ARMY RESEARCH LABORATORY

Target Acquisition and Engagement from an Unmanned Ground Vehicle: The Robotics Test Bed of Demo 1

Gary A. Haas
Phillip David
Bailey T. Haug

ARL-TR-1063 MARCH 1996

Approved for public release; distribution is unlimited.

MaxVideo™ is a trademark of Datacube, Inc.

Motorola®is a registered trademark of Motorola, Inc.

SPARCStation™ is a trademark of Sun Micro-Systems, Inc.

UNIX™ is a trademark of Bell Laboratories.

VxWorks® is a registered trademark of Wind River Systems, Inc.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE March 1996	3. REPORT TYPE AND DATES COVERED Final

4. TITLE AND SUBTITLE
Target Acquisition and Engagement from an Unmanned Ground Vehicle: The Robotics Test Bed of Demo 1

5. FUNDING NUMBERS
PR: 622618AH80

6. AUTHOR(S)
Haas, G.A.; David, P.; Haug, B.T.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
Weapons Technology Directorate
Aberdeen Proving Ground, MD 21010-5066

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
Weapons Technology Directorate
Aberdeen Proving Ground, MD 21010-5066

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
ARL-TR-1063

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The capability of robotic technology to perform dangerous military missions without exposing troops to hazard has been of significant interest to the U.S. Army. Much of this interest has focused on unmanned ground vehicles (UGVs). Perceived benefits include force multiplication, reduction of military hazard, and operation in nuclear/biological/chemical environments. UGVs have been discussed for a number of missions, including antitank, mine neutralization, physical security, smoke generation, scout, sentry, forward observer, and others.

This report describes and discusses one of the first military UGVs, the robotics test bed (RTB). The RTB is a teleoperated vehicle with on-board automatic target acquisition system and turret-mounted weapon surrogate. The perspective of this report is primarily on robotic target engagement, with other aspects of the program and equipment being covered as context. The history of the program provides context for system design. Major subsystems of the vehicle are described, with detailed description of the target acquisition and engagement subsystems. A description of the use of the system in a recent demonstration of the capabilities of military robotics illustrates how such a system might be used in war fighting. Issues in robotic target engagement are discussed and further work is proposed.

14. SUBJECT TERMS
automatic target acquisition
robotics test bed
unmanned ground vehicle

15. NUMBER OF PAGES
70

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT

INTENTIONALLY LEFT BLANK.

TABLE OF CONTENTS

LIST OF FIGURES

INTENTIONALLY LEFT BLANK.

TARGET ACQUISITION AND ENGAGEMENT FROM AN UNMANNED GROUND VEHICLE: THE ROBOTICS TEST BEDS OF DEMO 1

1. INTRODUCTION

1.1. Programmatic Context

Recognizing the potential payoffs for unmanned ground vehicles in force multiplication and the performance of mission tasks in areas of extreme risk to the soldier, the U.S. Army Laboratory Command (LABCOM) in 1987 initiated a program to consolidate the technologies necessary to develop a test bed unmanned ground vehicle (UGV) in a cooperative effort among the laboratories in the command. This was the beginning of the tech base enhancement for autonomous machines (TEAM), which became the robotics test bed (RTB) program. This unmanned ground vehicle program effected an intensive effort to develop and demonstrate robotics technologies to critical military decision makers in a scripted military scenario known as DEMO 1. In LABCOM's successor, the U.S. Army Research Laboratory (ARL), these efforts produced a solid infrastructure of people and test beds from which future military robotics initiatives are being launched.

1.2. Program History

A brief history of the TEAM/RTB program will help in the understanding of the decisions leading to the implemented design of the RTB vehicle.

The TEAM program sprang from a 1987 directive from Mr. Richard Vitali, Director of the U.S. Army Laboratory Command (LABCOM), that the LABCOM laboratories pool their budgets and efforts to demonstrate to the military community that their in-house talents could deliver capabilities, not just technologies. One proposal selected for implementation, submitted by a group led by Mr. Charles Shoemaker, was for a robotic tank killer. The concept was to showcase the technical capabilities of its participants, primarily

- Automatic target acquisition (ATA) from LABCOM's Harry Diamond Laboratories (HDL);
- Smart weapon system from LABCOM's Ballistic Research Laboratory (BRL);
- Operator interface technology from LABCOM's Human Engineering Laboratory (HEL);
- Robotic control system technology from the Department of Commerce's National Bureau of Standards (NBS), now known as the National Institute of Standards and Technology (NIST);
- Teleoperation technology from the Department of Energy's Oak Ridge National Laboratory (ORNL).

The TEAM concept was to be based on a teleoperated high mobility multipurpose wheeled vehicle (HMMWV) supporting an ATA system feeding target data to a weapon control system. The system could perform target acquisition and engagement autonomously or under supervisory control, depending on the desired level of operator involvement, but it was explicitly not targeted at autonomous mobility. This vehicle was envisioned as a tank killer, for application to ambush and mine field overwatch missions, working in cooperative groups of two or more vehicles with overlapping fields of fire. The capabilities of the TEAM concept were to be showcased in a scripted demonstration for a high level military audience, which came to be known as DEMO 1.

BRL's tasks in project TEAM were the selection and integration of a weapon system and the aiming and fire control of the weapon. Weapon systems anticipated to be available in the time frame were evaluated based on lethality, minimum and maximum range, flight time, weight, and appropriateness to implementation on a low cost military robot. Considerations in the latter category included tracking requirements, lethal footprint, and pointing accuracy required. The weapon of choice was a smart weapon of the "fly-over, shoot down" family represented by smart target-activated fire-and-forget (STAFF). STAFF's ability to sense the target over a wide path and self-forge for top attack kept turret-pointing requirements loose, so precision control of the launch was not necessary. The launch method chosen was a recoilless rifle, which offered faster delivery than a missile launch but without the complex recoil mitigation required by conventional guns. Unlike some of the competing rounds, the STAFF round was at that time on a development track that promised pre-production rounds in time for the demonstration.

As the TEAM program got under way, several parallel UGV programs were in progress at different commands. The teleoperated vehicle (TOV) was a program at Naval Ocean Systems Center based on a HMMWV using a fiber-optic communication link. The teleoperated mobile all-purpose platform (TMAP) managed by U.S. Army Missile Command (MICOM) was another UGV project. TMAP contractors had developed two designs of small (go-cart size) platforms to perform reconnaissance, using either fiber-optic or short-range line-of-sight radio frequency communication links. The Army's Tank-Automotive Command (TACOM) evidenced interest in robotics technology through a large number of small projects investigating technologies of utility to field robotics.

In 1990, Congress consolidated all the UGV programs under the policy and program direction of the Office of the Secretary of Defense (OSD). The TEAM program was linked to the joint unmanned ground vehicle (JUGV) program, led by a Marine Colonel as program manager (PM). The JUGV had at its heart the tactical unmanned ground vehicle (TUGV), a fullblown system destined for full scale development. The PM's office, torn between user requirements for small size platforms (such as TMAP) and large capacity platforms (such as TEAM and TOV) and still uncertain what doctrine the system might be required to implement, funded the construction of the surrogate teleoperated vehicle (STV). The STV was a limited production teleoperated UGV of intermediate size (e.g., large enough to be driven by a soldier, but small enough to be transported in the bed of a HMMWV). It was to be placed in the hands of the user as a learning aid to help him or her understand the capabilities of UGVs so that he or she could develop concepts of employment and define requirements for the production system. TEAM was renamed the RTB and was designated the principal near-term technology base for the JUGV.

As the JUGV program was being consolidated under OSD, additional changes were occurring that drastically altered the future direction of the mission module for the RTB. Congress barred expenditure of funds for the development of weapons systems based on robotic vehicles and withdrew funding so programmed. OSD accordingly directed that none of the funding for RTB be used for launcher or projectile development. Although this ban halted efforts specifically directed toward weapon work, it allowed the issues of control, communication, pointing and aiming, and system accuracy to be addressed without the complexity of addressing weapon safety, blast effects, and range safety of robotic vehicles. Since the design of the UGV and fabrication of the turret was well under way by this time, it was determined that a single copy of this turret would be completed, using non-weapon effectors to demonstrate its capabilities, and that a second version of the mission module would be developed in response to the new constraints and customer base.

The second version of the mission module was developed for DEMO 1 in response to the requirements of the PM-TUGV and the STV for technologies that would lend themselves to the smaller platform. The initial design of the BRL turret was driven by the anticipated weapon system and by the requirements of the target acquisition system being developed by HDL. Without the need to carry a large weapon system and with the improvements in the design of the target acquisition system, the size of the turret could be reduced and the leveling platform eliminated. A Cobra helicopter turret was selected as a mechanical basis for a smaller turret on a second HMMWV. Control of the smaller turret was enhanced to enable designation, a role more suited to a robot built on a smaller platform but consistent with the antitank mission.

The technical accomplishments of the TEAM program, embodied in the two robotic tank killers, were demonstrated at DEMO 1, conducted in the rolling hills of Aberdeen Proving

Ground's Churchville Test Course in May 1992. The scripted scenario also showcased robotics technologies from NBS and TACOM. The invitation list for the week's activities included decision makers from the Pentagon, Congress, and the Defense Advanced Research Projects Agency. With DEMO 1, the TEAM program per se had met its goals. The research products of the RTB program endure in programs such as the Advanced Research Projects Agency-funded DEMO 2 (a program focused on autonomous UGVs, slated for demonstration in 1996); in support activities and product improvements of the PM-TUGV; in support of robotics development efforts in mine clearing; and in development of robotics-based assists to the mechanized scouts.

1.3. Original TEAM Scenario

The original scenario for the TEAM vehicle motivated the design of the system and is described here as context for the engineering details to follow. The scenario represents a best guess of the original project team about what militarily significant technologies they could develop, integrate, and demonstrate in a 3-year time frame, in the context of a robotic tank killer. The scenario followed from the technologies and the design from the scenario.

The TEAM vehicle was to be driven in a teleoperated mode to its tactical position. In position, the mission module would deploy stabilizers from the bed of the HMMWV and self level. The operator, from his or her remote workstation, would select a field of view (FOV) for the target acquisition system and a corresponding field of fire for the weapon system. The ATA would look for targets in its area of responsibility and alert the operator when targets were detected. The system was to have the capability to operate autonomously, addressing any targets that appeared in the field of fire or under supervisory control, addressing only those targets selected by the operator, but in either case, the weapon system would aim based on the data provided by the ATA system. The design was to accommodate the sequential firing of rounds at multiple targets, a capability termed "ripple fire." Once the firing mission was completed, the mission module was to retract onto the vehicle and the HMMWV would, if desired, autonomously (though blindly) retrotraverse the last 500 m of the path traveled, to be ready for the next assignment or to initiate the return to the staging area.

2. ROBOTIC VEHICLE DESCRIPTION

2.1. Overview

The RTB system consists of the following major subsystems:

- Mobility platform, that is, the vehicle upon which it is based;
- Low level robotics package, which directly effects control of the vehicle;
- Electrical power system;
- Navigation sensor system, which senses the location of the vehicle in inertial coordinates;
- Operator control unit (OCU) in which an operator manipulates controls to direct the activities of the remote vehicle;
- Communications system, which connects the operator workstation to its remote vehicle;
- Safety system, which protects the vehicle from runaway;
- Mission package, a notionally interchangeable module that performs some military function. (The function implemented for DEMO 1 is pointing a weapon [or weapon surrogate].)

See Balakirsky et al. (1993) for an excellent description of a number of these subsystems.

2.2. Mobility Platform

The chassis chosen for the robotic vehicle is the HMMWV, which is available, affordable, capable, mobile, and military. It offers sufficient capacity to carry a reasonable payload as well as the robot electronics and can be driven manually or by means of robotic control. It has the size flexibility required to support prototypes of many novel driving and mission technologies that would not be available on a more narrowly focused mission-specific vehicle. Other benefits of selecting the HMMWV are user acceptance, reduced logistical support since the vehicle is already in the field, and the potential of a dual role vehicle performing as a conventional vehicle and an UGV.

