

AD-A185 507

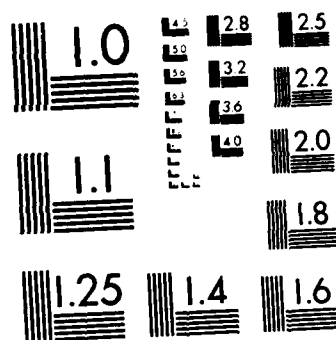
A QUERY DRIVEN COMPUTER VISION SYSTEM: A PARADIGM FOR
HIERARCHICAL CONTROL (U) MOORE SCHOOL OF ELECTRICAL
ENGINEERING PHILADELPHIA PA DEPT O R BAJCSV SEP 86
AFOSR-TR-87-1161 F49620-85-K-0018 F/G 12/9

1/1

UNCLASSIFIED

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

REPORT DOCUMENTATION PAGE

AD-A185 507

IS
CTE

1b. RESTRICTIVE MARKINGS

3. DISTRIBUTION / AVAILABILITY OF REPORT
Approved for public release;
distribution unlimited.

2b. DECLASSIFICATION

NOV 2 1987

4. PERFORMING ORGANIZATION REPORT NUMBER(S)

5. MONITORING ORGANIZATION REPORT NUMBER(S)

AFOSR-TR-87-1161

6a. NAME OF PERFORMING ORGANIZATION

6b. OFFICE SYMBOL
(If applicable)

7a. NAME OF MONITORING ORGANIZATION

University of Pennsylvania

AFOSR/NM

6c. ADDRESS (City, State, and ZIP Code)

7b. ADDRESS (City, State, and ZIP Code)

Computer & Info. Science Dept.
200 S. 33rd St.
Philadelphia, PA 19104-6389

AFOSR/NM

Bldg 410

Bolling AFB DC 20332-6448

8a. NAME OF FUNDING / SPONSORING ORGANIZATION

8b. OFFICE SYMBOL
(If applicable)

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

AFOSR

NM

F49620-85-K-0018

8c. ADDRESS (City, State, and ZIP Code)

10. SOURCE OF FUNDING NUMBERS

AFOSR/NM

PROGRAM
ELEMENT NO.

PROJECT
NO.

TASK
NO.

WORK UNIT
ACCESSION NO.

Bldg 410

61102F

2304

A7

Bolling AFB DC 20332-6448

11. TITLE (Include Security Classification)

A QUERY DRIVEN COMPUTER VISION SYSTEM: A PARADIGM FOR HIERARCHICAL CONTROL STRATEGIES DURING THE RECOGNITION PROCESS OF THREE-DIMENSIONAL VISUALLY PERCEIVED OBJECTS

12. PERSONAL AUTHOR(S)

Ruzena Bajcsy

13a. TYPE OF REPORT

13b. TIME COVERED

14. DATE OF REPORT (Year, Month, Day)

15. PAGE COUNT

Final

FROM 7/15/85 TO 7/14/86

9/19/86

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)

FIELD

GROUP

SUB-GROUP

→ natural language, computer vision, knowledge driven
recognition, scene analysis ←

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

We have developed a system called LANDSCAN, which is an integrated vision system and the recognition process is knowledge driven. This knowledge is generated by a query in English. The visual information is a stereo pair of images, and the descriptions are being made on 3-dimensional information. *Keywords: computer applications, image processing, optical images,*

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT

☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION

22a. NAME OF RESPONSIBLE INDIVIDUAL

22b. TELEPHONE (Include Area Code)

22c. OFFICE SYMBOL

Vincent G. Sigillito

(202) 767-4939

NM

DD FORM 1473, 84 MAR

83 APR edition may be used until exhausted.

All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

87 9 24 201

FINAL REPORT

to

AFOSR

AFOSR-TR- 87 - 1161

**A Query Driven Computer Vision System: A
Paradigm for Hierarchical Control Strategies During the Recognition
Process of Three-Dimensional Visually Perceived Objects**

Ruzena Bajcsy

September 1986



Accession For	
NTIS GRA&I	☒
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

Table of Contents

1. Introduction	0
1.1. Computer Vision	2
1.2. Natural Language	3
1.3. Special Purpose Computer Architecture	3
1.4. Outline	3
2. Computer Vision	4
2.1. Low Level Image Processing with Active Sensors	4
2.2. Recovery of the 3D data	5
2.2.1. Stereo	5
2.2.2. Optical Flow	6
2.2.3. Focus	6
2.2.4. Vergence Angle	6
2.3. Surface Reconstruction and Representation	7
2.3.1. Depth Point Interpolation - Filling in the Gaps	7
2.3.2. Reconstructing and Representing Surfaces	8
2.4. Computer Vision Bibliography	9
3. The Natural Language Issues	11
3.1. The hypothesis generation and object recognition	12
3.2. Natural Language Bibliography	15
4. IPON - Advanced Architectural Framework for Image Processing	17
4.1. Introduction	17
4.2. Overview of IPON	17
4.3. Current Status of IPON	20
4.4. Conclusions	22
4.5. Image Processing Bibliography	22

List of Figures

Figure 1-1: General City Scene	1
Figure 1-2: Engineering Quadrangle of the University of Pennsylvania in Philadelphia	2
Figure 4-1: Optical Network	19

A Query Driven Computer Vision System: A Paradigm for Hierarchical Control Strategies During the Recognition Process of Three-Dimensional Visually Perceived Objects

1. Introduction

In our proposal "Query Driven Computer Vision System: A Paradigm for Hierarchical Control Strategies During the Recognition of Three-dimensional Visually Perceived Objects", written four years ago, we set out to build a system which is able to interpret a natural language query and automatically generate a recognition strategy. We listed as key features of the proposed system:

1. automatic generation of recognition strategies
2. natural language query interface
3. hardware implementation of hierarchical architecture for real time processing, including real time stereo computation.

Since this is the final report, we shall first describe our accomplishments during the last three years.

This research is a part of a larger research effort conducted in the GRASP (General Robotics and Active Sensory Perception) Laboratory, which in turn is a part of the Center for Artificial Intelligence at the University of Pennsylvania. The Center for AI is supported by two large five year grants: one coming from NSF--CER (Computer Experimental Research), which goes from September 1983 through August 1988, and the other coming from the Army Research Office, which goes from September 1984 through August 1989. The principal investigators on both of these grants are Professor A.K. Joshi with R. Bajcsy as Co-PI, and a few other Computer Science Professors making various contributions. All the equipment in the GRASP laboratory, except for the IKONAS image display (which was purchased from this Airforce Grant) has been purchased from these two large grants. Needless to say that due to the Center for AI and its funding, the research proposed in this grant is well backed in terms of facilities, (see also the section on Facilities) but we need the support for people in order to carry out the work.

