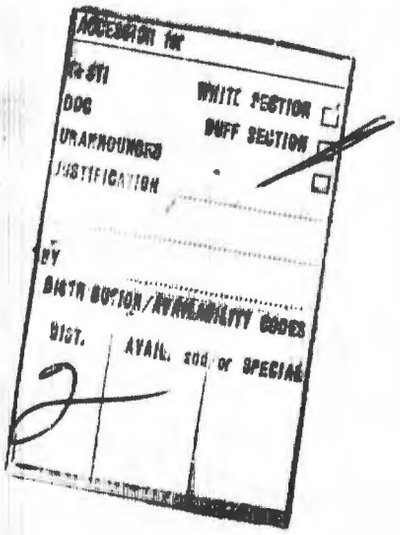
RADC-TR-68-605
Final Technical Report
February 1969

APPLICATION OF INTELLIGENT AUTOMATA TO RECONNAISSANCE

Nils J. Nilsson
Charles A. Rosen
Bertram Raphael
et al
Stanford Research Institute

Rome Air Development Center
Air Force Systems Command
Griffiss Air Force Base, New York

When US Government drawings, specifications, or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded, by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.



Do not return this copy. Retain or destroy.

APPLICATION OF INTELLIGENT AUTOMATA TO RECONNAISSANCE

Nils J. Nilsson
Charles A. Rosen
Bertram Raphael
et al
Stanford Research Institute

AFLC, GAFB, N.Y., 19 Mar 69-118

FOREWORD

This final report was prepared by the following personnel of Stanford Research Institute, Menlo Park, California, under contract AF 30(602)-4147, Project 4594, Task 459405; contractor number is SRI Project 5953:

Nils J. Nilsson	Leonard J. Chaitin
Charles A. Rosen	Sven E. Wahlstrom
Bertram Raphael	George E. Forsen

The RADC Project Engineer was Miss Patricia M. Langendorf (EMIRD).

This report has been reviewed and is approved.

Approved:
PATRICIA M. LANGENDORF
Project Engineer

Approved:
A. E. STOLL, Colonel, USAF
Chief, Intel & Info Processing Division

FOR THE COMMANDER:
IRVING J. GABELMAN
Chief, Advanced Studies Group

ABSTRACT

This report is a final report for the project "Applications of Intelligent Automata to Reconnaissance," under Contract AF 30(602)-4147. The primary goal of this project is to investigate techniques in artificial intelligence applied to the control of a mobile automaton in a real environment. The main emphasis is on the design of a hierarchy of computer programs that will accept visual and other sensory information gathered by the automaton and will direct the actions of the automaton in performing missions that require the abilities to plan ahead and to learn from previous experience.

A mobile automaton controlled over a radio link from an SDS-940 computer has been constructed and is operational. It uses a visual system and other sensors combined with a complex problem-solving system to enable it to navigate among simple objects in a laboratory.

Although this is a final report, another project (under Contract F 30602-69-C-0056) is using the results reported here to continue research in intelligent automata.

CONTENTS

ILLUSTRATIONS

BLANK PAGE

I INTRODUCTION

The primary goal of this project is to investigate techniques in artificial intelligence applied to the control of a mobile automaton in a realistic environment. The main emphasis is on the design of a hierarchy of computer programs that will accept visual and other sensory information gathered by the automaton and will direct the actions of the automaton in performing missions that require the abilities to plan ahead and to learn from previous experience.

The project began in March 1966 and since that time four Interim Reports[1,2,3,4]* have been written, and a number of papers concerning this work[5,6,7,8,9,10] have been presented. This report will summarize the work completed to date, referring to relevant sections of the previous reports for background.

First we will give a brief description of the automaton's capabilities. An experimental mobile vehicle controlled via radio and video links by an SDS 940 computer is operational. The main features of the vehicle are shown in Fig. 1. Essentially, the automaton can move around a large laboratory room containing a set of carefully chosen movable objects. In Fig. 2 we show a time-lapse photograph of the automaton in various positions among the objects in its environment. It can examine and extract information from its environment with touch, visual, and range-finding sensors. Such information forms the basis of constructing several abstract models of "its world" in its computer memory. By use of sensory and model information it can autonomously plan and execute a class of simple tasks related to exploring its environment. These tasks can be given to the automaton computer via teletype in the form of simple English sentences. Available software packages that are operational and integrated into the system include:

---

*References are listed at the end of the report.

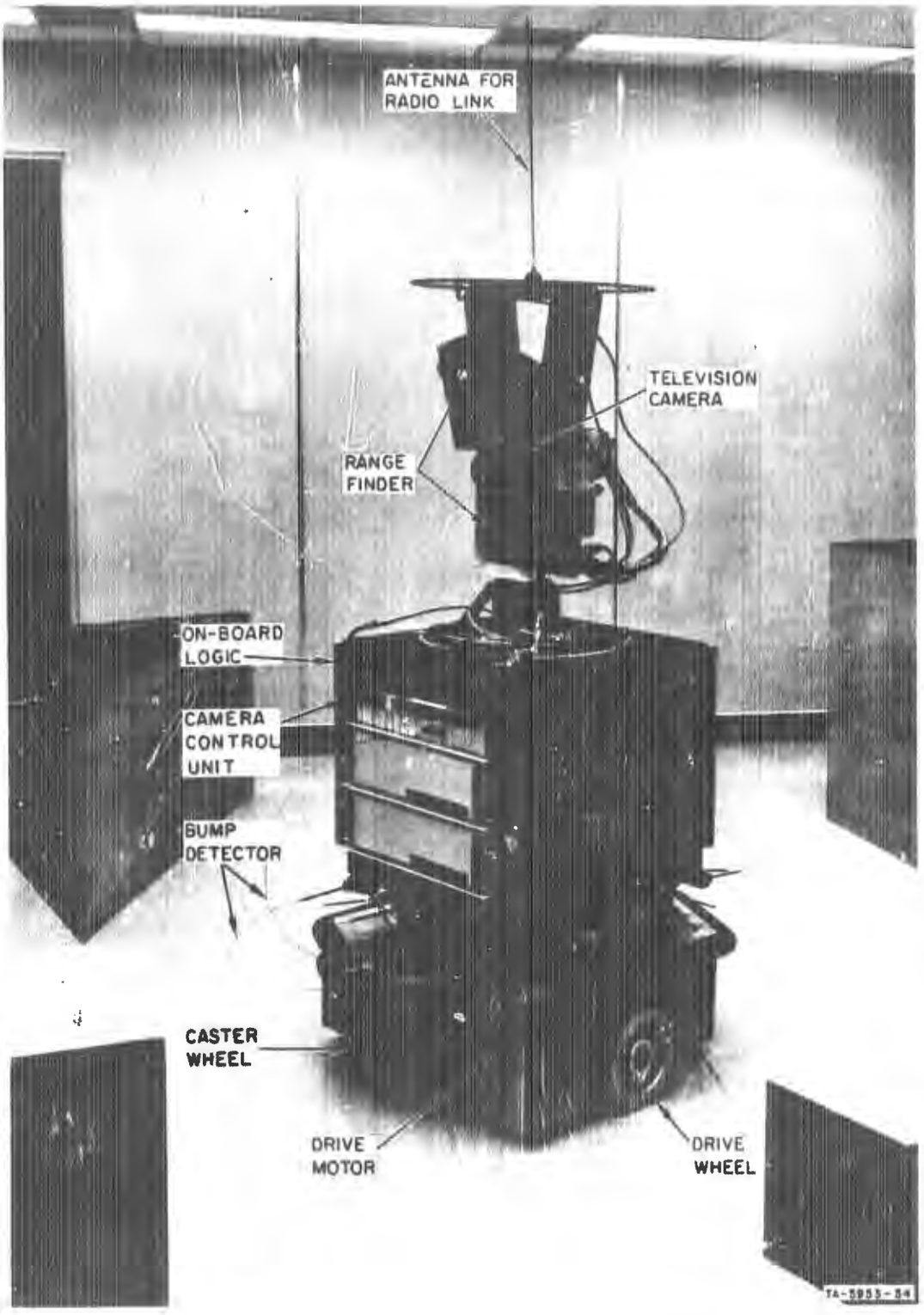


Figure 1. Automaton Vehicle

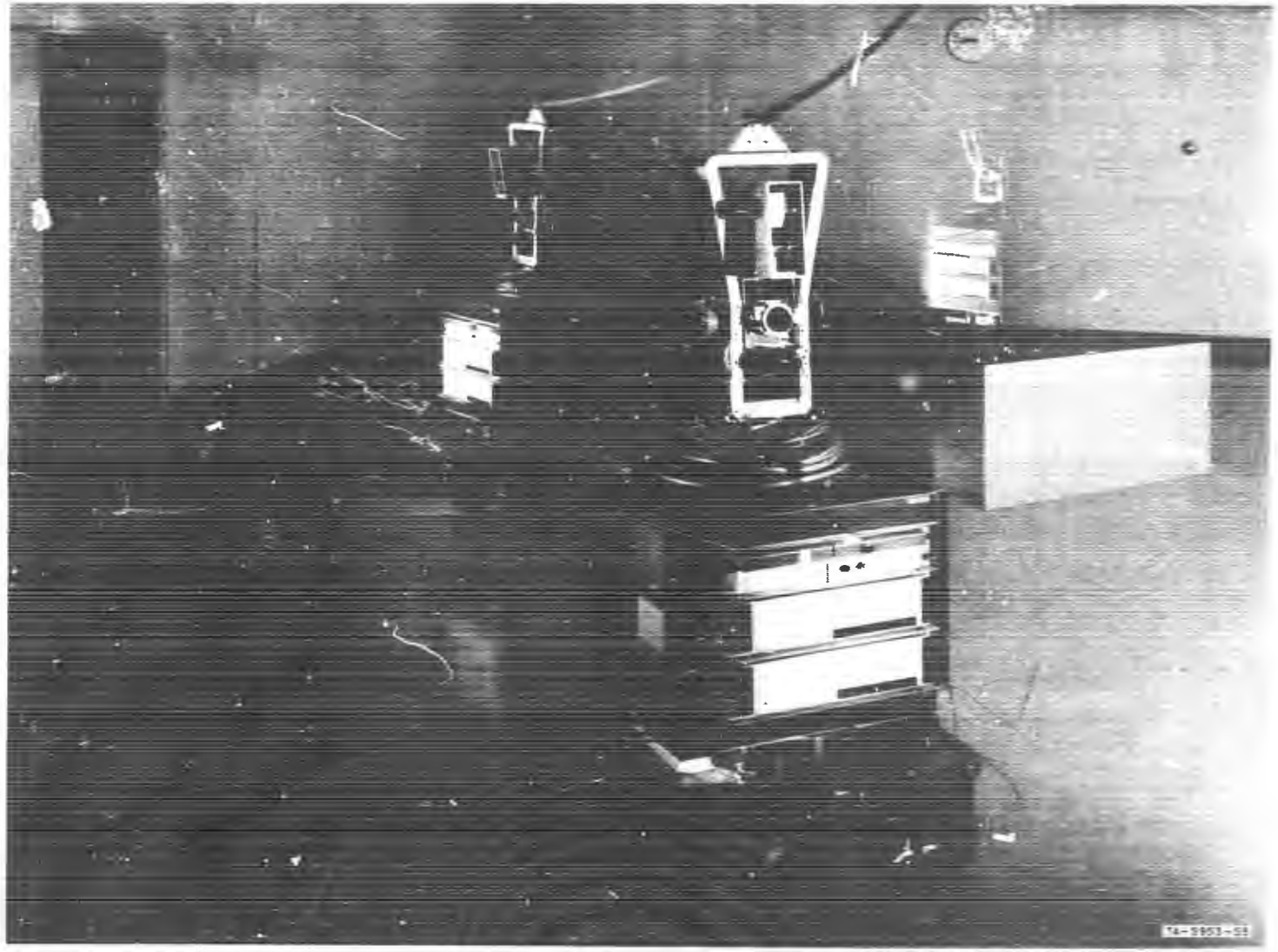


Figure 2. Automaton Vehicle in its Environment

(1) Low-level operating programs controlling the various automaton motor functions.

(2) Several tactical (planning) programs that direct the computation of efficient routes in cluttered environments. These include algorithms implementing formal and heuristic methods.

(3) Model-building programs that effect storage of pertinent information in two models--a Cartesian grid model and a property-list representation linked to the grid model.

(4) A question-answering program in which a body of facts stored in the representation, as well as other general facts entered directly by the experimenter, can be queried and logical implications obtained for use in planning routes, answering questions, etc.

(5) A natural language program that accepts simple English sentences (restricted to a small subset of English), performs syntactic and semantic analysis, and extracts and translates the meaning unambiguously into predicate calculus statements appropriate for the requirements of the question-answering program above.

(6) Picture processing and pattern-recognition programs. Video information from the TV camera is processed to obtain shape information (outlines) of objects in the environment as well as floor boundaries. Pattern-recognition programs are being developed using this information, and ultimately objects in the environment will be recognized and entered into the models. At present, information on gross shape and floor boundary is entered into the grid model, and the rough sizes of objects and the Cartesian coordinates of the centers of gravity are entered into the property list model.

The operation of the system can perhaps best be illustrated by describing what happens when a typical task is given the automaton, such as:

"Go carefully to a large box."

Let us assume that in some previous run a number of the objects in the environment, including one or more large boxes, have been discovered by tactile range-finder, and TV sensors and entered into the grid model; some properties of the objects, such as gross size and location have been entered into the property-list model together with the names of such objects supplied by the experimenter.

The English sentence is analyzed and its meaning made clear; it is first transformed into a statement in the first-order predicate calculus. In effect, a question is asked: "Does there exist an object "X" that is both a box and is large?" Using theorem-proving methods (in the question-answering program) operating on previously stored information a search is made, and if an affirmative answer is found, the location of the box is made the explicit goal location. The word "carefully" is interpreted to mean that a plan must be made to avoid bumping into known objects on the route to the goal.

The Automaton then "looks" in the direction of the now-known location of the goal, using video and range-finding sensors. The grid model is updated if new information about previously unknown obstacles is obtained. A route is planned, which will skirt the obstacles and will terminate in some clear space next to the designated box. The Automaton will then attempt to execute this plan by calling on low-level motor-running programs. Should its representations or models have been incomplete or imperfect and the Automaton bumps into something, a new procedure is initiated. Firstly, it makes use of an edge-following algorithm, using its tactile sensors, attempting to go around the obstacle until it is clear, and then it initiates the same sequence of steps as before--namely, it will take a new "look," plan a new route, and then attempt to execute its new plan. All new information will be added to its models, such as that acquired by bumping into an obstacle

and going around it, as well as the results of the second visual sweep. Thus, with increasing experience, the condition of knowledge about its world improves, requiring less sensory interaction and allowing more reliable planning to perform a given task.

A number of different experimental tasks have been performed similar to those described above. These experiments have served as a basis for planning additional capabilities and more sophisticated behavior in relation to the automaton's interaction with its environment. This project is being continued under Contract F 30602-69-C-0056, in which the prime research effort will be devoted to:

(1) Planning and problem-solving techniques that will rely heavily on applying formal theorem-proving methods to automaton problems.

(2) Question-answering research with special attention to the problem of representation and the organization of memory structure appropriate for storage and retrieval of a large body of facts and consequences of these facts.

(3) Improvements and extensions to natural language capabilities.

(4) Intensive effort will be made in improving the sensory capabilities, especially in the field of visual data processing. The goal is to be able to store a visual scene in terms of recognized objects and the relations between them.

In the remainder of this report we will discuss in detail the operation of the various subsystems that together enable the automaton to execute tasks similar to that described above.