2.3. Low Level Robotics Package

Remote operation of the vehicle requires mechanical actuation of the steering wheel, accelerator pedal, brake, parking brake, transmission shift lever, and transaxle shift lever. Electric motors with potentiometer feedback are used in all instances. A rotary servo motor is attached to the steering wheel by a toothed belt, while commercial linear actuators (servo motors acting through a ball screw) actuate the rest. Linkage to the shift levers is direct to the lever shafts, to the pedals by means of cables and miniature drive chains routed underneath the chassis, and to the parking brake through the stock cable actuator system under the chassis. An off-the-shelf electronics box containing motor drives, driving computer, communications links, and video electronics[1], is installed between the front seats. Output of traditional vehicle sensors (tachometer, speedometer) is routed to the driving computer. The starter is wired for remote and local operation.

2.4. Electrical Power System

Electric power for the substantial additional load must be provided, either from the HMMWV engine itself or from another on-board source. Two variations were built into the context of this program, one for each turret. One HMMWV was equipped with a 24-volt, 320-ampere (A) capacity, under-hood, engine-driven alternator. The other was equipped with a 6.5-kW, 110-volt AC, diesel-powered stand-alone generator, installed in a plywood enclosure in the bed of the truck. In either case, the output was converted to appropriate voltage levels and distributed as needed.

2.5. Navigation Sensor System

A modular azimuth positioning system (MAPS), an inertial navigation sensor developed for the artillery community, is installed on the vehicle and integrated to the driving computer for future use. The MAPS unit is to sense the location of the robotic vehicle in inertial coordinates. It will be used for locating the vehicle on the operator's map and for path retracing. This feature was demonstrated at DEMO 1 on another vehicle of similar architecture.

2.6. Operator Control Unit

For DEMO 1, the two ARL vehicles were controlled from an OCU integrated from the mobile control system (MCS) developed by NBS and the man-machine interface (MMI) developed by HDL. The MCS (Bostelman, Russell, & Wallace 1989) controls the driving functions, including starting, steering, braking, and shifting. The MCS is a custom controller featuring a miniature steering wheel, a small flat panel touch screen display, and various mechanical switches, built into a suitcase size box. The MCS communicates to the on-board vehicle controller through a dedicated spread-spectrum serial radio. The MCS can operate independently with a second suitcase containing a monitor.

---

[1] Video electronics (e.g., switching and transmitter) were housed in the mission package box for DEMO 1, because that is where the most space was available but were moved to the low level robotics box for subsequent exercises.

The MMI, a tabletop controller developed by HDL, gives the operator control of both vehicles by controlling one while monitoring the other. Based on a Versa Module European (VME)-format Sun SPARCStation™ 1, the MMI controls the mission packages and presents all video displays from the vehicle. The user-configurable graphical user interface (GUI) displays a virtual dashboard and status display; pull-down menus, which start subsystems, select operating modes, and initiate functions; and animated sliders, which point the weapon and adjust the cameras. An alarm is sounded when the vehicle detects a target or encounters an error condition. Live video overlays present views from the driving or target acquisition cameras. The MMI communicates to the vehicle mission package, described in Section 2.9, through a spread-spectrum radio transparently linking Ethernets at the base station and on the vehicle.

2.7. Communication

As shown in Figure 1, the communications subsystem consists of three completely separate radio links: a serial spread-spectrum radio linking the MCS to the on-board vehicle controller, an Ethernet spread-spectrum radio connecting the tabletop controller to the mission package elements, and an ultra-high frequency (UHF) video link conveying windshield-equivalent and target acquisition video imagery to the tabletop controller displays. One vehicle also was capable of transmitting compressed video imagery at reduced frame rate over the Ethernet spread-spectrum radio link. The implemented communications scheme was different from that described in Scott (1990).

2.8. Safety

The safety subsystem consists of a dedicated radio receiver aboard the vehicle that can seize control of the brake and engine solenoid circuits. It listens for two messages from its matched transmitter, which is in the hands of a (human) safety officer off board. The first message is a “keep-alive” message, which is broadcast once every several seconds. If more than one keep-alive message is missed, the safety system stops the engine and applies the brakes. The second message is a “kill” command. The kill command is transmitted when the safety officer pushes the emergency stop button on the transmitter box. Upon receipt of this command, the safety system stops the engine and applies the brakes. This stand-alone safety system is not a requirement for a fielded system but is necessary for testing a developmental system.

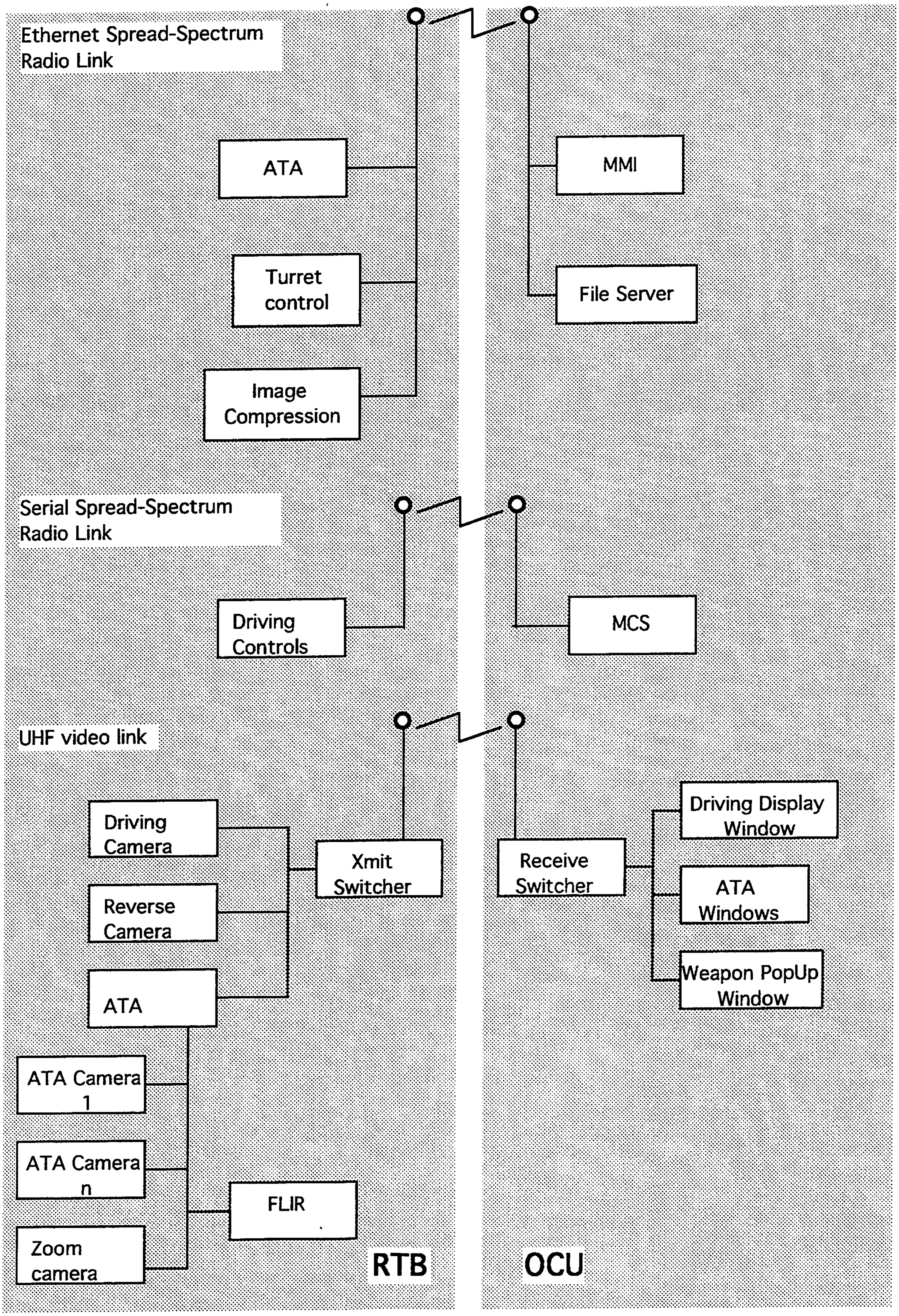

Figure 1. Communications Architecture.

2.9. Mission Package

The mission package for DEMO 1 is a target acquisition and engagement subsystem. Its major components are an ATA subsystem, which detects moving targets, and a turret subsystem, which points and shoots a weapon. The mission package is the focus of this report and is now described in more detail.

3. TARGET ACQUISITION AND ENGAGEMENT MISSION PACKAGE

The mission package for DEMO 1 was implemented in two versions. The so-called "tracking" and "pointing" versions differed in detail but are similar in architecture and purpose. The "pointing" turret is the initial design for the weapon-carrying TEAM program, specifically designed to point an 800-pound recoilless rifle firing smart ammunition. A specific design objective is the ability to fire rounds at several targets one after the other, a mode known as "ripple fire."

The "tracking" turret is a slightly upgraded design, built with available parts but designed to a smaller chassis and designed to track targets (keep the weapon pointed at a moving target for some period of time) with a small laser designator. While the chassis is quite capable, it has not been engineered for any specific weapon.

Differences in design between the two versions are discussed at the appropriate level of detail.

3.1. Automatic Target Acquisition (ATA) System

The controlling element of the DEMO 1 mission package is the ATA system. The ATA system detects and tracks moving targets using visible or infrared (IR) sensors. Pictures of detected targets are sent to the operator control station for the operator to view. If the operator determines that a target is real, he or she may command the ATA system to engage it with the surrogate weapon. The ATA system will then send target position data to the turret subsystem so that the surrogate weapon can be pointed and fired at the target.

The "pointing" turret system uses four TV cameras as its target acquisition sensors. Each of these cameras has a 15° FOV lens. These cameras are mounted on a dedicated sensor-pointing turret in such a way that they cover a 60° field of regard (FOR). Thus, a wide area of the battlefield can be viewed without having to pan the sensors; instead, the ATA system electronically switches from one camera to the next. The "tracking" turret system, on the other hand, has a single 15° FOV TV camera and a boresighted 14° FOV, 3- to 5-micron IR camera, both of which are mounted on a dedicated sensor turret that can be controlled independently of the weapon turret. This enables the "tracking" turret system to slew the cameras from one FOV to another to cover a wide FOR of the battlefield. As of this writing, however, both the "pointing" turret and "tracking" turret systems process only a single FOV at a time.

ATA algorithms use techniques from many fields including artificial intelligence, probability and statistics, and signal processing. Most ATA algorithms fit into the following paradigm (Bhanu 1983). Signal processing is first performed to improve target contrast and reduce sensor noise. Next, potential target locations are detected. Once detected, targets are segmented from the background. Features of each possible target are determined and are used to discriminate real targets from nontargets. To decrease the probability of a false alarm, potential targets may be tracked over a number of frames. Finally, high confidence targets are reported to a mission module for mission-specific functions.

The approach to ATA implemented on the RTB fits into the above paradigm. This approach is based on a system developed by Sandia National Laboratories (Eilers & Schnetzer 1989); the RTB's system includes enhancements to handle situations not addressed by the Sandia system. The system maintains an image of the stationary components of the scene. As each new image is acquired, changes between it and this reference image are detected. Moving targets appear as differences between theses two images. To ensure a high probability of target detection, the system must be such that many events other than moving targets cause changes to occur. Range information is used to eliminate obviously false target detections, and the remaining detections are tracked over a number of frames. The RTB system currently operates with one imaging sensor (either visible or IR) and a range map; future systems will process a number of different sensors simultaneously (visible, IR, range, and acoustic) and will integrate the results of each.

Difficulties arise even in the detection of moving targets. As with any target-detection algorithm, low contrast between target and background is a problem. Slowly moving targets and targets moving directly at the camera will often appear stationary. Furthermore, many forms of "clutter" are present in outdoor environments which might produce differences between an image and a reference image. Some of these are changes in scene illumination (e.g., because of cloud shadows), brush and tree movement, dust and exhaust, and birds and other animals. All these problems are addressed by the RTB ATA system; some are addressed by the detection algorithms and some by the tracking algorithms.

