.

With respect to attitude errors, flight test data revealed performance that was somewhat different from performance observed in simulation. These errors are depicted in Figure 130. Truth data confidence intervals are not presented on this plot because they were 0.1° for a single sample. As with position truth data, the 95% confidence interval of the mean attitude error in each dimension would be several orders of magnitude smaller than this. Hence, any computed errors in R7 attitude estimation were likely true errors.

Similar to simulation, the R7 pitch estimate exhibited a persistent negative bias (the algorithm tended to estimate greater pitch down than was present). However, standard deviations of the attitude estimates were on the order of about a degree, much lower than that predicted in simulation. Additionally, flight test yaw errors exhibited a positive bias (nose right) which was not observed in simulation. Table 28 presents the means and standard deviations of the attitude errors for this case. Table 29 presents the 95% confidence intervals for the mean attitude errors for this case. These results indicate that the R7 algorithm may be better at providing rough attitude estimates than was suggested by simulation results.

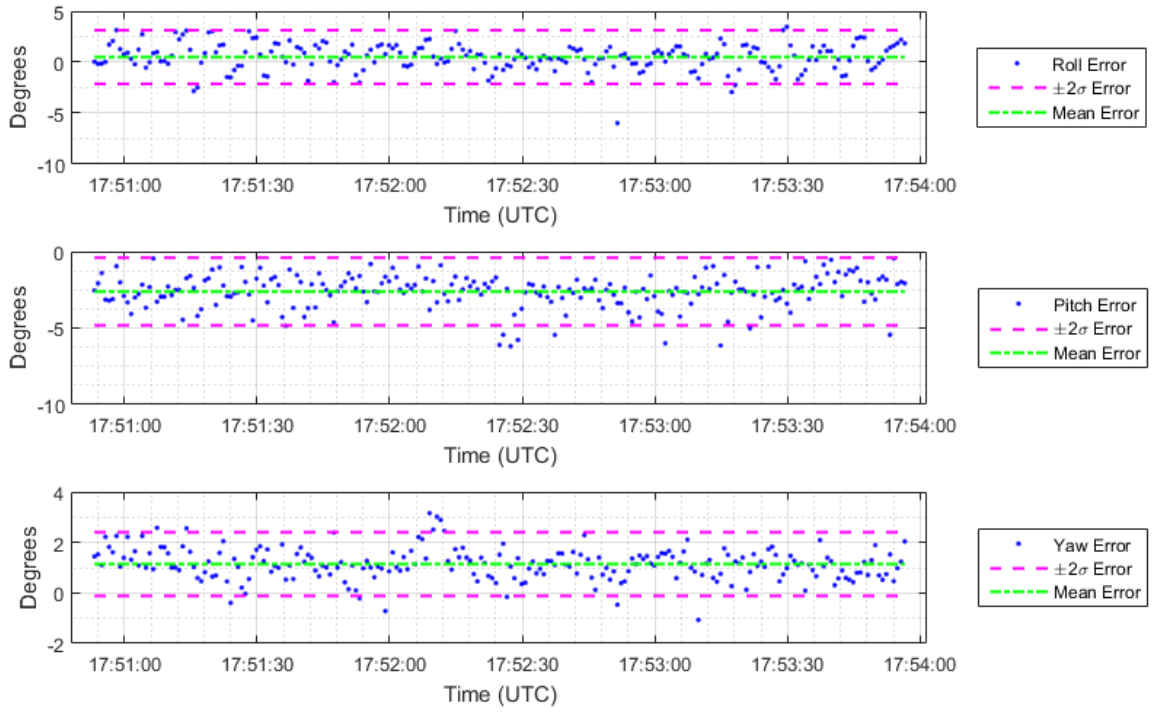


Figure 130. R7 Attitude Errors in Flight Test, Decimated Data Set, Tanker Frame.

Table 28. Flight Test Results for the R7 Algorithm Using the 221 Decimated Samples.

	Case 1		Case 2	
	Mean	Standard Deviation	Mean	Standard Deviation
Spherical Error (m)	0.326	0.147	0.270	0.119
x Position Error (m)	0.015	0.079	0.016	0.074
y Position Error (m)	0.076	0.094	0.045	0.090
z Position Error (m)	-0.283	0.165	-0.225	0.142
Roll Error (deg)	0.498	1.315		
Pitch Error (deg)	-2.614	1.105		
Yaw Error (deg)	1.144	0.630		

Table 29. Mean Error 95% Confidence Intervals for the R7 Algorithm on Flight Test Data. Intervals Indicated with a "*" Overlap with the 95% Confidence Interval of the Truth Data Mean Error.

	95% Confidence Interval of the Mean	
	Case 1	**Case 2**
Spherical Error (m)	[0.307, 0.346]*	[0.254, 0.285]*
X Position Error (m)	[0.005, 0.026]*	[0.006, 0.0256]*
Y Position Error (m)	[0.063, 0.089]	[0.033, 0.057]
Z Position Error (m)	[-0.305, -0.262]	[-0.244, -0.206]
Roll Error (deg)	[0.324, 0.672]	N/A
Pitch Error (deg)	[-2.761, -2.468]	N/A
Yaw Error (deg)	[1.060, 1.227]	N/A

6.2.3 Case 2: Attitude Feeding.

Next, the decimated data set of 221 observation image pairs was processed through the attitude fed variant of the R7 algorithm. Spherical error results are shown in Figure 131. As can be seen in the figure, mean spherical error was less than 30 centimeters and the distribution of the error was fairly consistent throughout the time series covered by this data set. All errors fell within the 95% confidence interval of the relative position computed with GLITE TSPI system data for a single sample. Additionally, the mean spherical error fell within the 95% confidence interval for the mean truth data system spherical error. Hence, no assertion of algorithm error can be made by an examination of spherical error alone. Both the mean spherical error and the standard deviation of the spherical error were lower than in Case 1. Table 28 shows this result. Table 29 shows that the 95% confidence interval for the mean spherical error in Cases 2 does not overlap with that from Case 1.

A paired t-test showed that the Case 1 and Case 2 mean spherical errors had a difference of statistical significance. At the 95% confidence level, the mean spherical error in Case 2 was 0.045 to 0.068 meters less than in Case 1. Similarly, a two-sample F-test showed that the variance of this error was significantly smaller in the attitude fed case. This result is the opposite of what was observed in simulation. The 95% confidence interval for the ratio of the variance in Case 1 to the variance in Case 2 was [1.181, 2.006]. These results indicate that the attitude fed version of the R7 algorithm can reduce mean spherical error and spherical error variance.

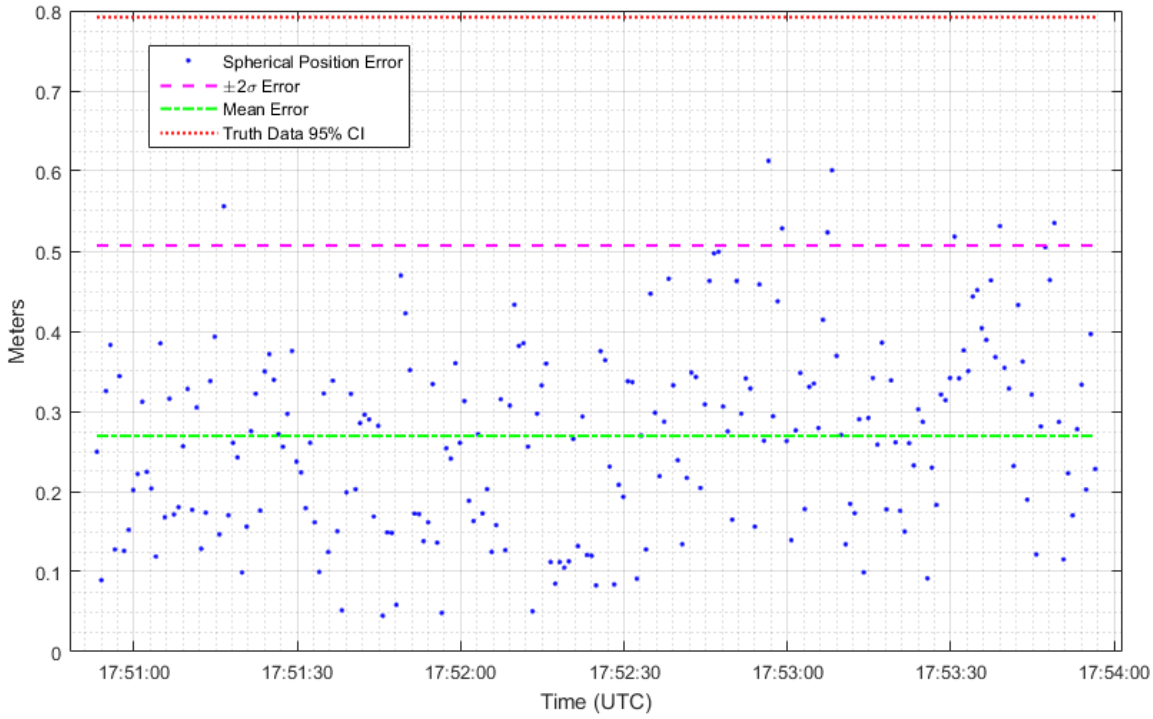


Figure 131. R7 Spherical in Flight Test, Decimated Data Set, Attitude Feeding.

Figure 132 plots the position errors for this case. As in Case 1, a strong bias was observed in the z-position estimate. Mean errors in the x and y-dimensions were much closer to zero. Based on the uncertainties of the truth data system, the observed z-axis bias in all likelihood represents a real bias. Again, as can be seen in Table 28, the mean and standard deviation for the z-component were lower in Case 2. Table

29 shows that the 95% confidence interval for the mean z-position error for Case 2 does not overlap with that from Case 1.

A paired t-test showed that z-position error differences between Cases 1 and 2 were statistically significant. At the 95% confidence level, the magnitude of the mean z-position error in the attitude fed case was 0.068 to 0.047 meters less than in the non-attitude fed case. Similarly, a two-sample F-test showed that the variance of this error was likely somewhat smaller in the attitude fed case. This result is the opposite of what was observed in simulation. The 95% confidence interval for the ratio of the variance in Case 1 to the variance in Case 2 was [1.026, 1.743].

Additionally, the mean y-position error was lower in Case 2. This can likely be attributed to the reduction in z-position error. A paired t-test showed that y-position error differences were statistically significant. At the 95% confidence level, the magnitude of the mean y-position error in the attitude fed case was 0.024 to 0.038 meters less than in the non-attitude fed case. Differences between other means and variances were not found to be statistically significant.

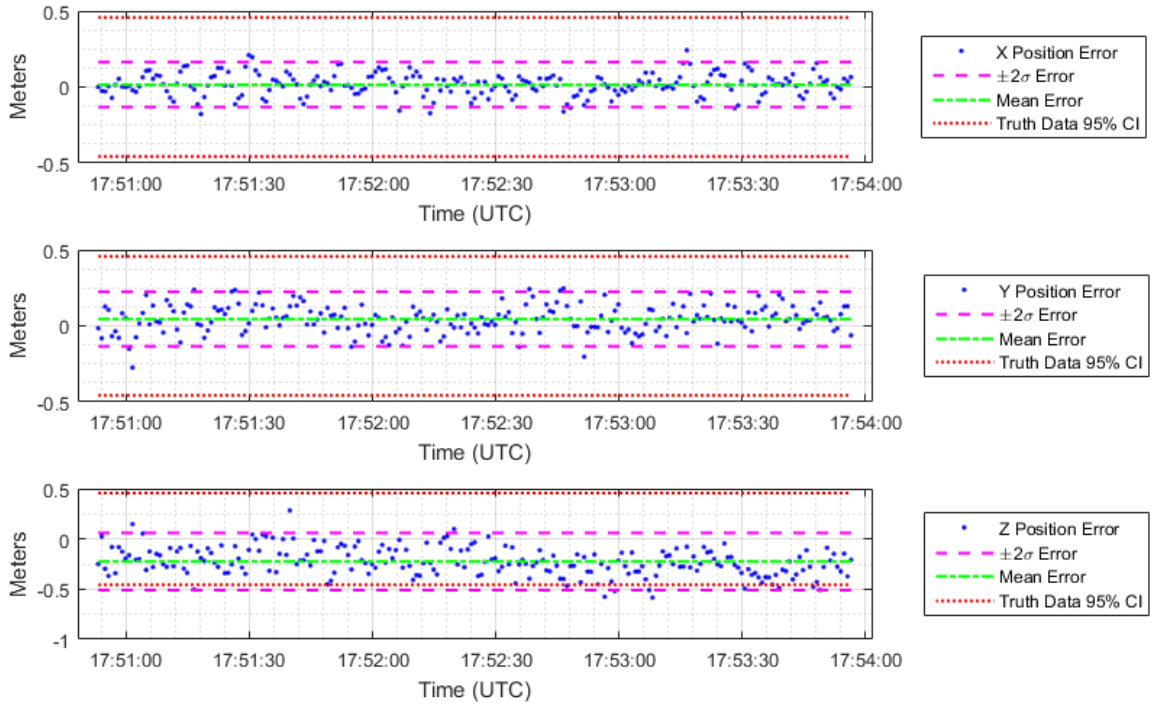


Figure 132. R7 Position Errors in Flight Test, Decimated Data Set, Attitude Feeding, Left Camera Frame.

Overall, the attitude fed variant of the R7 algorithm reduced mean errors and error standard deviations. Although these reductions were only on the order of centimeters, error reductions of this magnitude are significant in an AAR application due to the precision required. Most importantly, these differences were statistically significant in terms of error in the depth dimension (left camera frame z-position). Errors in depth directly contribute to errors in other position estimate components as detailed in Section 2.3.6.6. Hence, based on the results from this data set, the attitude fed variant of R7 results in better performance than the non-attitude fed variant.