II SOFTWARE SYSTEM ORGANIZATION

A. Introduction

In previous reports we have described various plans and ideas for the components and organization of the automaton software. Some of these plans--e.g., the grid model (Third Interim Report, Sec. III)[3]--have been implemented precisely as described. Some--e.g., several hierarchical levels of software (Second Interim Report, Sec. III)[2]--have been useful guides for the implementation, although they do not constitute rigid design specifications. Others, such as the line model (Third Interim Report, Sec. III), have been tabled as premature because other components of the system (e.g., vision) have not yet advanced to the point where these features would be useful. And finally, some plans, such as the specification language (Third Interim Report, Sec. VII), have been evolving into a form quite different from the one previously presented.

In this section we shall describe the components and organization of the automaton software as it existed at the conclusion of work on this project. Although it does not contain all the elaborations suggested at various times in previous reports, this final system indeed demonstrates how a software system for an intelligent automaton can be constructed and how its components can coordinate their activities.

The software system contains the following elements:

(1) An extension to the time-sharing monitor system.

(2) A set of elementary program packages.

(3) The grid model.

(4) The property-list model.

(5) Visual-perception programs.

(6) The FORTRAN Executive.

(7) VALET, the system executive.

(8) ENGROB, the LISP executive.

In the following sections we shall briefly discuss each of these components, describe how these components are related in the overall system, and trace in detail the actions of the components as the automaton carries out a complete task.

B. System Components

1. Extension to the Monitor System

The vehicle and all its associated hardware is viewed as a set of input/output devices for the SDS-940 computer. In order to operate these devices without disturbing the normal operation of the computer, several new instructions had to be added to the time-sharing monitor system. Also, it is necessary for the automaton software to communicate with the monitor in ways not previously available.

A convenient way to add new features to an SDS-940 system is to define a new SYSPOP (system programmed operator). The use of this extension of the assembly language (ARPAS) of the machine causes an automatic linkage to special subroutines that carry out the desired operations. The SYSPOP RBT has been added to our system to operate the automaton and related equipment. Detailed specificiations for RBT appear in Appendix A.

2. Elementary Packages

The "immediate-action routines" that allow simple vehicle actions to be executed from FORTRAN function calls were previously described (Third Interim Report, Sec. VI). In practice, a programmer frequently does not wish to descend to that level of detail. Therefore, a set of somewhat higher-level program packages has been prepared, each identified with a two-letter command. For example, the MØ package turns on power to the wheels, waits for the "power on" signal, issues a "move" command, waits for the move to be completed, and turns off wheel power.

Besides physically operating the hardware, some two-letter command packages perform other functions such as arming and disarming various interrupt signals, operating the display, storing information into the models (see below), and initiating the execution of entire sequences of command packages. The complete list of two-letter commands is given in Appendix B.

3. The Grid Model

The grid model is the hierarchical set of 4 × 4 arrays corresponding to regions of the floor plan (Third Interim Report, Sec. III) In addition to this "map," we usually think of a certain additional set of facts as being part of this "model," since they are necessary to form a more complete "mental picture" of the environment. These facts include such items as the vehicle's position and orientation, a goal position for the current task, and temporary representations for newly discovered items that have not yet been added to the models in standard form. All this information is kept with the grid arrays in FORTRAN "common" storage and on a drum file for transfer between program segments.

4. The Property-List Model

The grid model provides useful information for arithmetic calculations concerning geometric properties of the automaton's environment. It is not useful for describing other kinds of properties (shape, color, etc.) of objects and higher-level relations among objects. For this kind of data we have created the property-list model. The basic elements of this model are words that name objects; these names are of the form ØBJ1, ØBJ2, ØBJ3, etc. (Each name is an atomic symbol of the LISP programming system.) Associated with each name is a "property-list" of attributes and corresponding values for the named object. The current attributes are X, Y, and SIZE, which give the position in the grid and the approximate diameter of the object. The object names also appear as constants in the theorem-prover's memory (see "LISP executive," Sec. II-B-8 below) where additional facts about the objects may be stored.

5. Visual Perception Programs

The two-letter command "PI" causes the visual system to read a TV picture, reduce it to a line drawing, and extract certain information ("floor boundary") from the picture for insertion into the grid model. ("PI" also updates the grid model and displays results on the CRT.) Details of how this visual processing takes place are given in Sec. III of this report.

6. The FORTRAN Executive

This executive is the highest-level program in the portion of the software system that is coded in the FORTRAN language. Its principal function is to accept two-letter commands, either from a controlling teletype or from a command file on the drum, and execute the appropriate program packages. It also is responsible for anticipating unusual responses from the vehicle--e.g., a "bump" signal during an attempted move operation--by arming appropriate interrupts; and for processing these interrupts if they occur, without disrupting the overall operation of the system. The ability to arm and service interrupts at the FORTRAN programming level is a unique feature of our implementation.

7. VALET, the System Executive

The Automaton software system is about six times as large as the largest program that can normally be run on an SDS-940 computer. Also, it contains portions that operate under FORTRAN and portions that operate under LISP, although FORTRAN and LISP are incompatible systems. In order to coordinate these diverse computing requirements we use a program, called the VALET, that is a miniature monitor. It is written in ARPAS (the 940 assembly language), and operates under DDT (the standard ARPAS run-time executive). The FORTRAN and LISP programs are divided into segments, each of which can fit into the computer as a "core image," or single complete 940 program, at one time. The VALET is responsible for switching the other program segments in and out of core memory (by using the drum for intermediate storage), as well as for arming and disarming various hardware interrupts (as appropriate for

the currently running segment). It knows which operating system (LISP or FORTRAN) each program must run under, and it can load any segment either as a fresh program or from temporary storage as a continuation of a previous run. Thus, entire core images can be run as if they are subroutines to other core images.

8. ENGROB, the LISP Executive

This executive is the highest-level program in the portion of the software system that is coded in the LISP language. It controls three major subprograms: the natural language interpreter, the theorem-prover, and the property-list-model maintenance program.

Natural Language Interpretation. This package, based on independent work by S. Coles[9] accepts simple English statements (or commands) about the automaton's environment, and translates them into unambiguous formal terms that may involve first-order predicate calculus, two-letter commands, or other operations in the software system.

Theorem Prover. This package, adapted from the QA2 program developed under another project[11,12] stores facts and deduces answers in a predicate calculus data base by using formal theorem-proving techniques.

Model Maintenance. The results of either an interpretation of an English statement or a response by the theorem-prover might be a request for data from the property-list model. The model maintenance program supplies such information as ENGROB may require from the property lists to compose a two-letter command. It also can initiate a request, though the VALET, for new data from the grid model that should be added to the property-list model, and then update the property lists when the new data arrive.

C. Relations Between System Components

The software currently in use can best be described by considering all of the programs in two different ways. Figure 3 shows the control interfaces. This is the way the physical system is put together. It

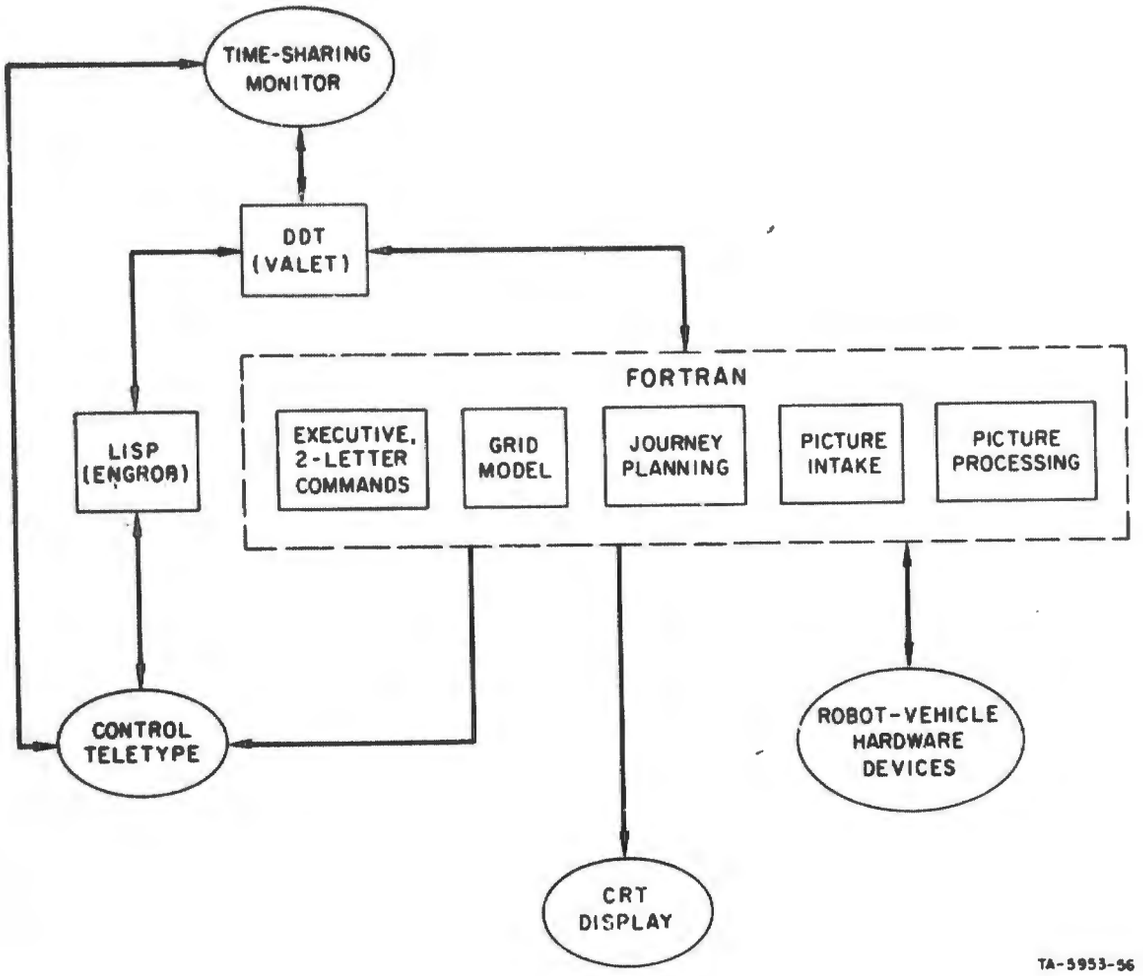


FIG. 3 SOFTWARE-CONTROL INTERFACES

shows the basic communication paths between LISP, FORTRAN, DDT, the controlling teletype, and the vehicle hardware.

Figure 4 shows the same set of programs. However, they are shown here as they are thought of in a logical sense--i.e., how they are organized to accomplish the tasks of the robot. This viewpoint makes the VALET invisible, and emphasizes the logical flow of control among the several FORTRAN packages. Figure 4 also shows in some detail how the vehicle devices are actually controlled. Certain two-letter commands from the FORTRAN executive call upon primitive subroutines (under DDT) that create separate programs, called "forks," that appear to operate in parallel with the main system of programs. These forks consist of

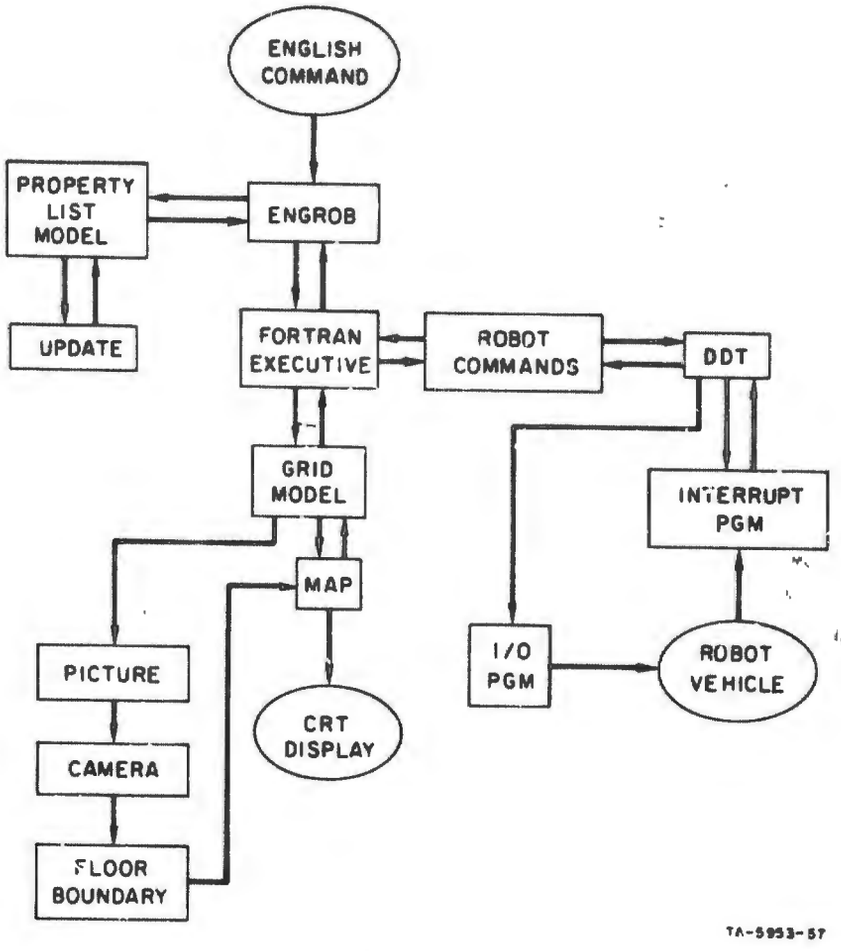


FIG. 4 SOFTWARE-SYSTEM LOGIC

"input/output programs" that do actual data transmission between the computer and the vehicle, and "interrupt programs" that receive and act upon various hardware-generated signals (such as bumps). These forks use the special RBT SYSPOPS.

D. Example

Suppose the robot's environment contains several large boxes and one wedge-shaped object (Fig. 5). Further, suppose that we initialize the system by building in knowledge of a preestablished coordinate system, but no information about the objects in the environment. We could then conduct the experiment given below, and observe the results described. (The conversation appears on the control teletype, with

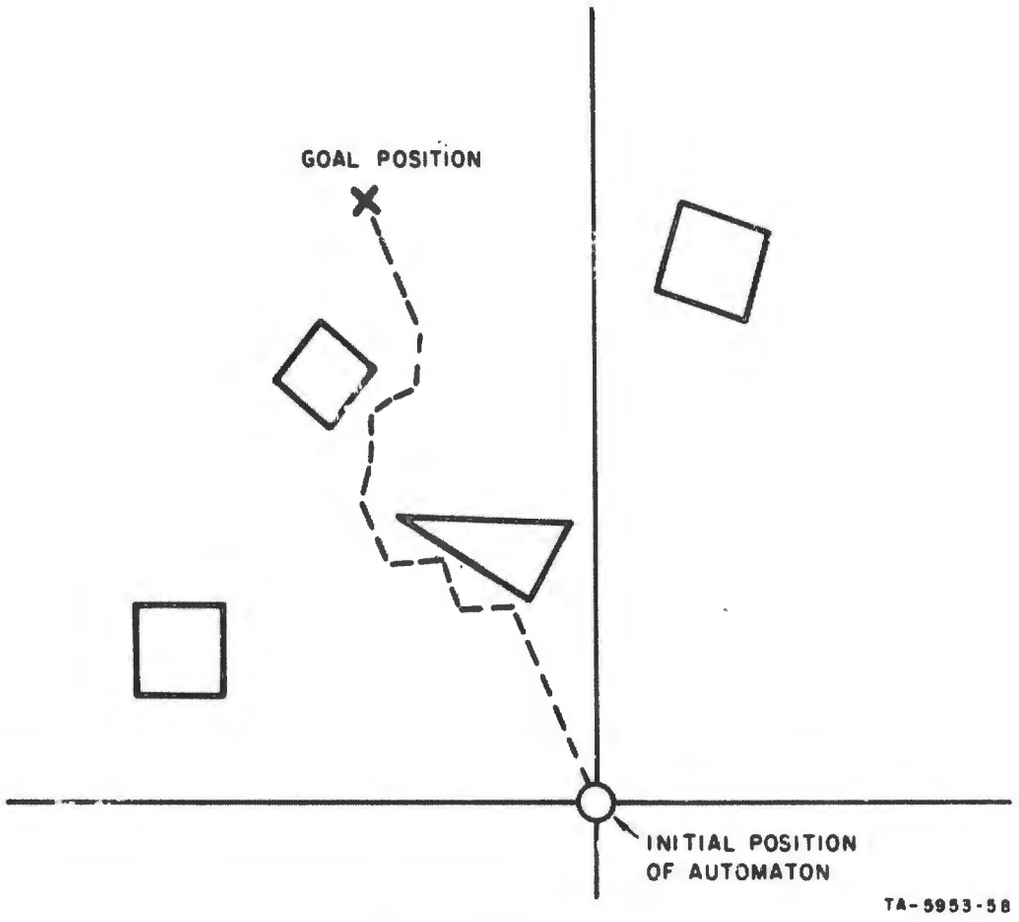


FIG. 5 SAMPLE JOURNEY

operator inputs indicated by underlining. Other observed effects are described parenthetically.) The experiment is as follows:

% <u>GO TO POSITION (12, -5)</u>.