Two images are generated at system set-up time. The first is a "change detection mask" created by the user, which defines regions in the scene where the system should look for targets. Regions of high noise or clutter may be masked using this image. The second image is a "range map," which gives a range value at each point in a two-dimensional (2D) grid that is evenly spaced over the scene. This may be manually created by the user (by drawing range contours and then letting the system interpolate between them) or automatically created if a range sensor is available.

The reference image is ideally an image of the scene without any targets present. The initial scene, which must be void of targets, is used as the basis of the reference image. The appearance of a scene typically changes slowly over time (e.g., from day to night); hence, the reference image must change to reflect these scene changes. Targets, however, must not become part of the reference image; if this were to happen, the ATA system would no longer be able to detect these targets. By requiring the reference image to change slowly, the ATA system will be able to acquire targets before they can become part of this image. Once the targets are acquired, selective reference updating will prevent a target from ever becoming part of the reference image. As each new image is acquired, the old reference image is replaced by a weighted sum of the old reference image and the new image. Pixels in the reference at which targets are located are not changed, however. Whether a target is located at a particular pixel is determined by higher level processing as described next. This selective updating of the reference image enables the system to detect slowly moving or stationary targets provided that they were initially moving (at nominal speeds) when they entered the ATA's FOV.

Targets are detected as regions in the scene where the "structure" of the input image is changing. These changes are detected by applying a local band pass filter to the difference between each new image and the reference image. The high pass part of this filter reduces the effects of broad illumination changes, and the low pass part reduces the effects of noise and other similar small changes. This filtered difference image must then be thresholded to find "interesting" regions. To this end, a sequence of noise images is created that assigns a noise/clutter value for each pixel in the scene. Noisy and high clutter regions in a scene cause high values to be generated at the corresponding pixels in this noise image, while low noise and clutter regions cause corresponding low values to be generated. The noise image is initialized to some constant, and with each new frame, the old noise image is replaced by a weighted sum of the old noise image and the latest filtered difference image. In a manner similar to the way that the reference image is updated, pixels in the noise image where targets are located are not changed. This ensures that targets will continue to be detected even if they stop. With this, the "interesting" regions in the filtered difference image are those pixels whose filtered difference value exceeds the corresponding noise value by a user-defined threshold. This results in a binary difference image.

The binary difference image is then subsampled on a 2D grid of evenly spaced pixels. This subsampling reduces the amount of remaining computations without significantly affecting the system's ability to detect targets. The subsampled image is next logically ANDed with the change detection mask to remove differences from "masked out" regions of the scene. Connected components (Ballard & Brown 1982) in the resulting image are then extracted and are used to create a set of symbolic objects representing regions in the scene that are changing. Each object has a number of features associated with it; some of these are area, width, height, centroid, and range. An object's range is the smallest range value (from the range map generated during system setup) occupied by any pixel of the object.

These objects representing regions of change are then filtered. Objects are discarded if there is a very high probability that the object cannot be a true target. Because this filtered set of objects is passed to the tracking system, this step significantly reduces the work load on the tracker. Objects are filtered, based on their size and range; an object whose size is not close to the expected size of a target at that range is discarded. Because poor segmentation of targets from background is likely to occur, these size constraints must be loose. Still, many spurious detections will be eliminated.

Target detection generates a set of potential targets. Some of these will not be true targets but will be the result of noise and clutter in the scene. With distant targets (generating small images) such as the ones this system must acquire, it is difficult to differentiate "target" or "clutter," based on the processing of just a single image. To help filter the clutter from the true targets, these objects may be tracked over a number of frames. This allows the system to better estimate the properties of each object and therefore make more informed decisions regarding the status of each object. A survey of various multi-target tracking algorithms is given in Bar-Shalom (1978).

The fundamental problem in multi-target tracking is that of determining the frame-to-frame correspondence of objects (potential targets). Because properties of objects change over time and because not all objects will be detected in every frame, this is a difficult problem. The tracking approach implemented on the RTB, based largely on Sandia's work (Eilers & Schnetzer 1989), uses a best-first search to determine an optimal correspondence of objects in the current frame with objects from past frames. The range of an object is used to limit its set of possible correspondence to past objects by applying a bound to the distance (in the image) that a target could have traveled since the last frame, based on the target's range and a priori maximum speed. To address the situation when a poor segmentation causes an object to appear to break apart, multiple detections in the current frame may be matched to a single track from previous frames. The goodness of a particular correspondence of old to new objects is based on a weighted difference of the features of these objects. This search monitors its progress so that when the search is taking too long, it switches to a much faster but suboptimal search. After determining corresponding objects, the system updates the parameters (position, velocity, size, etc.) of each object and then generates a single unified object list from the set of new and old objects.

After performing target tracking, the system examines the object list to determine if it must take any action. Objects that have been seen consistently during the last couple of frames and are exhibiting "target-like" properties are flagged as possible targets. Here, target-like properties are constraints on shape, size, and speed. If, after a few frames of analysis, these objects are still exhibiting target-like properties, then they will be marked as targets. If there are any new targets, the operator will be alerted to them. Pictures of the targets are sent to the operator control station so that the operator can perform the identify friend or foe (IFF) function. Targets designated as foe are engaged by the weapon system; the ATA system converts (using previously calculated calibration data, as described in Section 5.4) the image position and velocity of a designated target to azimuth and elevation position and velocity of the weapon turret. These data are sent to the turret control system either once (for point-fire type target engagement) or continuously (for some predetermined amount of time) for tracking type target engagement.

The ATA algorithms just described are implemented as a real-time system consisting of a single general purpose processor and a number of special purpose image processors. All of this hardware is 6U VME format. A single Motorola 68030 processor is used to control the image-processing hardware and to perform the functions of detection filtering, frame-to-frame target tracking, high level decision making, and target reporting. The image processing is implemented on Datacube, Inc., MaxVideo™ image-processing boards. The "tracking" turret system implements the ATA's image-processing algorithms on 15 MaxVideo™ cards and runs at 30 frames per second. The "pointing" turret system uses 6 MaxVideo™ cards to achieve a throughput of 10 frames per second. The architecture of the Datacube hardware is that of a pipeline processor (Hayes 1978). Each stage of the ATA algorithm is a stage in the pipeline. The input image resolution is 512x484 pixels with 8 bits per pixel. Most internal processing occurs on 16-bit deep images.

3.2. Turret Subsystem

The turret subsystem consists of the following elements:

- Superstructure,
- Turret mechanism and actuators,
- Control system,
- Weapon and weapon surrogate.

Since no weapon was ever installed on the RTBs, any subsequent reference to a weapon element shall be understood to refer to a weapon surrogate.

3.2.1. *Superstructure*

The superstructure of the two vehicles differs substantially. The "pointing" turret subsystem is built on a platform that can lift itself off the HMMWV truck bed to provide a relatively stable platform for the target acquisition sensors, isolated from engine vibration and from motion allowed by the vehicle suspension and tires. The platform lifts itself by deploying three legs through the truck bed, each equipped with a collapsible foot designed for rapid extension and retraction while maintaining road clearance of the vehicle. The legs are driven by stepper motor ball screw linear actuators. Micro-switches at each leg sense when the platform has cleared the truck bed and when the leg has fully retracted. Four equipment racks form the structural elements of the pedestal that support the turret on the platform. The air-conditioned racks provide housing for the computer systems, controllers, communication equipment, and power distribution subsystems for the mission package.

The "tracking" turret subsystem is fixed on the HMMWV chassis, with air-conditioned electronics cabinets mounted on the truck bed, and turret and rotator mounted concentrically on a rigid platform above the cabinets. This system has performed adequately without the chassis isolation built into the "pointing" turret. This may be because the ATA image-processing system can process 20 to 30 frames per second, while the "pointing" turret ATA system can process only about 7 frames per second. At the higher frame rate, the frame-to-frame differences are smaller, resulting in less noise from which to extract the desired signal.

3.2.2. *Turret Mechanism*

The "pointing" turret is designed to carry as much as 800 pounds of weaponry and to slew at 60°/sec in azimuth, 20°/sec elevation. The 36-inch-diameter turret was fabricated mainly in Government shops at Aberdeen Proving Ground. Azimuth drive is by means of a 1600-oz-in. stepper motor driving through a commercial ring gear, with provisions to add a second stepper motor if necessary. Azimuth position sensing is provided by a 13-bit absolute encoder which turns nearly one to one with the ring gear, resulting in a resolution of 0.7 milliradian (mrad). As-built accuracy, as measured by the encoder, is limited primarily by backlash in the drive gears and is about 2 to 3 mrad.

Elevation on the "pointing" turret is actuated by a stepper motor-driven linear actuator acting through a linkage and is sensed by a 13-bit encoder gear driven at the trunnion. Resolution of the elevation axis is 0.064 mrad. As-built accuracy is compromised by distortions in the mounting of the ring gear, an artifact of fabrication, which causes measurable and repeatable errors of as much as 1° in elevation. These errors were compensated in software, resulting in elevation errors measured at the encoder of probably less than 0.5 mrad.

The 28-inch "tracking" turret is based on the turret casting of a Cobra gunship. It is actuated by a small stepper motor on each axis, with 16 bit absolute encoders sensing position. The azimuth drive motor acts through the ring gear. As-built accuracy as measured at the spring-loaded encoders is limited in azimuth by roughly 1 mrad of backlash in the drive gear. The elevation motor acts on the platform through a gear mounted to the trunnion. Gravity loads the elevation gear mesh sufficient to virtually eliminate backlash.

3.2.3. *Turret Control*

3.2.3.1. Logic

Turret control is a computer-based system that accepts commands from the operator and from the ATA subsystem and generates appropriate control signals to turret hardware (the platform drive motors, turret drive motors, and weapon or weapon surrogate) to perform the following functions:

- Raise and lower the platform,
- Arm a weapon,
- Point the weapon,
- Fire the weapon.

It also performs support functions such as built-in test, clock synchronization, command status reporting, and system status reporting.

System status is comprised of device status for each major subsystem. Device status is a resource allocation flag used to prevent more than one command at a time from using a device. No command can be executed if the device that must execute is marked "busy." System status is reported to the operator when requested.

Commands to raise and lower the platform simply trigger hardware devices described elsewhere herein. Platform status is set to "busy" during the execution of platform operations. These commands are received only from the operator.

The command from the operator to arm a weapon selects the weapon to be fired. Options are

- No weapon ("safe"),
- Camera only,
- Point-fire weapon or surrogate,
- Laser designator or surrogate.

Commands to point the weapon, which may be issued by either the operator or the ATA system, may be one of three types:

- Point the weapon at a particular polar coordinate;
- Point the weapon at a target and fire the point-fire weapon;
- Track a target, that is, point the weapon at the target's current position and keep it pointed at the target as it moves.

Pointing the weapon at a polar coordinate is the simplest of these commands. The polar coordinates are converted to stepper steps, which in the "pointing" turret includes finding in a "look-up" table and interpolating compensation for significant nonlinearities. Then, the step generation circuitry is invoked and the stepper motors slew the turret to the designated position. The turret status is set "busy" while the motors are operating.

Pointing the weapon at a moving target requires a single command containing the current location and velocity of the target. The control system compensates for measured communication lags, then iteratively calculates target lead, consults its table of pre-calibrated rotational transit times, and predicts an intersection point at which to fire. Rotational transit times are nearly deterministic in a stepper motor-based turret. The weapon is pointed at the predicted intersection point as described in the previous paragraph. Upon completion of the turret move, if current position and time are close enough to the position and time predicted by the target lead algorithm, the firing circuitry of the selected weapon is triggered.