We emphasize the role of the **active sensor** in our research. By active sensor we mean a camera(s) which can move and serve as a probe rather than just a static recorder of the scene. This should not be confused with active sensors like sonar, radar, structured light, and laser range finders, which actually transmit a signal into the environment and receive its echos. The human analogy for the active sensor paradigm is a pilot in an airplane who can move his/her head and eyes in order to improve the recovery of 3D information by combining stereo with motion, improving the visibility of some details by control of zoom and focus, and their like. The activity is not in transmitting signals, but in positioning the sensor and optimizing its parameters for the signals being received.

The second aspect we emphasize is the Natural Language (NL) query system where the user is expected to be continuously interacting with the conceptual/linguistic system and the perceptual domain. The query represents the objects and their spatial relationships in the scene which must be translated into those components that the perceptual module can identify. This

of course implies a study of modularity and specialization and yet requiring interaction between the purely perceptual entities and the conceptual/linguistic entities.

The last but very important component of this research is the aspect of real time processing. Here we are interested in the analysis of established perceptual algorithms that can be converted into parallel algorithms, and in the development of high performance computer architecture for their implementation.

All this research, though basic, is also very experimental. The accomplishment is in the system analysis. Because of the complexity of the scenes, sensing apparatus, and the processing strategies, we are testing the system with both real life photos as well as on a scene mock-up, or model. This latter capability is provided by a controlled and verifiable experimental environment including arrangements of known objects to form the investigated scene. For this purpose we use two scale models: one of a general city scene (Figure 1) and another of the engineering quadrangle of the University of Pennsylvania in Philadelphia (Figure 2). The latter is scaled at 300:1 and the objects are quite detailed. The importance of the controlled scene is that we can test the "goodness" (including accuracy and precision) of our vision operators by making actual measurements of the objects and comparing them to the scale model. Furthermore, we can use these scenes as a testbed for comparative studies of our vision operators/algorithms with similar operators from other laboratories.

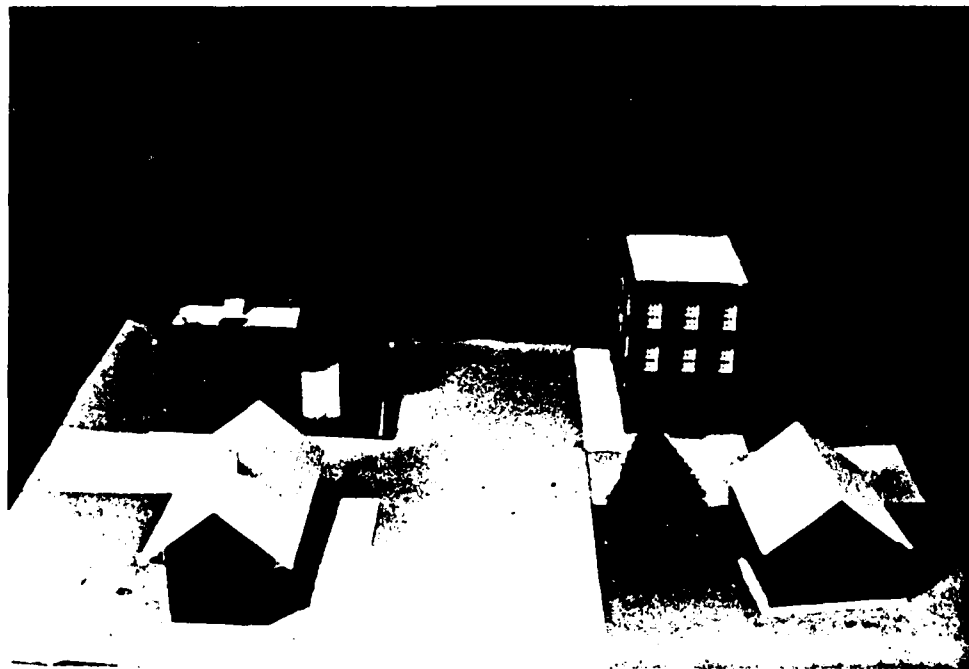


Figure 1-1: General City Scene



**Figure 1-2: Engineering Quadrangle of the University of Pennsylvania
in Philadelphia**

The basic research issues that we have been concerned with all along in this program are as follows.

1.1. Computer Vision

1. On the low level image processing we have investigated the robustness and the uncertainties of the low level visual operators, like the edge detectors, under different illuminations, different orientation, focus and zoom of the cameras.
2. For the recovery of three-dimensional information we are interested in how to combine redundant information and resolve conflicting data, such as what comes from stereo and optical flow.
3. Given 3D data points, we are concerned with how to identify 3D boundaries and surfaces, i.e., the 3D segmentation problem.
4. Rules for recognition strategies. Are there any principles? Can we separate the rules based on the knowledge about the camera parameters, the illumination and the semantics of the object?

1.2. Natural Language

1. Since this is a query driven system, the user can employ NL words to specify the spatial relations between the objects in the perceptual domain. One of the research issues then is to develop a computational model which maps these linguistic terms onto the perceptual model of the scene. This model must account for the meaning of the words which are related by the locative construct (i.e., spatial construct).
2. Also the user is expected to be continuously interacting with the conceptual/linguistic system and perceptual domain. We would like the system to behave in a cooperative manner with the user by correcting misconceptions, providing additional supportive information, etc. in much the same way as is done currently by NL interfaces to conventional databases. Here, however, we have extra degrees of freedom stemming from the active sensors, and their probing of the environment, that adds to the dynamics of this particular system. Thus one of the fundamental goals of this research is development of a **computational model** that accommodates this kind of interaction due to the capability of the perceptual module to acquire new data and/or reprocess already acquired data on demand from the query system.
3. Last but not least, the development of NL interfaces to an active perceptual module involves some key issues of knowledge representation, modularity, and communication between the linguistic/conceptual and perceptual components of the system.

1.3. Special Purpose Computer Architecture

1. We are investigating both hardware and software issues relating to the implementation of ultra-high performance systems for the execution of low and medium level image processing algorithms.
2. In terms of hardware, the Image Processing Optical Network or IPON is being developed as a high performance MIMD system based on a non-blocking optical interconnection network. A basic attribute of IPON will be the dynamically partitionable and reconfigurable network based on optical-hybrid technology for key components to provide high bandwidth communications, high capacity buffering, and certain types of high speed processing.
3. User level programming of IPON will be accomplished using the concept of process level dataflow control via an interactive graphical image processing language. Of fundamental importance here is the design of optimal strategies for the static and dynamic allocation of resources (processors, memory, communications links) and real-time scheduling.

1.4. Outline

In the subsequent chapters we shall describe in more detail our results for the last three years and our plans for new research. It will be divided into three parts:

- the computer vision investigation,
- the natural language problem, and
- the special purpose architecture development.

2. Computer Vision

The computer vision section will be further subdivided into three sections:

- the low level image processing with active sensor
- the recovery of 3D information;
- and the surface reconstruction, representation and interpretation.