OK

(After some delay, the vehicle turns to face floor coordinate position x = 12, y = -5, and starts moving toward it. It bumps into the wedge with the right front bumper, stops, backs up, turns left, moves a short distance, turns half-right, moves forward and bumps again. After a few repetitions, it finally gets clear of the wedge and continues toward its goal position. On the way it hits the corner of a box,

maneuvers around it, and eventually gets to its goal.)

DONE

% MAP THE ROOM.

OK

(A grid map of the environment appears on the display. It contains symbols indicating large unknown areas, a known-empty strip along the path just traversed, and several small known-occupied spots where bumps occurred. The display goes off when (CR) (carriage return) is typed.)

% WHAT ARE THE OBJECTS?

OBJ1 is at X = -2.5, Y = 5.1, SIZE = 2.1 FEET

OBJ2 is at X = -4.8, Y = 8.3, SIZE = 0.6 FEET

% OBJ1 IS A WEDGE.

OK

% OBJ 2 IS A BOX.

OK

% GO CAREFULLY TO A WEDGE.

OK

(The vehicle turns towards a spot next to the wedge, as the range-finder and television camera go on. Several stages of visual processing appear on the display, until each is terminated by (CR) . Finally the display shows a new floor plan, with much new area now indicated as known empty, and a broken line indicating a route to the wedge that would avoid bumping into the box. After the next (CR) , the vehicle turns and moves along its planned route until it stops next to the wedge.)

DONE

%

Now let us trace through this scenario again, except that this time we shall follow the activity of the internal program components instead of just the external behavior of the system.

ENGROB indicates it is ready for input by typing a percent sign. It then receives the first statement:

```
% GO TO POSITION (12, -5).
```

ENGROB calls the English interpretation program, which decides that the input calls for execution of the two-letter command GØ with 12, -5 as parameters. Thus it writes "GØ 12., -5.," on the command file on the drum, and returns to ENGROB, which types "OK" and calls the VALET to initiate the FORTRAN executive. This program reads the "GØ" command from the file, and starts the appropriate program package. This package blindly tries to get to the goal, taking appropriate evasive action when bump interrupts occur. The positions of the bumps, and the route followed by the vehicle, are placed in temporary storage associated with the grid model. When the interrupt programs indicate that a normal end-of-motion has occurred so that the goal has been reached, the "GØ" package types [DONE] and returns to the FORTRAN executive, which sees that the command file is empty and then calls the VALET, which causes ENGROB to be brought into core memory and continue running. ENGROB types [%] and waits for the next input. After MAP THE ROOM arrives, ENGROB calls the English interpreter, which decides that the two-letter command MA is called for and types [OK] . MA is written into the command file, and control passes through ENGROB, the VALET, the FORTRAN executive which reads the command, and finally to the MA package. This package needs grid-model maintenance programs that do not fit into memory along with the FORTRAN executive. Therefore the MA package writes the COMMON model onto a drum file, and signals the VALET to start up the FORTRAN grid-model package. This new core image reads the model from the drum, transfers the bump and route information from temporary storage into the grid, and displays the results. Upon receiving the (CR) signal, it writes the updated model on the drum and goes to the VALET. The VALET returns to the MA package, and the MA package reads the model from the drum and returns to the FORTRAN executive, then to the VALET, and finally to ENGROB which types [%] .

WHAT ARE THE OBJECTS? is interpreted by the English program to call for updating the property-list model, and control then transfers to the list-model maintenance program. The latter calls the VALET directly to get the FORTRAN grid-model core image (bypassing the two-letter command processing). A special FORTRAN program then scans the model for groups of known-full areas that are sufficiently close together to be considered a single object (e.g., if they are too close to each other for the vehicle to pass between them). For each such group, this program writes onto a response file on the drum three numbers: the x and y coordinates of the center of gravity of the group, and its maximum diameter. Then control passes back (through the VALET) to the LISP property-list program, which reads the response file, creates names for objects, stores the data on the property lists of the names, types out its information, and returns to ENGROB to type % .

The next two inputs

OBJ1 IS A WEDGE.

OBJ2 IS A BOX.

are recognized by the English interpreter to be new assertions. It translates them into predicate form

WEDGE (OBJ1)

BOX (OBJ2)

and passes them to the theorem prover to store in its memory before returning to ENGROB.

Finally, we get the command

GO CAREFULLY TO A WEDGE.

ENGROB gives this to its English interpretation program, which poses the logical question [EX(X) (WEDGE X)] to the theorem prover to identify a wedge. When the theorem prover replies "OBJ1," ENGROB asks model

maintenance to find the coordinates and diameter of OBJ1 from its property list. From this data ENGROB computes a goal position next to the WEDGE, puts the command "TE -5.6, 5.1," into the command file, and goes via the VALET and FORTRAN Executive to the TE program package.

TE is actually a "macro" command--i.e., it causes several other two-letter command packages to be executed in sequence. First it calls the PI package, with the goal position as its parameters. This package aims the vehicle and camera at the goal and takes a range reading. Then it moves control through the VALET to the picture-taking core image. The latter program segment displays each stage of processing as it reduces the picture to a line drawing. Then we again link through the VALET, this time to the "floor boundary" picture-processing segment, which extracts from the picture information about known-empty floor areas. Control then goes to the VALET, the PI package in the FORTRAN Executive, the VALET, and then the grid-model segment, where the floor boundary information is inserted into the grid. After control returns to the FORTRAN Executive segment, the PI package returns to the TE package which then calls the PL package. Optimum route planning, which is the job of PL, requires yet another FORTRAN core image, reached, of course, by means of the VALET. (The current grid model is transferred between these core-image segments, whenever necessary, by means of a drum file.) After generating a plan and placing it in the temporary-storage part of the model, the planning segment returns control to the TE package, which then executes the EX package to execute the plan. This package issues the move and turn commands that move the vehicle toward its goal as planned. The EX package also primes the system to take some appropriate action if unexpected bumps occur during the journey. Luckily, none do in our present example.

Eventually the vehicle reaches its goal and control passes from the EX package to the TE package, (which types DONE), the FORTRAN Executive, the VALET, and ENGROB, which types % and awaits the next command.

III VISUAL PROCESSING SYSTEM

A. Introduction

The visual sub-system is comprised of both hardware and digital computer software. The light sensor chosen is a standard vidicon closed-circuit TV camera. Special-purpose hardware has been constructed to interface the camera and the SDS 940 digital computer and to perform the first stages of data processing. A set of computer programs have been written to extract pertinent information from the digitized image. The current output of the visual subsystem is a list of connected line-segment end-points that constitute a line drawing of the open floor area. From this, the locations of observed objects can be obtained directly.

A ground rule of this project was to use available hardware and already published techniques for visual processing. Of the few efforts in perception of three-dimensional objects known at the time, the techniques of Roberts[13] appeared most applicable to our immediate problem. In the course of our work these methods have been somewhat modified to suit the constraints imposed by our particular system, but the current visual system for the Automaton largely reflects the sequence of processing first demonstrated by Roberts.

The initial environment of the Automaton was real, but contrived. It has been sufficiently simple to allow current visual capabilities to be useful to the Automaton, and sufficiently complex to indicate the weaknesses of current methods and to suggest areas of further research. Perhaps the most important result of our vision-research effort on the Automaton project is an appreciation of the potential complexity of the problem of vision when the real world is the subject matter, and a strong notion that the first step we have taken towards a general capability is very small indeed.

B. The Visual Environment

The current Automaton is restricted by its method of locomotion to move only on nearly flat surfaces. Initially its travel was limited by the length of cable connecting it and the computer. The addition of the radio links will allow the Automaton to travel further from the computer room, perhaps down hallways and into offices.

The first visual subsystem was designed to specialize in the planar-surfaced environment of our laboratory and office building. The objects in this environment are specially constructed rectangular parallelepipeds and wedges. The use of only the regularly spaced overhead florescent lights as well as light colored walls and floor allows us to essentially eliminate shadows and to limit the illumination to a 2-1/2: 1 range in the computer room.

The surfaces of the objects used are uniformly coated with red, grey, or white paint. Originally black was used to insure high contrast between adjacent surfaces. However the rangefinder relies on reflected light. Red replaced black because it is relatively dark to the TV camera and returns enough light to the rangefinder. Thus not only are the objects opaque, but also have non-specular surfaces. Furthermore no two-dimensional markings were put on the object surfaces. The floor tile was chosen so as not to have any detectable markings. The only two-dimensional marking purposely applied was a dark wall molding at the floor level. The floor has about the same reflectivity as the walls. There were verticle molding strips on one wall which were specular.

C. Functional Description of the Visual Subsystem

Figure 6 shows the current visual subsystem in block diagram form. The sequence of processing is as follows:

(1) Reflected light from the real environment is imaged on the sensor of the transducer (a vidicon TV camera). The sensor is electrically scanned to transform the image into a function of time.

(2) The resulting waveform is synchronously sampled and digitized.

(3) The resulting PCM data are processed (by pairs of lines) to "differentiate" the digitized image.

(4) The resulting differentiated image is stored in digital form in its entirety for subsequent processing.

(5) The differentiated image is then transformed into an array of features. These features are short line segments whose directions are transverse to the gradients in the original picture.

(6) Neighboring and nearly parallel short line segments are grouped together.

(7) The resulting lists of grouped features are then processed to form relatively long (possibly segmented) lines.

(8) These longer lines are then used to obtain a boundary of the navigable region of the floor. A simple geometric transformation relates image coordinates to floor coordinates.

In Fig. 7 we show a sequence of photographs illustrating these steps. Figure 7(a) is a sampled and digitized image of a wedge-shaped object. (120 × 120 lines, 4 bits gray scale.) Figure 7(b) shows the image resulting after applying the "differentiation" operator. The feature-masking operation then extracts the short line segments illustrated in Fig. 7(c). These are then grouped and illustrated as shown in Fig. 7(d). Each group is then fitted with a long straight-line segment as shown in Fig. 7(e). It is from the image of long straight-line segments that the floor boundary shown in Fig. 7(f) is extracted. The floor boundary is then used to update the Automaton's map of the world. In Fig. 8 we show the same sequence of operations for a slightly more complex scene.

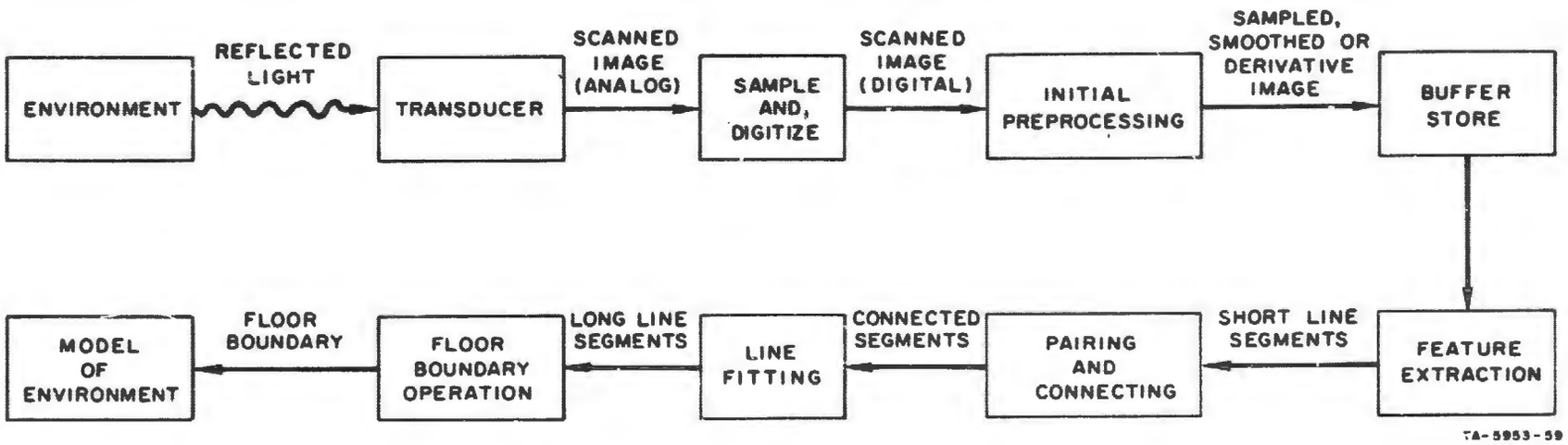
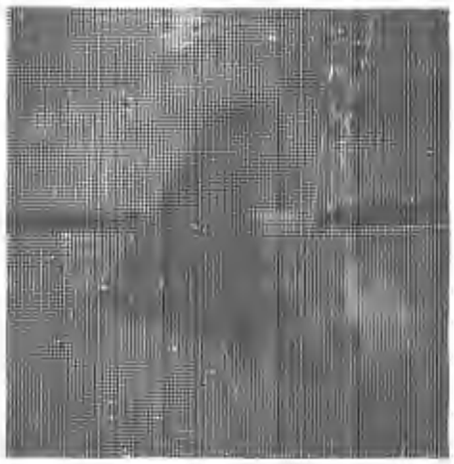
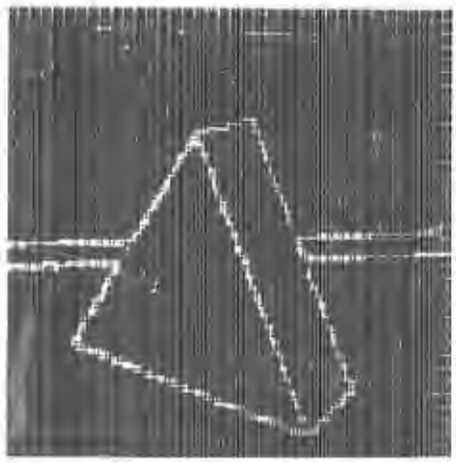