Tracking a target is a function implemented on the "tracking" turret only, in support of its laser designation function. It requires first that the weapon be pointed at the moving target, as described in the previous paragraph. This step can be envisioned as "locking on." The laser designator is then turned on. The command to track the target must be followed by a periodic (5 times per second, as implemented) stream of updates from the ATA revising target location and velocity. The control system calculates the velocity (rather than position) in polar coordinates, which will cause the weapon to cross the predicted target path midway through the next update period. It then looks up the closest step rate available from the step generator control circuitry. Then the step generation circuitry is invoked and the stepper motors slew the turret at the prescribed velocity until the next update is received.

3.2.3.2. Architecture

The control of each turret subsystem can be envisioned as in Figure 2. The control system consists of a board set residing in a VME chassis. An MV147 single-board computer boots a small program from its read only memory (ROM), which causes it to load the VxWorks® operating system. It loads the operating system either from a battery-backed RAM board on its own backplane, configured in DOS file format using VxWorks® utilities, or over the radio-link Ethernet from a remote development computer. The turret control software is loaded in the same fashion.

Up and running, the control computer receives UNIX™ socket-based commands over an Ethernet connection. Commands are routed by the local area network (LAN) manager process to a dispatcher process, which calls subroutines to execute the appropriate command. Subroutines are partitioned logically into "turret" level routines, which act on objects at a relatively high level of abstraction, and "driver" level routines, which depend on detailed knowledge of specific computer configuration. Commands representative of the turret and driver partitions are shown in Appendix A.

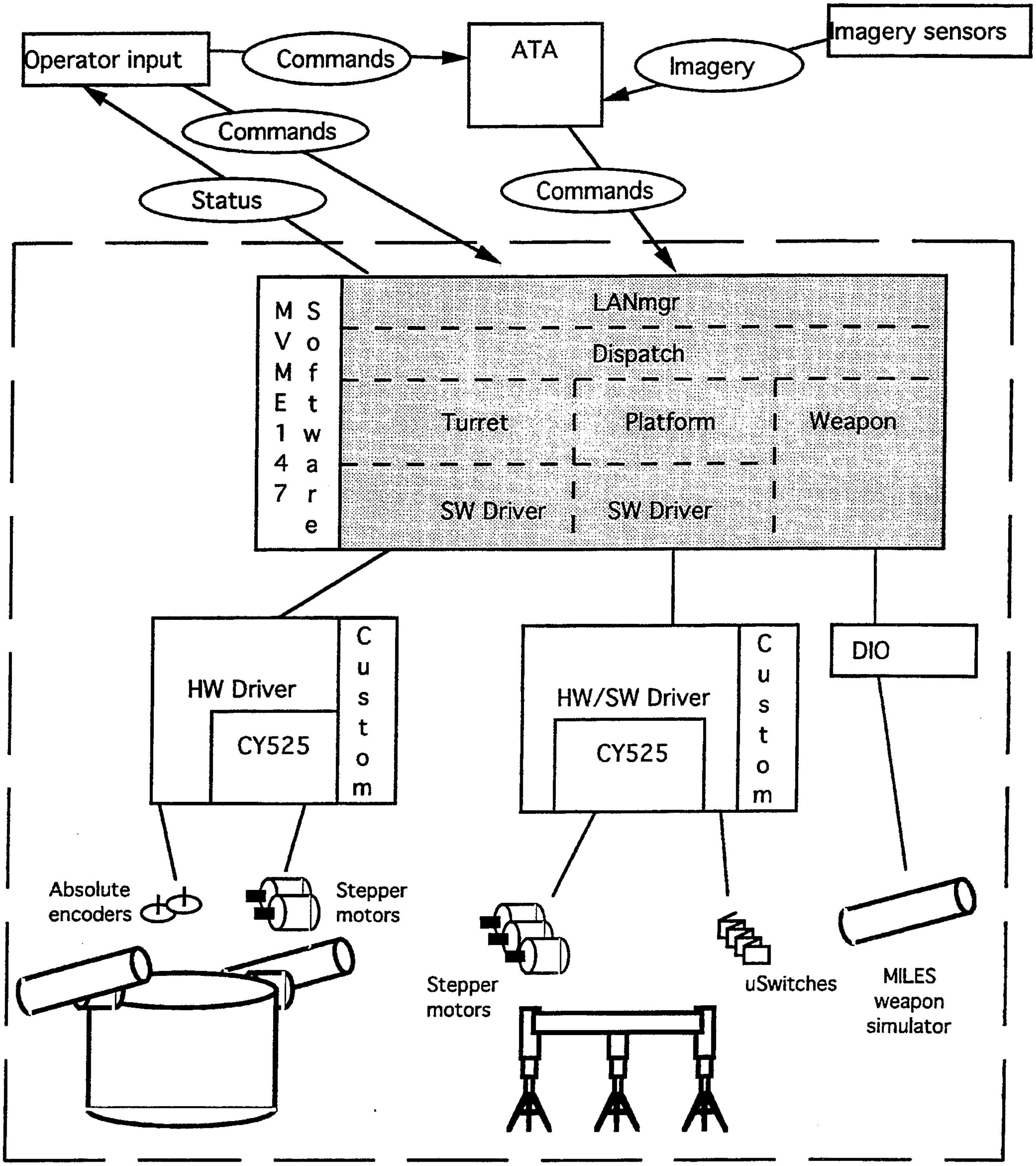

Figure 2. Demo 1 Robotics Test Bed Turret Subsystem.

Driver software acts across the VME bus to pass commands to Cybernetic Microsystems, Inc., CY525 intelligent stepper motor control chips residing on dedicated driver boards built on commercial prototyping boards. The driver routines detect and report status of the command back to the calling routines, which pass the command status back to the dispatcher. Errors are interpreted by a global error handler, which consults a look-up table about the impact of a specific error on equipment status. Command status and equipment status are reported back to the tabletop controller through the socket-based LAN manager as requested.

3.2.3.3. Driver Boards

The driver boards used on the two turrets are substantially different and are described separately. They also require software that differs substantially at the driver level. Careful configuration management of the software allows virtually all turret level software to be shared by the two systems, albeit with different data. Conditional compilation of the driver level software, implemented by the C language pre-processor, configures the executable files for the appropriate turret.

The driver boards used on the "pointing" turret, one each for the turret and the platform, have on-board Motorola® 68000 CPUs, which run software stored in ROM without an operating system. These boards initialize the board hardware, conduct initial and periodic built-in test, do error checking while transferring commands to the CY525s, and interrupt the MVME147 (through the mailbox interrupt) when the command is complete. In addition, the turret board is responsible for reading the encoders on command and placing the reading in dual ported on-board RAM. The platform board monitors the platform micro-switches, shutting off each motor as the platform reaches the appropriate position.

The driver board on the "tracking" turret buffers commands to the CY525 and encoder readings but has no on-board intelligence. The MVME147 must initialize the board hardware, generate control bytes, and monitor buffers for completion flags. However, access to the CY525 control bus is enhanced, facilitating the use of the chip in its velocity control mode. In velocity mode, the chip causes the stepper motor to slew at the commanded rate until the next command. This mode is used for the tracking function used in laser designation.

3.2.4. *Weapon*

The weapon originally intended for this vehicle was never built, but it provides an important context for design decisions that shaped the "pointing" turret in particular. It also defines the likely development path for next generation development efforts.

The original concept for this demonstrator was as precursor to an expendable, line-of-sight tank killer, capable of fearlessly going "toe to toe" with the finest armor of the Soviet army. The round selected for such a mission was a variation of the Army's developmental STAFF round. This round is designed to be fired in the direction of a target to sense the presence of a target near its flight path and to fire a warhead at the vulnerable areas of the target. The round is capable of killing an appropriate target and smart enough to compensate for errors in aim point resulting from the uncertain accuracy of untested ATA algorithms and from the low cost components appropriate to an expendable system. A recoilless rifle was deemed an appropriate launcher for such a round, as the launcher could be fixed on the turret without concern for mechanical mitigation of recoil effects. The blast effects inherent in recoilless rifles were judged an easier problem to address than were the forces of recoil.

Congressional concern about weapons aboard a "robotic" vehicle caused the weapon development portion of the program to be shelved, and resources were diverted to integration of a weapon surrogate. The surrogate was to provide a medium for illumination of issues related to pointing and aiming but without lethal capability.

3.2.5. *Weapon Surrogate*

In lieu of an actual weapon, each turret was equipped with a multiple integrated laser engagement system (MILES) and a video camera with long focal length lens, termed the "zoom" camera. MILES emulates a point-fire weapon or laser designator, while the video camera, with its wider (roughly 1°) FOV representing the footprint of a smart round, transmits a video image of what it is pointed at with superimposed cross hair. The zoom camera is also used in boresighting the ATA and turret subsystems.

MILES is a laser-based weapon simulator system used as a training device for war-gaming exercises. It "uses eye-safe laser bullets" (to quote sales literature) to allow soldiers to practice combat on each other without actually causing casualties. MILES equipment is issued as kits tailored to a specific battlefield entity, such as an individual soldier or a Bradley fighting vehicle (BFV). A MILES kit for a typical weapon system (for example a BFV) consists of a controller box, an IR laser transmitter that fires instead of the BFV main gun, a set of IR hit detectors that detect when the BFV has been hit by another MILES transmitter, and a flashing light that lights when the sensors detect a hit. The simulator for each type of weapon is tailored to represent certain essential characteristics of the real weapon. For example, the BFV main gun simulator transmission fades at about the same range that the BFV main gun loses effectiveness.

The BFV MILES kit comes with three transmitters, one each for the main gun, the coax machine gun, and the tube-launched optically tracked wire-guided (TOW) antiarmor missile launcher, with which a BFV might be equipped. The main gun and coax machine gun simulators send laser transmissions whenever the respective trigger circuits are closed and a MILES that detects the laser-encoded "bullet" message (which has encoded on it the type of weapon from which it was "fired") knows it has been hit. The TOW missile, on the other hand, is a slow missile. It takes some time for the round to hit the target, and the sights must be trained on the target for most of the flight. MILES attempts to replicate this by requiring the transmitter to be fired for 10 seconds (corresponding to an "average" flight time), and the detector scores a hit only if it receives transmissions over most of that time period. This is the same kind of behavior that would be required for the successful use of a laser designator.

Thus the TOW simulator from the Bradley MILES was selected as a reasonable substitute for a laser designator. The main gun transmitter was a logical choice as a simulator for the recoilless rifle, as the ranges for the two weapons were similar. For DEMO 1, detectors and hit indicators from the MILES were not used on the robotic vehicles, since the purpose was to determine how well targets could be engaged, not to simulate a situation where the targets return fire. The target vehicles were equipped with a MILES detector system only, the so-called MITS kit. To enhance the ability of the spectators to witness a simulated "kill," the MITS audio hit indication was used to trigger circuitry to fire a smoke charge.

Testing of MILES transmitters at ranges anticipated for DEMO 1 revealed that the transmitters reliably registered "kills" only if the sensors on the target vehicles were hit within about ±1 mrad. This corresponds to approximately 1 meter diameter target at the intended range of 500 m. This beam spread is substantially less than the footprint of the intended smart round and necessitated midcourse design corrections to tighten the resolution and accuracy of the turrets used in DEMO 1.

4. RESULTS FROM DEMO 1

The performance of the target engagement systems developed in the RTB program have not, to date, been rigorously tested. DEMO 1 provided the only opportunity for evaluation of system capability in any sort of challenging environment (meaning targets at design distance and crossing velocity). In this context, both turrets were successful in hitting moving target vehicles at 400 meters, crossing at speeds as fast as 20 mph.

DEMO 1 was a 2-week demonstration of the capabilities of military robotics presented to high level decision makers from throughout the U.S. Department of Defense, as well as interested parties from Congress and foreign allies. Conducted in the spring of 1992 in the rolling hills north of Baltimore at Aberdeen Proving Ground's Churchville Test Course, DEMO 1 featured five experimental robotic vehicles demonstrating their military potential in a scripted, tactical scenario. Two of the vehicles in DEMO 1 are those of this report. Their role in DEMO 1 is described in this section. The remainder of DEMO 1 has been described in other forums.