2.1. Low Level Image Processing with Active Sensors

Traditional approaches with static images use much low level image processing which concentrates on filtering and edge detection. In the context of active sensing we are seeking measurements from the current scene to feed back and control the various parameters of the active camera: size of the lens aperture, positioning of the head, orientation and the viewing angle, zooming in on the area of interest and converging on some points of interest with the vergence control of the stereo camera.

We have investigated several edge detectors and filters in the domains of both time and space. In particular, we have experimented with a non-directional edge detector very much like the Laplacian of Gaussian function, a directional edge detector using the Gabor filter, another directional edge detector approximating the first derivative of intensity [3;David84] and features of the intensity functions, such as the first and second derivatives, very much like Haralick's Topographic Primal Sketch [6;Crowley;Bajcsy86].

It is very clear that different filters and features are suitable depending on the scene, its illumination and the opening and closing of the camera aperture (iris). The open issues are:

- a) what is the feedback signal for the camera in terms of opening and closing the aperture with respect to the optimal contrast.
- b) How should differently scaled filters and their corresponding edges be combined in order to obtain the "best boundaries" of objects. Here we define contours as 2D outlines that are obtained from edges, and the label boundary denotes the true 3D boundaries of objects.

For this we propose the following study: a laboratory set-up with a fixed scene, for example a mock-up of a fictitious city (see Figure 1) with a 10-channel illumination setup which can be precisely computer controlled. What one wishes to measure is a function of the magnitude of an edge with respect to changes of two parameters: first, the illumination of the scene, the size of the aperture; second, the scale (bandwidth, standard deviation) [14;Terzopolous82] of the filter which is used before the edge detector is applied.

We hope to prove or disprove two hypotheses: one, that for every scene (depending on the material of objects in the scene) and the illumination there is an optimal degree of opening of the camera's aperture; the other is that the scale on which the edge is detected the "best" is proportional to the size of the object and to the detail that the observer is interested in.

Other low level image processing consists of linear and non-linear filtering (see Appendix 2).

2.2. Recovery of the 3D data

In this section we wish to study how to recover the 3D information from a stereo pair of images, a series of images taken in time, and controlling the vergence angle between a stereo pair of cameras.

2.2.1. Stereo

The problem of stereo is traditionally divided into two parts: the correspondence problem (which is the difficult one), and computing the true (in some absolute coordinate system) depth value. We assume that the camera calibration problem has been solved, including the problem of scan line registration [9]. First we shall deal with the problem of correspondence and matching. The computation of the true depth value will be treated when we discuss the use of the vergence angle.

The stereo matching problem: During the last year or so we have experimented with a combined edge-region matcher (Appendix 1). Although the results were encouraging, we wished to understand the inherent limitations of a stereo matcher of static scenes. Hence, we embarked on the following problem: Given two 2-D projected views of a 3-D scene which differ by an arbitrary but known transformation, one needs to find unique matching between corresponding points. We assume that the input data for both images is a series of edge maps recovered through different filters and/or features.

There are two possible errors:

1. features in each image that should be matched but are not--the true negatives;
2. the features in each image that should not be matched but are matched--the false positives.

Furthermore, from the total number of features not all have a match, due to partial occlusions. So the total number of matchable features is less than the total number of features in either image.

What are the parameters or features upon which matching may occur?

1. edge points
2. edge segments
3. two edges and their relationship (corners, intersection,...)
4. more than two edges
5. enclosed contours.

To test the feature based stereo as opposed to edge point we developed a line based stereo. While here point based stereo is limited to the parallel camera setup (since it uses matching or scene line by scene line base) vergence angle = 0, it is ? in principle to any larger distances. The line based stereo is a general point based stereo matcher. We are currently evaluating its robustness and efficiency.

The selection of the particular feature from the above list (and there could be more) depends on two criteria:

- *Uniqueness*, i.e., we wish to have such a feature that uniquely finds its corresponding match; and

- **Robustness**, i.e., we need such a feature which will not be sensitive to the camera transformation.

From the uniqueness condition it would appear that the feature should be as rich as possible (ideally the whole object). On the other hand, from the robustness condition would follow the requirement for as small feature as possible. Our task is to find the optimum compromise between the two extreme criteria.

2.2.2. Optical Flow

The problem: Given a series of images and a particular feature in time, the problem is to compute the vector (its magnitude and direction) of the feature spatially displaced over time. The problem is similar to stereo computation in that the issue is to find the proper features upon which one can match and then solve the correspondence problem. The problem is different from the stereo in that while in stereo there is an angular disparity, in the time sequence when sampling rate is high the positional disparity between the consecutive images is purely translational.

For the computing of optical flow we have investigated the following features:

- No features--the Horn and Schunk method; [8]
- Motion energy--Adelson's method [1]
- Burt's correlation method [7].

The advantage of the first two methods is that there is no need for solving the correspondence problem. However, the price for that is high! In Horn and Schunk's method the smoothness constraint is a terribly limiting factor. In Adelson's method we are getting only the motion energy and the movement direction left and right, no other. This method uses filters sensitive to space/time oriented intensity changes. This work is in progress and it still remains to be seen whether we will be able to use this method for recovery of 3D from motion parallax [13].

2.2.3. Focus

Three-dimensional data can also be recovered from a scene using "depth from focus". We are building hardware to automatically control focus. We have developed four different techniques for measuring focus sharpness, including (in increasing computational complexity) scan-line sum-modulus-difference of intensity, grey-level population entropy, grey-level variance, and power spectrum energy distribution analysis (via radial histogramming) [10].

These techniques will be implemented and compared with respect to their effectiveness in improving focus to the extent that one point in the visual field can be said to be in focus, and from the position of that point on the image plane, the camera focal length, and the diameter of the aperture, we can precisely and uniquely determine the range of that point.

2.2.4. Vergence Angle

The last method in the recovery of 3D information is the use of vergence angle. This is a direct way of reading out the distance once the correspondence of the point has been established. The method is essentially triangulation. We are building hardware to both control and measure the vergence angle between two cameras. With this angle, the exact distance to

any point fixated in both visual fields can be discovered. Given this exact distance, the relative depth maps returned from stereo and optical flow can now be fixed as absolute depth maps [11].

We propose to use this device (designed and under construction) for accurate and unique absolute distance mapping of the visible surfaces and the stereo and the optical flow for filling in the gaps, which return *relative* distances.

2.3. Surface Reconstruction and Representation

From the previous section it should be clear that no matter how hard we shall work on various algorithms to obtain as perfect as possible 3D data, there is an inherent limit, due to well known physical limitations (occlusion, illumination, focus, zoom, orientation and the visible aspect of the object, to name a few) to the completeness with which 3D information can be recovered. So the next problem is how to supplement the missing data. The obvious answer is that some kind of interpolation method needs to be applied.