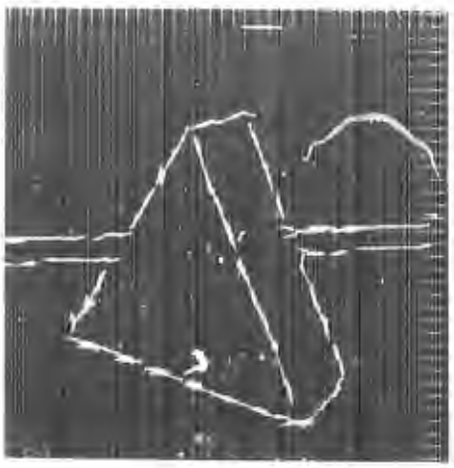
FIG. 6 FUNCTIONAL BLOCK DIAGRAM OF THE VISUAL SUBSYSTEM



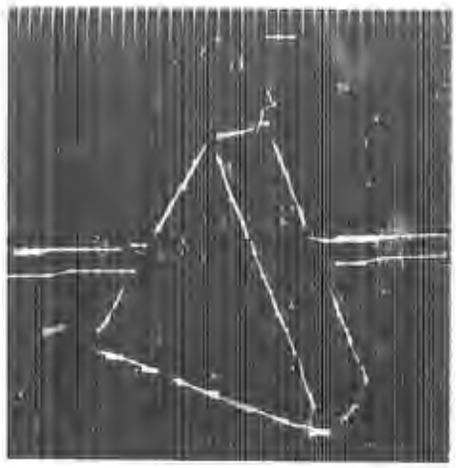
(a) Digitized Image



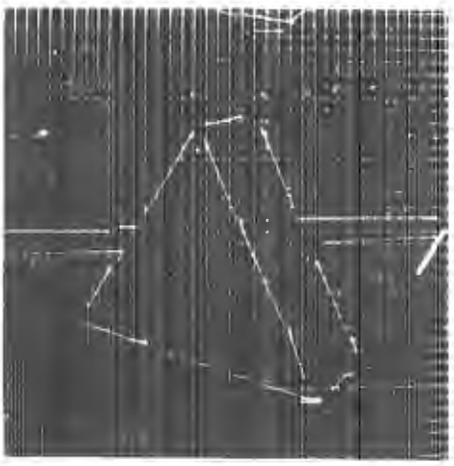
(b) Differentiated Image



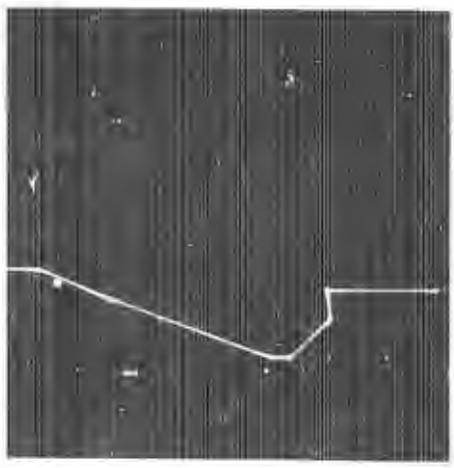
(c) Line-Segment Mask Responses



(d) Grouped Line Segments



(e) Long-Line Fits



(f) Floor Boundary

TA-5953-52

Figure 7. Example of Visual Processing Steps

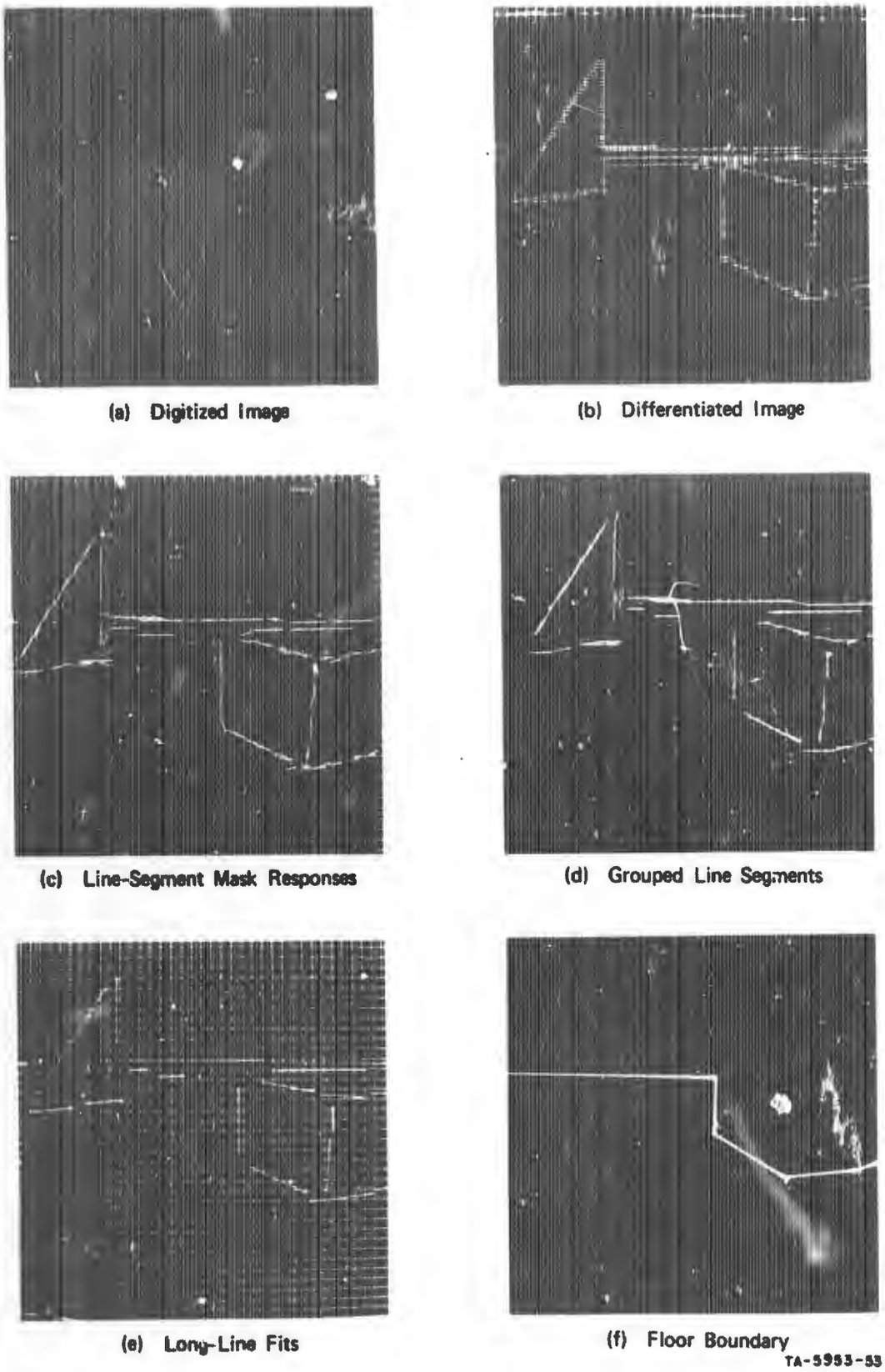


Figure 8. A Second Example of Visual-Processing Steps

From the line segments produced by the line-fitting routine, a program called FLOOR BOUNDARY forms an irregular polygon. The corners of the polygon are projected onto the floor and the resulting polygon is taken to be the known free floor area.

The following nomenclature is used to describe the operation of FLOOR BOUNDARY: A point chosen from the line segment end-points that is "confidently" on the floor boundary is called a "trace point." The line segment of which it is one end-point is called the "test segment." An "interior pointer" is a vector that points from the "trace point" into the interior of the floor boundary and is perpendicular to the test segment.

The program attempts to follow those line segments on the periphery of the free floor areas from the original "trace point" first to the left-hand edge of the picture and then to the right-hand edge of the picture. A box is drawn around the current "trace point," and segments with only one end-point within the box are considered candidates for immediate inclusion in the floor boundary outline. If no points are within the box, the closest segment that appears to be an extension of the test segment is considered. If no lines are found, a line is extended to the edge of the picture. The list of candidate line segments thus found are then treated as vectors pointing radially from the trace point. These are then reduced in number by keeping only those segments which, as vectors, form essentially a positive dot product with the test segment vector. The test segment vector points to the current trace point. The line segment among the reduced set with the smallest angle deviation from the "interior vector" is taken to be on the floor boundary. The segment thus chosen becomes the new "test segment." The "interior vector" becomes the appropriate normal to this line. The new "trace point" is the end-point of the new test segment which is farther from the old "trace point." The old trace-point and the end-point of the new test segment that is closer to the old trace point are averaged to have identical locations or extended to form a corner so as

to connect the old test segment to the new test segment and thus extend the floor boundary by one more line segment.

The algorithm depends mainly on the ability to find the base board in the picture to connect isolated objects. However, an object may lie entirely within the interior of the floor boundary first found. To take care of this circumstance the program replaces any line segments that have line segments below them.

After this procedure, the segments found are joined to the foreground and sides of the picture to form the irregular polygon.

A straightforward geometric transformation relates the boundary points thus found in the image plane in a one-to-one manner to the corresponding points on the floor plane. This simple procedure for obtaining depth from a single view assumes that the camera height above and the tilt angle with respect to the floor plane, and the acceptance angle of the TV camera lens/scanning system are known constants.

D. Problems in Automated Vision

There are three broad classes of problems in scene analysis. These are discussed in more detail in Appendix C. The first problem area is common to any three-dimensional environment and should be considered first. This area includes the "figure-ground problem," the "occlusion problem," and the "geometric invariance problem." The basic underlying problem in all these is the unresolved question, can the component parts of objects be identified as belonging to their respective whole self under a variety of arrangements and points of view in the environment?

The second problem area relates to upgrading the complexity of the environment. The human everyday environment has non-planar surfaces, considerable complexity, and much greater ranges of intensity, spectra, and detail than typified by our present laboratory environment.

The third problem area is the lack of special equipment that has the resolution, sensitivity, adaptability, size, and power consumption of, say, the human eye. Human vision involves a hierarchy of increasingly sophisticated processes. But it has the advantage of being able to

start processing data of higher quality and larger amounts than we can now obtain from electro-optic transducers that can be mounted on an Automaton. It is furthermore suspected that ultimately to increase the capabilities of mechanized vision to acceptable levels for useful applications, different computing structures will have to be utilized. It is anticipated that in addition to special preprocessing hardware, associative computers with content-addressed operations and multiple processors may be required, to enhance future performance.

The following processing techniques should be used, to aid mechanized vision:

(1) Directives from higher-level programs so that stored experience can be utilized;

(2) Partial sets of descriptors to make initial hypotheses for decisions, and use of these until further evidence refutes them;

(3) Basic shapes of areas and volumes in addition to points and lines;

(4) More descriptor types such as color, texture, depth, and relative velocity;

(5) Qualitative word descriptors for associating objects or parts of objects--e.g., "support," "next to," and "behind;"

(6) Multiple views from different points of view and with different optics and/or preprocessing.

This is but a partial list but it should be sufficient to suggest that there is much work that can still be done to improve mechanized vision.

BLANK PAGE

IV HARDWARE SYSTEMS

A. Hardware System Overview

The first and foremost goal for the hardware task was to develop equipment of a quality high enough to permit programming and software development to progress with a minimum of equipment breakdowns. The basic design concepts used in modern computer design, including usage of integrated circuits for all logic, were applied. This, plus careful selection of components and subsystems, contributed to achieving a high system reliability.

The second goal for the task, without infringing on the first goal, was to provide a sufficient variety of functions in the system so that rich, nontrivial experiments could be undertaken by the programming tasks. Some of the functions specified in the First Interim Report were later deleted or modified. It is our experience that these modifications, possibly with the exception of the deletion of the arms, have not changed the richness of experiments.

On the following pages is a description of the Automaton system hardware as it exists upon completion of the present project. Footnotes for some of the headings indicate where additional information can be obtained in previous reports.

Figure 9 shows a block diagram of the hardware system. The system consists of a stationary part interfacing with the SDS 940 computer and the mobile vehicle which is remotely controlled from the fixed equipment via a full duplex radio link.

Commands to the vehicle are transmitted in digital form preceded by a module address referring to the module on the vehicle that is expected to act. Each module is equipped with its own register. The register holds bits specifying information on desired direction of motion, speed, requested distance, and other special functions. When action is requested, the action starts and continues until completed or interrupted

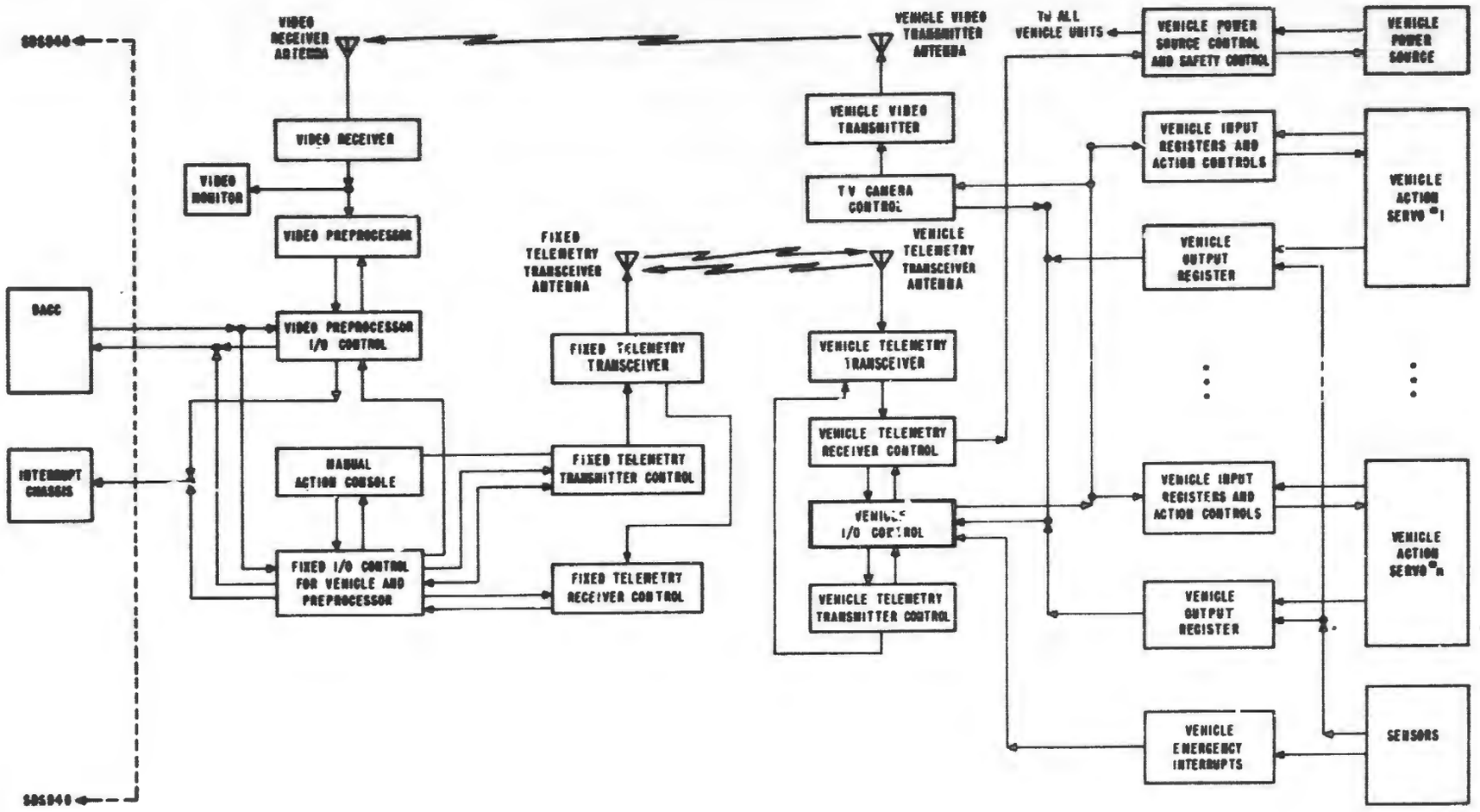


FIG. 9 AUTOMATON-SYSTEM BLOCK DIAGRAM

TC-5053-53

by other control functions in the system. End-of-action or other control interrupts are transmitted back to the stationary equipment in coded form, where they are decoded and sent as interrupts to the computer. Interrupts of a similar nature are ORed together to limit the number of interrupts. Status registers are therefore provided on the vehicle so that status can be interrogated from the computer any time the source of the interrupt is in question.

Special registers for the sensors, such as the range finder, bumpers, etc. are available and can be interrogated by a read operation in the same manner as reading from the module register.

The hardware for the visual system uses the same interface to the computer. The power for the TV camera and the special transmitter for the videodata is controlled from the power-control register on the vehicle. The rest of the visual system is quite independent.