The relevant part of DEMO 1 can best be described by dividing activities into set-up activities, which preceded the scripted scenario, and those of the scripted scenario itself. During setup, the vehicles were deployed in positions near the pre-planned positions of the scenario. These positions were selected in part because the ATA rotator was not working on either vehicle, so the FOV of the ATA cameras could be pointed at the field of fire only by positioning the chassis of the vehicle itself. Fortunately, given an appropriate pre-planned position, it was not difficult to aim the ATA camera, given the view from the driving camera.

If the ATA had been modified/repaired since the previous execution of the boresighting procedure (described in Section 5.4), the boresighting procedure was run. Otherwise, the boresight was checked and, if necessary, small correction factors were added. If the vehicle's pre-planned position had been changed since the previous range map activity, the range map was entered. Cameras were checked for appropriate iris settings, and all systems were checked to see that they were working. The vehicles were then moved to their scripted starting positions.

At the beginning of scripted activities, the two vehicles were driven in unmanned mode to the pre-planned positions. The "pointing" turret was directed to raise its platform off the vehicle chassis. Then, in accordance with the script, target vehicles appeared on a dirt road crossing the field of fire, at a range of roughly 400 m and a crossing speed of roughly 10 mph. As the targets were detected by the ATA, the operator was alerted by an audio cue. When he selected the ATA video window of the "pointing" turret to appear on the OCU display, field of fire imagery appeared with targets highlighted. The operator visually confirmed the identity of the target vehicle and assented to the engagement by moving the cursor to the vicinity of the target vehicle and clicking the trackball button. The "pointing" vehicle responded by pointing its MILES transmitter at the target and firing, simultaneously snapping a freeze frame video from its zoom camera, for transmission to the OCU. The freeze frame, with superimposed cross hairs, appeared in a window at the OCU, confirming the shot. Given an accurate shot, a charge on the target vehicle emitted a puff of smoke and a light flashed, triggered by the MITS hit detector system on the target vehicle.

In a similar sequence, the "tracking" turret signaled its acquisition of a target. Upon concurrence of the operator, the "tracking" turret pointed its MILES TOW simulator transmitter at the target vehicle and tracked it, firing the TOW transmitter and transmitting live video from its zoom camera. The live video with superimposed cross hairs was displayed at the OCU until the operator canceled the window. Typically, the live video showed the target vehicle, remaining steadily in the middle of its window with cross hairs dead center, as the scenery passed behind it. After about 10 seconds, the MITS box on board the target vehicle determined that it had been tracked by a TOW long enough for a kill to have occurred; a puff of smoke appeared, and the target vehicle's hit indicator light began flashing.

In a final sequence demonstrating ripple fire, three target vehicles were detected by the "pointing" turret vehicle. With rapid key clicks, the operator assented to the engagement of each target, and the "pointing" turret pointed and fired its surrogate weapon at each in turn, in a sequence that lasted only a few seconds.

By the end of the demonstration, snapshots taken by the zoom camera consistently showed cross hairs on target, and the MILES transmitters were usually successful in triggering the MITS hit indicator. The live video from the zoom camera on the "tracking" turret was especially impressive when the turret was in tracking mode. However, in the context of a demonstration, data collection was not the paramount priority, and constant adjustment of the system confounded any attempts to acquire useful and consistent data. Consequently, significant features remain to be evaluated, several vital subsystems are known to be candidates for redesign, and there has been no opportunity to analyze performance of constituents of the system.

Despite shortcomings in the available test data, program participants have become acquainted with some issues influencing performance and have (for some issues) developed measures for implementation in the next phase of the program. The list of issues presented in the next section is not intended to be comprehensive but representative of the concerns that have been addressed in the program to date.

5. ISSUES IN ROBOTIC TARGET ENGAGEMENT

5.1. Automatic Target Acquisition

The ATA system provides input to the target engagement system. For the purposes of this discussion, issues in ATA are confined to the accuracy of its assessment of target location in its own coordinate system.

The ATA system uses image-processing algorithms to process pixels and selects the aim point based on the geometry of the region of changed pixel luminance values. The impact of the ATA system on target engagement performance is principally through the accuracy and realism with which the ATA system can find the kill zone of the target in the ATA coordinate system. Accuracy (meaning, in this implementation, the effectiveness of the ATA in identifying the pixel location of the centroid of the region) is primarily a function of how well the centroid of the region coincides with the centroid of the target. It can be affected by the algorithm used to define region boundaries and by the resolution of the pixel map used in image processing (which may be less than the resolution of the imager itself). Realism, however, is affected by target shape. For example, the geometrical centroid of a flatbed truck might be in the air above the bed. It would certainly not be the cab, where a human gunner would aim for a kill. The error between the gunner-selected aim point and the target-region location calculated by the ATA system is a metric of interest in assessing system performance. Either of these measures can be extracted from videotape of the ATA imagery with target overlay, which is a normal output of the ATA.

Target velocity, as estimated by the ATA, is based on time-wise differencing of target aim point. Velocity estimates are affected by ATA resolution, precision, and the sampling rate. Error metrics such as the difference between target velocity estimate and actual target velocity are more difficult to measure. Subjective assessment of ATA performance in DEMO 1 and other informal field trials supports the hypothesis that low sampling rates (four frames per second) result in poorer tracking performance than 20-Hz sampling, but factor isolation and quantification are needed.

5.2. Fire Control

The only fire control function implemented at present is target lead. Calculation of lead depends on the ATA's estimate of velocity (discussed previously), a stable, high resolution time base, and effective clock synchronization with the ATA. The time base used for calculating target lead is the VxWorks® system clock, set at a resolution of 60 Hz. Jitter is not thought to be a problem, but synchronization can be no better than clock resolution. This issue justifies further analysis in the context of the target range and velocity profiles. That is, how precisely must clocks be synchronized to hit targets with the expected range and velocity profiles? How much jitter will cause the weapon to miss the target? Are ATA estimates of velocity accurate enough to assure high probability of hit of a target with the expected range and velocity profiles?

5.3. Motion Control

Turret motion control design is driven by the need to point the weapon at the target and track target motion. Requirements for resolution, accuracy, speed, power, torque, and control system bandwidth were typical of control system design and were anticipated. The dynamic range of turret rotational velocity needed on the "tracking" turret was unexpected because of the unanticipated requirement for target tracking. The requirement for fast gross motion and the narrow dynamic range of the already purchased motion controller chip made smooth tracking of distant targets impossible. Also, the cables to the motor and encoder on the upper (elevation) stage of the turret, which had to wrap around the base of the rotator as the turret moved in azimuth, provided an unexpected packaging challenge. Otherwise, this mature technology is not challenged by the target engagement application.

5.4. Boresight

Boresight between the ATA sensors and the weapon system was an unexpected challenge. The boresighting procedure calls for the operator to choose a landmark visible in the ATA camera imagery and put a cursor on it, recording the pixel coordinates of the cursor. He or she then searches for the landmark by moving the turret until the landmark is visible in the zoom camera co-located with the weapon, recording the turret coordinates read from turret encoders. A number of boresight points must be so collected. In practice, a boresighting session seldom required less than an hour and usually required an operator at the OCU and a spotter, who helped the operator find the landmark in the zoom camera, at the vehicle. (Use of a zoom lens controllable from the OCU may simplify the process in a future version.)

Early work in subsystem development, with ATA and turret systems not geometrically fixed with respect to each other, required that parameters for the coordinate transformation between the reference frame of the ATA and that of the weapon turret be measured anew each time the turret or ATA camera was moved. A simple routine interpolating polar coordinates between pixel coordinates in the neighborhood was used initially but lacked mathematical rigor and had to be redone each time the vehicle was moved.

A new technique, which uses a genetic algorithm (from the artificial intelligence domain) to calculate elements of the coordinate transformation matrix, was introduced during DEMO 1. A genetic algorithm is an optimization procedure. In this application, the elements of the coordinate transformation matrix are the optimization variables, and the objective function to be minimized is the error term between turret aim points calculated by the transformation matrix and the corresponding points from the boresighting routine. The coordinate transformation technique appeared robust, surviving several vehicle moves without needing to be re-boresighted. The coordinate transformation technique, regardless of how its elements are calculated, can be extended as the ATA rotator (with two additional degrees of freedom) is implemented.

5.5. Weapon and Weapon Control

The most critical issue in firing a lethal round from a robotic vehicle is the IFF issue. It is the probable reason that the weapon subsystem for RTB was banned. IFF is a difficult problem even in conventional (manned) combat. The Army is seeking solutions to this problem in another program not discussed here. The approach of the RTB was to require the operator to give concurrence to fire on a potential target, effectively leaving him in the control loop for the IFF function. Another solution considered was to use the RTB in a role similar to that of a mine, that is, in a free fire zone where the only moving objects anticipated in the field of fire are enemy. Robotic weapon systems are customers for new IFF technologies. More sophisticated ATA technologies that use image understanding for target identification (e.g., distinguish between an M-2 and an M-3 Bradley) will help, but stand-alone IFF technologies will probably remain necessary for likely future wars against former arms customers, where an M-2 may be either friend and foe. Future military robots must use the best of the developing IFF technologies. Military robots will not, however, drive the requirements for such a system. The Army needs a bolt-on IFF system for its manned systems. Military robots will use it as a peripheral device to the robot control computer.

Safe-and-arm issues were not significant in DEMO 1 since no lethal weapon was implemented. However, separate commands were required to arm and fire the MILES, each command activating separate circuitry. Were the weapon real, more resources would have been devoted to assuring that these circuits were fail-safe. Safe-and-arm practices for robotic weapon systems need not differ at the technology level from those used in today's electronically fired weapons. The peculiar focus for a robotic weapon system is on the operating procedure that determines at what point in the mission and during what circumstances the weapon is armed. In a teleoperated weapon system, this critical decision remains in the hands of the soldier. His or her training in the use of the robotic weapon and the detailed doctrine under which it is deployed must address safe and arm as a critical issue. Algorithmic implementation in more autonomous systems needs thorough evaluation, both analytical and in simulation, to assure that mistakes are not made.

A weapon for a robotic vehicle should probably be inexpensive, since part of the appeal of a robot is that it can be sacrificed. If it is expensive, it should offer some significant performance benefit over a manned version. It should be adaptable to a small chassis, a characteristic desired by the infantry user. It can generate lots of overpressure or hazardous by-products, since no soldier needs to be nearby when it fires. It can require precision in its control, since robotic systems can be precise, internally. It can require speed in its control because robotic systems can respond to input in milliseconds or less.

A round from a weapon wielded by a robotic vehicle needs a lethal footprint broad enough to strike the target's vulnerable area, given the characteristics of the ATA and the turret. While the lay error of weapons aimed by soldiers is well understood, this is not the case with ATA systems.

6. FUTURE WORK

6.1. Automatic Target Acquisition

In the RTB, targets are located in only two degrees of freedom, that is, in the plane of the ATA camera. It is desirable to add the third degree of freedom, that is, range, to the target for use in ballistics calculation and target position reporting. Laser rangefinder technology is reasonably mature, affordable, and accurate, and will be added in future work. The rangefinder will logically be installed on the weapon turret to allow range measurements to be taken at several points in the ATA image.

The target acquisition approach implemented in the RTB is a simple one: it uses a single stationary sensor and requires targets to be moving to be detected. This is very restrictive. The ability to use multiple sensors to track while moving and also to acquire stationary targets is a necessary future extension of this system. Automatic target recognition (ATR) algorithms are improving and could be integrated into the RTB as a way to detect stationary targets, to further detect false targets, and to assist the operator in the IFF function. By processing multisensor data simultaneously and combining the results, one should be able to obtain a system with a higher probability of detection and lower probability of false alarm. Cooperative target acquisition from multiple RTB vehicles is another way in which the ATA system can be improved, that is, in addition to being able to have one RTB verify the ATA results of a second RTB, multiple, non-collocated RTBs can report targets to each other as targets enter and leave each RTB's FOV.