2.3.1. Depth Point Interpolation - Filling in the Gaps

The research issue for any scheme of filling the gaps is the trade-off between the measurements and the *a priori* information. We elaborate on this trade-off with an example. Let us suppose that we have a sparse array of 3D points after a stereo and/or optical flow computation. Remember we are left with some points that have not been matched either in the stereo matching nor in optical flow computation. In order to fill in the gaps we have several possibilities:

- a) we can ignore the unmatched points, i.e., have confidence only in those points (measurements) that have been matched. Then assume, let us say, a linear (or any polynomial) model (the *a priori* information about the local surface). Based on this we perform linear (or polynomial) interpolation between the neighboring points.
- b) An alternative to the case a) is instead of assuming the linear or polynomial models, which are inherently local, neighborhood models, assume a global smoothness constraint, which, using variational calculus, tries to fit the smallest and smoothest surface over the sparse data. [5].
- c) The third possibility is to assume a local smoothness constraint in the depth values. Then reexamine the unmatched points (match them with the closest edgels in the other image) and check whether their depth value would satisfy the smoothness constraint with the neighboring points.
- d) Finally, if, for example, from the outline we can identify measured objects, then clearly the "fill in gaps" process can use this information. An example of this case can be sidewalks or roads in aerial views.

As usual in machine perception, there is no one technique that works uniformly well in all cases. We believe that this is an integral part of the surface interpretation. One clearly needs all the above techniques available and then having a rule-based system use whichever give the "best" results. For example if we have one object in the view, then perhaps the third method is the "best". If one has reason to assume that one deals with objects that have only planar

surfaces, then the first method might be adequate. The third method is the most versatile since it uses the most measurements and the least *a priori* information. The cost is in computation. We have implemented all of these, and some partial results are shown in [12].

2.3.2. Reconstructing and Representing Surfaces

Having a rich set of depth points available, the next problem is how to find closed boundaries, and from them, surfaces, and finally description of objects.

Finding boundaries of objects versus their surfaces are two complementary mechanisms which work simultaneously in a cooperative fashion. For the problem of boundaries there are two problems that we wish to differentiate: one is to find the boundary of an object in a complex scene, that is to singulate (or segment) an object; the other is to identify boundaries among surfaces in the same object. In the first case the problem is of a decomposition of the 3D visible space into individual objects, for example, by finding the smallest enclosing convex polyhedron. In the second case we are concerned with finding enclosed curves or connected segments of lines that enclose a continuous surface.

While the problem of singulation of an object is the Ph.D. thesis of E. Krotkov (see his proposal), in this paper we shall report on the program for finding boundary lines, also called wire frames. Naturally, we assume that all visible boundaries are true physical boundaries. The process starts with looking for points of high curvature and corners. From these points, a divide-and-conquer method of recursive decomposition finds that line which has the lowest curvature and shortest path. Another method for finding contours which instead of divide and conquer first generates all possible contours and then uses graph search for finding the "best" contour in terms of some cost function was investigated by Heeger [7]. This work, though interesting as a plausible computational model for the psychophysical phenomenon of subjective contours, is inefficient for practical implementation with current sequential hardware. For the future we need to improve our corner finder! (See Appendix 1 for a discussion of how edge detection may also directly identify "edgels" as corner features). After obtaining lines in between the corners and/or high curvature points we still need to know which of these contours are closed. The process that performs this task also creates a graph (a linked list of vertices, edges, and faces) which serves as the basic data structure for further, higher-level processing.

All the above procedures get leverage by virtue of the fact that our objects are polyhedral. What remains an open research question is how to proceed when the surfaces within boundaries are not planar. One method we shall investigate is converting the set of 3D points into two images, one representing the surface normals and the other the range information. Then by applying region growing and/or edge detection techniques one should be able to discriminate between planar and curved surfaces [4]. The curvature of curved surfaces can be represented using splines [2].

How to go from the low level to identification of an urban scene is the work of Helen Anderson, as described in our proposal to AFOSR, titled **LANDSCAN - A Query Driven Recognition System**, submitted in June of this year. We assume to have an explicit graph--a tree of objects expected in an urban scene organized with respect to their height and shape priority. These two attributes guide the search in the image base.

2.4. Computer Vision Bibliography

- [1] Adelson, Edward, H. and Bergen, James, R.
Spatio-Temporal Energy - Models for the Perception of Motion.
Computer Science Princeton N.J., RCA, David Sarnoff Research Center, October, 1984.
- [2] Allen, Peter.
Surface Descriptions From Vision and Touch.
International Conference on Robotics (pp. 394-397), Atlanta, August, 1984.
- [3] Canny, John, F.,
Finding Edges and Lines in Images.
Computer Science MIT AI-TR-720, University of Pennsylvania, 1984.
- [4] Dane, Clayton Albert III.
An Object-Centered Three-Dimensional Model Builder.
Computer Science MS-CIS-82-50, University of Pennsylvania, 1982.
- [5] Grimson, W.E.L.
j: From Images to Surface: A Computational Study of the Human Early Vision System.
MIT Press, Cambridge, 1981.
- [6] Haralick, Robert, M., Layne, Watson T., Laffey, Thomas J.,
The Topographic Primal Sketch.
The International Journal of Robotics Research , 1983.
- [7] Heeger, David.
Filling In The Gaps: A Computational Theory of Contour Generation.
Master's thesis, University of Pennsylvania, 1984.
- [8] Horn, B.K. and Schunck, B.G.
Determining Optical Flow.
Artificial Intelligence Vol. 17:185-203, 1981.
- [9] Izaguirre, Alberto, Pu, Pearl and Summers, John.
A New Development in Camera Calibration Calibrating a Pair of Mobile Cameras.
Computer Science GRASPLAB, University of Pennsylvania, March, 1985.
Proc. 2nd Int. IEEE Conf. on Robotics, St. Louis.
- [10] Krotkov, E.P., J.F. Summers, and F. Fuma,
Computing Range with an Active Camera System.
In *To appear in International Conference on Pattern Recognition.* 1986.
- [11] Krotkov, E.P. and J.P. Martin,
Range from Focus.
In *Proceedings of IEEE Robotics and Automation Conference*, pages pp. 1093-1098.
April, 1986.
- [12] Smitley, D., Goldwasser, S., and Lee, I.
IPON - Advanced Architecture for Image Processing.
In *12th International Symposium on Computer Architecture.* Boston, MA, June, 1985.
- [13] Turner, M.R.,
Gabor Functions and the Detection of Differences Between Textured Regions.
To be published in Biological Cybernetics , 1986.

- [14] Witkin, Andrew, P.
Scale-Space Filtering.
Eighth IJCAI (1019-1022), 8-12 August, 1983.
Karlsruhe, West Germany.

3. The Natural Language Issues

One of our major tasks is the development of a natural language (NL) query system interface to a visual (perceptual) system. The reason for using NL is not because we want to construct a cute interface, but rather because the use of NL provides "flexibility" to the user. There are many aspects of "flexibility" that make such interfaces attractive for conventional databases or knowledge bases, and, of course, these will carry over to the perceptual domain also. However, the particular aspects of "flexibility" that are directly relevant to our domain are as follows.