B. Visual System Hardware

1. TV Vidicon Camera

a. Camera Mechanisms

The TV camera consists of one control unit mounted on the platform of the vehicle and one camera head mounted on a pedestal in the center of the vehicle. The camera can be turned ± 180 degrees around a vertical centerline, and it can be tilted + 60 degrees and -45 degrees around a horizontal axis located below and perpendicular to the optical axis of the camera. The camera is equipped with a manually replaceable lens. The lens mounts in a mechanism with two motors for control of iris and focus. The control of all degrees of freedom of the camera and its lens system is accomplished by stepping motors. The rotation of the camera around the vertical shaft is under control of a servo similar to that used for the wheels of the vehicle. The control from the computer is in the form of LEFT or RIGHT commands of a given number of steps. The camera has one left-rotational terminal switch at +180 degrees rotation and one right-rotational terminal switch at -180 degrees rotation. When these switches close, the rotation in the direction in process is interrupted. The switches also signal the emergency

circuit, causing an interrupt signal at the computer. Associated with the shaft rotation, there is also a pan distance counter. The content of the counter can be transmitted to the computer. The tilt of the camera is controlled by a stepping motor operated at a constant step rate. The motor reacts to a TILT UP or TILT DOWN command for a given number of steps. The tilt mechanism has limiting switches up and down. The limit switches stop the tilt and signal the interrupt circuits in the computer. The content of the tilt counter can be transmitted to the computer. A brake mechanism locks the camera in its tilt position when power is removed from the motor.

Only one lens is presently used. Focus is controlled by one stepping motors and iris by another. The rotation is limited by limit switches. The limit switches preset the counters at maximum focus and minimum iris associated with the stepping motors.

The control logic has an up-down counter for distance and direction.

b. Choice of the Vidicon*

We chose to use a vidicon TV camera as the light transducer, on the basis of performance, weight, power consumption, and availability as a commercial product at moderate cost. Five vidicon TV cameras were tested and evaluated for resolution, smearing, geometric distortion, field uniformity, sensitivity, noise, convenience, low-voltage dc conversion, serviceability, modular replacement, availability of technical assistance, and cost. A sixth, the Plumbicon camera, though recommended, was not available for demonstration. The highest-performance camera (RCA TK22) was ruled out as being too expensive, bulky, and without convenient dc conversion. It was tested mainly for performance comparison. Of the remainder, the Granger V-1000 came closest to meeting the performance of the RCA TK22 in resolution and field uniformity, and was superior to other cameras on most other tests.

---

*See also Appendix I of the First Interim Report.[1]

The other possible types of light transducers considered were (1) image dissector, (2) flying-spot scanner, (3) image orthicon, (4) solid-state vidicon, (5) parallel photocell arrays, and (6) a mechanical scan of a linear photocell array. These were ruled out for at least the following reasons: The image dissector would have been either too slow for the desired signal-to-noise ratio, resolution, and operation time with the chosen illumination, or too bulky to be mounted on the Automaton. The flying-spot scanner would require too much power and equipment on board the Automaton to produce the required light intensity. The image-orthicon would have been too expensive, bulky, and fragile. The solid-state vidicon was not yet sufficiently developed as a commercial product. Parallel photocell arrays would have been either too slow or too expensive depending on the device used, and would have required gain adjustments on each cell. A mechanical scan of a linear array would have lacked resolution in the direction of scan.

The vidicon image transducer is a parallel integrating device and thus, between readouts, tends to smooth the statistical fluctuation of light itself. Circuit noise is often the limiting factor on gray-scale resolution but spatial nonuniformity in the transducer surface can also contribute detectable noise. Signal-to-noise ratio is the limiting factor for determining a realistic number of quantization levels. The specified noise figures for the chosen camera (with an IEEE standard roll-off filter) will allow a twenty-to-one intensity range to be quantized into thirty-two levels, with a probability of 0.9 correct quantization (when the signal value equals the mean of adjacent quantization thresholds).

2. The Visual Preprocessor*

The main unit in the visual system is the fixed TV-preprocessor. There, the video signal from the TV camera on board is received and stored in a core memory after first going through a network for spatial differentiation. The differentiated picture is processed on commands from

---

*See also pp. 68-91 and Appendix J of the First Interim Report and pp. 34-35 of the Second Interim Report.

the computer. The processing consists essentially of matching subfields of the picture against a set of masks. Best match is registered and stored in another section of the external core memory. The result of the preprocessing can be transferred any time after the completion of the operation. Special interrupts are reserved for the preprocessor to indicate end of preprocessing, end of picture acquisition, and end of transfer of masks between external core memory and the scratch-pad memory. In this section we shall summarize some of the details of the preprocessor operation.

a. Analog-to-Digital Conversion

As discussed in Sec. III, the visual processing is based on a spatial "derivative" of the input picture.

The computation of this derivative along a given scan line requires temporary storage of the signal from the previous scan line. The scan line could be stored either in its original analog form or first converted to digital form and stored. In either method it was important to maintain the original quality of the video signal. Our preliminary investigation showed that analog storage of one scan line using delay lines and associated circuitry having bandwidths, signal-to-noise ratios, and delay stabilities comparable to the qualities of the chosen vidicon TV camera would have been just as expensive as a digital store of the same information at the quantization and sampling rates initially specified. The differentiation operation would not reduce the amount of data unless it were string coded, which itself would require considerable additional hardware. Thus it was decided to digitize the analog TV signal and do the scan-line storage and all the subsequent processing with digital hardware and computer techniques. This decision also reflects to a certain extent the experience and bias of the staff who would be entrusted with the job of realizing the interface between the TV camera and the computer.

It has been suggested above that thirty-two levels of quantization (five-bit PCM digitization) would be consistent with the specified signal-to-noise ratio of the chosen vidicon TV camera. These

figures were realistic for optimal camera operation and for using new vidicons, and with a low-pass filter with a slow roll-off characteristic starting about 4×10^6 Hz. The camera actually had a total bandwidth of 3×10^7 Hz, but much of the circuit noise was in the higher frequencies. Aperture-correction techniques were not employed, so as to retain the specified signal-to-noise ratio.

Horizontal resolution of a TV camera is related to its video bandwidth. Other limiting factors are scanning-beam spot size and local properties of the transducer. Assuming bandwidth to be the limiting factor and using the 4×10^6-Hz figure and a single RC-time-constant low-pass filter approximating that specified for the noise measurements, and assuming a step change in light intensity, the time it would require the camera video output to respond to one part in thirty-two away from its final value is about 130 nanoseconds (about 3-1/4 time constants). If the step change were from one extreme of light intensity to the other, the error at 130 nanoseconds would correspond to one quantization level.

It was decided to use a square format for sampling the TV image field. This is because normal operation of the Automaton would have the camera tilted down to view the foreground as well as objects further away. Also this removes the sampling region from the corner areas of the optical and electric images that have less sensitivity and resolution. Thus the 4:3 aspect ratio of the standard TV scanning raster was not used, and only the interior three-fourths of a given scan line is sampled. The sampling of one scan line requires about 40 microseconds. If the video were digitized every 130 nanoseconds, slightly less than 300 samples would be obtained for each scan line. For reasons of hardware compatibility it was decided to use 240 samples per scan line, thus requiring about a 160-nanosecond sampling period. About 240 scan lines cover one TV field in 1/60 second. Exact figures and more details can be found in the First Interim Report. It is thus felt that these conversion rates are a good match to the chosen TV camera.

No A/D convertor was commercially available that could digitize to five-bit accuracy in 160 nanoseconds. The usual serial

techniques would require 30-nanosecond decisions per bit, which at the time would have been pushing the state of the art.* The usual serial A/D convertor digitizes to more than five bits. It was decided that parallel conversion using thirty-two comparators and a 32-to-5 encoding logic could have the speed, stability, and accuracy desired for this application, and be built at reasonable cost. A temporary version together with word assembly and computer interface logic has been constructed mostly from stock items and has been operational for most of the project. It has proved useful for inputting digitized TV images directly into the core memory of SDS 940. However, this computer only allows a 13.7×10^6-bit-per-second transfer rate. The A/D convertor, TV camera system is capable of producing meaningful data at a 3×10^7-bit-per-second rate. To comply with the computer rate it was necessary to drop the lowest-order bit from each digitized sample and to skip every other sample. The hardware preprocessor, which has its own core memory, is designed to make use of the full capabilities of the A/D convertor design and includes essentially duplicate conversion hardware. The rest of the preprocessor is briefly described below.

b. Initial Preprocessing

The function of the preprocessor is to perform two transformations on the raw data. Figure 10 is a block diagram showing these basic functions. This section describes the first of these. Three spatial local operators--sampling, spatial smoothing or differentiating, and thresholding--transform the digitized samples into a set of picture elements that are then stored either for subsequent transfer to the SDS 940, or for the second stage of preprocessing described below. As mentioned above, the video waveform is always sampled and digitized to five bits 240 times per scan line. This first operation can be applied either to a full field of 240×240 5-bit digitized samples at one-quarter resolution (using alternate samples from alternate scan lines) or to a translatable quarter-field of 120×120 samples at full resolution,

*Commercially available products with adequate speed and resolution have since become available.

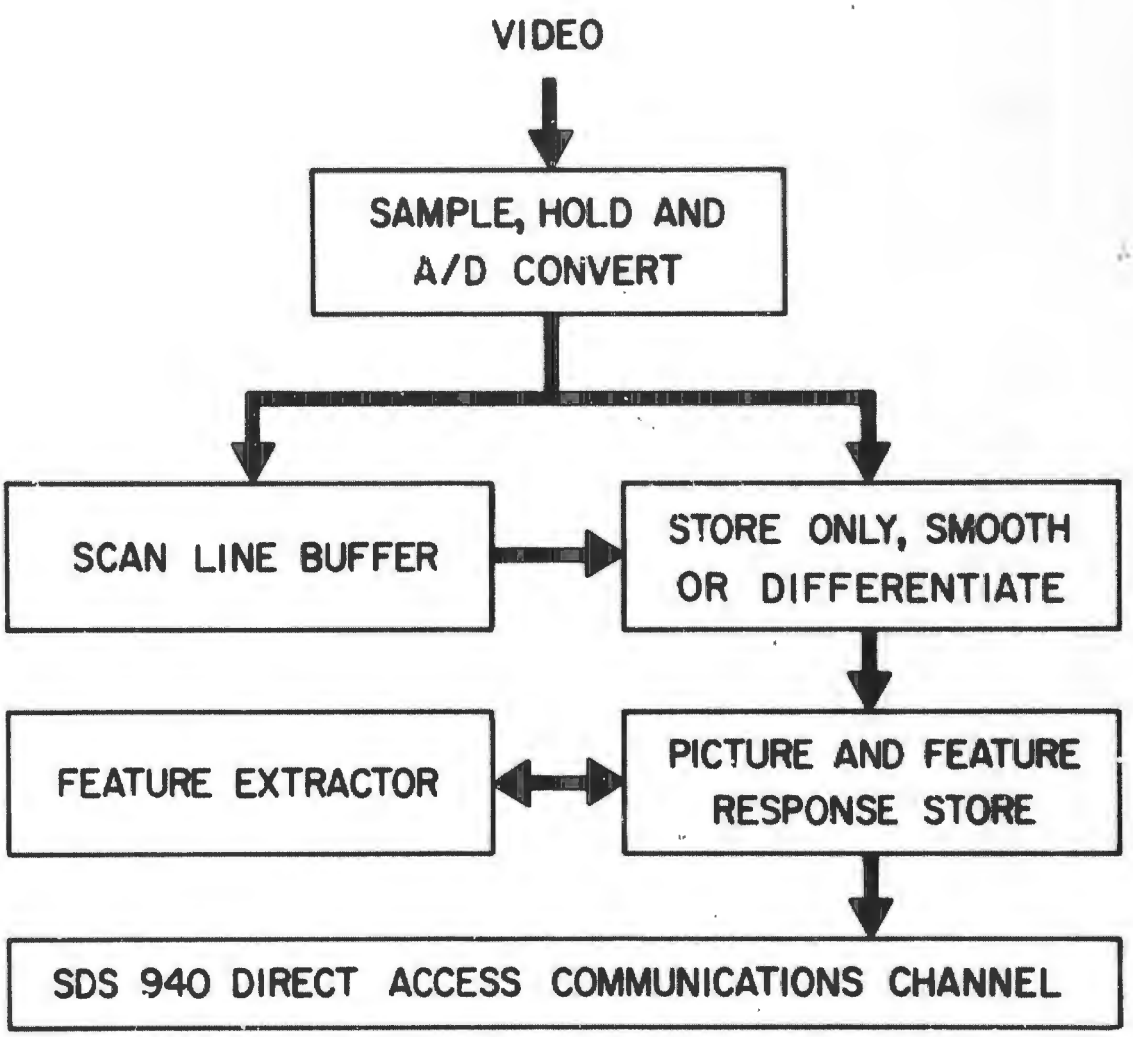


Figure 10. Visual-Data-Preprocessor Hardware

or to a stereo attachment field of 60 scan lines using every fourth scan line and all 240 samples per line. In each case, a total of 14,400 picture elements are stored and packed into 2400 preprocessor buffer store words. Figure 11 shows the three sampling formats. Figure 12 shows the spacing of the samples, and Fig. 13 shows the expressions for the spatial sampling, smoothing, and differentiation operations. The spatial differentiating operator used is an approximation to the one suggested by Roberts.[13] The operator is applied to square arrays of four sample points. (The operator could be easily extended to nine-point square arrays requiring temporary storage of two scan lines.) The sampling-only operation merely selects the upper left sample, and drops the lowest-order bit. The averaging operation adds all four samples and drops the three lowest-order bits of the sum. The differentation operation adds the absolute magnitude of the differences of the values at both sets of diagonally opposite corners of the square and truncates at level 15. Furthermore it sets to zero all those points below a computer set threshold level. This transformation will be done by the hardware as fast as the data are produced, about six million picture-element computations per second. This is equivalent to 18 million additions and/or subtractions, plus an average of 9 million complements, plus, if differentiating, six million overflow tests and six million comparisons plus six million word-packing operations a second.

It was initially intended that the averaging operation would be used in gross area considerations where local intensity information was desired. Because five-bit data were used for each picture-element computation, the resulting four-bit average is considered to be essentially free from the effects of quantization noise. The differentiation operation was planned for use in the construction of line-drawing representations, and for texture analysis. It is particularly important for line-drawing synthesis to threshold the resulting derivative value so that subsequent preprocessing operations can be simplified and speeded up. Thresholding was originally intended to remove the effects of minor local variations in intensity in the light reflected from an otherwise smooth surface, and also to minimize the effect of whatever noise would be present.