6.2.4 R7 Flight Test Processing Times.

Image processing times in the 3DVW were longer when reading flight test images from disk than when using simulation generated imagery. It took approximately 69.75

seconds to process the 221 observation images analyzed in the decimated data set, an average of 0.316 seconds per image. In a system designed to work in real-time, these processing times would need to be reduced. However, no attempt was made in this test effort to optimize processing times. In the future, speed improvement could be accomplished in part by directly feeding captured imagery into R7 image processing routines or, more generally, with architecture specifically designed to optimize speed.

MATLAB® processing times were similar to those observed in simulation, requiring 144.08 seconds to process 5,501 point clouds returned from the 3DVW. This is an average of 0.026 seconds per point cloud. The R7 algorithm's speed, in absence of any speed optimization efforts, would ease implementation in a real-time system.

6.3 V5 Data and Analysis, Flight Test

The same decimated set of 221 flight test observation images used to analyze the R7 algorithm were used to analyze the V5 algorithm. As described in Section 5.4.5, a reference image database was created specifically for comparison against flight test observation images. Errors were analyzed in the left camera frame. An integrity risk level of 0.05 was utilized in the analysis. With this integrity risk level and sample size of 221, an average of 11.05 protection level violations would be expected.

Based on the simulation results outlined in Section 4.1, two configurations of the V5 algorithm were tested against flight test data. Table 30 describes these two configurations. The uniform prior probability distribution was identical to that used with simulation data. By contrast, the Gaussian prior did not make use of truth data, but instead relied upon the V5 algorithm's estimate at the last epoch.

With respect to the Gaussian prior, $\boldsymbol{\mu}_i$ is the mean of the prior probability distribution used on sample i, $\hat{\mathbf{x}}_{i-1}$ is the composite position estimate from sample $i-1$,

and $\boldsymbol{\Sigma}$ is the covariance matrix. When using the Gaussian prior, the first sample used a uniform prior probability distribution identical to that used with simulator data.

Table 30. Configurations of the V5 Algorithm Used on Flight Test Data.

Prior Probability Distribution	Likelihood Function	Pixel Intensity Threshold	Relative Position Estimator	Integrity Risk Level
Uniform	Gaussian $\mu = 0$ $\sigma = 0.1$	0.05	Composite	0.05
Gaussian $\boldsymbol{\mu}_i = \hat{\mathbf{x}}_{i-1}$ $\boldsymbol{\Sigma} = \begin{bmatrix} 1.1 & 0 & 0 \\ 0 & 1.1 & 0 \\ 0 & 0 & 1.1 \end{bmatrix}$	Gaussian $\mu = 0$ $\sigma = 0.1$	0.05	Composite	0.05

6.3.1 Case 1: Uniform Prior.

In Case 1 a uniform prior probability distribution was used to apply the V5 algorithm to flight test data. Figure 133 shows that mean spherical error was less than a meter, and that no protection level violations were observed. Using this configuration, protection levels greatly exceeded error levels as can be seen in Table 31. Since these results were returned with from an integrity risk level of 0.05, this implies that the PMF resulting from this configuration was assigning too great a probability mass to relative positions distant from the estimated position. However, spherical error magnitudes were similar to those observed in simulation. Table 31 presents the means and standard deviations of these error components. Table 32 presents the 95% confidence intervals of the mean spherical error and the mean protection level.

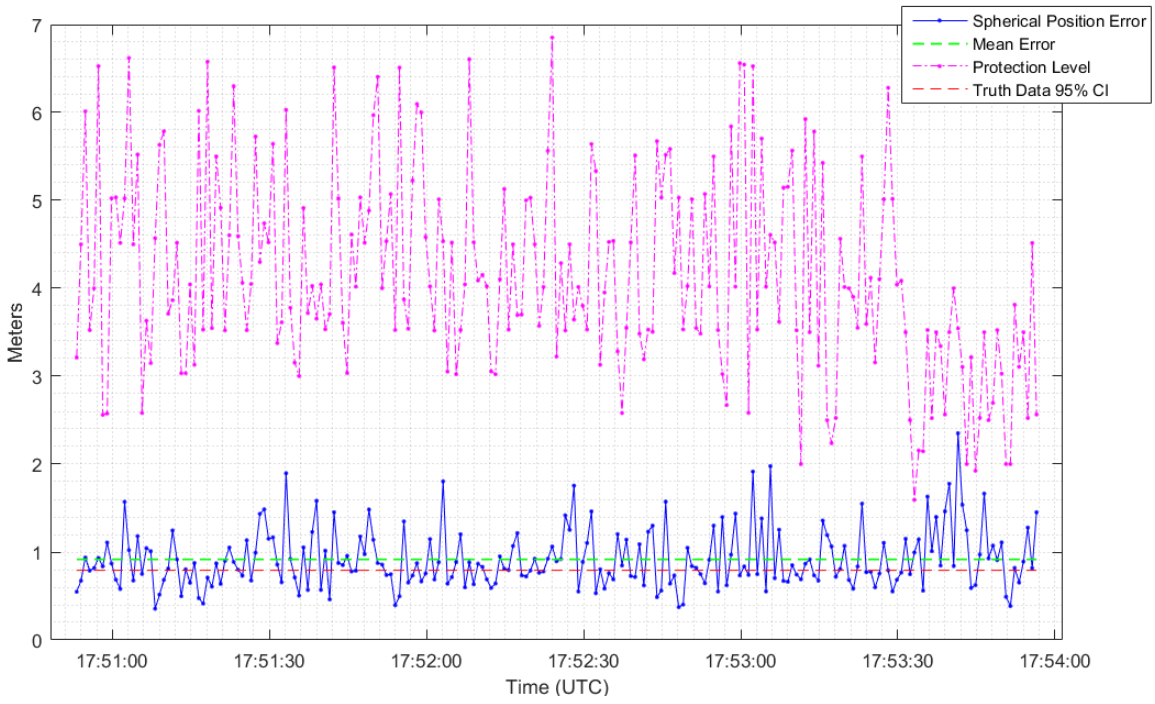


Figure 133. V5 Spherical Position Error on Flight Test Data Using a Uniform Prior.

Figures 134 and 135 show the left camera frame component position errors returned in this case. As the figures show, there was a strong x-position bias. This phenomenon was not observed in simulation where the V5 algorithm was an essentially unbiased estimator of x and y-positions in the camera frame. As with simulation data, the most variability was present in the z-position errors. However, the mean error in this case was very close to zero. Table 31 presents the means and standard deviations of these error components. Table 32 presents the 95% confidence intervals of the mean position errors.

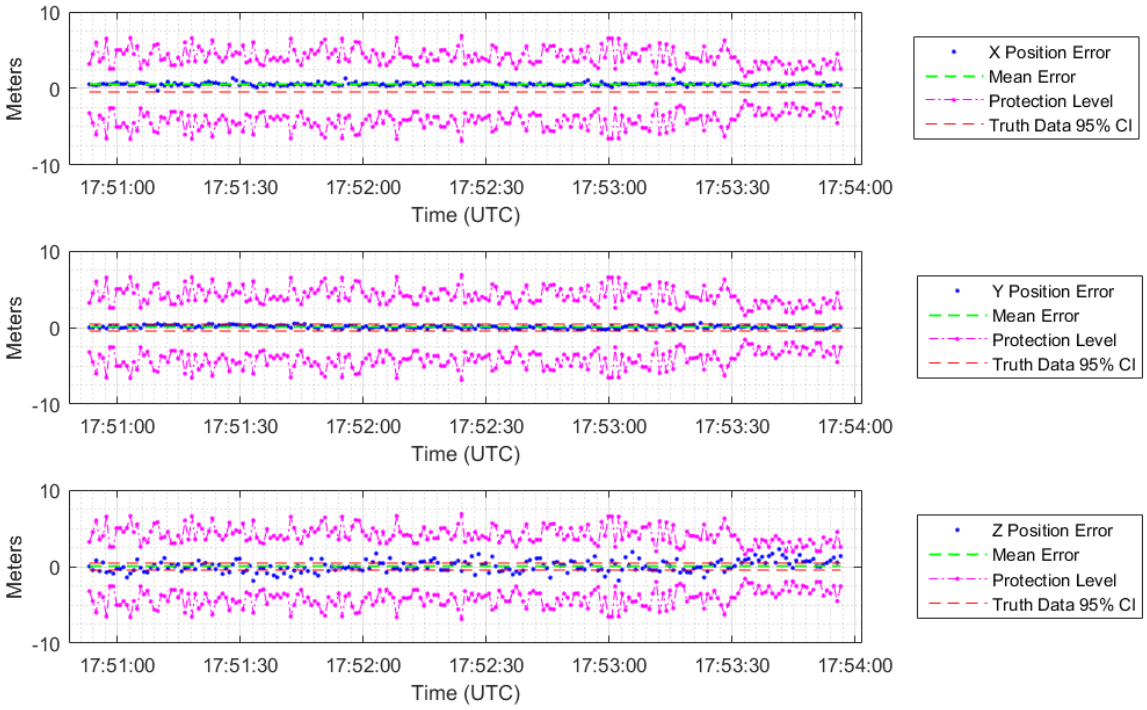


Figure 134. V5 Position Error on Flight Test Data Using a Uniform Prior, Left Camera Frame.

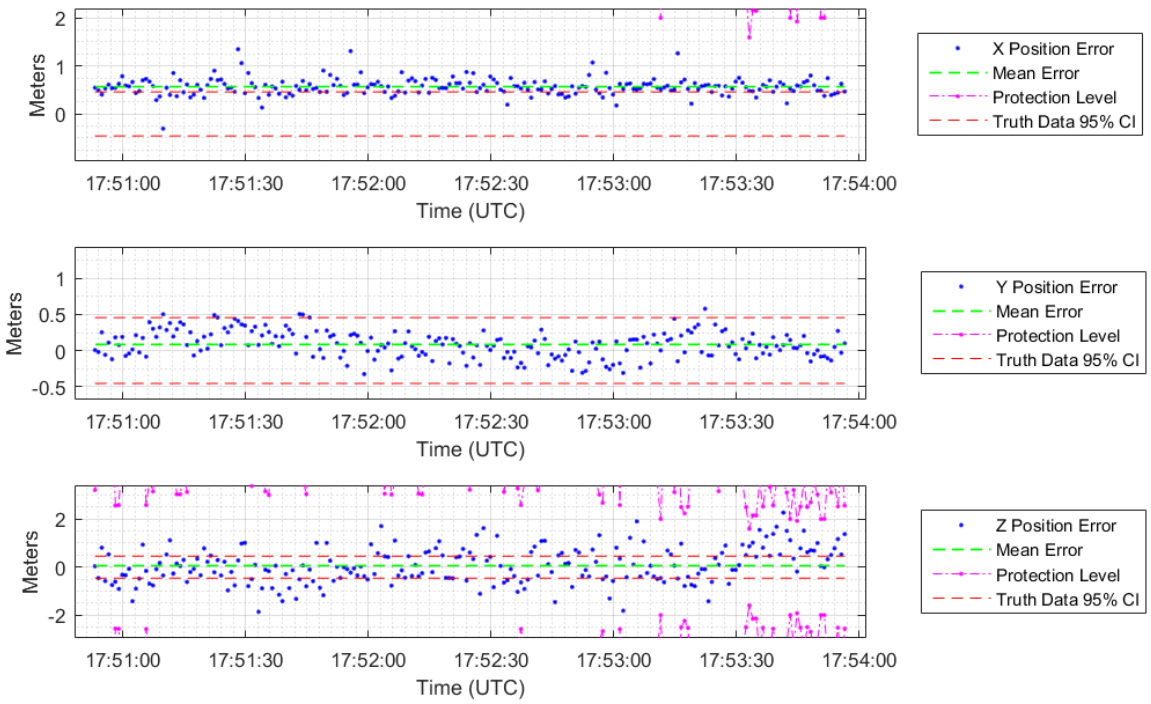


Figure 135. V5 Position Error on Flight Test Data Using a Uniform Prior, Zoomed In, Left Camera Frame.

Table 31. Flight Test Results for the V5 Algorithm Using the 221 Decimated Samples, Left Camera Frame.

	Uniform Prior		Gaussian Prior	
	Mean	Standard Deviation	Mean	Standard Deviation
Spherical Error (m)	0.917	0.337	1.018	0.430
x Position Error (m)	0.567	0.180	0.535	0.124
y Position Error (m)	0.084	0.185	0.113	0.181
z Position Error (m)	0.063	0.746	0.178	0.919
Protection Level (m)	4.152	1.143	2.462	0.682
Protection Level Violations	0		11	

Table 32. Mean Error and Mean Protection Level 95% Confidence Intervals for the V5 Algorithm on Flight Test Data, Left Camera Frame. Intervals Indicated with a "*" Overlap with the 95% Confidence Interval of the Truth Data Mean Error.

	95% Confidence Interval of the Mean (m)		
	Case 1: Uniform Prior	**Case 2: Gaussian Prior**	**Test Case: Uniform Prior, Reference Image Pixel Sampling Only**
Spherical Error	[0.872, 0.961]	[0.961, 1.075]	[0.904, 1.003]
X Position Error	[0.544, 0.591]	[0.519, 0.552]	[0.533, 0.584]
Y Position Error	[0.059, 0.108]	[0.089, 0.137]	[0.094, 0.142]
Z Position Error	[-0.036, 0.162]*	[0.056, 0.300]	[-0.140, 0.075]*
Protection Level	[4.001, 4.304]	[2.371, 2.552]	[3.388, 3.702]

6.3.1.1 Flight Test PMF Analysis.

Figure 136 shows an example PMF generated with flight test images. A bias in the positive y-direction of the v-frame (equivalent to positive x-direction in the left camera frame) is present. Neglecting the bias, most of the probability mass is concentrated along the camera line-of-sight. The resulting distribution was multi-modal. These two properties were also observed in simulation. As a result of the multi-modal nature of the distribution, on average the maximum likelihood estimator was an inferior estimator to the composite position estimator. This is discussed further in Section 6.3.1.6. Additionally, though not presented in this section, use of the combined PMF resulted in better performance than single camera PMFs.