The user can employ NL terms (words) to specify the spatial relations (and later actions in the robotics domain) in the perceptual domain. It is in these terms the user can best characterize the domain. The system then has the responsibility to map successfully these terms on to the terms (or composites of them) to the perceptual module of the system.

The semantics of spatial relational words (eg. spatial prepositions) is extremely complex. Determining the proper interpretation of a spatial preposition is not merely a matter of matching a preposition with a single representation. The interpretation of spatial constructs depends heavily on the entities which are related by that construct [4] [7]. For this reason, the system will have available to it the linguistic properties of the objects which may appear in the domain as well as a set of interpretations for the location of constructs based upon the semantic values of the entities it relates. The linguistic properties are those features which affect the usage and interpretation of a spatial construct (phrases describing the spatial relations between objects). Since the domain is a visual one, each object in the domain will have a "place" associated with it. This is what Herskovits calls the canonical geometric description of a spatial entity (objects) [Herskovits84]. Ordinary solid objects (buildings, vehicles, people) are bounded closed surfaces. Geographical objects are entities with slightly imprecise boundaries - roads, rivers, and fields. Some other properties which must be represented are a prototype shape and the allowable deviations from it, the relative size, and characteristic orientation - i.e. a table stands on its legs normally. The typical geometric conceptualization will also affect the choice of spatial construct - is the object normally viewed as a point or line. Along with the typical geometric conceptualization is the typical physical context of an object. For instance, a door is normally viewed as being in a wall. The normal function of an object, its relative size, its functionally silent parts and the actions commonly performed with an object will also be necessary for analyzing the spatial constructs.

For example, proper use of the preposition IN as in A is IN B involves not only computing containment (or partial containment) of A in B, but also assuring that B is in its normal orientation. Thus, in asking "Is the coin in the cup?" the user is assuming that the cup is in its normal orientation. If that is not the case and, say, the cup is upside down and the coin is under it, a response by the system "Yes" would be misleading, as it will tend to confirm the user's false presumption that the cup is in its normal orientation. An appropriate response is at least "No", but preferably (more cooperatively), "No, it is under the cup, the cup is upside down". Thus the system has to be sensitive to the normal orientation of objects in order to fully capture the semantics of IN.

The kind of cooperative behavior described above has been studied extensively in the context of NL interfaces to conventional databases or knowledgebases. Much of this theory and

technology for these domains can be successfully carried over to the perceptual domain. However, NL spatial terms have not been systematically studied from the point of view of developing interfaces for perceptual domains. A rather preliminary study has been done. However, this study is incomplete in many ways, especially in terms of the development of a computational model without which it is of no great value to our proposed task. Thus, one of our fundamental goals is the development of an appropriate computational model for the kind of interactional we want to support.

The second aspect of "flexibility" we call the query driven system. Given the number of relevant spatial relations between objects in a perceptual domain, it is impossible to precompute all the necessary relations. Our approach is "query driven" in the sense that, as a result of a query being asked, the system will compute the needed information from perceptual database as necessary. This dynamic behavior is not limited to just making some additional computations on already collected data, but will also involve acquiring new data, for example, by taking an additional view from a different angle (or getting new information from another modality), etc. The user is not constrained by what information has been collected already and what predicates have been precomputed. His queries will determine what information is needed to properly answer the query, and if that information is not available, then it will so inform the perceptual module. The perceptual module can then determine whether this new information can be computed from the data already gathered or whether it will require to get new data. Such behavior is initiated by the failure of the query at some level of interpretation. Such an opportunity is rarely available in the conventional databases, and even when it is available, it is of a very limited kind, as in the case of updatable databases.

If the reasoning processes fail to produce a positive response (the query fails to have an answer although it is syntactically correct), two types of query failure analysis are performed. The first type of query failure involves a query violating the global knowledge known about the domain. In this case, the system will respond with a message indicating that the query is conceptually ill-formed in this domain and why it is ill-formed. For instance, if the query asked how many walls the street had, the system would respond that streets do not have walls and that for that reason, the query is ill-formed. The other type of failure involves not finding the information requested in the scene model. In this case, rather than simply responding that the system was unable to find the data in question, of the scene with the old one in order to obtain a positive response to the query.

Thus the development of interfaces to an active perceptual module involves some key theoretical issues of knowledge representation, modularity, and communication between the linguistic/conceptual and perceptual components of the system.

3.1. The hypothesis generation and object recognition

The goal of the LandScan system is to perform query driven analysis for urban scenes. This places two constraints on the object recognition process: it must have top-down control structure, finding only those objects referenced in the query, and must encode global knowledge about a domain in which objects of the same type may have very different appearances. We have considered several different schemes for the representation of the global knowledge

necessary to perform object recognition such as frame based [5, 2], production systems, [6] and their like. We have finally settled for Augmented Transition Network (ATN) formalism because it enables the global knowledge to be encoded as a generative model for constructing objects from the primitives in the scene while driving the recognition in a top-down fashion [10].

The ATN formalism [1], [9] has been chosen to perform object recognition. Despite earlier failures using syntactic object recognition we have found that a higher level syntactic approach works well in the urban environment. It appears that there are "rules" to describe the recognition of objects in the urban, aerial domain. These objects appear to be composed of planes in fairly regular fashion even though their appearances may be quite different. For example, while two buildings may appear quite different, the relations between the planes which comprise each may be the same. Earlier attempts at object recognition using a syntactic approach failed because the primitives which were combined were too low level (edges, etc), the matching sequences were too strict, and the domains were not appropriate for a syntactic approach. In LandScan, the primitives used are higher level (surfaces) and thus have more information associated with them. Unlike other syntactic pattern matching systems, the grammar rules in LandScan do not specify a strict matching sequence. Instead they specify the properties which must hold between the simpler components of an object. Since the rules are more general there are fewer in the system thus simplifying the recognition process. The grammar enables the global knowledge about object appearances to be encoded as a generative model for objects of indefinite appearances. This also differs from the Tropt and Walters ATN for 3-D object recognition [8] first generates an hypothesis and then uses the ATN to verify the hypothesis is correct. The ATN operates using a top-down control structure - enabling the object recognition to be a query-driven process. In LandScan the control structure used in recognition has been separated from the global knowledge used in the recognition process. Thus finding additional object types only involves adding syntactic rules for recognizing these objects. It also implies that the control strategy used can be changed as long as it can still use the grammar rules.

The Augmented Transition Network (ATN) is composed of three parts: the grammar, a dictionary, and an interpreter. The grammar represents the *a priori* or world knowledge that the system must have in order to recognize objects and assign labels to subset of the scene. The dictionary presents the actual data which will be used in the recognition process- the surface model described above. The third component of the recognizer is the Lisp program which provides the control structure for the process. An object is recognized by traversing a network successfully.

