N73-31146

BOLT BERANEK AND NEWMAN INC

CONSULTING · DEVELOPMENT · RESEARCH

BBN Report No. 2615

"ROBOT COMPUTER PROBLEM SOLVING SYSTEM"

Quarterly Progress Report No. 6
For The Period 1 April 1973 to 30 June 1973
Contract No. NASW-2236

E. William Merriam
Joseph D. Becker

15 July 1973

Submitted to:

National Aeronautics and Space Administration
Headquarters Contracts Division
Code DHC-4 / Negotiator: CBSpann
Washington, D.C. 20546

BBN Report No. 2615

"ROBOT COMPUTER PROBLEM SOLVING SYSTEM"

Quarterly Progress Report No. 6
For The Period 1 April 1973 to 30 June 1973
Contract No. NASW-2236

15 July 1973

Submitted to:

National Aeronautics and Space Administration
Headquarters Contracts Division
Code DHC-4 / Negotiator: CBSpann
Washington, D.C. 20546

TABLE OF CONTENTS

INTRODUCTION

The following is a report on progress made on NASA Contract No. NASW-2236 for the period 1 April 1973 to 3Ø June 1973. This contract concerns research, development, and consulting in the areas of:

I. Research and development on a "robot" computer problem-solving system
II. Assistance with the NASA-JPL robot project

The progress made in each of these areas may be summarized as follows:

I. "Robot" Computer Problem-Solving System

(A) Design and implementation of a new "Mars-like" environment for the simulated robot
(B) Implementation of new environmental features, particularly the visual occlusion of objects by other objects
(C) Establishment of movement and vision capabilities of the robot in the new environment, using the LISP/FORTRAN interface for computational efficiency
(D) Development of a complete graphical display and plotting capability for the robot and its environment

II. Assistance with the JPL Robot Project

(A) Continued progress in converting the programming language SAIL to run under the TENEX monitor
(B) Consultation on a number of issues involving the simulation of a Mars-like environment
(C) Demonstration of the Martian-simulation program as described in I. above

These topics are described in the following pages, using the same outline as that given above.

I. The "Robot" Computer Problem-Solving System

A. The New Simulated Environment

In previous work on this contract, our robot simulation has been structured to resemble an automobile-and-driver navigating through the streets of a city. We have now designed and implemented a completely new simulated environment, which represents a robot exploration vehicle on the surface of Mars. The reasons for creating a new simulation were:

1. To make the simulated environment intuitively more similar to that which will confront the hardware robot being developed by our co-workers at JPL,
2. To allow the inclusion of several improvements in the simulation that we had developed (these are discussed in the previous progress report, and in Section I.B of this report), and
3. To demonstrate that the problem-solving capabilities of our system were not dependent on the restricted characteristics of our original city model.

The new environment is shown on the computer-generated plot reproduced on the next page. It represents a section of "Martian" landscape that is nominally 400 feet square. This planar area is partially bordered by the edges of three mountains, and includes within it two hills and four craters (one a double crater). The plane (to which the robot is restricted) is randomly sprinkled with 5Ø rocks, 5Ø "unidentified objects" (which are small objects that resemble each other), and a single unique "instrument package". The rocks appear as little squares in the plot, the "unidentified objects" as little crosses, and the "instrument package" is drawn as a unique symbol which appears to the right of center in the figure. Although the symbols on the plot take up space, the objects represented are considered to be points with no spatial extent.