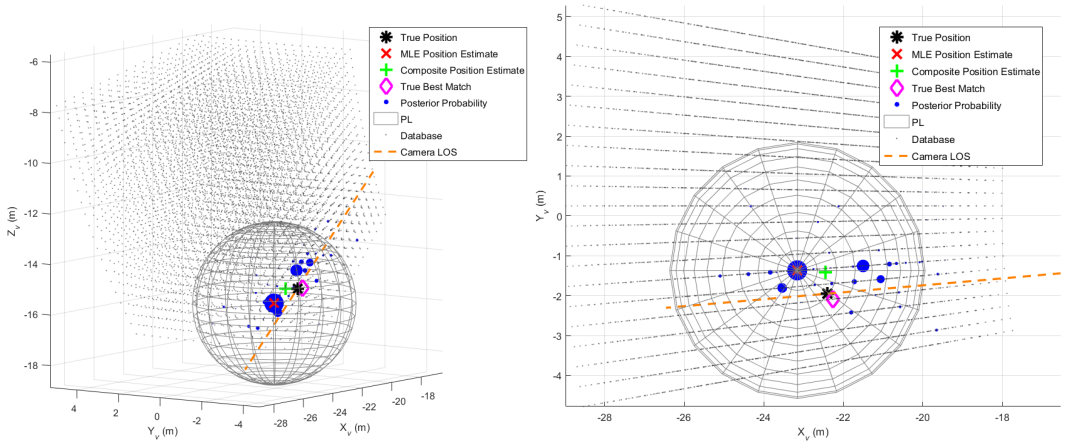


Figure 136. Example V5 PMF and V5 Position Estimates Returned from Flight Test Data.

6.3.1.2 Test Case.

In an attempt to address the observed biases, flight test data was analyzed with an altered form of the V5 algorithm. In this configuration, pixel intensity differences were only evaluated at pixels corresponding to edges in the reference image. This pixel sampling scheme was intended to reduce the error induced by noise present in the sample images. The results from this scheme are listed in Table 32 in the "Test Case" column.

As can be seen in the table, error results proved to be statistically similar to those observed in Case 1. However, reduced protection levels were observed (which also resulted in two protection level violations). The V5 likelihood function and thresholds were not optimized for only sampling based on edge pixels identified in the reference images. Despite these limitations, the Test Case performed essentially as well as Case 1 and seemed to better estimate protection levels. These results indicate that the Test Case pixel sampling scheme could improve performance in the future. The results also indicated that pixel sampling was an unlikely source of bias.

6.3.1.3 Bias Analysis.

As can be seen in Table 32, only the z-component of the V5 position estimate was likely unbiased in Case 1. One contributing factor to the non-zero mean errors observed in the x and y axes was the discrete nature of the rendered image database. At a range of 30 meters, each adjacent x and y-position in the database was separated by approximately 0.52 meters. As was discussed in Sections 2.2 and 3.4, the discretized nature of the database limits achievable position resolution even when using the composite position estimator. Thus, much of the y-position bias could be attributed to database resolution limits. However, the magnitude of the x-position bias suggests that other causes likely increased mean error.

Figure 137 shows a flight test reference image and a flight test observation image. The reference image shown most closely depicts the position captured by the observation image. As can be seen in the figure, a fair amount of noise is present in the observation image. Additionally, the right wing of the receiver did not manifest in the processed observation image. The noise present in the processed observation images collected in flight test could drive algorithm error, but other causes are likely more dominant as will be detailed below. An improved image processing sequence specifically designed to eliminate noise could yield superior results in the future. This could be accomplished with better speckle filtering or better tuning of other disparity map algorithm parameters described in Sections 2.3.6.5 and 3.5.3. Image processing for the V5 algorithm was performed in MATLAB®. Utilizing other image processing libraries, such as the OpenCV library used with the R7 algorithm, could yield improved results.

Figure 138 superimposes the processed left camera observation image (sample 20) with the reference image that most closely models the true position of the receiver. This reference image is referred to as the "true best match." Figure 139 superimposes

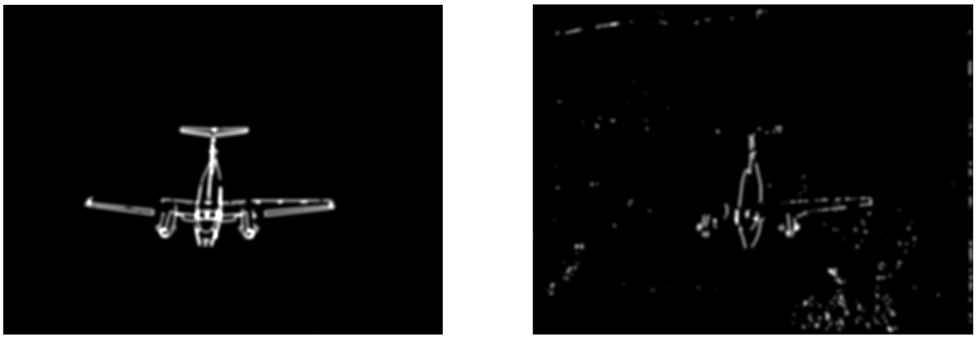


Figure 137. At Left, Reference Image Corresponding to the Observation Image Shown at Right.

the same observation image with the reference image that most closely models the composite position estimate. Figure 140 superimposes the same observation image with the reference image that yielded the highest posterior probability. The position depicted by this reference image would be the position solution returned when using the maximum likelihood estimator.

In this case, error was actually significantly smaller when using the composite position estimator (0.515 meters spherical error) than when using the maximum likelihood estimator (2.593 meters spherical error). This is the opposite of what is suggested by the figures. In the left camera frame, the composite and maximum likelihood position estimate errors in this case were:

$$\text{Composite Position Error (m)} = [0.364, 0.321, 0.173]^T \tag{127}$$

$$\text{Maximum Likelihood Error (m)} = [0.612, 0.527, 2.464]^T \tag{128}$$

Based on the figures, the V5 algorithm did an excellent job of identifying an image with a high pixel-to-pixel correspondence as the maximum likelihood image. This implies that the noise present in the observation images was not a major source of

error. However, this reference image depicts a position that is actually significantly different than the position captured in the observation image. The maximum likelihood estimate also strongly exhibited the depth ambiguity problem, as can be seen in the z-component error. Additionally, based on the Figure 139, the x-position error of the composite position solution should be negative and not positive. This is because the the solution image is to the left (negative left camera frame x-direction) of the observation image.

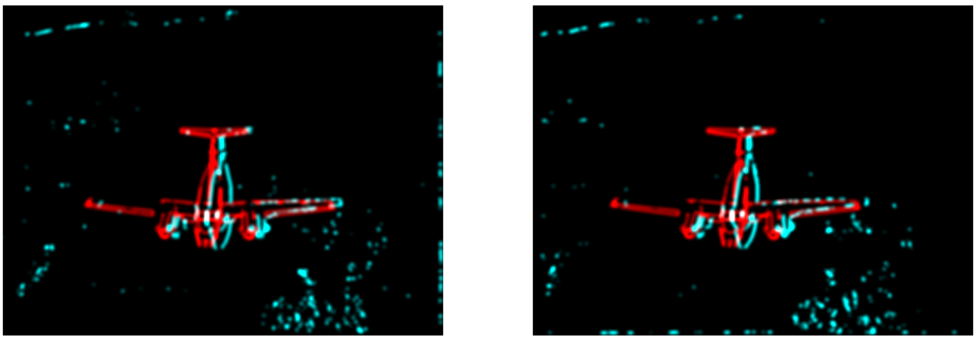


Figure 138. True Best Match Reference Images Superimposed Onto Observation Images. Reference Images are Shown in Red, Observation Images are Shown in Blue.

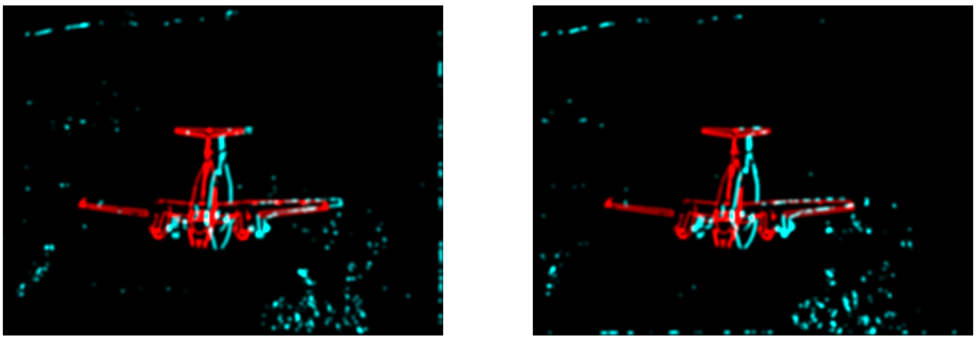


Figure 139. Reference Images Most Closely Depicting the Composite Position Estimate Superimposed Onto Observation Images. Reference Images are Shown in Red, Observation Images are Shown in Blue.

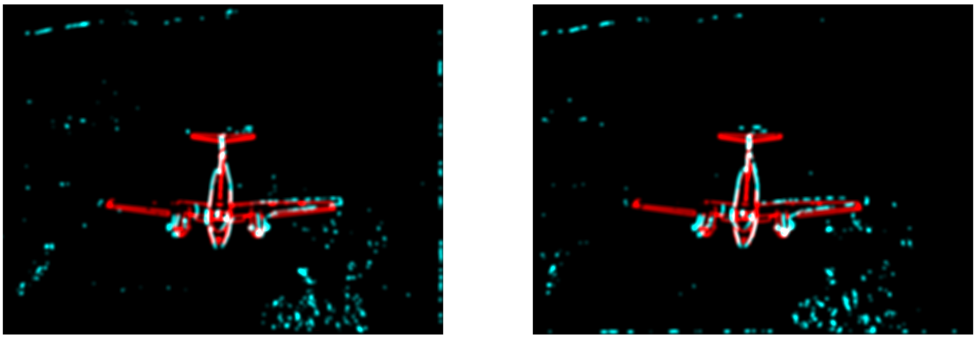


Figure 140. Reference Images for the Maximum Likelihood Estimate Superimposed Onto Observation Images. Reference Images are Shown in Red, Observation Images are Shown in Blue.

These discrepancies can most likely be attributed to one of two causes. First, as outlined in Section 5.4.5.2 different rectification mappings were applied to the reference and observation images. This difference existed because the 3DVW cameras used to generate the reference images did not perfectly model the optics of the real-world cameras. As is discussed in Section 2.3.6.4, both the rectification DCMs and the rectified camera calibration matrices are used to generate the pixel mapping for image rectification. These parameters depend upon both the stereo camera geometry as well as the intrinsics of both stereo cameras. Hence, two stereo camera systems that exhibit either different geometry or different camera intrinsics will have different rectification mappings. In the future, this source of error could be overcome with a reference image database that better models the optics of the real-world cameras collecting the observation images. Otherwise, systematic error, as was observed in flight test, is likely to result. Rectification mapping could be confirmed as a primary source of error by comparing the mapping functions applied to the reference images to the those applied to the observation images. Due to time constraints, this analysis was not undertaken in this thesis.

By contrast, simulation results yielded no such bias in the left camera frame x or y-position estimates. In simulation the rectification mapping used to generate the reference and observation images were identical and the extrinsic camera calibrations were known perfectly. Hence, results suggest that differences in rectification mapping were likely a primary contributors to V5 estimate bias.

Second, errors in the orientations measured in the extrinsic camera calibrations would cause the images in the reference image database to have an an incorrect perspective. Since a flight test specific reference database was generated based on these extrinsic measurements, errors would directly lead to bias in the V5 algorithm. However, were substantial extrinsic camera calibration errors present, the R7 algorithm would have exhibited similar biases. Since the mean R7 x and y-errors in the left camera frame were small, extrinsic camera calibration error is not a likely source of bias in this case.

Overall, observation image noise, while problematic, was likely not a major contribution to algorithm error. Instead, the problem is likely related to the reference image database. Generating a reference image database that appropriately models real-world observation images is a significant challenge to V5 algorithm development. This challenge is caused by the fact that the V5 algorithm utilizes image rectification to obtain a disparity map. Image rendering and template matching approaches that utilize a single camera do not encounter the same problem since no disparity map is generated. However, as discussed in Section 3.4.3 when using a camera configuration that points to a region of significant clutter, a disparity map is essential to eliminating image features other than the receiver aircraft. The image rectification mappings used to generate these disparity maps will differ between the reference and observation image sets unless the simulated cameras used to generate the reference set perfectly model the real-world cameras. Future efforts on the algorithms similar

to the V5 algorithm should focus on ensuring that a reference image database can be generated that appropriately models real-world observation imagery.

6.3.1.4 Processing Times.

Processing times similar in magnitude to simulation were observed with flight test data. Figure 141 shows the processing times for flight test data examined in Case 1. These processing times reinforce the conclusion that the V5 algorithm would require a substantial improvement in speed to be implementable in a real-time system.