6.2. Fire Control

Given the on-board sensors for sensing position and attitude in geographical coordinates, a common terrain data base, and a communications infrastructure, fire control may assume a new dimension as a target sensed by one vehicle can be reported to another for the kill. Many trade-offs necessary in designing a target acquisition and engagement system can be mitigated and design opportunities opened if acquisition and engagement can be packaged separately. For example, an expensive and covert acquisition system can feed the exact location of targets to cheap, blind (e.g., no on-board ATA), and fearless shooters (or perhaps artillery). Depending on the accuracy of the navigation sensor system, the shooter system might need a less capable acquisition system of its own to refine the relative position of the target sufficient to hit it with a round, or a more expensive smart round might be able to find the target even if the shooter system knows its location only approximately. Many combinations are possible, and an understanding needs to be developed of the capabilities, limitations, and relative costs of these emerging technologies.

6.3. Motion Control

The ability to fire while moving is a highly desirable feature, currently available only on top-of-the-line military systems. It places additional demands on the motion control subsystem, however, because it must not only point the weapon, but also overcome the noise function of the vehicle's motion over terrain. New technologies for sensing this motion at high bandwidth are necessary so that the control loop can be closed in the inertial coordinate frame. (Sensor requirements for this application are similar to those of the inertial reticle application being studied at ARL's Weapons Technology Directorate.) This type of sensor would permit targets to be assigned to a robot that fires at the assigned position while moving across rough terrain yet has no target acquisition capability itself.

6.4. Boresight

The next frontier in boresighting is boresighting target acquisition on one vehicle with a weapon on another vehicle, termed "target hand-off." Such a boresight requires knowledge of the location of each vehicle with respect to the other in six degrees of freedom, so that location of the target with respect to the target acquisition vehicle can be converted to location of the target with respect to the weapon-carrier vehicle. This presents a severe challenge to the state of the art in position and attitude sensing. The state of the art must be evaluated against the requirements, sensitivities determined, ingenuity applied to simplify the problem, and anticipated error terms quantified. This technology would enable an expensive ATA vehicle to discretely report targets to expendable weapons carriers, which would bear the brunt of return fire.

6.5. Weapon and Weapon Control

Future military robots must use the best of the developing IFF technologies. It is unlikely that military robots will be armed before the advent of a reasonably mature IFF system. The military robotics community must identify itself as a customer of the IFF community and must provide support as required.

Algorithms implementing safe-and-arm decisions in autonomous systems need thorough evaluation, both analytical and in tactical simulation, to assure that mistakes are not made.

The weapons community must develop a new understanding of requirements and opportunities for weapons applications to robotic vehicles, which is free of preconceptions based on the limitations of manned vehicles. The requirements should address characteristics essential to a set of likely military robots, for example, a HMMWV-based robot, a small chassis robotic vehicle, and a robotic howitzer. The opportunities should be examined for potential benefits from robotics and the characteristics of a weapon. Of particular interest is the division of intelligence among the ATA subsystem, the weapon pointing subsystem, and the round.

7. CONCLUSIONS

The RTB has shown the potential of robotics technologies to the acquisition and engagement of moving ground targets. Initial testing indicates that military targets can be acquired and engaged, but the limited amount of data collected to date have not allowed the project team to understand the system's capabilities, limitations, and internal error budget. Much more testing is required to learn the lessons this test bed is capable of teaching.

Continued interest in military robots assures an eventual requirement for an unmanned weapon-using capability. Today, the RTB is DoD's best mechanism for learning how to successfully implement this capability.

8. REFERENCES

Balakirsky, S., David, P., Emmerman, P., Fisher, F., Gaylord, P., Semi-Autonomous Mobile Target Engagement System, Symposium of the Association for Unmanned Vehicle Systems, June 1993.

Ballard, D., and Brown, C., Computer Vision, Prentice Hall, 1982.

Bhanu, B., Evaluation of Automatic Target Recognition Algorithms, Proc. of SPIE, Vol 435, 1983, pp. 18-67.

Bar-Shalom, Y., Tracking Methods in a Multitarget Environment, IEEE Trans. on Automatic Control, Vol. 23, No. 4, August 1978, pp. 18-27.

Bostelman, R., Russell, R., and Wallace, W, A Portable Driver Control Station for a Remote Vehicle, unpublished National Institute of Standards and Technology Internal Report, Dec., 1989.

Eilers, D., and Schnetzer, E., 2-Dimensional Video Motion Detection System, Sandia National Laboratories Report, 1989.

Hayes, J., Computer Architecture and Organization, McGraw Hill, 1978.

Scott, H., Vehicle-Command Center Communication of a Robotic Vehicle System, International Journal of Robotics and Autonomous Systems, October, 1990.

INTENTIONALLY LEFT BLANK.

9. BIBLIOGRAPHY

David, P., Balakirsky, S., and Hillis, D., A Real-Time Automatic Target Acquisition System, Conference on Unmanned Vehicle Systems, July 1990.

DePiro, F. W., Noell, T. E., Gee, T. F., Remote Driving with Reduced Bandwidth Communication, Sixth Annual Space Operation, Application, and Research Symposium, August 6, 1992.

Haug, B. T., Mission Module for Robotic Testbed, Proceedings of the Robotic Vehicle Working Group, Albuquerque, NM, Feb 1991.

Haug, B. T., "Robotics Testbed-Mission Module, Proceedings of the Robotic Vehicle Working Group, Pasadena, CA, Mar 1992.

Murphy, K., Navigation and Retro-Traverse on a Remotely Operated Vehicle, Proceedings of the IEEE Singapore International Conference on Intelligent Control and Instrumentation, Singapore, February, 1992

Murphy, K., Juberts, M., Legowik, S., Nashman, M., Schneiderman, H., Scott, H., Szabo, S., Ground Vehicle Control at NIST: from Teleoperation to Autonomy, Seventh Annual Space Operations, Applications, and REsearch Symposiumm Houston, TX, August 3-5, 1993.

Shoemaker, C. M., Low data rate control of unmanned ground vehicles, Proceedings of the AUVS-90, Dayton, OH, July 30-Aug. 1 1990.

Szabo, S., Murphy, K.N., Scott, H.A., Legowik, S. A., Bostleman, R. V., Control System Architecture for Unmanned Land Vehicles. Proceedings of the Association for Unmanned Vehicle Systems (AUVS), Huntsville, AL, June 21-24, 1992.

Szabo, S., Scott, H. A., Kilmer, R. D., Control System Architecture for the Team Program, Proceedings of the Second International Symposium on RObotics and Manufacturing Research Education and Applications, Albuquerque, NM, November 16-18, 1988.

Szabo, S., Scott, H. A., Kilmer, R. D., Control System Architecture for Unmanned Ground Vehicles, Proceedings of the AUVS-90, Dayton, OH, July 30-Aug. 1 1990.

Szabo, S., Scott, H.A., Murphy, K.N., Legowik, S. A., Control System Architecture for a Remotely Operated Unmanned Ground Vehicle, Proceedings of the 5th IEEE International Symposium on Intelligent Control, Philadelphia, PA, September, 1990.

Szabo, S., Scott, H.A., Murphy, K.N., Legowik, S. A., Bostleman, R. V., High-Level Mobility Controller for a Remotely Operated Unmanned Land Vehicle, Journal of Intelligent and Robotic Systems, Vol. 5, pgs 63-77, 1992.

INTENTIONALLY LEFT BLANK.

APPENDIX A

SOFTWARE

INTENTIONALLY LEFT BLANK.

SOFTWARE

The appendix is composed of subroutine descriptions representative of three levels of the software architecture.

Dispatch level

```
/************************************************************************/
/************************************************************************/

/* dispatch.c

Contains top-level routines involved in dispatching commands (in CMD_STRUCT
        format) to the turret level

*/

/************************************************************************/

/************************************************************************/

/*  dispatch(msgIn)

        calls the subroutine indicated in the CMD_STRUCT msgIn.  Checks for
        acknowledge request, and sends ACK_REPT of command status if requested.



        Variables set:
        Variables used:
        Functions called:  setTime,sendStatus, turretMoveTask, shootTarget,
        platformGo, trackSetUp, shutdown, cmdPrt, printErr, netSend
        See also:

*/

/************************************************************************/

/* setTime()

        synchronizes clock

        Returns:  status
        Variables set: turretStatus
        Variables used:  turretStatus
        Functions called:  setTimeZero()
```

```
*/

/************************************************************************/
/*turretMoveTask(pPose)

    Claims turret resource.  moves turret to position in command pPose.
    Reads encoder on arrival.


    Returns:  status
    Variables set: turretStatus
    Variables used: disPrt,turretStatus,
    Functions called: turretSetDest(),turretGoTo(), RdEnc()
    Error:  ERR_DIS_TURRET_NOT_READY
    See also:

*/

/************************************************************************/

/*  shootTarget()

    Claims turret resource.  Moves turret to intercept (moving) target.
    If target is properly intercepted, fires weapon.


    Returns: status
    Variables set: turretStatus
    Variables used: turretStatus , disPrt
    Functions called: lockOn(), fireWeapon()
    Error: ERR_DIS_TURRET_NOT_READY
    See also:

*/

/************************************************************************/

/*  lockOn(target)

    Intercepts (moving) target.  If turret reaches the calculated
    interception point on time, it returns status OK, otherwise
    reports error.


    Returns:
    Variables set: timeOfDispatch, tNow, timeToTarget
```

```
     Variables used: tgtAzWindow, tgtElWindow, tgtTimeWindow
     Functions called: RdEnc(),getRelTime(), cvtToLead(), tgtLead(),
        cvtFromLead(), turretGoTo(),
     Error:  ERR_DIS_MISSED_TARGET
     See also:

*/

/*********************************************************************************/


/*  cmdPrt(msgBuf)

     prints formatted data of msgBuf.  If not a known (or supported) command,
     does hex dump of buffer.

*/

/*********************************************************************************/

/*  trackSetUp(target)

     Claims turret resource.  initializes latestTrackPoint, previousTrackPoint
     and msgsMissed.  Sends turret to intercept target.  On successful
     intercept, spawns "track" process.


     Returns:  status
     Variables set: trackTask, latestTrackPoint, previousTrackPoint, msgsMissed,
       turretStatus
     Variables used:  turretStatus
     Functions called:  lockOn(
     Error:  ERR_DIS_CANNOT_SPAWN_TRACK
     See also:

*/

/*********************************************************************************/
```