The grammar as written is a two level network (this is considerably simpler than most ATN's which handle natural language utterances.) The bottom level concerns itself with the recognition of "simple objects." An object is simple if its further decomposition into parts will result in no entity which is in the domain of objects. For example, decomposing a building with a pitched roof will result in two halves of a pitched roof. Neither of these entities are considered objects in the domain - they are parts of objects. This level consists of the networks SIMPBUILD, SIMPSTREET, SIMPFIELD, and SIMPSIDEWALK. The top level combines the simple objects which were recognized in the first level of the network into "complex objects". A complex object is decomposable in a nontrivial way into at least one simple object. Each grammar rule represents

the components and relations which must hold between those components in order to be considered an object or "sub-object". The components are specified by the arc type - either an object primitive (surface) or a simpler instance of the object. The tests associated with the arcs encode the relations which must hold between the components as well as providing further checking for component features.

As objects are recognized, a dynamic model of the scene is incrementally built by adding more information to it as further image analysis occurs. The scene model in 3-D MOSAIC [3] is also incrementally derived as more data becomes available but the modelling process is data driven. LandScan builds a model using a query driven control. In other words, the modeller obtains more data as the user directs the vision system to analyze other areas of the scene which are of interest to him/her. Thus the Scene Model reflects the user's interest in the scene. The LandScan dynamic scene model is especially useful because it is flexible. the accuracy of the scene model increases as new data is acquired. Thus old hypotheses can be discovered false, deleted, and the scene model updated to reflect the more accurate understanding of the scene. In LandScan, when the scene analysis of a new image begins the scene model is empty. As questions are asked, the scene analyzer/constructor searches for the entities whose existence is in question using the object recognizer described above. As soon as the objects queried are found they are added to the Scene Model. Thus the Scene Model also reflects the history of the user's interest in the image. The dynamic scene model is composed of two components: a list of objects currently known to be in the scene and a set of matrices representing the p primitive relations hold between the objects on the object list. This design facilitates updating the scene model. To update the model the new object is simply added to the object list and the primitive relation matrices are expanded to include the relationship of the new object to all other objects in the model.

The first component of the scene model is the object list. The elements on this list are those objects which have been recognized during previous scene analysis operations. These objects are represented only by polyhedral surfaces, conceptually the most primitive component of an object. Thus to the high level reasoner it appears that objects are composed of only bounded planes - primitives at one level of representation. The use of a single primitive at one level of representation. The use of a single primitive (or a set of primitives which are not composed from one another) is conceptually clean to work with and is adequate for modelling objects in this domain. Each instance of an object in the scene has the information associated with it which was determined necessary to facilitate further scene analysis. The components of an object record are a name, the list of faces (polyhedral surfaces) comprising the object, its location in Euclidean three space (average of the centroids of all the faces comprising the object), and a subtype which gives more specific information about the expectations one can have about the object.

The relations in the scene model represent the primitive relations or topological properties between objects in the scene. The relations are ADJACENT, CONTIGUOUS, LOOKSADJACENT, LOOKSCONTIGUOUS, ABOVE, and CONTAINS. They are defined over the set of all objects currently recognized in the scene. These relations are defined similarly to their counterparts in the Surface Model. The relations are represented by their adjacency matrices because the adjacency matrix is easily updated and makes composition of relations simple. The composition

becomes a simple matter of boolean matrix multiplication for which there are many fast and efficient algorithms.

The combined use of the Scene Model and the object recognizer facilitates the following scene analysis operations: determining the relations, both complex and simple, among objects; locating and identifying specific objects and object parts. The existence of objects will be resolved in one of two ways - finding the object in the scene model by searching the object list, or using the recognizer to find a new instance of the object. To find an object part its face list will be searching until the part is found using the global knowledge about parts embodied in the object model. As for resolving the interpretation of locative constructs, the relations allow objects to be located relative to other objects in the scene using the matrix operations specified by the semantics of the spatial constructs. Suppose the question were asked, "Is there a car on the street?" An object of type CAR is ON an object of type STREET if the following primitive relations hold:

CONTAINS(STREET,CAR)
ABOVE(CAR,STREET)

The reasoner would determine if the CAR is ON the STREET by calculating the following relation composition:

CONTAINS * ABOVE^T

which would be calculated by a simple matrix multiplication of the CONTAINS adjacency matrix and the transpose of the ABOVE adjacency matrix. So the understanding of relational expressions will be accomplished by composing the primitive relations.

3.2. Natural Language Bibliography

- [1] Bates, Madeleine.
The Theory and Practice of Augmented Transition Network Grammars.
In Leonard Bolc (editor), *Natural Language Communication with Computers*. Springer-Verlag, 1981.
- [2] Glicksman, Jay.
Using Multiple Information Sources in a Computational Vision System.
In *Proceedings of the 8th International Joint Conference on Artificial Intelligence*. 1983.
- [3] Herman, Martin, Takeo Kanade, Shigeru Kuroe.
The 3D MOSAIC Scene Understanding System.
In *Proceedings of the 8th International Joint Conference on Artificial Intelligence*. 1983.
- [4] Herskovits, Annette.
Space and the Prepositions in English: Regularities and Irregularities in a Complex Domain.
PhD thesis, Department of Linguistics, Stanford University, 1982.
- [5] Hwang, V., T. Matsuyama, L. Davis, A. Rosenfeld.
Evidence Accumulation for Spatial Reasoning in Aerial Image Understanding.
Technical Report, Center for Automation Research, University of Maryland, October, 1983.

- [6] Rosenthal, David.
An Inquiry Driven Vision System Based on Visual and Conceptual Hierarchies.
UMI Research Press, 1981.
- [7] Talmy, Leonard.
How Language Structures Space.
Technical Report 4, Berkeley Cognitive Science Report, January, 1983.
- [8] Tropf and Walters.
An ATN for 3-D Recognition of Solids in Single Images.
In *Proceedings of the 8th International Joint Conference on Artificial Intelligence.* 1983.
- [9] Winograd, Terry.
Language as a Cognitive Process.
Addison-Wesley Publishing Co., 1983.
- [10] Zwarico, Amy.
The Recognition and Representation of 3D Images for a Natural Language Driven Scene Analyzer.
Technical Report, University of Pennsylvania, 1984.

4. IPON - Advanced Architectural Framework for Image Processing

This section outlines the organization and implementation of IPON in terms of both the hardware and programming environment, the progress to date, and our future plans for this research effort. Additional details can be found in [4] and [5].

4.1. Introduction

One fundamental computational problem with image processing is the time needed to execute typical algorithms. This is especially severe with the types of image processing required for interactive image understanding applications. These algorithms deal with extraordinarily large quantities of data. A typical two dimensional image (512 x 512) consists of approximately a quarter megabyte of data. Voxel (3D) and time sequenced images consist of much greater amounts of data. Even the most powerful contemporary processors become ineffective when presented with such quantities of data. Many related applications such as a mobile robot trying to avoid obstacles as it moves require real-time processing capability (one image every thirtieth of a second). The use of ACTIVE SENSORS further increases this computational load since processing may need to be performed quickly at several different levels of detail or on slightly different data.