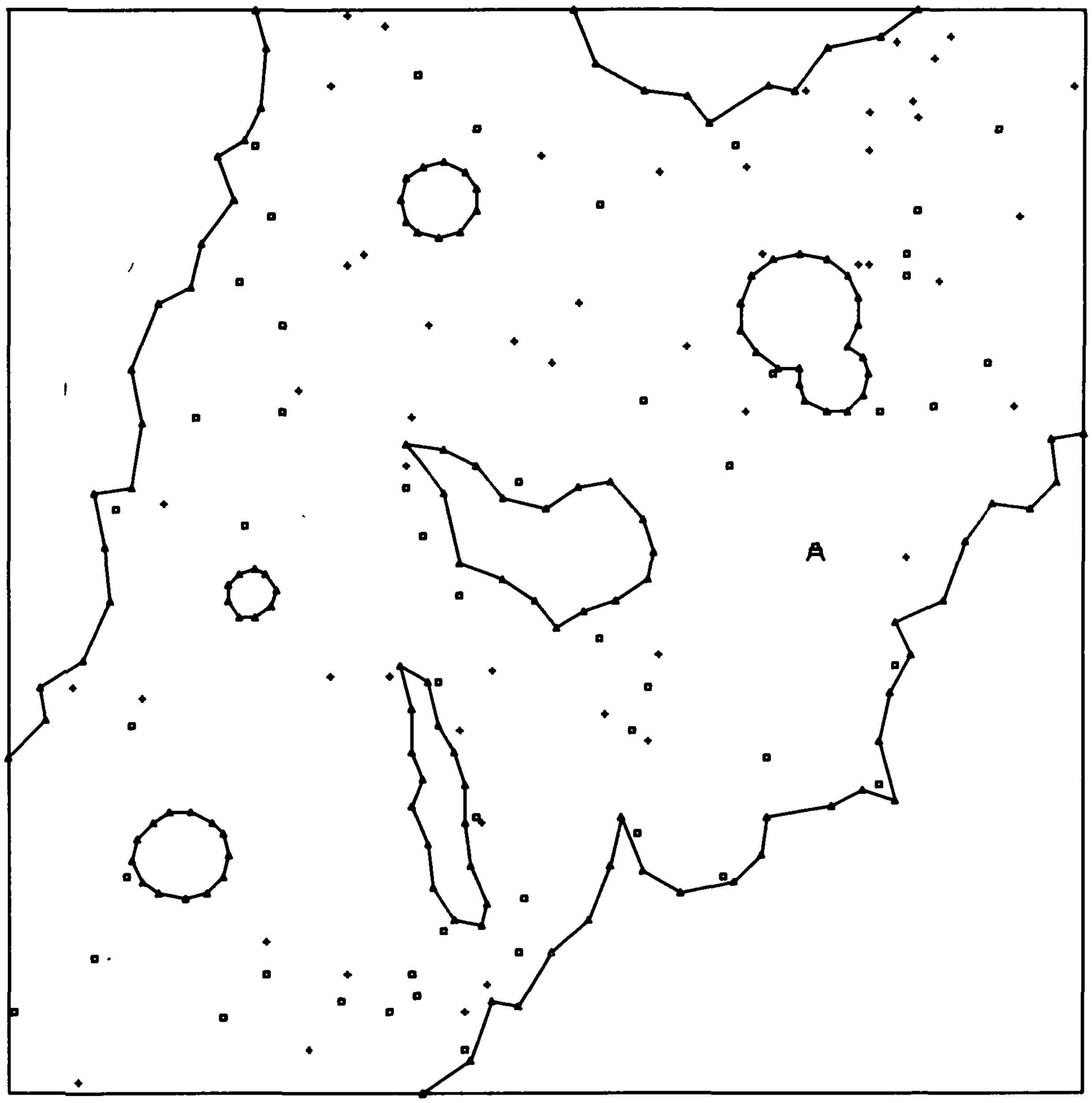

FIGURE 1

This landscape is of course fictitious, but it does resemble lunar terrain, and similar cratered portions of the Martian surface. The mountains, hills, and craters were designed by hand, but the smaller objects were distributed on the basis of computer-generated random coordinates. The "unidentified objects" are provided so that there can be another type of object which is more distinct from rocks than rocks are from each other. The "instrument package" is provided so that we may experiment on the effect of having a unique object in the environment.

The line segments which make up the shapes of the mountains, hills, and craters are _not_ visible to the robot, but the vertices at which these segments meet _are_ visible to it. These points, which we call "terrain feature points" might be thought of as a _sampling_ of the edge of a continuous object, representing points of extremal or of typical curvature, of extremal or of typical color, and so on. In other words, these points represent the ultimate result of discretizing a continuous form, which will probably be necessary in a robot system, and indeed is probably necessary in living perceptual systems.

In our current model, the mountains, hills, and craters do not have a height that enters into their shape as far as the robot is concerned. Basically, it is as though the robot is unable to raise its eye above the horizontal -- a restriction that we have imposed in order to reduce the dimension of the coordination problems that our system must solve in its present version. In spite of this restriction, we have allowed a kind of "peripheral" height perception, in that the robot can see mountains behind hills and craters, and it can see hills behind craters, whereas in all other combinations, a background object is occluded (visually blocked) by an object in front of it (in particular, the rear surfaces of mountains, hills, and craters are occluded by their own front surfaces). This matter of occlusion is important enough to merit a more extended discussion in Section I.B.3.

B. Relationship of the New Environment to the Old

B.1. Operational Improvements

Our new Martian environment is modeled in Cartesian coordinates, unlike the rather cumbersome system of relative coordinates that was used in the city model. Besides its simplicity, both conceptual and computational, the new system removes certain restrictions on the old environment:

* Objects are scattered about a plane, rather than being lined up along the edges of streets,
* The former arbitrary restrictions on the range of the robot's vision have been removed,
* The robot itself will ultimately be able to move freely in the plane, although at present its motion is still restricted to a proscribed path, and
* The robot's path may contain arbitrary curves.