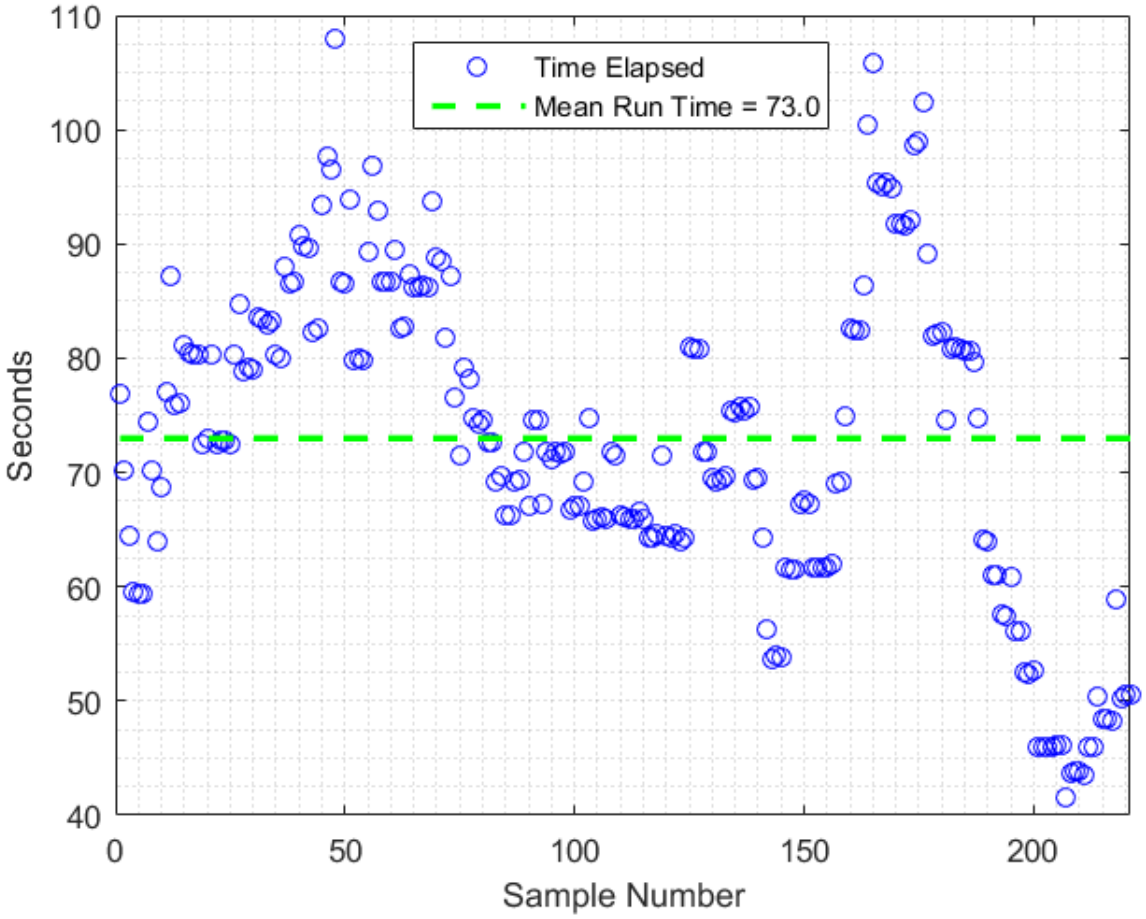


Figure 141. V5 Processing Time on Pre-Processed Flight Test Images.

6.3.1.5 Integrity Risk Level Effect.

As expected, higher integrity risk levels led to higher protection levels. Figure 142 shows this result for integrity risk levels of 10^{-6} and 0.05. For the 10^{-6} case, the mean and standard deviation of the protections level were 5.717 meters and 1.011 meters, respectively. For the 10^{-6} case, the mean and standard deviation of the protections level were 4.152 meters and 1.143 meters, respectively. A two-sample F-test showed that the protection level variances for these two integrity risk levels did not have a statistically significant difference. This reinforces the conclusion that protection level fluctuations of similar magnitude would be expected when using integrity risk levels of 0.05 or greater.

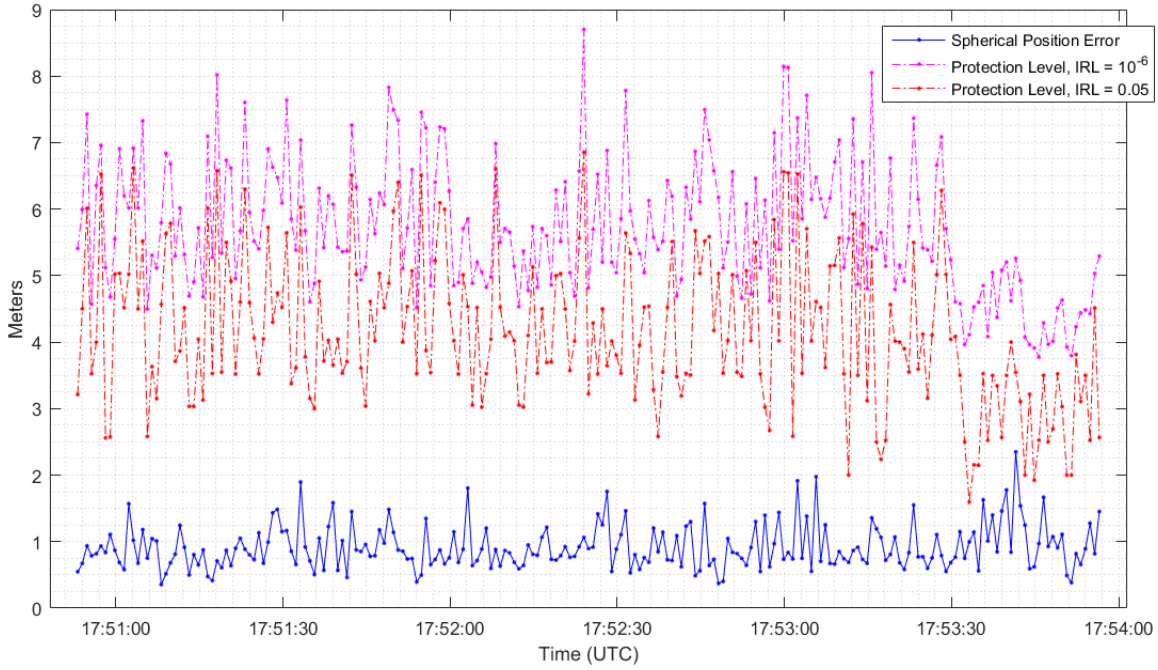


Figure 142. Integrity Risk Level Effect on the V5 Protection Level in Flight Test.

6.3.1.6 Maximum Likelihood Estimator.

As was seen in simulation, the maximum likelihood estimator performed worse, on average, than the composite position estimator. Figure 143 and Table 33 show this

result. Both mean error and the standard deviation of the error were higher when using the maximum likelihood estimator. As was discussed in Section 6.3.1.3, much of the translational error (x and y-dimensions in the left camera frame) could be attributed to differences in rectification mapping between the reference and observation images or to errors in the extrinsic camera calibrations.

Besides the translational errors, depth ambiguity remained a problem when using the maximum likelihood estimator. This could be overcome by using a reference image database that deliberately staggers images at different depths. Using this scheme, the location depicted by each image in the reference image database would fall on a unique vector from the camera origin. This would help eliminate depth ambiguity since the depiction of a receiver in any given reference image would overlap less strongly with the depictions in other reference images that fall along similar line-of-sight vectors.



Figure 143. V5 Spherical Position Error on Flight Test Data Using a Uniform Prior and the Maximum Likelihood Estimator.

Table 33. Comparison of Maximum Likelihood Estimator and Composite Position Estimator Errors, Left Camera Frame, Flight Test Data.

	Maximum Likelihood Estimator		Composite Position Estimator	
	Mean	**Standard Deviation**	**Mean**	**Standard Deviation**
Spherical Error (m)	1.766	0.887	0.917	0.337
X Position Error (m)	0.565	0.405	0.567	0.180
Y Position Error (m)	0.112	0.373	0.084	0.185
Z Position Error (m)	0.197	1.802	0.063	0.746

6.3.2 Case 2: Gaussian Prior.

Case 2 utilized Gaussian prior scheme described in Table 30. This scheme differed significantly from the informative priors used with simulation data.

Rather than setting the mean of the prior to the known true relative position of the receiver, the mean of the prior was set to the V5 algorithm's most recent composite position estimate. In doing so, the algorithm was not artificially relying on truth data in order to generate a PMF with an informative prior. The accuracy achievable from this scheme was limited by the fact that it did not account for any receiver or tanker dynamics in the time between samples. Properly accounting for these dynamics would improve performance.

The Case 2 covariance matrix was identified in a tuning process designed to yield 11 protection level violations, since on average 11.05 violations would be expected for an integrity risk level of 0.05 and 221 samples. The covariance matrix was chosen in this fashion to facilitate a qualitative assessment of how the V5 protection level was

affected when using an informative prior with flight test data. A diagonal covariance matrix was chosen to simplify the tuning process.

Figure 144 shows the spherical error and protection level resulting from the use of this prior. As can be seen in the figure, protection levels were substantially lower than when using a uniform prior. Additionally, there appears to be greater time correlation of errors. This result is expected because the relative position of the receiver was fairly stable during data collection. Since formation dynamics were not modeled, any estimated position biased the next several estimates toward its estimate resulting in time correlation. Table 31 presents the means and standard deviations of these spherical error and protection level. Table 32 presents the 95% confidence intervals of the mean spherical error and the mean protection level. Both these tables are found in Section 6.3.1.

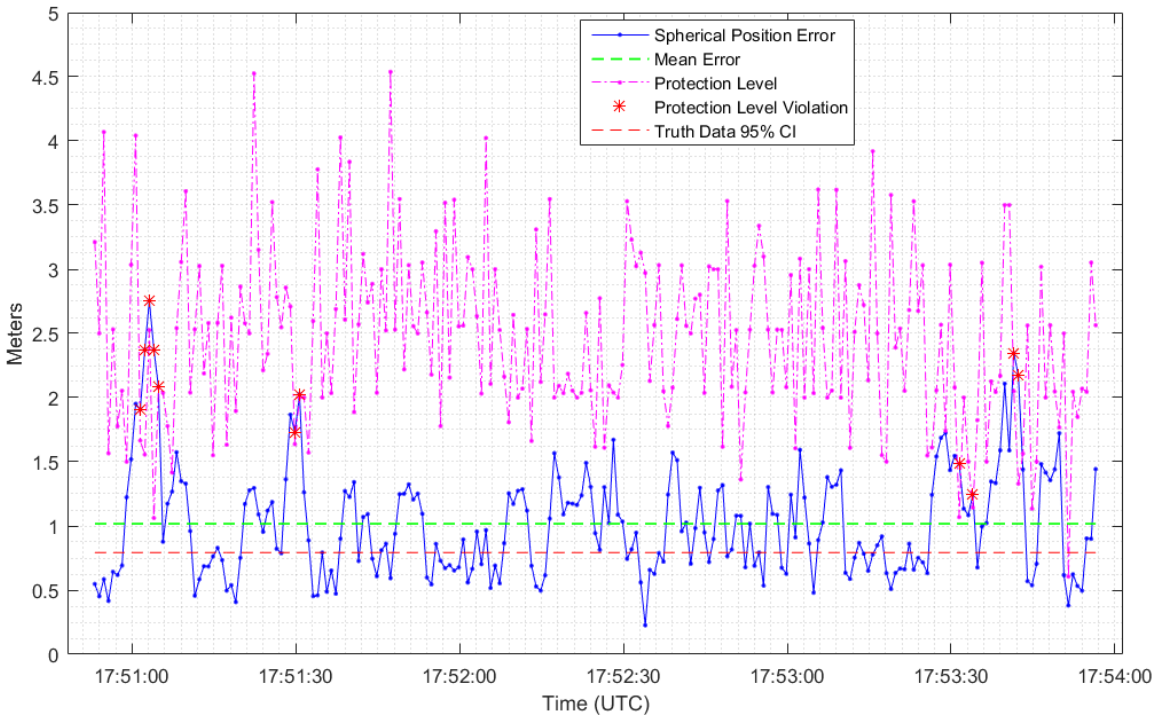


Figure 144. V5 Spherical Position Error on Flight Test Data Using a Gaussian Prior.

Figures 145 and Figure 146 shows the position errors resulting from this prior. As can be seen in the figures, most variability and time correlation were present in the z-position error, and the x-position error exhibited the same bias observed with a uniform prior. Additionally, most of the protection level violations can be attributed to z-estimate errors. Table 31 presents the means and standard deviations of these error components. Table 32 presents the 95% confidence intervals of the mean position errors.

The data show that V5 errors were fairly similar in Case 1 and Case 2. However, protection levels and protection level variance were significantly reduced when using the Gaussian prior. A paired t-test showed that Case 2 had lower mean protection levels than Case 1 at a confidence level $\gg 99\%$. The 95% confidence interval for the protection level reduction in Case 2 was 1.546 meters to 1.835 meters. A two-sample F-test showed that the 95% confidence interval for the ratio of the uniform prior protection level variance to the Gaussian prior protection level variance was 1.546 to 1.835.

Overall, Case 2 results indicate that an informative prior can significantly improve V5 protection level computations without much increase in error, even when failing to account for formation dynamics. Accounting for formation dynamics would likely lead to significant performance improvement when using the Gaussian prior. Additionally, the informative prior in Case 2 was limited by the fact that its search space was centered on the V5 estimate from the last epoch. Sequential epochs were separated in time by approximately 0.83 seconds and formation dynamics in this interval were typical of what could be expected from experienced pilots performing AR. Despite these limitations, the true aircraft relative position of each sample fell within the attenuated reference image database which was centered on the relative position estimate from the previous epoch. This result is significant because it sug-

gests that reference image database attenuation is a feasible approach in a real-time system. Results would be expected to improve significantly if the system accounted for formation dynamics.