Turret level

```
/************************************************************************/
/************************************************************************/


/*
 * turret.c author:  Gary Haas, after Tom Haug
      purpose:  A set of routines which convert between engineering
      units and native hardware units for the TEAM turret.
 */
/*********************************************************************

/*DegToCnt()

/*
 * Converts turret orientation in native encoder counts to degree engineering
 * units
 *
      Returns: degree units
      Variables used:  Cnt2Deg[], zero[]

*/

/************************************************************************/



/*CntToDeg()
/*
 * Converts turret orientation in degree engineering units to native encoder
 * units
 *


      Returns:  encoder units
      Variables used:  Cnt2Deg[], zero[]
*/

/************************************************************************/


/*calcMove()

/*
 * calculate the number of motor steps required to move to pose designated in
```

```
* the Cnt (or, for EL, elActuator) structure.  Store result in Cnt structure. *
*
*

      Variables set: Deg[].Delt,Cnt[].Delt,Cnt[].Dir
      Variables used:  Cnt2Step[]
      Functions called:   RdEnc()


*/

/***********************************************************************/


/* elFloatComp()

      Calculates the amount of the elevation measurement which varies as
      a function of azimuth at a fixed elevation actuator setting.  This
      so-called "float" arises from variations in the flatness of the
      bearing race.   Tis phenomenon appears only in the "pointing" turret;
      the "tracking" turret program simply returns a zero.


      Returns:  "float" error at the azimuth measurement of the argument,
        in encoder count units
      Variables used: floatLkup[]
*/

/***********************************************************************/


/*turretGoTo()

      Sends turret to location in Cnt[].Dest.  Calls routines to read
      encoders, calculate steps, and generate hardware signals.  Returns
      when turret has arrived at destination.

      Returns:  status
      Functions called:  RdEnc(),calcMove(), XYCgo()
      See also:  turretSendTo()


*/

/***********************************************************************/
```

```
/*turretSendTo()

        Sends turret to location in Cnt[].Dest.  Calls routines to read
        encoders, calculate steps, and generate hardware signals.  Returns
        immediately.  If turret is busy, returns with no error.

        Returns: status
        Functions called:  RdEnc(); calcMove(); XYCsend();
        See also:  turretGoTo()

*/

/*****************************************************************************/

/*turretSetDest()

        Puts arguments into Deg[].Dest and Cnt[].Dest buffers.  Calculates
        elActuator.Dest.  Checks destination to assure that destination
        is within the turret operating envelope.

        Returns:  status
        Variables set:  Deg[].Des, Cnt[].Dest, elActuator.Dest
        Variables used:  limAzHi, limAzLo, limElHi, limElLo
        Functions called:  elFloatComp()
        Errors:  ERR_TUR_SET_LIMITS

*/

/*****************************************************************************/

/*tgtTrack()

        This is the workhorse routine for target tracking.  It assumes a
        stream of target track points at a regular time interval.  Given
        such a track point as a parameter, the routine extrapolates from
        trackpoint position and velocity the current target position, and
        its position halfway to the time of the next update (the "lead").
        It also calculates current aiming error.  It then sends the turret
        to the "lead" location in either point-to-point mode or in a mode
        which chooses the closest available (piecewise) constant velocity
        to come near the "lead" location.


        Operating modes:
          pToPMode = 1 uses the indexer's point-to-point mode.
          pToPMode = 0 uses the indexer's velocity mode.
```

```
      prtOnError = 1 prints Cnt/Deg buffers on error
      abortOnError = 1 deletes the track task on error
      trkPrt = 1 prints the aimError values at each track point


    Variables set: tNow, Cnt[].Delt, Deg[].Delt, Cnt[].Vel, Deg[].Vel,
      Cnt[].Dir
    Variables used: tNow, Deg[].Posn, ataUpdateRate, pToPMode, Cnt2Deg[],
       Cnt2Step[], prtOnError, abortOnError, trkPrt
    Functions called: getRelTime(),turretSendTo(), RdEnc(), rateCode(),
      shutDown(), velocity_run(),prtDB()
    Errors: ERR_TUR_VEL_LIMITS


*/

/*************************************************************************/


/*track()

    This task, spawned from the dispatch subroutine on receipt of a
    TARGET_TRACK command, polls latestTrackPoint for a non-null
    value.  If no track point has been received, the task delays to
    avoid monopolizing the CPU.  When latestTrackPoint indicates receipt
    of new data, track runs tgtTrack on the new data.  It then checks to
    see if any track points have been missed (indicating that the CPU is
    overtaxed) and so reports if in trkPrt mode.

    Modes:  Reports track points missed if trkPrt = 1


    Variables set: latestTrackPoint, previousTrackPoint
    Variables used:  latestTrackPoint, trkPrt, previousTrackPoint,
      msgsMissed
    Functions called:  tgtTrack()
    See also:

*/

/*************************************************************************/



raisePlatform()
```

Sets “raise” bit in platform command port

/***/

lowerPlatform()
Sets “lower” bit in platform command port

/***/

turretInit()
Initializes the driver board, creates watchdog timers, creates and initializes tracking task

/***/

int getSysCondition(sysCond)
Retrieves, interprets, and formats status of turret subsystem components in global status data structure

/***/

Driver level

```
/*
 * driver.c for the XYCOM085 board author:  Gary Haas & Minh Tran
 purpose:  This source
 * code provides routines for reading the encoders, sending commands to the
 * driver hardware, and monitoring the hardware interface port.  A provision
 * for testing the software in the absence of the turret interface board is
 * provided (see "driver.h").
 *
 */

/**************************************************************************

/*
 * rateCode calculates the CY525 rate code corresponding to a rate in steps/s.
  Most variables used are in driver.data.

       Returns:  CY525 rate code, or -1 if the rate is out of the linear range.
       Variables set:
       Variables used:  rateCompFactor
       Functions called:
       See also:

*

/*************************************************************************/


/*
 * RdEnc() reads the encoders into the Cnt[].Posn structure.  It also
 * converts the value to degrees to maintain Deg[].Posn consist with the new
 * Cnt values.
 *
 *
 *     Variables set: turret.XYC_STATUS, turret.XYC_COMMAND, Cnt[].Posn
       Variables used:  az_encoder, el_encoder
       Functions called:  CntToDeg(),encoder(),elFloatComp()
       Errors:  ERR_DRV_ENC_LIMIT
       See also:
 *
 */
/*************************************************************************/

/*
 * XYCgo() is a simple passthru to routine pointToPointRun
 *
```

```
 *
 *
 *    Returns:
 *    Functions called: pointToPointRun()

 *
 */
/************************************************************************/
/*
 * XYCsend() moves the Cnt data structure to the XYCOM port and issues the "go"
 * command.
 *
 *

/************************************************************************/


grnTurret()
/************************************************************************/

/*

/* drvInit() initializes the PIT-68230s and cy525s.   525's are
      initialized to point-to-point mode.


      Variables set: ip1_68230, ip2_68230, cy_1, and cy_2 arrays
      Variables used: PtoPRaz, PtoPFaz, PtoPSaz, PtoPRel, PtoPFel, PtoPSel
      Functions called: cy1_transfer(), cy2_transfer(),encoder()
      Errors: ERR_DRV_XYC_FATAL
      See also:

*/

/************************************************************************/


/* cy_reset()
      resets CY525 chips

*/

/************************************************************************/


/* cy1_transfer() transfers data from the PIT-68230s to azimuth cy525.
```

```
        Times out if 525's freeze up.

        Returns:  OK, or ERR_DRV_CY_XFER_TIMEOUT on timeout.
        Variables set: perror
        Variables used:
        Functions called: cy_reset();
        See also:

*/

/************************************************************************/


/* cy2_transfer() transfers data from the PIT-68230s to elevation cy525.
        Times out if 525's freeze up.

        Returns:  OK, or ERR_DRV_CY_XFER_TIMEOUT on timeout.
        Variables set: perror
        Variables used:
        Functions called: cy_reset();
        See also:


*/

/************************************************************************/

/*
/*
 * encoder() reads the encoders. Places values in global variables.
 *       Returns error if encoder data bits are all 0 or all 1
 *
 *      Returns:  OK, or  ERR_DRV_XYC_FATAL
 *       Variables set: az_encoder, el_encoder;
 *      Variables used:  testmode, ipX_68230 arrays
 *      Functions called:
 *
 */

/************************************************************************/

/*


/* shutDown() stop the turret at anytime, using the CY525 abort line.

        Variables set:
```

```
        Variables used:
        Functions called:
        See also:

                !!!!!!!!!!!!!   needs a timeout    !!!!!!!!!!!!!!!

*/

/**********************************************************************/

/*

/*  velocity_run() runs the turret in the velocity mode.


        Variables set:
        Variables used:
        Functions called:  cy1_array(); cy2_array(); cy1_transfer(m); cy2_transfer(m);

*/

/**********************************************************************/
/*

/* cy1_array() sets up array for velocity mode xfer to CY chip.  Uses
        rate and direction arguments, F and S velocities from global
        variables


        Variables set: cy1 array
        Variables used: VelFaz, VelSaz
        Functions called:
        See also:

*/

/**********************************************************************/


/* cy2_array() set up cy2_array  for velocity mode xfer to CY chip.  Uses
        rate and direction arguments, F and S velocities from global
        variables


        Variables set:
        Variables used: VelFel, VelSel
```

```
        Functions called:
        See also:

*/

/*****************************************************************************/

/*
/*
 * pointToPointRun() runs the turret in point-to-point mode.  The program
 * loops until the turret has reached its position.  During each loop, the
 * routine executes a 1-count taskDelay to allow the CPU to keep up with
 * communications.
 *
 *
 *      Variables set:
 *     Variables used:
 *     Functions called:  cy1_transfer(), cy2_transfer(m) *
 */

/*****************************************************************************/

/*
 * pointToPointSend() checks to see if the turret is busy, and if not,
 *      sends the turret  to a point and returns.  It does not wait until
 *     the turret has reached its position
 *
 *     Variables set:
 *     Variables used:
 *     Functions called:  cy1_transfer() cy2_transfer()
 *
 */

/*****************************************************************************/


platformGo() sets a bit on the platform driver board which causes
the current platform command (raise or lower) to be executed by
the platform drive chipset



/*****************************************************************************/


emergencyStop() executes shutDown, stopping the turret drive motors
```

```
/********************************************************************************/
```

int getPlatformStatus() retrieves the status register of the platform driver board, interprets it, and places in global status data structure

```
/********************************************************************************/
```

prtBd() prints the registers of the turret control board

```
/********************************************************************************/
```

APPENDIX B

PHOTOGRAPHS

INTENTIONALLY LEFT BLANK.

Figure B-1. The "Pointing" Turret in its Original Paint Scheme, with Dummy Sensor Package, Aboard an RTB.

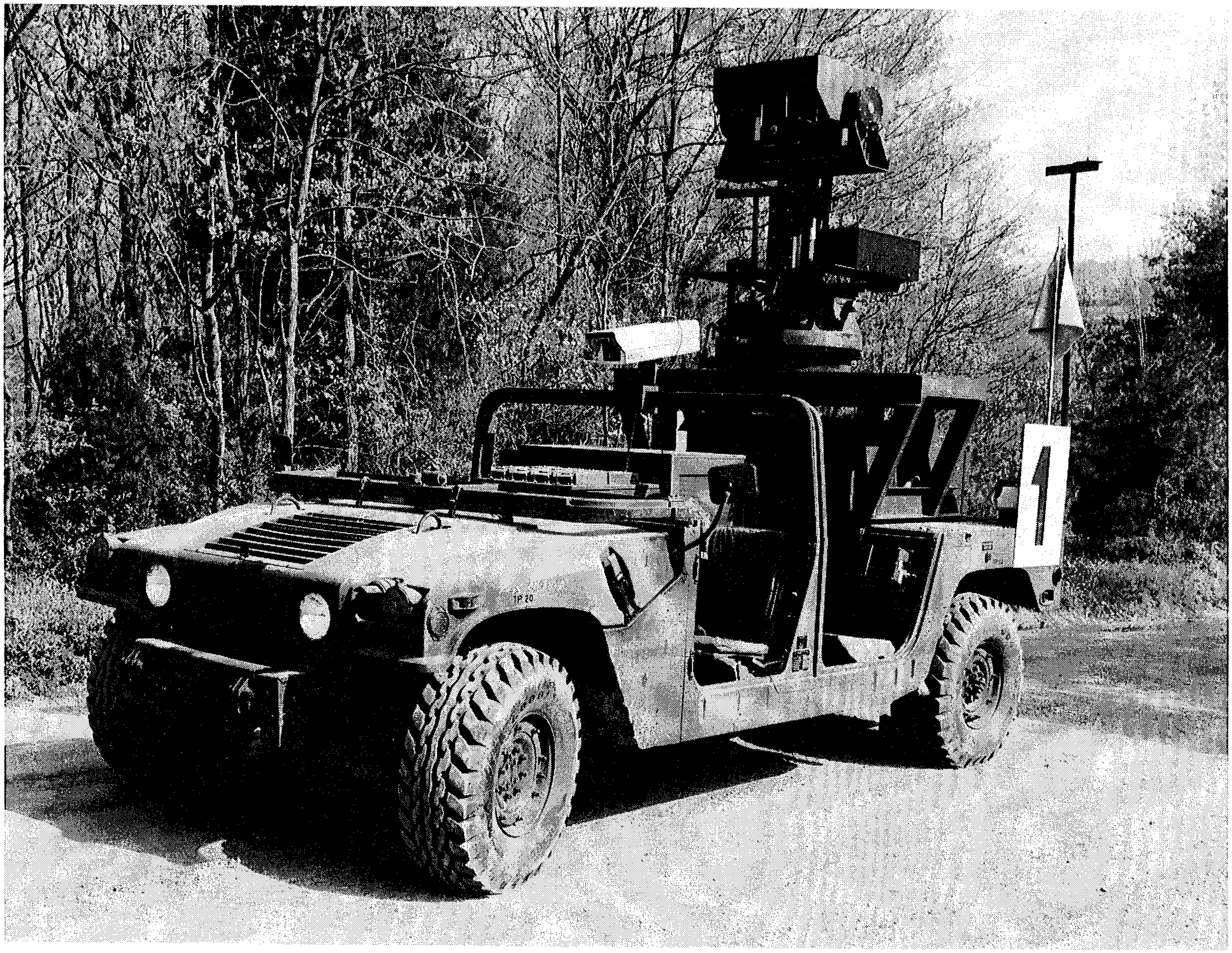

Figure B-2. The "Tracking" Turret Aboard RTB #1 at DEMO 1.