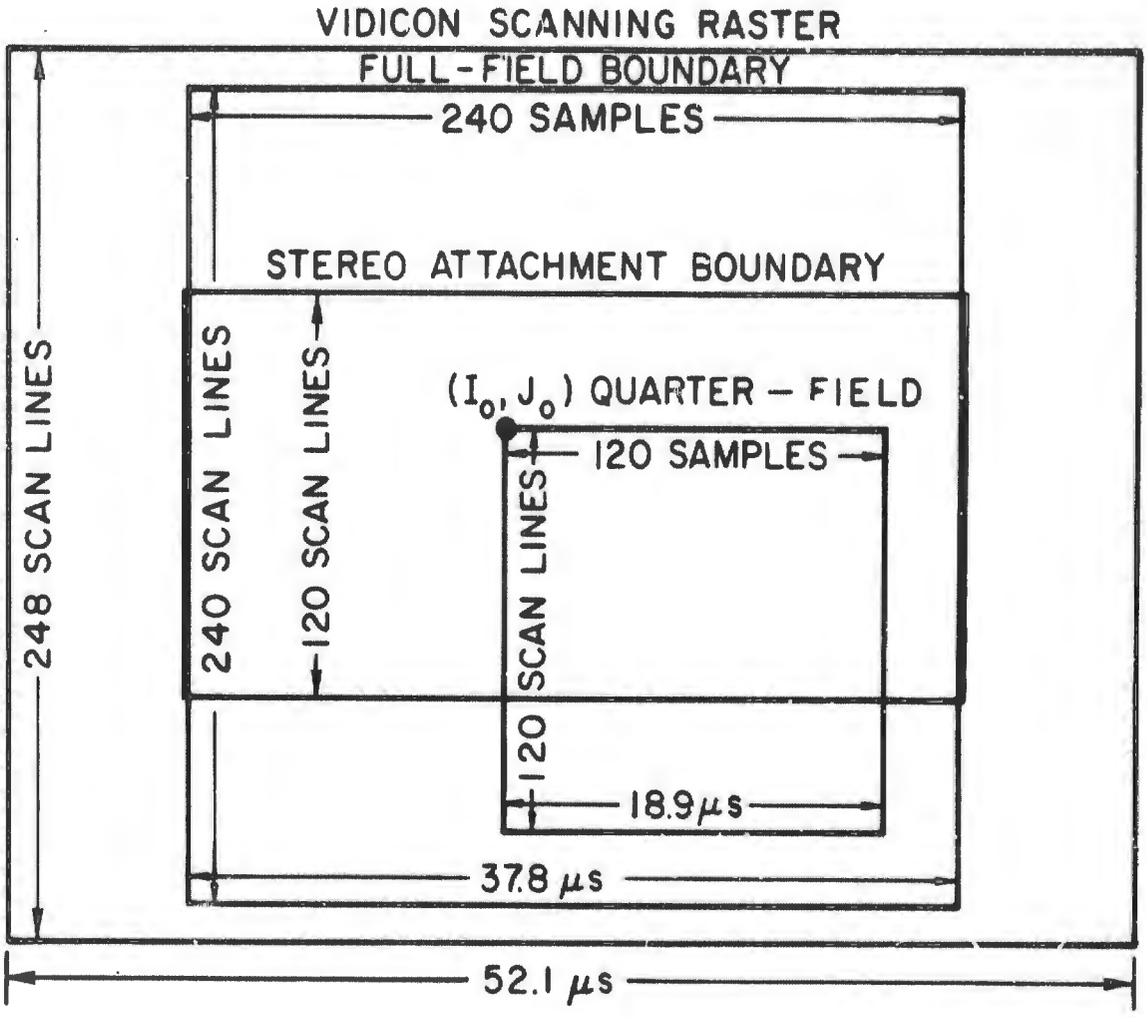


Figure 11. Video Sampling Formats for Full, Quarter, And Stereo Fields

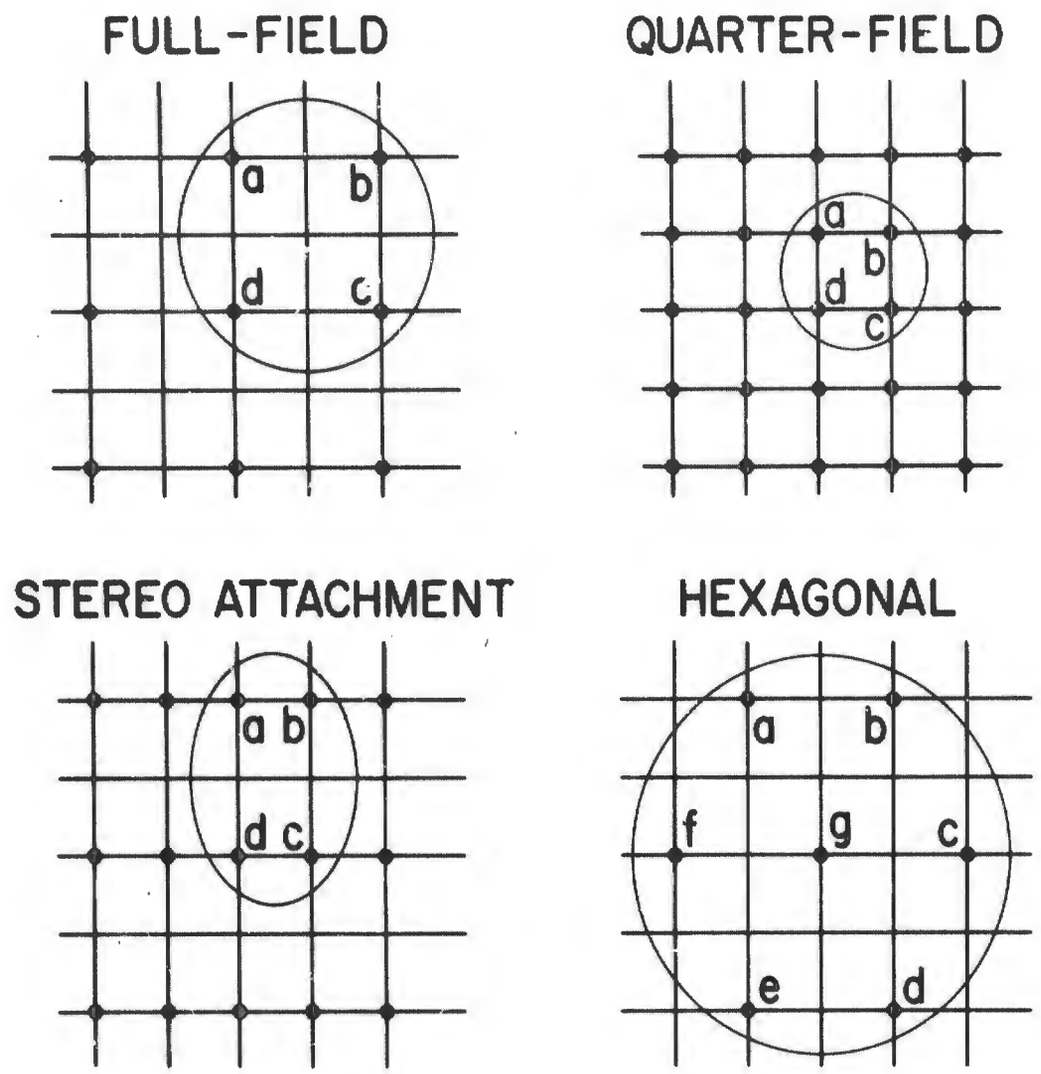
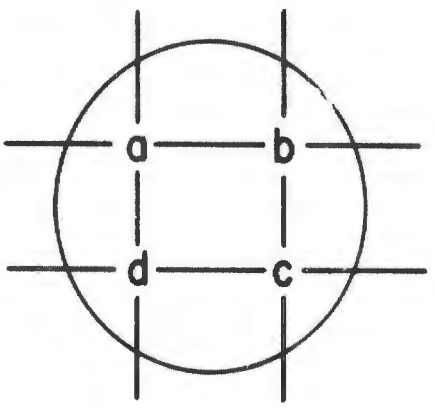


Figure 12. Sample Spacing for Full, Quarter, Stereo, and Hexagonal Formats



PICTURE ELEMENT VALUES

MODE	EXPRESSION COMPUTED
STORE ONLY	$y = a$
SMOOTH	$y = \|a + c\| + \|b + d\|$
DIFFERENTIATE	$y = \|a - c\| + \|b - d\|$

PICTURE ELEMENT COMPUTATIONS

Figure 13. Expressions for Spatial Sampling, Smoothing, and Differentiation

Our experience with quantization of the original video into sixteen levels has suggested that circuit noise is not a problem when the camera and transmitter are operating properly. What has been a serious drawback is the necessity to remove the derivative response due only to quantization. A smooth surface may change many quantization levels in reflected light intensity from one side to another. It is common for this change to be systematic, thus often producing a derivative value of two merely from going from one quantization level to another in neighboring picture elements. Furthermore, we have noticed that important edges are often missing from the derivative data because the reflected light intensities from the two adjoining surface differ by less than two quantization levels out of the total of sixteen. The lack of gray-scale resolution has probably been the factor that has caused the greatest difficulty in subsequent processing. Many of the special ad hoc techniques in the subsequent processing that try to recover this lost information could be simplified or eliminated with a noticeable decrease in computation time if more reliable levels of quantization were used.

The hardware preprocessor uses all thirty-two quantization levels to compute the value of the spatial derivative. The same numerical value of threshold that is used to eliminate the effects of quantization using sixteen total levels could be used to eliminate the effects of quantization using thirty-two levels. This would allow an increase in sensitivity to legitimate responses by a factor of two. One quantization level out of thirty-two corresponds approximately to a change of fifty percent in light intensity.

c. The Preprocessor Memory as a Buffer

The use of the preprocessor memory as a block data buffer avoids many serious problems that would have arisen had we attempted to use the main SDS-940 core memory in which to store raw visual data. The buffer can first synchronize with the camera, store raw or preprocessed data, then synchronize with the computer in transferring that data, and immediately, if desired, store another TV picture.

Without the use of this buffer, inputting one TV field via a temporary channel places extraordinary demands on the SDS-940 time-sharing system. The temporary TV input channel shares a common port-to-memory with the drum files which are in constant use for file storage and user program swapping. The data transmission from the TV camera to the computer cannot be interrupted once it has begun, because the camera scanning is not under computer control. Thus the camera must wait until the queue of drum file requests is satisfied before a picture can be recorded. Although complete transfer is accomplished in 1/60 second after the start of transmission, the port must be dedicated to the camera for a time that includes not only the transmission time of the data but also an initial random delay to wait for the beginning of the camera field to occur. Furthermore, because the transmission capacity of the computer is the limiting factor it is not possible to "steal" cycles for other possibly slower devices. In addition to this inconvenience to the time-sharing system, it is necessary to "lock-up" several 2048-word pages of core memory and allocate them specifically to the TV data file. These pages are not available to other users during transfer. Another device that shares this port is the SDS-940 display. This display does not have its own buffer memory and must be constantly refreshed from the SDS-940 core. This refreshing must also be interrupted when data are transferred via the temporary TV input. Recently a large disc file has been added to the SDS 940 system. This complicates the operation even further. In addition, the Automaton itself shares the same channel. Once the transmission of the TV data has taken place, time-sharing resumes its normal mode of operation. It is quite probable that the program that called for a TV picture would then be swapped out. It has not been determined how long it would be before another picture could be taken, because that would depend on the number and type of other user jobs on the system. It has been suggested by our system programmers that the period between two consecutive pictures could vary from a small fraction of a second to on the order of ten seconds. Thus, relative-motion on tracking experiments are not practical with the temporary TV input, because timing is important.

d. The Preprocessor: Feature Extraction

Figure 14 shows the visual system hardware design in block diagram form. Most of the hardware shown is used for the two preprocessor functions. As mentioned above, the first is to apply a sampling, a spatial-smoothing, or a spatial-differentiation local operator over the entire field at standard TV scan rates. The resultant data are stored for the second function, which is the extraction of local features--e.g., short line segments--over any computer-specified rectangular portion of the field.

The second special transformation of the preprocessor is to emulate the first layer of processing in the cat's visual cortex. The purpose of this hardware is to reduce the large amount of data in the differentiated (or unprocessed) digitized image to a smaller amount of meaningful features. The conceptual design has been discussed in detail in the First Interim Report in Appendix J. A brief review is given below for completeness of this report.

The method of operation of the feature-extraction hardware is as follows: At each picture-element location we assign the feature of a set of computer-generated features that produces the highest "correlation" with the data in the neighborhood of that element. A neighborhood consists of a 7 × 7-picture-element square array. The correlation function is a normalized, forty-nine-dimensional, scalar product between the feature vector and the corresponding picture-element vector. The components of the feature vectors are limited to three values: +1, 0, and -1.

The feature responses are subsequently reduced in number, but not in accuracy of position, by two methods. First, the value of the maximum correlation is compared with a threshold. If the threshold is not exceeded, the associated candidate feature is considered as noise, and omitted. This threshold is the larger of a computer-generated value and a value computed by the preprocessor from the picture elements in correspondence with the feature. The threshold must be a function of

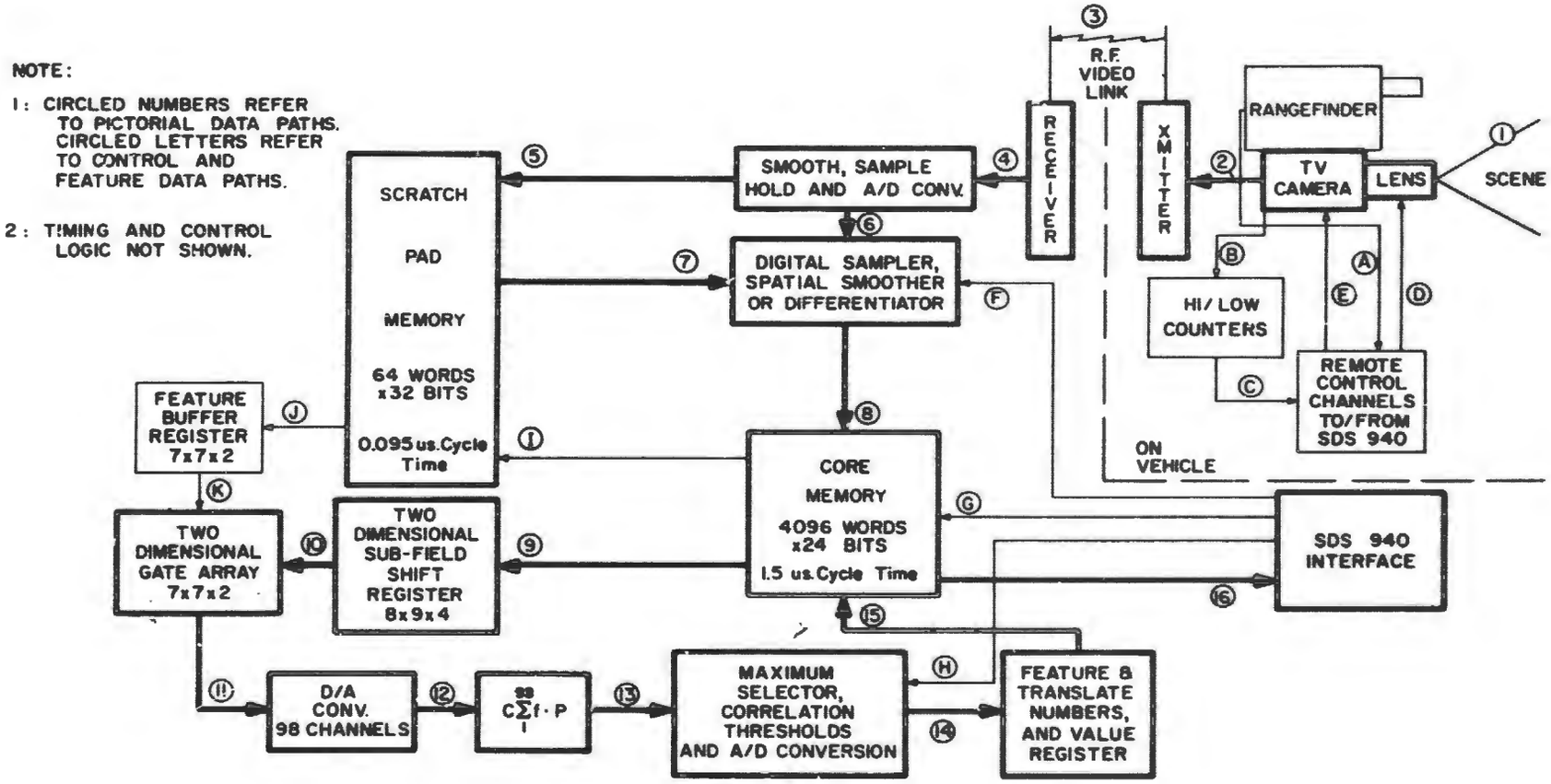


Figure 14. Block Diagram of Visual-System Hardware

local contrast for a differentiated picture or local average value for a non-differentiated picture.