The objective of the IPON (Image Processing Optical Network) project was to investigate possible solutions to these problems. An architectural framework is evolving from this effort which is usable on current computation systems and will be directly applicable to emerging advanced technology as it becomes available in the future.

The realization of real time image processing has long been a goal of many researchers in computer architecture. Towards this end many different architectures have been developed. The applicability of MIMD, SIMD, pipelined and data flow processors have been investigated [3] and each found to have the following types of problems:

1. Lack of flexibility (Pipelined and SIMD processors).
2. Complex awkward programming (MIMD).
3. Implementation Difficulties (MIMD and Data Flow).
4. Limited areas of efficient application (SIMD, Systolic array).

Image processing represents one of a class of computation applications which requires the manipulation of extremely large datasets. Traditional computer architecture including Von Neumann (SISD) machines as well as pipeline or systolic arrays, SIMD, and MIMD networks falls far short of the performance required for the real-time needs of machine perception, image analysis, certain types of image related computer graphics, object tracking, etc. Inherent in these approaches are bottlenecks associated with network communications and data storage.

4.2. Overview of IPON

The Image Processing Optical Network represents an architectural framework consisting of two major parts: the IPON hardware configuration and optical interconnection network and the integrated IPON software environment.

IPON is a computer system built around an optical interconnection network. Optical interconnection networks such as the one which we are designing provide solutions to many of the problems associated with the use of traditional electronic networks. Communicating through this network are a number (< 1000) of heterogeneous processors which need not be 'silicon' based.

The IPON programming environment facilitates the development and debugging of parallel image processing algorithms. The hardware and the software of IPON have been designed in such a way that programs written using the IPON program development system can be efficiently executed on the IPON hardware as well as on other multiprocessors or conventional superminicomputers.

It was essential to develop a system that is easy to program and debug while still providing parallel execution for increased throughput. The IPON hardware configuration represents a machine on which actual image processing algorithms will be implemented and used by vision and robotics researchers. Towards this end, IPON embodies the following, which make it a powerful system for developing real time image processing algorithms. IPON is a system of hardware built around an optical network which is:

1. Completely connected
2. Non-blocking
3. High speed
4. Dynamically reconfigurable
5. Expandable at a linear cost

These characteristics:

- Allow for maximum utilization of any number of ultra-high performance heterogeneous processors which can be easily integrated into the IPON system.
- Reduce the concern over the time taken to transmit data from one processor to another. This can reduce the difficulty of task scheduling since the transmission of data is not as costly as it is in traditional MIMD systems.
- Allow for the use of distributed control flow as opposed to a centralized token matcher or task dispatcher.
- Make IPON expandable. The network complexity increases linearly with the number of processors, not at the rate of n -squared. Algorithms written for a given machine configuration do not need to be rewritten when the machine is expanded.

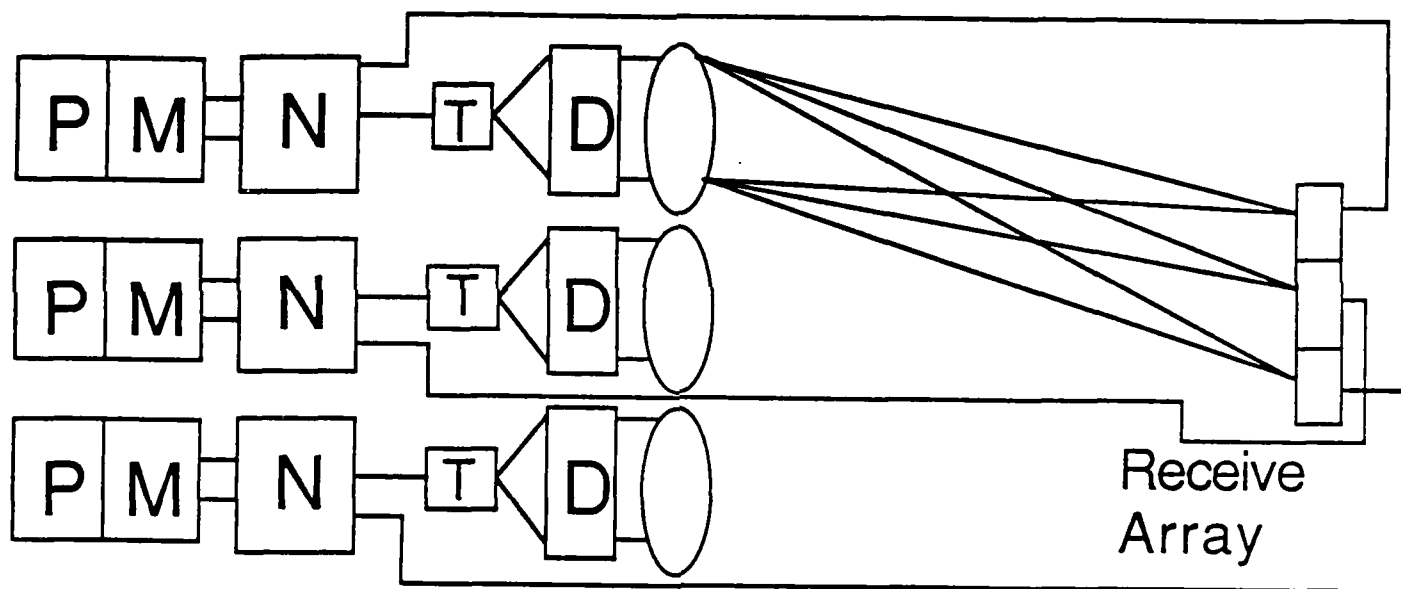
IPON's programming environment is based on process level data flow which:

- Gives rise to modular programs which can be used as building blocks for more advanced algorithms.
- Reduces any possible communication bottleneck due to the fact that data is only transmitted at the completion of a process as opposed to the completion of an instruction.
- Allows one to exploit inherent parallelism amongst processes.
- The data flow execution paradigm is enforced only upon the processes themselves.

Internally, the process can use any other appropriate flow of control paradigm to efficiently execute the algorithm.

- IPON is programmed in a graphical, hierarchical programming language which eases the development problem associated with parallel algorithms.

The optical network, which allows any processor to communicate with any other processor and allows any number of such conversations to take place simultaneously, is diagrammed in (Figure 3). The network consists of n optical transmitters (laser diodes), n acousto-optic deflectors (AOD, Bragg cells) and n photo sensitive receivers (photodiodes). Each processor is attached to one or more transmitters and receivers. The AOD devices serve as beam steerers; they deflect an incoming laser beam at an angle proportional to the frequency applied to the device. For applications where high speed dynamic reconfigurability is not required, low cost mirror based deflection systems based on galvanometers, servomotors, or piezoelectric devices can be used.