Figure B-3. The "Pointing" Turret Aboard RTB #2 at DEMO 1.

Figure B-4. "Tracking" Turret with Mission Package Electronics Exposed.

Figure B-5. RTB Personnel Working on ATA Sensors.

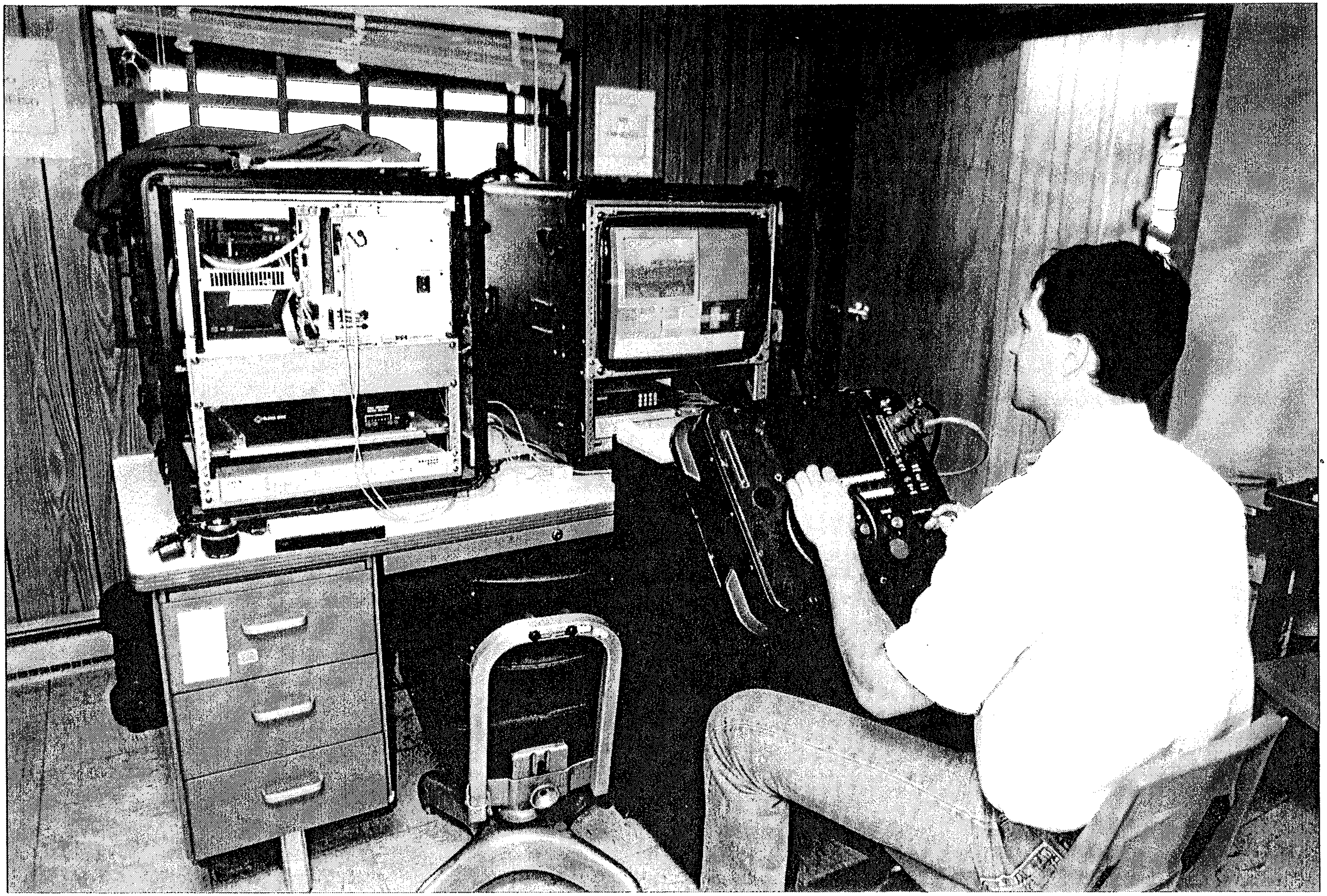

Figure B-6. RTB Personnel Operating OCU.

NO. OF
COPIES ORGANIZATION

2 ADMINISTRATOR
DEFENSE TECHNICAL INFO CENTER
ATTN DTIC DDA
8725 JOHN J KINGMAN RD STE 0944
FT BELVOIR VA 22060-6218

1 DIRECTOR
US ARMY RESEARCH LABORATORY
ATTN AMSRL OP SD TA/
RECORDS MANAGEMENT
2800 POWDER MILL RD
ADELPHI MD 20783-1197

1 DIRECTOR
US ARMY RESEARCH LABORATORY
ATTN AMSRL OP SD TL/
TECHNICAL LIBRARY
2800 POWDER MILL RD
ADELPHI MD 207830-1197

1 DIRECTOR
US ARMY RESEARCH LABORATORY
ATTN AMSRL OP SD TP/
TECH PUBLISHING BRANCH
2800 POWDER MILL RD
ADELPHI MD 20783-1197

12 DIRECTOR
US ARMY RESEARCH LAB
ATTN AMSRL IS SA
FRANCIS N. FISHER JR
PHILLIP DAVID
MARK COLEMAN
THOMAS MILLS
DAVID HILLIS
STEVE BALAKIRSKY
LAUREL SADLER
PHILLIP GAYLORD
SFC PAUL IMES
ROBERT LYNCH
JOHN M. VAN ETTEN
STEWART YOUNG
2800 POWDER MILL RD
ADELPHI MD 20783-1145

1 DIRECTOR
US ARMY RESEARCH LAB
ATTN AMSRL IS
PHILLIP EMMERMAN
2800 POWDER MILL RD
ADELPHI MD 20783-1145

NO. OF
COPIES ORGANIZATION

6 DIRECTOR
US ARMY RESEARCH LAB
ATTN AMSRL IS PA
SEAN HO
JAMES BOWDEN
STEVEN BRAGONIER
RICHARD R GREGORY
JOHN LOCKER
TIMOTHY ROSE
2800 POWDER MILL RD
ADELPHI MD 20783-1145

1 US ARMY TACOM
M/S AMSTA ZR
ATTN BRUCE BRENDLE
WARREN MI 48397-5000

1 US ARMY TACOM
M/S AMSTA TR V
ATTN PAUL LESCOE
WARREN MI 48397-5000

1 NCCOSC
RDTE DIV 531
ATTN BART EVERETT
53406 WOODWARD RD
SAN DIEGO CA 92152-7383

1 OAK RIDGE NATIONAL LABS
PO BOX 2008
ATTN JOE HERNDON
OAK RIDGE TN 37831-6305

3 OAK RIDGE NATIONAL LABS
PO BOX 2008
ATTN DR WILLIAM HAMEL
DENNIS HALEY
BRAD RICHARDSON
OAK RIDGE TN 37831-6304

6 NIST
ROBOT SYS DIV
BLDG 220 RM B-124
ATTN KARL MURPHY
STEVE LEGOWIK
MARIS JUBERTS
SANDOR SZABO
HARRY SCOTT
DR JAMES ALBUS
GAITHERSBURG MD 20899

NO. OF
COPIES ORGANIZATION

1 US ARMY MICOM
UGVS/JPO
ATTN AMCPM-UG
COL JEFF KOTORA
REDSTONE ARSENAL AL 35802

1 US ARMY WAR COLLEGE
STUDENT OFFICER DET
ATTN LTC RICK LYNCH
CARLYLE BARRACKS PA 17013-5050

1 US ARMY - BELVOIR
ATTN AMSEL RD NV MN
LTC HARRY MCCLELLAN JR
10221 BURBECK RD STE 430
FORT BELVOIR VA 22060-5806

1 DR ERIK G METTALA
UNIV OF TEXAS - ARLINGTON
COLLEGE OF ENGINEERING
PO BOX 19019
ARLINGTON TX 76019-0019

1 MICHAEL TOSCANO
DEPT OF DEFENSE
PENTAGON RM 3B1060
OUSD AT STTS/LW
WASHINGTON DC 20301

1 LOCKHEED MARTIN ASTRONAUTICS
PO BOX 179
M/S H8380
ATTN MARK ROSENBLUM
DENVER CO 80201

1 UNIVERSITY OF PENNSYLVANIA
GRASP LABORATORY
ATTN RUZENA BAJCSY
34TH & SPRUCE STREET
PHILADELPHIA PA 19104

1 TOOELE ARMY DEPOT
ATTN SDSTE AE FRED ELDREDGE
BLDG 1005
TOOELE UT 84074-5004

1 THE ASSOCIATION FOR UNMANNED
VEHICLE SYSTEMS - INTERNATIONAL
1735 N LYNN STREET SUITE 950
ATTN BARBARA LINDAUER
ARLINGTON VA 22209-2022

1 ARPA
3701 N FAIRFAX DR
ATTN THOMAS M STRAT
ARLINGTON VA 22203

NO. OF
COPIES ORGANIZATION

2 RAND CORPORATION
M/S DTP-2
ATTN RANDALL STEEB
JOHN M MATSAMURA
1700 MAIN ST PO BOX 2138
SANTA MONICA CA 90407-2138

ABERDEEN PROVING GROUND

5 US ARMY RESEARCH LABORATORY
TECHNICAL LIBRARY
ATTN AMSRL OP AP L
BLDG 305 APG AA

1 DIRECTOR USARL
ATTN AMSRL WT
I MAY
BLDG 328

1 DIRECTOR USARL
ATTN AMSRL WT
DR C MURPHY
BLDG 120

8 DIRECTOR USARL
ATTN AMSRL WT WC
J ROCCHIO
C M SHOEMAKER
DR J BORNSTEIN
R VON WAHLDE
C L HENRY
R M PHELPS
T T VONG
N M WOLFE
BLDG 120

9 DIRECTOR USARL
ATTN AMSRL WT WC
G HAAS (5)
B T HAUG
J J SPANGLER
M TRAN
J T MCLAUGHLIN
BLDG 1120A

2 DIRECTOR USARL
ATTN AMSRL WT WC
T L BROSSEAU
M D KREGEL
BLDG 390

1 DIRECTOR USARL
ATTN AMSRL IS SP
BARBARA BROOME
BLDG 394

NO. OF COPIES	ORGANIZATION
1	DIRECTOR USARL AMSRL WT WA MR HARRY ROGERS BLDG 120
1	DIRECTOR USARL AMSRL WT WB DR W P D'AMICO JR BLDG 120
1	DIRECTOR USARL AMSRL WT WE MS J LACETERA BLDG 394
1	DIRECTOR USARL AMSRL WT WF MR G HORLEY BLDG 1121
1	DIRECTOR USARL AMSRL WT WG DR L PUCKETT BLDG 120
4	DIRECTOR USARL AMSRL HR SC C C STACHOWIAK J W GOMBASH T W HADUCH D R SCRIBNER BLDG 459