The number of responses is further reduced by retaining on the first pass only the maximum response for each contiguous 3 × 3 picture-element subfield. Thus, the original 240 × 240 samples of raw data are reduced to 38 × 38, or a total of 1444 response fields, each of which may or may not contain a single extracted feature. It is expected that the hardware will perform this second transformation at a rate of two million correlations per second. Using, for example, sixteen features in a set, the entire field could be transformed in a maximum of 100 milliseconds. By using the same technique in the hardware that allowed the simulation to be speeded up--namely, omitting the correlation calculation for zero derivative picture elements--the hardware processing time can also be reduced. If one correlator were assigned to a single feature, the total time could be reduced to about ten milliseconds. It is conceivable that the preprocessor could then keep pace with standard TV frame rates.

It was intended that the feature-extraction hardware would be used in the following possible way: One pass through the entire digitized gray scale or derivative image would extract a set of features that have the highest correlation exceeding threshold, in their respective neighborhoods. Judging from the way this alone has worked on real data, it will often be advisable to use the second-pass capability of the hardware preprossor. The second-pass would produce the highest correlation exceeding threshold in each neighborhood but would inhibit any response at the location of the feature extracted on the first pass. This will have the effect of filling out possible voids between adjacent responses from the first pass.

Another capability of the hardware should be noted. The feature set is computer-generated and stored as data in the External Core Memory for subsequent use by the Scratch Pad Memory. It was intended that after the first (and possibly the second) pass over the entire image, which would require a fraction of a second, the responses

so obtained would be used by computer programs to form longer line segments. At corners of objects and/or at breaks in what should be long segments, it would be useful to refer back to the digitized gray scale or differentiated picture. The hardware would locally correlate with new features that were determined by the computer program to join the longer segments at corners or at breaks. There are too many possible types of corner junctions and possible angles for each example to be included in the primary set of features. The synthesis of the connected line drawing would be made more accurate by the use of the detailed digital image data again in just those small areas of ambiguity where longer line segments could be joined. It is more efficient to compute the feature set for these areas based on the first line synthesis. Fortunately the hardware has the capability of extracting computer-generated features over a computer-specified local sub-image as well as over the total image.

3. Optical Range Finder*

Mounted with the camera is an optical range finder. It consists of a light source deflected by a rotating mirror and mounted above the camera. The light beam is swept at a constant angular speed vertically, beginning from an angle of -45° from the optical axis of the camera lens and a receiving lens physically mounted below the camera. When the light beam strikes an object on the optical axis of the receiving lens, a signal caused by reflected light is detected by a light detector. The time for the beam to sweep from the -45° position, which is detected by a separate detector, to the reception of the above-mentioned signal is a measure of the distance to the illuminated object. This count is transmitted to the computer and a table gives the corresponding distance.

---

*See also pp. 28-29 of the Second Interim Report.

C. Vehicle*

1. Wheel Arrangement

The vehicle is equipped with two driving wheels paced on each side of the main vehicle body. One caster wheel is placed under the front end of the vehicle. The driving wheels and the front caster wheel are, in normal operation, supporting the whole weight of the vehicle. A second caster wheel is placed at the rear end of the vehicle, attached to the rear end of a battery carrier. The front of the battery carrier is hinged below the vehicle platform and in front of the drive wheels. If in operation, the center of the load falls behind the driving wheels, this caster will take the load from the rest of the vehicle. The rear caster wheel is designed to allow the vehicle to climb a ramp having a 1/10 slope with the floor plane.

2. Wheel Operation

Each driving wheel is equipped with a stepping motor. These motors are individually controlled by a digital servo. The servo is designed to perform two modes of operation in a stable manner. One translation mode allows for moving the vehicle a specified number of steps forward or backward. One rotational mode allows for rotating the vehicle around its center a specified number of steps.

The motion of the two wheels is completely independent and controlled by separate registers. A coupled mode, however, provides direct mechanical coupling between the two wheels via electromagnetic clutches, one for linear motion and one for rotational motion.

Each wheel is equipped with a parking brake. This brake is on when power is off. The brake is released when the motor is energized.

Each of the two drive wheels on the vehicle is driven by a stepping motor. On command from the computer, the motor can drive the wheel forward or reverse a given number of steps. To overcome the

---

*See also Appendix A of the Third Interim Report.

special limiting effect on the stepping motor imposed by a high-inertia load, a special inner-loop feedback system is used. This synchronizes the stepping rate with the actual vehicle speed. The vehicle can move forward and in reverse by applying the same direction command on the two wheels or it can rotate by application of opposite direction commands.

D. Action Logic*

The action-logic modules consist primarily of 24-bit registers and associated gates and flip-flops. Each action module on the vehicle has an input register receiving data from the computer and/or an output register from which data is transferred to the computer. Each module has its own address as specified by the computer. First the computer gives the address of an input-output device via the Direct Access Communication Channel. The data word transmitted to the selected unit begins with a module address. This module address is broadcast to all modules on the vehicle, but activates only the one that has been assigned the address.

Every action module is equipped with a digital servo. In most of the modules, stepping motors are operated at a constant stepping rate. The input register includes a binary counter that tells how many steps to proceed. The stepping continues until a zero count is reached. The output register of the action modules is the same as the input register.

In the wheel servo system, where the motors operate with high inertial loads, a continuously variable stepping rate is available, controlled by a feedback signal from the motor shaft. The input register of the selected module is filled with data from the computer. These data have bits reserved for mode control, direction control, distance of displacement, threshold information, etc. A typical module requires between 6 and 18 bits of information. Provisions have therefore been

---

*See also Appendix H of the First Interim Report.

made to transmit up to three 6-bit characters to each selected module. Likewise, information on the status of a module can be transmitted back to the computer in groups of up to three 6-bit characters.

Some modules are reserved for overall vehicle control. For instance, to preserve power, a number of power switches controlling major power-consuming sectors of the vehicle are controlled by one register that can be addressed. Another register is reserved for action status. Each module has an action flip-flop that is set when an action is in process. The contents of up to 18 such flip-flops can be read out to the computer. The output gate of one such group of 18 flip-flops is addressed in a fashion similar to the way in which other action modules are addressed.

Special provisions are made for emergency situations. An emergency exists when certain limit switches are closed, when the range and speed of the vehicle call for emergency stop, when abnormal conditions on the vehicle and its environment are in effect, when the battery voltage is below a safe level, etc. Any of these conditions results in special interrupt signals that are transmitted to the computer. Each emergency situation has one flip-flop assigned in an emergency register on the vehicle. When an emergency interrupt has reached the computer, the emergency register will be read in for inspection and necessary action. Some of the emergency signals are acted upon locally on the vehicle. For instance, touch sensors on the bumpers of the vehicle bring the vehicle to a stop.

The local safety circuits on the vehicle must sometimes be over-ridden by commands from the computer. For instance, if an object is to be pushed by the vehicle, the above-mentioned emergency stop must be disabled. An emergency override register is available. This register can be set by signals from the computer under strict restrictions. This register is automatically reset on the vehicle after preset delays or other local conditions expected to occur frequently, such as a zero count in the wheel-distance register.

In order to make most efficient use of available space and power, a local timesharing on the vehicle was introduced. to 16 modules

with similar value can share a common shift register holding .1 data for the modules. Only addressing logic and some special functions are uniquely wired for the individual modules.

E. Data Communication*

Data are communicated to and from the vehicle via a full duplex radio link. The system on two frequencies--140.385 MHz from vehicle to the stationary station and 163.4625 MHz from the stationary station to the vehicle. The video signal from the TV-camera on board the vehicle is transmitted as a frequency-modulated signal at 1.7 GHz.

*See also the Appendix of the Fourth Int[illegible] Report.[4] A console for manual opera[illegible] of the Automaton is de[illegible]d in Appendix B of the Third Interim [illegible]ort.

Appendix A

DESCRIPTION OF RBT SYSPOP

by

L. J. Chaitin

Appendix A

DESCRIPTION OF RBT SYSPOP

Because of the necessity of the Automaton communicating with or through the time-sharing monitor, a new SYSPOP -RBT- was built into the time-sharing system. To use it in ARPAS code one writes RBT n, where n is a number from one to eleven. The significance of n is as follows:

n=1 Handles SKS commands. This checks the status of either the vehicle or the preprocessor.

n=2 Handles system EØD commands. This is to check channel status and initiate certain kinds of I/Ø.

n=3 Handles EØD; PØT sequences. This is for certain one-word transfers, and to ready the channel for reading.

n=4 Handles DACC transfers. This command is the one that does all I/Ø. The devices accessible with this command are the Automaton, preprocessor, Rand tablet, and 910 link. It is necessary to specify the characters to be written (or read) and the core address of the transfer.

n=5 RBT 5 is the mechanism by which data are stored between the monitor and certain software (primarily the Automaton). It accesses a set of words, each one being a flag or switch of some sort. The information is transferred via the A register. The B register contains:

(1) Read word into A

(2) Store word into A

(3) Increment word

(4) Decrement word

(5) Set bits on in word that are on in A

(6) Turn off bits in word that are on in A

(7) Read address of word into A.

The X register contains the number of the word to be accessed:

(1) Used by the Automaton system to dismiss and awaken interrupt and command programs

(2) Incremented when a bump occurs

(3) Incremented when a limit occurs

(4) Tells which actions have been initiated

(5) Tells which actions have been completed

(6) Used by television

(7) Used to dismiss and awaken I/Ø programs

(8) Used to dismiss and awaken I/Ø programs

(9) Used by preprocessor

(10) Keys iris position

(11) Keys focus position

(12) Keys tilt position

(13) Keys pan position

(14) Used to control I/Ø

(15) Used to dismiss and awaken the VALET

(16) Keeps status of power register

(17) Incremented when a read error occurs

(18) Incremented when a write error occurs

(19) Used by 910 interface

(20) Used by Rand tablet software.

n=6 Locks in a page and uses data therein to start the CRT display.

n=7 Shuts off CRT display and unlocks page.

n=8 Arms various Automaton interrupts.

n=9 Disarms various Automaton interrupts.

n=10 Arms the light gun used with the CRT.

n=11 Returns with all the PMT bytes that are currently assigned.
Used by the VALET.

Appendix B

LISTING AND SUMMARY OF TWO-LETTER COMMANDS

by

L. J. Chaitin

Appendix B

LISTING AND SUMMARY OF TWO-LETTER COMMANDS

The two-letter commands used in the system are listed below. In these commands n is a parameter to the command. If n is negative, the opposite direction is assumed. Such things as setting and arming bumps and limits, etc. are not directly controlled by the two-letter commands except for UN.

MØ - Move forward n feet.

TU - Turn counterclockwise n degrees.

PA - Pan the head counterclockwise n degrees.

TI - Tilt the head up n degrees.

RI - Turn counterclockwise about the left wheel n degrees.

LE - Turn clockwise about the right wheel n degrees.

IR - Adjust iris control n increments.

FØ - Adjust focus control n increments.

ØV - Sets the overrides on the wheels--i.e., sets vehicle to ignore bumps--according to n. Sets left wheel override (n=1), sets right wheel override (n=2), sets both overrides (n=3), or turns off overrides (n=0).

RA - Samples the range finder and prints value in feet.

UN - Disarms a Automaton interrupt according to n.

XR
YR - Sets Automaton position to XR, YR in grid model.

XG
YG - Sets goal position to XG, YG in grid model.

AN - Sets Automaton orientation as angle n degrees with respect to x axis. By definition AN = 0 when Automaton "looks" along x axis.

PL - Plan a journey. Treat unknown space as empty (n=0), full (n=9.9), or make "fullness" a function of n. XG and YG must have previously been given. Generates best possible path from known information. Display plan on scope.

EX - Execute the plan generated by PL.

SC - Turn 360 degrees. Turn back 360-n degrees. Print range. Repeat until turn back = 360 degrees total.

PI - Take a picture, and generate a floor boundary by changing TV picture into a line drawing. Display floor boundary and update the model.

MA - Map the model onto the display, having first updated the model.

GØ - Given XG and YG, go to goal. Does not use the model. Gets around obstacles by "touch" only.

WH - Prints location (X,Y,Ø) of the Automaton.

RE - Prints position of pan and tilt, setting of Automaton interrupts, setting of iris and focus, and status of cat whiskers.

AL - Sets time alarm to n seconds; generates an interrupt after n seconds.

ST - Stop, dump model to file /ØLDMØDEL/, disarm all hardware interrupts.

WR - Set pan, tilt, focus, or iris, depending on n to the value m.

CL - Clears model; sets it to completely unknown.

TE - Given XG and YG, takes a picture of the gaol (gets floor boundary and updates model), generates a plan and executes it. If an unexpected bump occurs, the Automaton tries to clear the object and the above procedure is repeated.

Appendix C

IMPROVEMENTS IN THE AUTOMATON'S VISUAL SYSTEM

by

G. E. Forsen

BLANK PAGE

Appendix C

IMPROVEMENTS IN THE AUTOMATON'S VISUAL SYSTEM

I INTRODUCTION AND SUMMARY

Stanford Research Institute is currently engaged in the construction of and programming for a computer-controlled Automaton. The Automaton will have a visual sensor with some pattern-recognition ability. The visual system will be used directly to classify objects in the field of view, and/or indirectly to assist the Automaton in performing other functions such as navigation in or modeling the environment.

The Automaton's visual system and its present visual environment have been designed to be mutually compatible. The present system will attempt to classify isolated objects in the viewed three-dimensional scene on the basis of shape descriptors synthesized from features extracted from the measured intensity of light incident on a single two-dimensional projection. The present design specified techniques for which precedents existed, and equipments that were readily available at allowable cost. The system will be composed of a vidicon TV camera, a hardware preprocessor that extracts features, and programs in an SDS 940 computer for combining these features into descriptors for classifying the descriptor sets and/or assisting the navigation or modeling routines of other programs. However, these methods and equipments are expected to perform satisfactorily only in a restricted laboratory environment.

The visual system will be able to differentiate between several classes of geometrically simple objects when each object is visually distinct from its background and is not partially or totally occluded by other objects. The system will depend on the Automaton's executive program to move the vehicle and/or camera pan and tilt to obtain better views when necessary. This action is considered to be costly and in some circumstances may be impossible. The present system specializes

in objects that have only a small number of planar, nonspecular, opaque, evenly illuminated surfaces. Environments whose objects can be so described would rarely be found outside of the confines of the present Automaton's "world."

In order for the visual system to provide useful performance in its present environment, additional capabilities should be provided. In particular, the visual system, aided by an executive program that constructs models and provides relationships, should be capable of scene analysis.* It should be able to isolate and name what objects are in the field of view and describe their physical relationships. It is also desired to increase the complexity of the Automaton's current artificial environment, to anticipate more realistic applications. The following sections discuss problem areas requiring solution before specific applications are considered, general techniques thought to apply, and specific methods proposed for investigation.

II SPECIFIC AREAS FOR POTENTIAL IMPROVEMENT

The following are now considered to be important areas in which any improvement of the Automaton's visual system would be useful. There are at least two classes of problems--those due to the variability of the environment and the system and those due to increasing complexity of the environment. The first class is more fundamental and needs solutions for any nontrivial three-dimensional environment.

(1) In order to facilitate scene analysis, the so-called "figure-ground problem" must be resolved. In order to minimize the effect of the background it is necessary to isolate the part of the field that belongs to a given object (figure) from other objects as well as from the

---

*The related tasks of tracking and modeling are also important but will not be mentioned below explicitly in order to make this note brief. Improvement in performance in these will follow as a consequence of work on scene analysis. However, it is anticipated that their implementations (both hardware and software) would differ considerably for reasons of speed and efficiency.

background. This problem has yet to be solved, except for certain manufactured data.