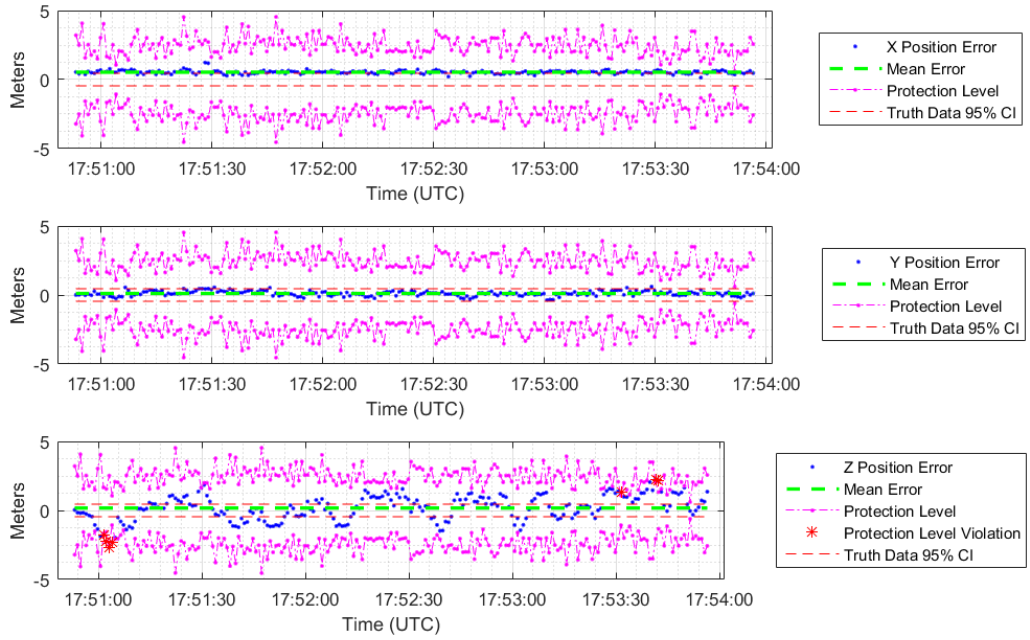


Figure 145. V5 Position Error on Flight Test Data Using a Gaussian Prior, Left Camera Frame.

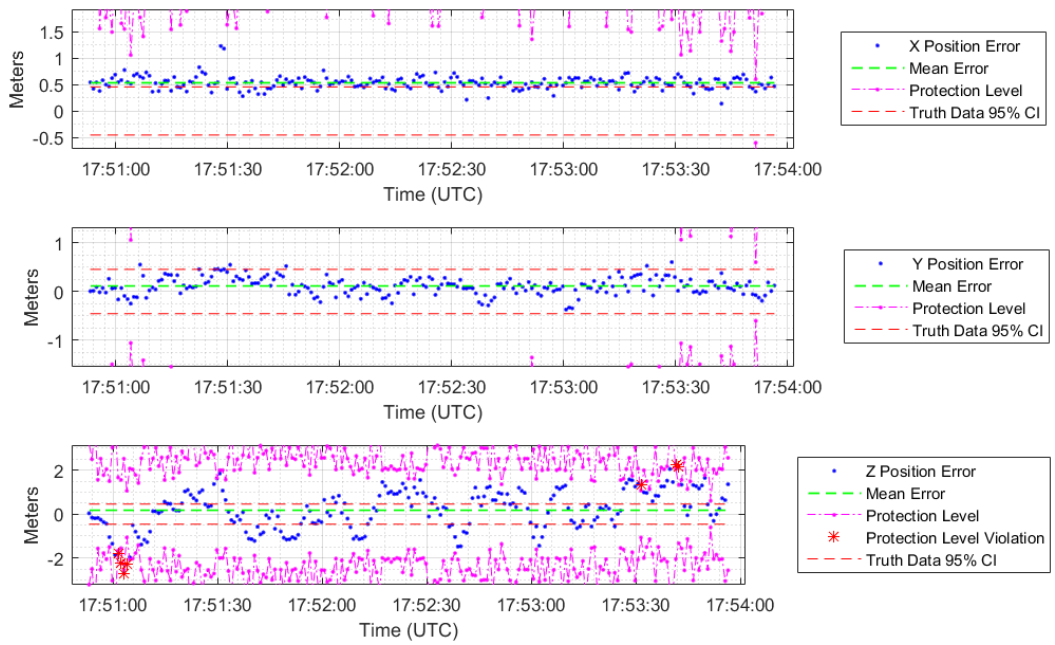


Figure 146. V5 Position Error on Flight Test Data Using a Gaussian Prior, Zoomed In, Left Camera Frame.

6.4 R7 and V5 Comparison

In terms of speed, both flight test and simulation results supported the conclusion that as implemented the R7 algorithm is significantly faster than the V5 algorithm. However, no effort at speed optimization has been attempted in either case.

Figure 147 plots the spherical errors observed in flight test for the R7 and V5 algorithms. This comparison was made using the non-attitude fed version of R7 (Case 1) and in the uniform prior configuration of the V5 algorithm (Case 1). As can be seen in the figure, R7 errors were substantially lower overall than V5 errors. Much of this difference can be attributed to the x-bias that was present in the V5 algorithm as well as the greater variance in that algorithm's estimate errors.

Table 34 presents the 95% confidence intervals of both algorithm's errors, obtained with a one-sample t-test. The table also shows 95% confidence interval resulting from

a paired t-test comparing the R7 and V5 errors. In the final column, the results of a two-sample F-test are shown, comparing the variance observed in the two algorithms.

The data show that overall all, the R7 algorithm had lower mean error, at a statistically significant level, in the x-dimension of the left camera frame. This result could be attributed to the unresolved bias in the V5 algorithm. The R7 algorithm also had more than 0.5 meters less spherical error than the V5 algorithm at a statistically significant level. On the other hand, the V5 algorithm had lower mean error, at a statistically significant level, in the z-dimension of the left camera frame. This result could be attributed to the unresolved bias in the R7 algorithm. All error variances, spherical and in each position, were lower at a statistically significant level with the R7 algorithm.

These results indicate that the R7 algorithm was overall more accurate and had less variability than the V5 algorithm. Both algorithms were prone to some degree of bias in certain dimensions. Additionally, V5 variances may have been smaller if a more dense reference image database had been utilized. However, a more dense database would have resulted in longer processing times.

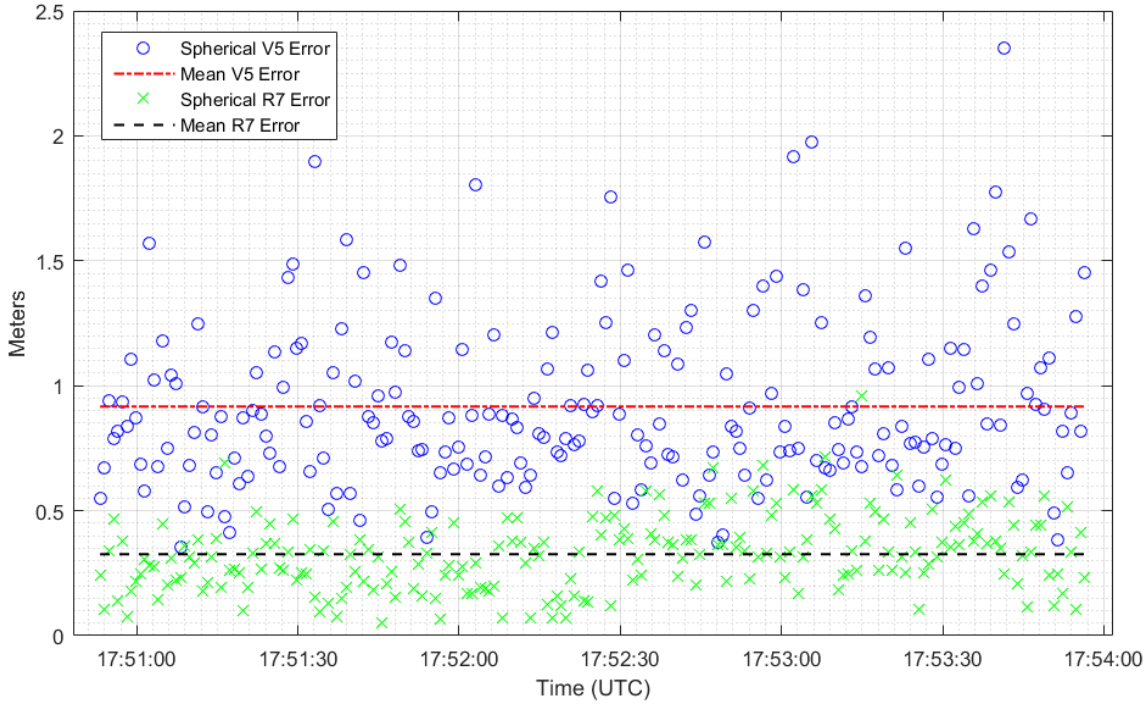


Figure 147. Comparison of R7 and V5 Spherical Errors in Flight Test, Left Camera Frame.

Table 34. Statistical Comparison of R7 and V5 Error Distributions, Flight Test.

	95% Confidence Interval			
	R7 Mean, No Attitude Feeding (m)	**V5 Mean, Uniform Prior (m)**	**R7 - V5 Mean (m)**	**Ratio of R7 Variance to V5 Variance**
Spherical Error	[0.307, 0.346]	[0.872, 0.961]	[-0.673, -0.544]	[0.147, 0.249]
X Position Error	[0.005, 0.0256]	[0.544, 0.591]	[-0.576, -0.529]	[0.147, 0.250]
Y Position Error	[0.063, 0.089]	[0.059, 0.108]	[-0.033, 0.017]	[0.199, 0.338]
Z Position Error	[-0.305, -0.261]	[-0.036, 0.162]	[-0.450, -0.242]	[0.037, 0.063]

VII. Conclusions and Recommendations

In this thesis, two relative navigation measurement algorithms, V5 and R7, were designed and analyzed. Both algorithms were developed in the context of AAR and were designed to be applicable to the KC-46 program. More generally, both algorithms provide relative navigation measurements from a stereo camera system. The AAR context was chosen to facilitate analysis and to support development of a robust USAF AAR capability that is not reliant on the GPS constellation.

Both algorithms were analyzed in simulation and in flight test. This chapter summarizes the major findings and contributions resulting from these efforts. Commensurate with these findings, recommendations are made on how to improve algorithm operation and on areas to focus on in future research. These recommendations are made with the goal of implementation in a real-time system in mind. This is done first for the V5 algorithm and second for the R7 algorithm. The chapter concludes with a rough sketch for an algorithm that would leverage the strengths of the V5 and R7 algorithms. This sketch is intended to suggest a possible way forward in developing an algorithm capable of providing a relative navigation measurement as well as an independent assessment of navigation integrity in real-time.

7.1 V5 Algorithm Contributions and Recommendations

The V5 algorithm was designed principally by building upon Calhoun's Bayesian inference integrity monitor developed in [2] and [5]. Utilization of Calhoun's Bayesian inference techniques enabled the V5 algorithm to provide an independent measure of navigation integrity in the form of a protection level. However, the AAR problem addressed by the V5 algorithm differed from the AAR problem addressed by Calhoun in two key ways. First, in V5 development, camera systems were considered to be

mounted on a tanker aircraft looking down at a receiver aircraft. This geometry dictated that the V5 algorithm had to be designed to contend with background clutter. Second, the V5 algorithm examined the use of a stereo camera system. Based on this change to the problem space, the V5 algorithm was specifically designed to leverage the stereo camera configuration to filter out background clutter. This was achieved with stereo image processing techniques.

The V5 algorithm was analyzed with both simulation and flight test images. Overall, the V5 algorithm output PMFs that adequately characterized the uncertainty of its relative position estimates. This was evidenced by the fact that the expected number of integrity risk violations were observed during both simulation and flight test analysis. Regardless of the error levels achieved, this is a finding of importance. It demonstrates that stereo vision-based Bayesian inference can be used to obtain navigation integrity (via protection levels) even in the presence of substantial background clutter.

For reference in the discussion below, a subset of V5 simulation and flight test results are summarized in Table 35. These results were generated with the V5 configuration specified in Table 36. Based on simulation and flight test data analysis, this thesis yielded seven primary contributions with respect to the V5 algorithm. These contributions are outlined in the subsections below and are accompanied by commensurate recommendations.

7.1.1 Image Processing.

First, the V5 stereo image processing technique was shown to effectively eliminate background clutter. The resultant images were shown to be edge-space images of the target of interest. In the simulation environment, the resultant images were free of noise stemming from the background or poor pixel matching. When using flight test

images, noise artifacts were present in the images, but analysis revealed that noise levels were likely not substantial enough to skew the the reference-observation image matching process. In future work, noise in the processed images could be eliminated with better speckle filtering or with point cloud filters similar to those developed with the R7 algorithm. R7 results, which performed image processing with OpenCV in C++, indicate that use of image processing libraries other than those found in MATLAB® may yield superior results.

7.1.2 PMF Estimation.

Second, analysis showed that the PMF estimated by probabilistically combining the left and right camera PMFs was superior to the PMF estimated by either camera in isolation. The combined PMF led to lower error levels and superior protection level estimation. In simulation, the combined PMF resulted in approximately 10 centimeters less mean spherical error than the best performing single camera PMF. Moreover, both single camera PMFs yielded extremely high numbers of integrity violations while the combined PMF yielded the expected number for the applied integrity risk level. The KC-46 program is designed to have two stereo camera pairs. Future work could examine algorithm gains from incorporating measurements from all four camera systems.

7.1.3 Relative Position Estimates.

Third, the composite position estimate developed in this thesis was shown to outperform the maximum likelihood estimate used in previous work. The composite position estimate operated by using the mean of the V5 PMF rather than the mode to estimate the relative position of the receiver aircraft. This resulted in substantial improvements in relative position estimation. Simulation results showed that use

of the composite position estimator reduced error variance in all components at a statistically significant level and reduced mean spherical error by nearly one meter. Similar results were observed in flight test, where mean spherical error was reduced by approximately 80 centimeters.