P - Processors, M - Memory, N - Network Interface Module
T - Optical Transmitter, D - Deflector (AOD or Mirror)

Figure 4-1: Optical Network

Connected to this network are a number of homogeneous processors. These processors need not be typical digital processors; indeed one of the motivations behind the development of IPON was to allow integration of non-traditional image processing devices into a more traditional (in terms of programming and use) image processing system. The reason for this is the fact that digital computers are not always the ideal devices for doing image processing. Alternative image processing devices include coherent and non-coherent optical devices [6] that enable the

computation of complex functions such as the Fourier transform to proceed at the speed of light. Hybrid analog-digital systems [2] have also been developed that perform many image processing functions which, if performed using purely digital techniques, would require orders of magnitude more hardware to produce the same result in the same amount of time. More traditional machines capable of increased throughput, such as SIMD computers, can also be integrated into the IPON system. While many of these approaches are at the present time extremely primitive, the important point is that they can be easily integrated into IPON as the technology matures.

IPON programs are written in a *graphical* data flow language. The language is also hierarchical, allowing the programmer to view a program at any level of detail he desires. We are choosing to use a graphical language in the hopes that a graphical representation of an algorithm consisting of a number of cooperating parallel processes will be easier to understand, hence easier to construct and debug. It is interesting to note that in most texts describing parallel systems, the system is first represented graphically and then it is shown how to convert this graph to a one dimensional representation, i.e., a program written in a language that supports parallel flow of control operators such as fork and join [1]. While this program retains the same semantics of the original graph, it is no longer as easy to visualize just what function it performs. We feel that it is this *linearizing* of parallel programs, which makes writing and understanding such programs the difficult task that it is today. IPON attempts to reduce this difficulty.

4.3. Current Status of IPON

Substantial progress has been made in the time since the IPON project was initiated. Some of the accomplishments of the first phases of the IPON effort are listed below:

- Architectural design of IPON.
- Functional emulation of IPON structure.
- Preliminary graphical programming interface.
- Initial investigation of optical network implementation.
- Determination of requirements for distributed control.
- Organization of optical data link interface processor.

Note that most of these areas of research are quite general in nature. Thus, although our immediate objectives relate to IPON, the results obtained with these investigations are applicable to other multiprocessor and dataflow systems - especially in the areas of optimal resource allocation and scheduling on MIMD and dataflow systems.

We have investigated the following aspects of IPON:

- Implementation of prototype optical network.
- Optimal network control and task allocation.
- Use of shared high capacity storage.
- Performance evaluation and optimization.

- Graphical programming system development.
- Hierarchical image database management.
- Integration of special purpose or hybrid processors into IPON.

Current work is centered around the development and analysis of the optical network. We have constructed a small prototype of the hardware and have evaluated the resultant network in terms of speed, reliability, and cost. Furthermore, we have developed the necessary control algorithms through which the processors will interface to the network.

The simulator allowed us to investigate various network control and task allocation strategies and determine their effect on overall performance. Once the optimum strategies have been determined we plan to implement them on a network of VAXes and measure the real world performance of such a system. This network of VAXes will initially be connected through the use of a high speed Ethernet, but as development proceeds on the hardware for the optical network the Ethernet will be phased out.

One of IPON's features is the use of heterogeneous processors, each tailored for efficient execution of certain image processing tasks. These processors are interconnected in such a manner that if a portion of a given image processing algorithm can be executed in an extremely efficient manner on a certain processor, then an attempt should be made to execute that task on that processor. Several problems arise when attempting to perform this sort of optimization. One problem is that of measuring what the performance of a processor is when presented with a specific task. The performance of a processor depends on many factors and what is needed is a way of expressing these factors in such a fashion that a task allocator can rapidly determine how well a processor can perform a given task. Another problem concerns the task allocator itself. Even if a processor's performance can be ascertained, the task allocation problem remains a NP-complete problem and heuristics must be used to reduce the time taken to determine task to processor allocation. An algorithm to perform such allocation has been developed but experiments need to be performed to determine its effectiveness.

Development of IPON's programming system is proceeding concurrently with the development of the hardware. The graphical programming language is being expanded to provide a complete set of programming language constructs. The expanded language will allow for the expression of highly parallel image processing algorithms in a manner comprehensible to the programmer. In addition to expanding the language, work is needed in the area of the user interface. This includes determining the most effective manner of interactively manipulating graphical symbols and presenting these symbols in a form which is understandable to the programmer.

Hierarchical access to multi-spectral image data at variable resolution, size and resolution is a characteristic of many complex image processing algorithms. IPON will support such access through the use of generic image processing tasks. A generic task will be able to process any size or resolution image. To accomplish this, image access will be provided in terms of an arbitrary number of rectangular sub-images or *segments* which may be configured with respect to one another without altering the actual image data. Images can then be treated as a list of Segment Descriptor Blocks (SDBs) through which image processing tasks access the actual

data. Using SDBs, a given image processing task can be written in such a way that it can process a large variety of image formats without need for modification. Research into the question of how to efficiently interpret the SDBs in the IPON environment is to be conducted.

4.4. Conclusions

IPON is meant to be both a tool to design image processing algorithms and a system which can execute these algorithms in real time. We are taking the approach that there exist machines that offer efficient solutions to certain image processing tasks and what is needed is a way to easily and coherently integrate these machines so that they can work together to efficiently execute complex image processing algorithms. Another function of IPON is to demonstrate that digital electronics is not the only way to implement image processing algorithms. The system is to allow experimentation with hybrid digital, analog and optical image processing techniques to determine the advantages and disadvantages associated with such an approach. It is through the use of an *ideal* network, that a system providing the desired capabilities of IPON is possible.

Initial results, both in the design of the software and the design of the network, encourage us to believe that IPON is a viable concept.

Development of the concepts for IPON are finished and implementation and evaluation are nearing completion.

4.5. Image Processing Bibliography

- [1] Dennis, J.B. and Horn, E.C. Van,
Programming Semantics for Multiprogrammed Computations.
Communications of the ACM Vol. 9(No. 3):pp. 143-155, October, 1976.
- [2] Dood, G. and L. Rossol, ed.,
Computer Vision and Sensor Based Robots.
Plenum Press, New York, 1979.
- [3] Etchells, D.
A Study of Parallel Architectures for Image Understanding Algorithms.
Image Understanding Research Report Vol. 104:pp. 133-176, October 19, 1983.
University of Southern California.
- [4] Goldwasser, S.
The Image Processing Optical Network: Advanced Architecture for Image PRocessing.
Computer Science Grasplab report, University of Pennsylvania, July, 1984.
- [5] Smitley, D., Goldwasser, S., and Lee, I.
IPON - Advanced Architecture for Image Processing.
In *12th International Symposium on Computer Architecture*. Boston, MA, June, 1985.
- [6] Stark, H., ed.,
Applications of Optical Fourier Transforms.
Academic Press, New York, 1982.

END

DATE
FILMED

DEC.

1987