(2) A related problem is the "occlusion problem." This occurs when an object is partially obscured from view by another in front of it. It may first be necessary to solve the figure-ground problem to isolate foreground objects.

(3) Invariance to perspective, translation, and other geometric transformations is another problem area. Because only a projection can be sensed, the shape of an object as well as its background varies with the viewing location and direction. The shape descriptors are invariant only to small amounts of translation, rotation, scale change, distortion, and/or noise. Thus, the classifier must have sufficient capacity for, and be trained with, examples of a large variety of views for each class. The present system will probably be required to treat topologically different views of each object as subclasses of a given class in order to differentiate between objects of different classes having similar shapes for certain views. This limits the number and complexity of classes for a given system size and/or response time.

(4) Another problem area is the restriction to planar surfaces. Although a linear boundary is the simplest to synthesize, there are few real objects whose boundaries are entirely linear. The addition of higher-degree descriptors would allow objects with curvilinear boundaries to be described more accurately and/or efficiently. The next higher level of complexity is to include compound objects. These objects can be described as if they were constructed from simple solids. The next level might be described as irregular but smooth (e.g., telephones and ash trays); and yet a more difficult class would include irregular and rough (e.g., trees and bushes).

(5) The present preprocessing scheme extracts local features--for example, short line segments corresponding to gradients of incident light intensity. There will most likely be feature errors of commission and omission. Shadows will introduce extra lines, and surfaces of nearly equal intensity adjacent in the visual field will cause lines to be lost. The resulting larger and/or more complicated descriptors that are formulated from the resulting feature set will not, in general, completely correspond to the boundaries of the surfaces of the objects.

(6) Camera noise limits the sensitivity to intensity gradients for a fixed intensity range (or vice versa). Related problems are uncertainty in position of vehicle, and in the accuracy of the stored model(s). These problems may require a statistical view of the environment and/or a description of visual scenes that do not depend on precise geometric information.

(7) A basic problem relates to the wide ranges of visual data (light intensities, spectra, and detail) found in natural environments. In order to detect fifty percent changes in intensity, the total range of intensity is presently limited to 20:1 for a given field of view. Overall intensity range for a normal office environment may exceed 10,000:1. The present system can just detect a door-frame shadow when the full frame is in the field of view, but doesn't have enough resolution for all the surface clutter on a desk, for example.

(8) A general consideration, associated with all the above topics is potential cost in terms of computation time, storage space for both programs and data, and hardware. The present visual system is not efficient in many ways. The solutions to the above problems will result in even greater complexity. Presently available computing power and mass storage facilities rule out the more direct solutions. Thus, not only must schemes be developed that are potentially more efficient,

but also their method of implementation must be chosen so as to keep computer usage by the visual system commensurate with its usefulness to the overall performance of the Automaton.

III POSSIBLE TECHNIQUES TO INVESTIGATE

It is now anticipated that the following techniques will enhance the solution of the above problems:

(1) Use of directives from higher-level programs using learned or stored environment models. The higher-level program would anticipate certain responses from the visual system under certain conditions. The visual system would "look for" what is expected rather than passively respond.

(2) Use of partial sets of descriptors which make tentative classifications based on partial views. Instead of requiring a description based on a complete view of an object, classification may require only sets of descriptors that are sufficient to distinguish one object from another.

(3) Use of basic shapes of not only contours but also surfaces and possibly elementary solids. These basic shapes, or distinguishing parts, are sensitive to the particular environment. It is proposed, therefore, that adaptive techniques be developed that use feature data to synthesize or identify the shape of these parts. Thus, trainable classifiers would categorize parts of objects, rather than whole objects.

(4) Use of other decriptor types in addition to light-intensity gradients, which would be a more ambitious effort. For example, depth, relative motion, texture, and possibly color can also be used to determine contours more reliably when used collectively, and to enhance the association of areas in the visual field with their corresponding objects. Using other types of descriptors may help resolve contour errors when missing or extra intensity gradients occur. They are

also either more directly related to geometric transformation (e.g., depth and relative motion are simply related to translation), or more invariant (e.g., color and texture remain nearly constant until a different source of illumination is encountered).

(5) Additional techniques, which are proposed to help solve these and other problems--in particular, the development and use of qualitative descriptors. The positional concepts of "next to," "behind," "around," "far away," etc. are imprecise, but possibly sufficient for most relationships between objects. The prevalence in natural language of these qualitative descriptors suggests their usefulness. They may provide an efficient way to represent imprecise models and learned or stored associations.

The use of functional concepts such as "support," "contain" (i.e., the particular use to which it is being put) should be explored. Broad classes of objects are often more efficiently described in natural language by their function, rather than their shape, color, etc. For example, a chair is used for sitting. Imagine the variety of shapes, sizes, and colors of chairs.

(6) Multiple views, and/or recomputation with new features and descriptors. In particular, if noise is a problem, the input data from several readings of the same scene can be averaged. If light-intensity range is a problem, transducer gains (camera and/or lens iris settings) can be set for a different region of the field each reading. If the descriptor synthesis program suggests a corner at the implied intersection of two lines, then the corresponding picture elements can be preprocessed with a corner feature. If resolution is a problem, and if a zoom lens is available, the focal length can be increased. If a zoom lens is not available and a wide-angle scene is required, a lens of long focal length can be used

for detail and several contiguous fields can be read for the wide-angle requirement. If still unable to describe or analyze the scene, or recognize an object, as a last resort the vehicle can be moved. Repetitive decisions from consecutive viewpoints may also improve the recognition confidence.

There are at least two approaches to the realization of visual processing techniques. They are (a) trainable structures, and (b) preprogrammed structures. The former attempts to define functions on the basis of the experience of the system itself. The latter directly relates to the experience of the programmer, and his a priori notions about the nature of the environment. It is expected that:

(7) A combination of preprogrammed and trainable structures, which will provide a more efficient, higher-performance visual system. The specific assignment of type of structure to subsystem function will require considerable experimentation.

The classifier, whether it be a trainable structure or list-matching model-transformation technique of Roberts is a classifier that does not require prenormalization to translation, rotation, etc. It is proposed that experimentation be undertaken to resolve the issue of what subsystem has the responsibility for providing invariance, and that newer programming languages be studied and/or developed in order to provide more convenient tools for the programmer and others to improve the efficiency of the processing.

Similarly, anticipating a continued effort in this field, we suggest that other (potentially) available computing, storage, and transducing media be reviewed. It is felt that the specific hardware ultimately used will strongly influence the choice of visual system techniques applied.

IV SPECIFIC INVESTIGATIONS TO IMPROVE THE AUTOMATON'S VISUAL SYSTEM

A review of the association matrix between the above proposed techniques and the observed problem areas shows several techniques that

are related to most problems. Only a few techniques are related to specific problems. This would suggest a technique rather than problem-oriented investigation. However, the specified purpose of this effort is to improve a particular system. Thus the strategy will be to select certain aspects of the above-mentioned techniques, study areas, and hardware changes that will have more immediate payoff with respect to the Automaton program. It should be mentioned, in passing, that the proposed hardware additions are relatively straightforward and can be obtained with little study and not much more effort, while the investigation of the proposed techniques and study areas appears at the moment to be more or less open-ended. The following lists specific investigations related to problem areas, which have been selected for more immediate improvement to the Automaton's visual system, with only minor additions to the present equipment.

1. The use of models of the environment and objects to differentiate figure from ground. The simplest nontrivial figure-ground problem is one unobscured object distinct from its background. The goal is to isolate the object, thus eliminating the effects of background. A proposed method to isolate the object is to assume a geometrically precise model of the environment having location, orientation, and identity of objects stored in a higher-level program. This latter program will offer the hypothesis that at a particular location in the field of view, a known object at specified orientation should be visually sensed. Assuming the camera position and direction are known, the visual system will use the above data with its own stored model of the specified object to eliminate data anticipated to be background. The remaining data will be used to check the identity of the object. This method is expected to be most useful when the particular object is being repeatedly observed.

2. The use of depth (and other) clues to differentiate figure from ground. Another proposed method is to use depth information and assume that the object is one body and does not extend "far" beyond its observed surfaces. Initial use will be made of the Automaton's range-finder equipment. Related clues are areas of near-equal velocity relative to

the visual projection field when relative motion occurs, areas of constant intensity, texture, and/or color. This study will begin with contrived data until the necessary changes in and additions to the equipment are made and related feature extraction and descriptor synthesizing programs are sufficiently developed. For the purposes of the figure-ground study, the various descriptor types will not be refined, as this represents considerable effort in itself.

3. Depth, relative motion, texture, and color descriptors. Horizontal disparity of vertical lines will be investigated to provide depth descriptors. A study of constant relative-velocity features will provide relative motion descriptors. Spatial harmonic analysis is expected to be useful for texture descriptors. A two-color scheme will be investigated for color descriptors. The purpose of this task is to provide only rudimentary descriptors of these types, although this investigation (as well as some others listed herein) could easily be continued beyond the currently proposed initial study.

4. Collective use of several descriptor types. It is proposed that the coincidence of boundaries of nearly equal intensity, depth, texture, and possibly color will be more reliable than any one alone. Combinatorial and priority assignment methods will be studied.

5. Partial list matching to identify partially occluded objects. The simplest occlusion occurs when an object has already been differentiated from its background as well as the foreground objects. If the classifier is a list-matching structure and only partial lists are available as a result of occlusion, the programming languages CONVERT and FLIP are expected to be useful tools. It is proposed that these be obtained for the SDS 940 time-sharing system to investigate their efficiency and convenience. Initially it will be assumed for the purposes of this study that no descriptor errors exist. A related method is the use of superimposed codes.

6. Ternary-valued inputs to trainable structures. A trainable structure will be trained using ternary-valued inputs. A +1 or -1 input stands for the occurrence or absence of a particular descriptor,

and zero stands for the data occluded or the background. Related experiments will use more geometrically invariant inputs.

7. Invariant topological descriptors. Classifiers will be tried that use descriptors invariant to geometric but not topological transformations. Both list matching and trainable structures will be tried. The selection of descriptors will be by trial and error.

8. Second-degree shape descriptors. The addition of second-degree, in addition to piecewise-linear descriptors, will enhance the classification of objects by boundary information and perhaps improve the classifier's efficiency by reducing the number of descriptors required. It will allow the extension of the environment to include nonplanar-surfaced objects.

9. Use of models to select descriptor and feature sets. In this experiment, the selection of descriptor and feature sets will be made according to the scene anticipated by the model of the environment.

10. Modify Roberts' technique. Roberts' technique for transforming a gray-scale picture into a black and white line drawing and classifying objects by fit criterion on transformations of stored models, will be modified to allow some line-segment errors in the model-matching data.

11. Directive to higher-level program. In order to obtain repeated looks at an object from different (but neighboring) points of view, an investigation of differentiation techniques, to produce directives to higher-level Automaton programs to produce the required motion, will be studied.

12. Cost-performance tradeoffs in sequential operation. Quantitative measures of cost and performance will be developed to evaluate optimum apportionment of mechanical vs. electronic operations related to the visual system.

13. Averaging to reduce noise. New Scratch Pad Memory (in the visual preprocessor) formats and repeated averaging techniques will be used to reduce camera and other random noises.

REFERENCES

1. C. A. Rosen and N. J. Nilsson (Eds.), "Application of Intelligent Automata to Reconnaissance," First Interim Report, Contract AF 30 (602)-4147, SRI Project 5953, Stanford Research Institute, Menlo Park, California (November 1966), RADC-TR-66-822. 817189

2. C. A. Rosen and N. J. Nilsson (Eds.), "Application of Intelligent Automata to Reconnaissance," Second Interim Report, Contract AF 30(602)-4147, SRI Project 5953, Stanford Research Institute, Menlo Park, California (March 1967), RADC-TR-67-171. 820989

3. C. A. Rosen and N. J. Nilsson (Eds.), "Application of Intelligent Automata to Reconnaissance," Third Interim Report, Contract AF 30 (602)-4147, SRI Project 5953, Stanford Research Institute, Menlo Park, California (December 1967), RADC-TR-67-657. 827938

4. N. Nilsson, B. Raphael, and S. Wahlstrom, "Application of Intelligent Automata to Reconnaissance," Fourth Interim Report, Contract AF 30 (602)-4147, SRI Project 5953, Stanford Research Institute, Menlo Park, California (May 1968).

5. C. A. Rosen and N. J. Nilsson, "An Intelligent Automaton," IEEE International Convention Record, Part 9 (1967).

6. B. Raphael, "Programming a Robot," Proc. IFIP Congress 68, Edinburgh, Scotland (August 1968).

7. G. E. Forsen, "Processing Visual Data with an Automaton Eye," in Pictorial Pattern Recognition (Thompson Book Company, Washington, D.C., 1968).

8. C. Green, "Theorem-Proving by Resolution as a Basis for Question-Answering Systems," Machine Intelligence 4, B. Meltzer and D. Michie, (Eds.), Edinburgh University Press (Edinburgh, Scotland; to appear 1969).

9. L. S. Coles, "An On-Line Question-Answering System with Natural Language and Pictorial Input," Proc. 1968 ACM Conference, Las Vegas, Nevada (August 1968).

10. C. Green and B. Raphael, "The Use of Theorem-Proving Techniques in Question-Answering Systems," Proc. 1968 ACM Conference, Las Vegas, Nevada (August 1968).

11. C. Green and B. Raphael, "Research on Intelligent Question-Answering Systems," Scientific Report No. 1, Contract AF 19(628)-5919 SRI Project 6001, Stanford Research Institute, Menlo Park, California (May 1967).

12. B. Raphael, "Research on Intelligent Question-Answering Systems," Final Report, Contract AF 19(628)-5919, SRI Project 6001, Stanford Research Institute, Menlo Park, California (May 1968).

13. L. G. Roberts, "Machine Perception of Three-Dimensional Solids," Optical and Electro-Optical Information Processing (MIT Press, 1965).

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)*

Stanford Research Institute
Menlo Park, California

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

3. REPORT TITLE

APPLICATION OF INTELLIGENT AUTOMATA TO RECONNAISSANCE

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Final Report 17 March 1966 - 5 December 1968

5. AUTHOR(S) *(First name, middle initial, last name)*

Nils J. Nilsson Charles A. Rosen Bertram Raphael

Leonard J. Chaitin Sven E. Wahlstrom George E. Forsen

6. REPORT DATE

February 1969

7a. TOTAL NO. OF PAGES

84

7b. NO. OF REFS

13

8a. CONTRACT OR GRANT NO.

AF 30(602)-4147

b. PROJECT NO.

4594

c. Task No:

459405

d.

9a. ORIGINATOR'S REPORT NUMBER(S)

SRI Project 5953

9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*

RADC-TR-68-605

10. DISTRIBUTION STATEMENT

This document is subject to special export controls and each transmittal to foreign governments, foreign nationals or representatives thereto may be made only with prior approval of RADC (EMIRD), GAFB, NY 13440.

11. SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY

Rome Air Development Center (EMIRD)
Griffiss Air Force Base, New York 13440

13. ABSTRACT

This report is a final report for the Project "Application of Intelligent Automata to Reconnaissance," under contract AF 30(602)-4147. The primary goal of this project is to investigate techniques in artificial intelligence applied to the control of a mobile automaton in a real environment. The main emphasis is on the design of a hierarchy of computer programs that will accept visual and other sensory information gathered by the automaton and will direct the actions of the automaton in performing missions that require the abilities to plan ahead and to learn from previous experience.

A mobile automaton controlled over a radio link from an SDS-940 computer has been constructed and is operational. It uses a visual system and other sensors combined with a complex-problem-solving system to enable it to navigate among simple objects in a laboratory.

Although this is a final report, another project (under contract F30602-69-C-0056) is using the results reported here to continue research in intelligent automata.

DD FORM 1 NOV 68 1473

UNCLASSIFIED

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Artificial Intelligence						
Pattern Recognition						
Automata						
Modeling						
Visual Perception						