7.1.4 Pixel Intensity Thresholds and Protection Level Computations.

Fourth, analysis showed the criticality of identifying a pixel intensity threshold when using the V5 algorithm. In simulation, use of a 0.05 pixel intensity threshold was shown to concentrate probability mass along the camera system's line-of-sight. This effect substantially reduced variance in the camera frame x and y-component errors in comparison to the zero threshold case. Use of this non-zero threshold reduced mean spherical error by approximately 60 centimeters or 41%. Moreover, this threshold was shown to decrease mean protection levels by approximately 45 centimeters while still yielding an acceptable number of integrity risk violations.

Given the fact that probability mass was concentrated along the line-of-sight, an alternative protection level computation scheme could be developed that leverages this fact. Additionally, future work could estimate separate protection levels for each camera frame dimension. This would result in protection levels that provide more information to a user about which specific dimensions account for the bulk of algorithm uncertainty and hence which dimension poses the greatest risk for a safety violation. Future work could also leverage the superior left camera frame x and y-position estimation capabilities to improve z-position estimation. This could be accomplished by deliberately staggering the reference images generated at sequential depths. Using this scheme, the location depicted by each image in the reference image database would fall on a unique vector from the camera origin. This would help eliminate depth ambiguity since the depiction of a receiver in any given reference

image would overlap less strongly with the depictions in other reference images that fall along similar line-of-sight vectors.

7.1.5 Informative Priors.

Fifth, analysis showed that an informative prior can substantially improve protection level estimation even when failing to account for formation dynamics. When using an informative prior on the flight test data set, the mean protection level was 2.462 meters, a reduction of approximately 1.7 meters from the uniform prior case. The protection level standard deviation with the informative prior was 0.682 meters, a reduction of approximately 0.46 meters from the uniform prior case. Additionally, use of an informative prior on flight test data yielded an acceptable number of integrity risk violations and did not result in significantly worse errors. These results were achieved despite not taking formation dynamics into account between observations. This result is important because the key advantage of the V5 algorithm is the degree of navigation integrity it is designed to provide. Tighter, reliable protection levels are essential to its utility. Future work should focus on using an informative prior that is based on the output of a recursive estimation algorithm, such as a Kalman filter. In this manner, the prior would take into account formation dynamics. This would likely lead to smaller algorithm errors and smaller, but still reliable, protection levels.

7.1.6 Rectification Mapping and Flight Test Errors.

Sixth, flight test data revealed a persistent left camera frame x-position estimate bias. Differences between the reference image database and observation image rectification mappings were assessed to be the most likely source of this error. In future work rectification mapping could be confirmed as a primary source of error by comparing the mapping functions applied to the reference images to the those applied

to the observation images. Additionally, rectification mapping differences could be overcome by developing simulation cameras that are capable of closely imitating the optics of the real-world cameras. Preferably, mapping differences could also be overcome by adding an additional step to the image processing routine. In this step, the processed, rectified images would be re-mapped back into non-rectified images. In doing so, the background filtering properties of the image processing routine would still be retained. This would obviate the requirement to develop better simulation camera models.

7.1.7 Algorithm Processing Times.

Seventh, the V5 algorithm was shown to require extensive processing times. On a fundamental level, the multiple pixel sampling and template matching techniques at the heart of the V5 algorithm are time intensive. This is an obstacle to real-time implementation. To reduce processing times, future work should focus on developing selective sampling techniques that reduce processing times while still yielding acceptable PMF returns. Techniques applied in particle filters and unscented Kalman filters may offer insights [52].

Table 35. V5 Algorithm Simulation and Flight Test Results, Left Camera Frame.

	Simulation Results		**Flight Test Results**	
	Mean	**Standard Deviation**	**Mean**	**Standard Deviation**
Spherical Error (m)	0.896	0.527	0.917	0.337
X Position Error (m)	0.004	0.374	0.567	0.180
Y Position Error (m)	0.008	0.174	0.084	0.185
Z Position Error (m)	-0.197	0.935	0.063	0.746
Protection Level (m)	2.956	1.138	4.152	1.143

Table 36. Configuration of the V5 Algorithm for Data Shown in Table 35.

Prior Probability Distribution	Likelihood Function	Pixel Intensity Threshold	Relative Position Estimator	Integrity Risk Level
Uniform	Gaussian $\mu = 0$ $\sigma = 0.1$	0.05	Composite	0.05

7.2 R7 Algorithm Contributions and Recommendations

The R7 algorithm was designed to leverage well established stereo vision techniques and the ICP algorithm to obtain relative position and attitude measurements of an AR receiver. This approach was taken to facilitate real-time implementation in the future. Algorithm development also included the addition of several features to avoid negative effects from clutter, such as a boom filter. However, unlike the V5 algorithm, the R7 algorithm was not designed to provide an independent assessment of navigation integrity.

The R7 algorithm was analyzed with both simulation and flight test images. Overall, the R7 algorithm was shown to be capable of rapidly producing accurate relative position measurements. Additionally, R7 errors were shown to be low variance. These results bode well for implementation of similar approaches in a real-time system.

For reference in the discussion below, R7 simulation and flight test results are summarized in Table 37. The simulation results in the table were generated in Case 5 which simulated receiver depths equivalent to those observed in flight test. Based on simulation and flight test data analysis, this thesis yielded six primary contributions with respect to the V5 algorithm. These contributions are outlined in the subsections below and are accompanied by commensurate recommendations.

7.2.1 Depth Error.

First, depth error (error in the left camera frame z-position estimate) proved to be the largest driver of R7 relative position estimate error. Both simulation and flight test data revealed a persistent negative bias in the z-position estimate indicating that the R7 algorithm tended to underestimate the depth of the receiver. In flight test, this bias was approximately 30 centimeters. Depth error is problematic because it is the primary driver of error in the other two dimensions [32]. Analysis in this thesis suggested that disparity estimation error was likely a key driver of depth estimate error. Hence, future work should examine methods to improve disparity estimation. Particular attention should be paid to improved sub-pixel disparity estimation techniques.

7.2.2 Attitude Fed ICP.

Second, the attitude fed version of the ICP algorithm was shown to reduce R7 depth error and depth error variance in flight test. This reduction was approximately 0.06 meters in depth bias and a roughly 25% reduction in depth error variance. Despite these small magnitudes, this finding is significant because, as stated earlier, depth error is the primary driver of algorithm error. Reductions in depth error and error variance also lead to reductions in spherical error. Moreover, using the attitude fed version of the R7 algorithm reduced R7 processing times by approximately 5%. These speed savings could ease real-time implementation. Similarly, future work could develop an attitude bounded version of the ICP algorithm. This would be accomplished by setting a maximum allowable rotation limit. This could confine the attitude search space to a window more realistic for the AAR application and preclude estimation of unlikely attitudes.

7.2.3 Boom Filtering.

Third, simulation results with a boom in the camera fields of view showed that the R7 boom filter was effective at eliminating boom-related clutter in all but one of 6,000 test cases. R7 errors did increase when using the filter in comparison to cases examined with no boom in the fields of view. However, these increases in error means and standard deviations were not drastic; the largest increase in mean error was 0.2 meters in the x-component. Moreover, using the attitude fed version of ICP with a boom in the field of view essentially eliminated the increase in error biases as well as much of the increase in error variances. This finding indicates that the boom filter designed in this thesis is a good starting point for implementation in a real-world flight test. It also indicates that the attitude fed version of ICP may be important to reducing error in the presence of boom occlusion. Future work on boom filtering should focus on incorporating the actual extension, azimuth, and elevation data output from real-world USAF tanker booms. This could be done in simulation or with bus data from real-world boom systems. This would facilitate development of a boom filter capable of using boom measurements in real-time.

7.2.4 Flight Test Attitude Errors.

Fourth, flight test attitude error variances were substantially lower than those observed in simulation. While the flight test errors were also more biased, this result indicates that the R7 algorithm has greater potential to provide some attitude estimation capability than simulation results suggest. For instance, while the algorithm's attitude measurements may not be nearly as accurate as those obtainable from IMUs, they could provide a means to monitor for unusual attitudes. Future work on developing an attitude bounded version of the ICP algorithm could also improve R7 attitude estimates.

7.2.5 R7 Error Variance.

Fifth, simulation and flight test results showed that R7 algorithm errors had low standard deviations. In flight test, error variance was low enough that the mean R7 spherical and x-position estimate errors fell within the uncertainty of the truth data system. This finding indicates that R7, or a similarly constructed algorithm, has the potential to provide low uncertainty measurements to a Kalman filter or other algorithm. In order to be implemented in a filter, R7 algorithm errors need to be characterized. Appendix C suggests one possible method for error characterization in simulation. More generally, methods to characterize R7 and ICP error distributions should be examined including ones that take into account the stochastic nature of disparity measurements [53].

7.2.6 R7 Processing Times.

Sixth, the R7 algorithm was shown to have very rapid processing times. This finding indicates that an ICP based approach to stereo vision relative navigation (such as the R7 algorithm) is a good candidate to be implemented as a real-time measurement system. Future efforts on developing a real-time protection level estimate should leverage the speed inherent in the ICP algorithm. A rough sketch for an algorithm that does so is presented in Section 7.3.

Table 37. R7 Algorithm Case 5 Simulation and Flight Test Results, Left Camera Frame.

	Simulation Results		Flight Test Results	
	Mean	Standard Deviation	Mean	Standard Deviation
Spherical Error (m)	0.517	0.123	0.326	0.147
X Position Error (m)	-0.018	0.048	0.015	0.079
Y Position Error (m)	0.149	0.048	0.076	0.094
Z Position Error (m)	-0.491	0.118	-0.283	0.165
Roll Error (deg)	-0.796	7.907	0.498	1.315
Pitch Error (deg)	0.292	4.539	-2.614	1.105
Yaw Error (deg)	-0.074	1.300	1.144	0.630

7.3 Future Work: Leveraging ICP and Bayesian Inference

As was stated in Chapter I, the ultimate vision for both algorithms would be to incorporate their measurements into a vision-based AAR control system as depicted in Figure 148. In such a system, algorithm measurements would be taken from stereo image pairs captured at rates on the order of 30 Hertz. These measurements would be intended to replace GPS measurements in cases of GPS non-availability. In such a system, the algorithm measurements would be incorporated in a recursive estimation algorithm, such as a Kalman filter, whose state estimate outputs would be used to execute the control laws of a relative navigation control algorithm in real-time. Future work should continue efforts to implement a vision-based AAR control system (to include the formation control laws) in real-time. For purposes of navigation integrity, it is highly desirable that an independent, real-time protection level could also be generated as a safety precaution.

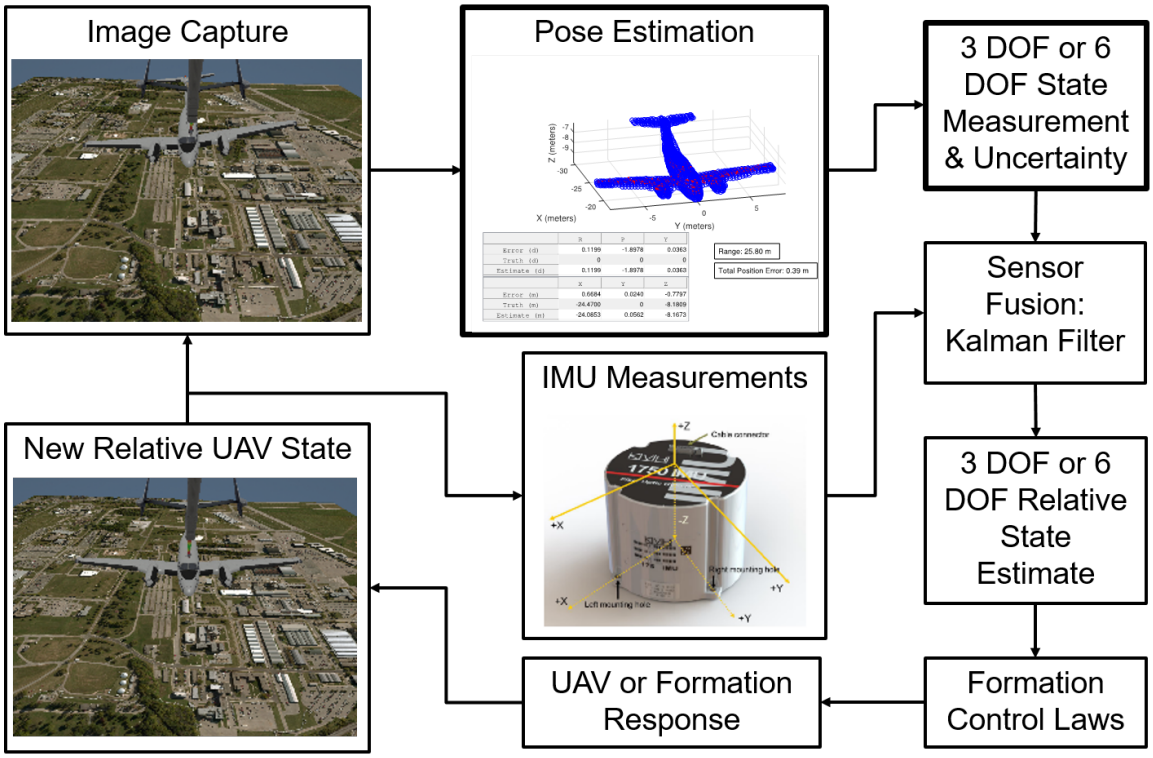


Figure 148. Hypothetical AAR System Block Diagram.

Wit this goal in mind, data analysis revealed that the ICP-based approach taken with the R7 algorithm is capable of achieving much more rapid processing times than the template matching approach taken with the V5 algorithm. The ICP-based approach was also shown to be more accurate for the algorithm configurations developed and tested in this thesis. However, the R7 algorithm is capable only of providing a relative navigation measurement, it is not capable of providing an independent measure of navigation integrity. Navigation integrity is highly desirable in the context of AAR due to the close proximity of the two aircraft in formation. Hence, an algorithm that possesses both the speed of the R7 algorithm as well as the V5 algorithm's ability to provide a protection level would be desirable.

One could begin design of such an algorithm with the general architecture suggested below. This algorithm would be most easily implemented with the attitude-fed version of ICP developed in this thesis.

1. Utilize an algorithm similar to R7 to obtain a relative navigation measurement by leveraging the ICP algorithm. The ICP algorithm operates by computing the Euclidean distance, D, between every observation point cloud point and its nearest neighbor in the model point cloud [8]. Retain these measurements.

2. Construct a likelihood function, $L(D)$, that describes the likelihood of observing the distance, D, in a perfectly matched model and observation point cloud. Using this approach, a likelihood could be computed for each point in an observation point cloud. As in the V5 algorithm, this likelihood function would have to be developed prior to algorithm implementation. One could also examine the use of a likelihood function that utilizes a measure of the entire point cloud, such as the root mean square of all the computed D values. In this case only a single likelihood could be computed for an observation-model point cloud comparison. Machine learning could be usefully employed to identify a parameterization for the likelihood function that minimizes estimate errors [52].

3. With $L(D)$, compute the likelihood that the solution obtained in Step 1 produces an ideal match between the observation and model point clouds.

4. Selectively render the model point cloud at positions other than that obtained in Step 1. Compute the distance between every observation point cloud point and its nearest neighbor in these selectively rendered model point clouds. Compute the likelihood that these positions have produced an ideal match with the observation point cloud with the likelihood function $L(D)$. These positions should be selected based upon the most recent estimate of the PDF describing the po-

sition of the receiver. Selected positions should be most heavily concentrated in the vicinity of the solution obtained in Step 1. Sparser sampling should be done as distance from the solution obtained in Step 1 increases. Distribution sampling methods used in Unscented Kalman Filters or particle filters may also offer useful insights.

5. Optionally, assume a distributional form for the likelihoods present in the gaps between the samples taken in Step 4. Using this assumption will enable generation of a PDF rather than a PMF. For instance, as distance from the observation point cloud increases, the growth rate of the distance metrics measured in Step 4 will eventually become essentially equal for all points. In this case, since one has knowledge of how the growth rate of the D values, one can predict what the likelihood will look like at points far from those sampled. Additionally, growth of D values or a similar metric can be used to indicate when the model and observation point cloud no longer overlap. This or a similar indicator can be used to bound the search space.

6. Using the likelihood distribution obtained in Step 4 or Step 5, use Bayesian inference to compute probability distribution describing the relative receiver position. Using the method in Step 4 would result in a PMF, using the method in Step 5 would result in a PDF. Either method would enable the computation of a protection level.

7. Optionally, iterate the sampling process to reduce error and, possibly, reduce the number of required computations. Iteration could be based on the composite position estimate obtainable from the PMF or PDF obtained in Step 6. Alternatively, iteration could be based on the likelihoods observed in Step 4 and could be initiated prior to applying Bayesian inference.

Appendix A. Camera Calibration Results

This appendix presents the camera calibration results for the Prosilica GT1290C EO cameras used in flight test. All results, except for the extrinsic parameters, were yielded from Bouguet's MATLAB® stereo camera calibration software. The presented intrinsic camera calibration results follow the convention yielded by Bouguet's MATLAB® software [34]. See Sections 2.3.4 and 2.3.6.3 for more details.

The intrinsic camera calibration matrix and distortion coefficients for the left EO camera were computed to be:

$$\mathbf{K}_{L,EO} = \begin{bmatrix} 1181.2 & 0 & 655.0 \\ 0 & 1182.6 & 491.8 \\ 0 & 0 & 1 \end{bmatrix} \tag{129}$$

$$\mathbf{d}_{L,EO} = [-0.0852, 0.1468, 0.0007, -0.0009, 0] \tag{130}$$

The intrinsic camera calibration matrix and distortion coefficients for the right EO camera were computed to be:

$$\mathbf{K}_{R,EO} = \begin{bmatrix} 1181.6 & 0 & 629.7 \\ 0 & 1182.4 & 513.4 \\ 0 & 0 & 1 \end{bmatrix} \tag{131}$$

$$\mathbf{d}_{R,EO} = [-0.0837, 0.1477, 0.0015, -0.0004, 0] \tag{132}$$

The translation vector (expressed in millimeters) and DCM describing the geometry of the EO stereo camera system were computed to be:

$$\mathbf{T}^{R,EO} = [-500.2, -0.4313, 0.6761]^T \tag{133}$$

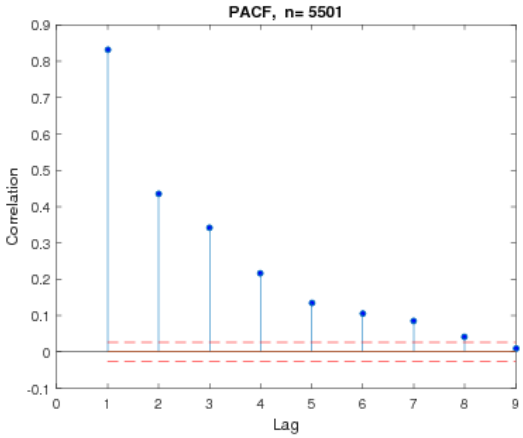
$$\mathbf{R}_{R,EO}^{L,EO} = \begin{bmatrix} 1.0000 & -0.0017 & 0.0040 \\ 0.0017 & 1.0000 & -0.0018 \\ -0.0040 & 0.0018 & 1.000 \end{bmatrix} \tag{134}$$

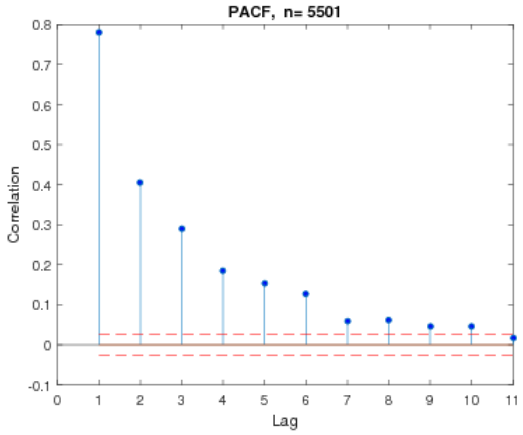
Processing of USAF Test Pilot School instrumentation team measurements were used to perform the extrinsic camera calibration of the left EO camera. The vector (expressed in meters) and DCM describing this camera's orientation and position with respect to the C-12C pseudo-tanker's GLITE TSPI system were computed to be:

$$\mathbf{l}_{EO}^{p} = [1.9550, 0.6686, 0.4339]^{T} \tag{135}$$

$$\mathbf{R}_{L,EO}^{p} = \begin{bmatrix} -0.0258 & 0.4214 & -0.9065 \\ -0.9993 & 0.0150 & 0.0356 \\ 0.0285 & 0.9067 & 0.4208 \end{bmatrix} \tag{136}$$

Appendix B. PACF Returns





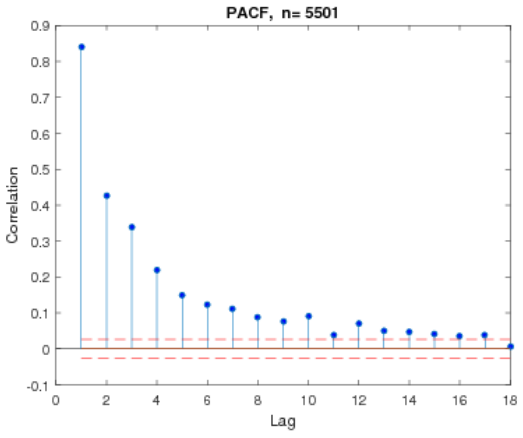
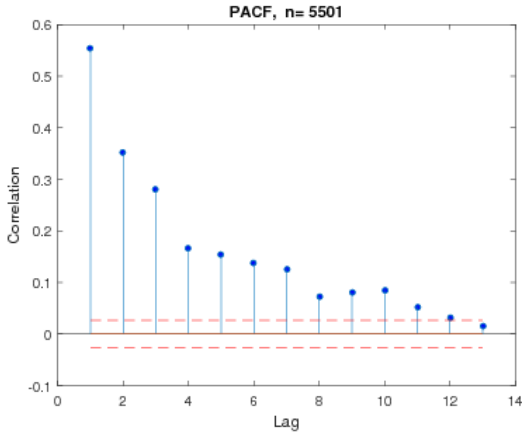
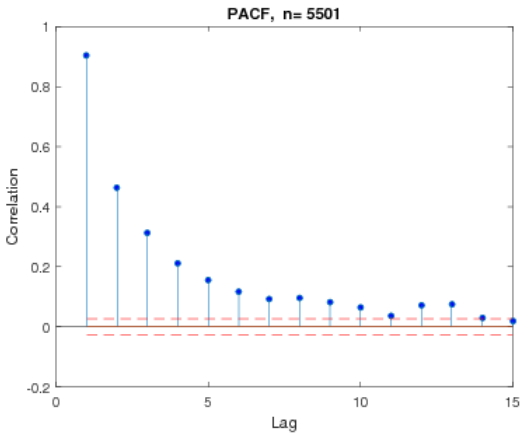


Figure 149. From Top to Bottom, PACF Results for X, Y, and Z Position Errors, Left Camera Frame.





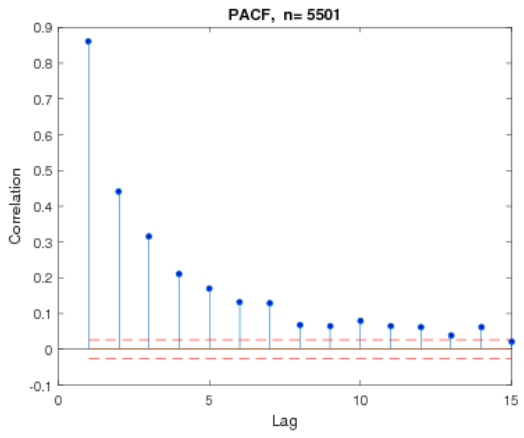


Figure 150. From Top to Bottom, PACF Results for Roll, Pitch, and Yaw Attitude Errors, Tanker Frame.

Appendix C. R7 Error Characterization

One can conceive of the aircraft position estimate returned from R7 as being analogous to a single point that has been projected into three-dimensions from the rectified stereo camera image pair. Hence, the error in the R7 algorithm's relative position and attitude estimates can be characterized in simulation according to the aircraft's z-position in the left rectified camera frame. This can be done in the 3DVW by repeatedly rendering the receiver aircraft at random positions and attitudes within a given z-plane. Within each left rectified camera z-plane, the error in the algorithm's estimate of the receiving aircraft's z-position (Z_{L_r}) would be fit to a normal distribution with mean $\mu_{Z_{L_r}}$ and variance $\sigma^2_{Z_{L_r}}$. This would serve to characterize the error in Z_{L_r}. Alternatively, distributional fits other than a normal distribution could be used.

Recall from Section 2.3.6.6 that the x and y-coordinates of a point in the left rectified camera frame are linearly related to the z-coordinate of that point in the left rectified camera frame per the equation:

$$\mathbf{X}^{L_r} = \begin{bmatrix} X_{L_r} \\ Y_{L_r} \\ Z_{L_r} \end{bmatrix} = \begin{bmatrix} \frac{\left(x_{L_r} - c_{x_{L_r}}\right) Z_{L_r}}{f_c} \\ \frac{\left(y_{L_r} - c_{y_{L_r}}\right) Z_{L_r}}{f_c} \\ -\frac{f_c T_x^R}{d} \end{bmatrix} \tag{137}$$

Hence, if a normal distribution is assumed for the error in Z_{L_r}, then the error in X_{L_r} and Y_{L_r} will also be normally distributed if the pixel coordinates x_{L_r} and y_{L_r}, the principal point $(c_{x_{L_r}}, c_{y_{L_r}})$, and focal length f_c are assumed to be non-random constants. The parameterization of these distributions will be directly related to the parameterization of the error distribution of Z_{L_r}. The parameter values for the error distribution in X_{L_r} will be:

$$\mu_{X_{L_r}} = \frac{\left(x_{L_r} - c_{x_{L_r}}\right)}{f_c} \mu_{Z_{L_r}} \tag{138}$$

$$\sigma^2_{X_{L_r}} = \left[\frac{\left(x_{L_r} - c_{x_{L_r}}\right)}{f_c}\right]^2 \sigma^2_{Z_{L_r}}, \tag{139}$$

Where $\mu_{X_{L_r}}$ is the mean and $\sigma^2_{X_{L_r}}$ is the variance.

Similarly, the parameter values for the error distribution in Y_{L_r} will be:

$$\mu_{Y_{L_r}} = \frac{\left(y_{L_r} - c_{y_{L_r}}\right)}{f_c}\mu_{Z_{L_r}}, \tag{140}$$

$$\sigma^2_{Y_{L_r}} = \left[\frac{\left(y_{L_r} - c_{y_{L_r}}\right)}{f_c}\right]^2 \sigma^2_{Z_{L_r}}, \tag{141}$$

.

Hence, one can ascertain from the above equations that the X_{L_r} and Y_{L_r} coordinates of point position estimates will have greater errors the farther the aircraft is from the principal point in the image.

In practice, R7 would return a three-dimensional relative position estimate of the aircraft in the p-frame. Below, estimated point coordinates are indicated with a hat. This could be transformed into the left rectified camera frame according to the equation:

$$\hat{\mathbf{X}}^{L_r} = \begin{bmatrix} \hat{X}_{L_r} \\ \hat{Y}_{L_r} \\ \hat{Z}_{L_r} \end{bmatrix} = \mathbf{R}_L^{L_r}\mathbf{R}_p^L \begin{bmatrix} \hat{X}_p \\ \hat{Y}_p \\ \hat{Z}_p \end{bmatrix} \tag{142}$$

With these values in hand, one can obtain the pixel coordinate $(\hat{x}_{L_r}, \hat{y}_{L_r})$ needed to obtain the error distributions of the three-dimensional coordinate estimates $\hat{X}_{L_r}$ and $\hat{Y}_{L_r}$ per the equations:

$$\hat{x}_{L_r} = \frac{\hat{X}_{L_r} f_c}{\hat{Z}_{L_r}} + c_{x_{L_r}} \tag{143}$$

$$\hat{y}_{L_r} = \frac{\hat{Y}_{L_r} f_c}{\hat{Z}_{L_r}} + c_{y_{L_r}} \tag{144}$$

With these pixel coordinates, error distribution estimates in all three left rectified camera frame dimensions could be obtained from an analysis conducted along the lines described in this section. Since these distributions are all assumed to be normal, one could transform these distributions into an arbitrary frame that is linearly related to the left rectified camera frame and still retain normality. These error distribution estimates could then be incorporated into a Kalman filter utilizing the outputs of the R7 algorithm. Alternatively, if the pixel coordinates x_{L_r} and y_{L_r}, the principal point $(c_{x_{L_r}}, c_{y_{L_r}})$, or the focal length f_c were not assumed to be non-random, then an alternative analysis would be required.

Bibliography

1. S. M. Calhoun, J. F. Raquet, and G. Peterson, "Vision-Aided Integrity Monitor for Precision Relative Navigation Systems," in *Proceedings of the 2015 International Technical Meeting, ION ITM 2015*, 2015, pp. 756–767.

2. S. M. Calhoun, "Integrity Determination for Image Rendering Vision Navigation," Ph.D. dissertation, Air Force Institute of Technology, 2016.

3. United States Air Force Test Pilot School, "Electro-Optical Systems," in *Systems Phase Textbook*. United States Air Force Test Pilot School, 2002.

4. J. Raquet, "Vision-Aided Navigation," Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 2014.

5. S. M. Calhoun and J. Raquet, "Integrity determination for a vision based precision relative navigation system," in *2016 IEEE/ION Position, Location and Navigation Symposium (PLANS)*. IEEE, apr 2016, pp. 294–304. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7479713

6. C. Parsons, "Improving Automated Aerial Refueling Stereo Vision Pose Estimation Using a Shelled Reference Model," Master's thesis, Air Force Institute of Technology, 2017.

7. T. Stuart, S. Bellingham, T. Grace, J. Lambach, and D. Welch, "Have Vision: Flight Test Evaluation of Stereo Vision Algorithms for Relative Navigation and Automated Aerial Refueling," United States Air Force Test Pilot School, Edwards Air Force Base, California, Tech. Rep., 2017.

8. P. Besl and N. McKay, "A Method for Registration of 3-D Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.

9. M. L. Fravolini, M. Mammarella, G. Campa, M. R. Napolitano, and M. Perhinschi, "Machine vision algorithms for autonomous aerial refueling for UAVs using the USAF refueling boom method," *Studies in Computational Intelligence*, vol. 304, pp. 95–138, 2010.

10. G. Campa, M. L. Fravolini, and A. Ficola, "Autonomous Aerial Refueling for UAVs Using a Combined GPS-Machine Vision Guidance," *Aerospace*, no. August, p. 5350, 2004.

11. G. Campa, M. R. Napolitano, and M. L. Fravolini, "Simulation environment for machine vision based aerial refueling for UAVs," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 45, no. 1, pp. 138–151, 2009.

12. M. L. Fravolini, V. Brunori, A. Ficola, M. L. Cava, and G. Campa, "Feature Matching Algorithms for Machine Vision Based Autonomous Aerial Refueling," *Analysis*, no. Mv, 2007.

13. M. L. Fravolini, G. Campa, and M. R. Napolitano, "Evaluation of Machine Vision Algorithms for Autonomous Aerial Refueling for Unmanned Aerial Vehicles," *Journal of Aerospace Computing, Information, and Communication*, vol. 4, no. 9, pp. 968–985, sep 2007. [Online]. Available: http://arc.aiaa.org/doi/abs/10.2514/1.17269

14. M. Mammarella, G. Campa, M. R. Napolitano, M. L. Fravolini, Y. Gu, and M. G. Perhinschi, "Machine vision/GPS integration using EKF for the UAV

aerial refueling problem," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 38, no. 6, pp. 791–801, 2008.

15. M. Mammarella, G. Campa, M. R. Napolitano, and B. Seanor, "GPS / MV based Aerial Refueling for UAVs," *AIAA Guidance, Navigation and Control Conference and Exhibit*, no. August, pp. 1–16, 2008.

16. M. Mammarella, G. Campa, M. R. Napolitano, and M. L. Fravolini, "Comparison of point matching algorithms for the UAV aerial refueling problem," *Machine Vision and Applications*, vol. 21, no. 3, pp. 241–251, 2010.

17. D. B. Wilson, A. H. Göktogan, and S. Sukkarieh, "A vision based relative navigation framework for formation flight," *2014 IEEE International Conference on Robotics and Automation*, pp. 4988–4995, may 2014. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6907590

18. W. R. Williamson, G. J. Glenn, V. T. Dang, J. L. Speyer, S. M. Stecko, and J. M. Takacs, "Sensor Fusion Applied to Autonomous Aerial Refueling," *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 1, pp. 262–275, jan 2009. [Online]. Available: http://arc.aiaa.org/doi/abs/10.2514/1.34589

19. D. B. Wilson, A. H. Goktogan, and S. Sukkarieh, "Experimental Validation of a Drogue Estimation Algorithm for Autonomous Aerial Refueling," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2015, pp. 5318–5323. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7139941

20. J. M. Howard and M. J. Veth, "Image Aided Relative Navigation for Air Vehicles Using a Passive, Statistical Predictive Rendering Approach," in *24th International Technical Meeting of the Satellite Division of the Institute of*

Navigation 2011, ION GNSS 2011, vol. 5, 2011, pp. 3546–3556. [Online]. Available: http://www.scopus.com/inward/record.url?eid=2-s2.0-84861411879&partnerID=tZOtx3y1

21. S. M. Calhoun, J. Raquet, and J. Curro, "Flight Test Evaluation of Image Rendering Navigation for Close-Formation Flight," in *Proceedings of the 25th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS 2012)*, Nashville, TN, 2012, pp. 826–832.

22. GMV Inc., "Integrity," 2014. [Online]. Available: http://www.navipedia.net/index.php/Integrity

23. RTCA, "DO-208: Minimum Operational Performance Standards (MOPS) for Airborne Supplemental Navigation Equipment Using Global Positioning System (GPS)," RTCA-159, Washington D.C., Tech. Rep., 1991.

24. ——, "DO-245A: Minimum Aviation System Performance Standards for the Local Area Augmentation System (LAAS)," RTCA SC-159, WG-4, Washington D.C., Tech. Rep., 2004.

25. R. Gonzalez, R. Woods, and S. Eddins, *Digital Image Processing Using MATLAB*, 2nd ed. United States of America: Gatesmark Publishing, 2009.

26. D. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*, 2nd ed., M. Hirsch and T. Dunkleberger, Eds. United States of America: Prentice Hall, 2012.

27. R. Brunelli, *Template Matching Techniques in Computer Vision: Theory and Practice*. Wiley, 2013.

28. S. Miller and D. Childers, *Probability and Random Processes*, 2nd ed. United States of America: Academic Press, 2012.

29. A. Gelman, J. Carlin, H. Stern, D. Dunson, A. Vehtari, and D. Rubin, *Bayesian Data Analysis*, 3rd ed. Boca Raton, Florida: CRC Press, 2013.

30. C. Olson, "Maximum-likelihood image matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 6, pp. 853–857, 2002.

31. W. M. Wells III, "Statistical Object Recognition," Ph.D. dissertation, Massachusetts Institute of Technology, 1993.

32. G. Bradski and A. Kaehler, *Learning OpenCV*, 1st ed. United States of America: O'Reilly Media, 2008.

33. R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge, UK: Cambridge University Press, 2003.

34. J.-Y. Bouguet, "Camera Calibration Toolbox for Matlab." [Online]. Available: https://www.vision.caltech.edu/bouguetj/calib_doc/

35. MathWorks, "What Is Camera Calibration?" [Online]. Available: https://www.mathworks.com/help/vision/ug/camera-calibration.html

36. D. C. Brown, "Close-Range Camera Calibration," *Photogrammetric Engineering*, vol. 37, no. 8, pp. 855–866, 1971.

37. J. Heikkila and O. Silven, "A Four-Step Camera Calibration Procedure with Implicit Image Correction," *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1106–1112, 1997. [Online]. Available: http://ieeexplore.ieee.org/document/609468/http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=609468

38. Z. Zhang, "A Flexible New Technique for Camera Calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11,

pp. 1330–1334, 2000. [Online]. Available: http://ieeexplore.ieee.org/document/888718/

39. R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer, 2011.

40. A. Fusiello, A. Fusiello, E. Trucco, E. Trucco, A. Verri, and A. Verri, "A compact algorithm for recti cation of stereo pairs," *Machine Vision and Applications*, vol. 12, no. 1, pp. 16–22, 2000.

41. K. Konolige, "Small Vision Systems: Hardware and Implementation," in *Proceedings of the Eighth International Symposium on Robotics Research*, Y. Shirai and S. Hirose, Eds. Shonan, Japan: Springer, 1997, pp. 203–212.

42. OpenCV, "Camera Calibration and 3D Reconstruction," 2013. [Online]. Available: http://docs.opencv.org/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html

43. S. Perri, D. Colonna, P. Zicari, and P. Corsonello, "SAD-Based Stereo Matching Circuit for FPGAs," in *2006 13th IEEE International Conference on Electronics, Circuits and Systems*. IEEE, dec 2006, pp. 846–849. [Online]. Available: http://ieeexplore.ieee.org/document/4263499/

44. C. Chang and S. Chatterjee, "Quantization Error Analysis in Stereo Vision," *[1992] Conference Record of the Twenty-Sixth Asilomar Conference on Signals, Systems & Computers*, pp. 1037–1041, 1992. [Online]. Available: http://ieeexplore.ieee.org/document/269140/

45. P. Misra and P. Enge, *Global Positioning System: Signals, Measurements, and Performance*, 2nd ed. Lincoln, Massachusetts: Ganga-Jumana Press, 2012.

46. D. Titterton and J. Weston, *Strapdown Inertial Navigation Technology*, 2nd ed. London: The Institution of Engineering and Technology, 2004. [Online]. Available: http://digital-library.theiet.org/content/books/ra/pbra017e

47. W. K. Pratt, *Digital Image Processing*. Hoboken, NJ, USA: John Wiley & Sons, Inc., jan 2007. [Online]. Available: http://doi.wiley.com/10.1002/0470097434

48. A. Massey and S. J. Miller, "Tests of Hypotheses Using Statistics," Providence, RI. [Online]. Available: https://web.williams.edu/Mathematics/sjmiller/public_html/BrownClasses/162/Handouts/StatsTests04.pdf

49. Snedecor, G. W., and W. G. Cochran, *Statistical Methods*, 8th ed. Ames, Iowa: Iowa State University Press, 1989.

50. J. R. Taylor, *An Introduction to Error Analysis The Study of Uncertainties in Physical Measurements*, 2nd ed. Sausalito, California: University Science Books, 1997.

51. T. Jorris, "MATLAB Stats and DOE App," United States Air Force Test Pilot School, Edwards Air Force Base, California, 2017.

52. E. A. Wan and R. V. D. Merwe, "The Unscented Kalman Filter," *Kalman Filtering and Neural Networks*, p. 62 pages, 2001.

53. G. Sibley, G. Sukhatme, and L. M. Robotic, "The Iterated Sigma Point Kalman Filter with Applications to Long Range Stereo." [Online]. Available: http://www.roboticsproceedings.org/rss02/p34.pdf

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704–0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202–4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
03–22–2018	Master's Thesis	Sept 2015 — Mar 2018

4. TITLE AND SUBTITLE

Integrity Monitoring for Automated Aerial Refueling:
A Stereo Vision Approach

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Stuart, Thomas R., Maj, USAF

5d. PROJECT NUMBER

18G153

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-MS-18-M-062

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Lab Aerospace System Directorate Power and Control Div
2130 8th Street, Bldg 45, Room 190
WPAFB OH 45433-7765
COMM 937-938-4617
Email: ba.nguyen@afit.edu

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL/RQQC

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

This paper develops two algorithms that provide relative navigation measurements for automated aerial refueling solely from a stereo image pair. Algorithms were analyzed in simulation and then in flight test using two C-12C aircraft. The first algorithm, the Vision and Bayesian Inference Based Integrity Monitor (V5), uses Bayesian inference and template matching to return a probability mass function (PMF) describing the position of an observed aircraft. This PMF provides a relative position estimate as well as a protection level, thus providing a degree of navigation integrity. Using both simulation and flight test data, mean V5 spherical error was less than one meter and protection levels reliably characterized algorithm uncertainty. The second algorithm, relative pose estimation with computer vision and iterative closest point (R7), uses stereo vision algorithms and the iterative closest point algorithm to return relative position and attitude estimates. Using both simulation and flight test data, mean R7 spherical error was less than 0.5 meters. Additionally, in flight test, mean R7 attitude errors were less than 3° in all axes.

15. SUBJECT TERMS

Stereo Vision, Automated Aerial Refueling, Relative Navigation, Iterative Closest Point, Bayesian Inference, Computer Vision, Integrity Monitor, Navigation Integrity

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Scott Nykl, AFIT/ENG
U	U	U	U	352	19b. TELEPHONE NUMBER *(include area code)* (937) 255-3636, x4395; scott.nykl@afit.edu