The total number of (point) objects in the world has been raised from 68 to 243 (including terrain feature points), so that the robot's environment is much richer than it was before. Also, we intend to make the sets of visual features assigned to point objects be more varied and challenging to the robot.

B.2. Terrain Objects and Terrain Feature Point Objects

The new world simulation is like the old in that all "objects" perceived by the robot are taken to be spatially localized to a single point. This of course is not a property of the real world, but rather it simulates a property that we believe will be found in the sensory systems of both animals and machines, namely: the continuous information in the world is broken down into discrete information by the sensory system before it is passed to the perceptual (cognitive) system. That is, we hypothesize that even when a human looks at, say, a mountain, he does not gather an infinitude of data, but rather his sensory system automatically extracts a sampled set of data points that are either extremal (e.g. a maximum of curvature, the edge of a shadow) or typical of an area (e.g. a typical point in the middle of an arc of constant curvature or in the middle of an area of constant color). (Figure 2 shows some possible feature points on a curve that might represent the profile of a mountain.) Obviously the density of such sampling is much greater in human vision than in the model of our robot, but hopefully this quantitative difference is not great enough to qualitatively affect the perceptual processes involved or our intuitions of what the robot is seeing.

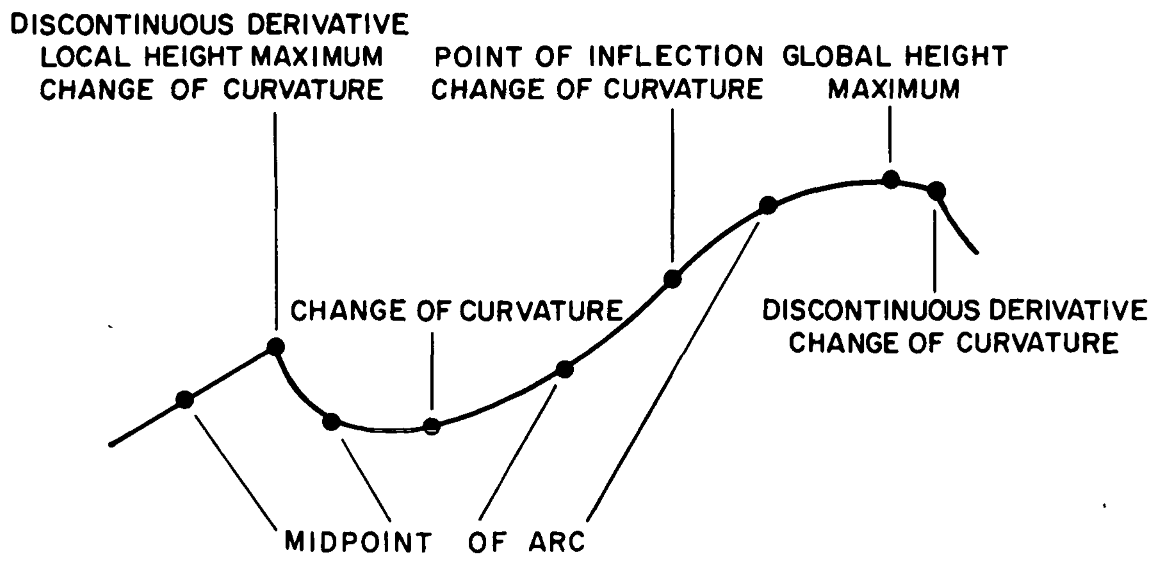

FIGURE 2: Feature Points Along A Continuous Curve

In the new environment, certain groupings of point objects are associated with each other in a way that is obvious to us as human observers, namely those points which form the border of the nine "terrain objects" (3 mountains, 2 hills, and 4 craters). We wish to make clear that this association is <u>by no means obvious to the robot</u>, nor is it pre-programmed into the robot's conceptual system. Rather, it is one of our main research goals to give the robot a system of procedures the gist of which is, "find interesting clumps of point objects and consider whether they could usefully be regarded as entities in themselves".

Such procedures are very likely to find useful groupings of point objects that are <u>not</u> terrain objects, such as the cluster of rocks near the bottom of the world-map shown in Figure 1. In other words, the point objects that we call "terrain feature points" could be grouped into recognizable terrain objects on the basis of clustering principles (proximity, distinctiveness of the grouping, etc.), but such clusterings could also be made of point objects other than terrain feature points.

There is, however, one property of terrain objects that <u>does</u> distinguish them from other groupings, which is that they are visually <u>opaque</u>, meaning that they <u>occlude</u> objects standing behind them, including their own rear sides. This property of occlusion, which is found in the new Martian environment, corresponds to <u>additonal</u> perceptual grouping procedures that must be supplied to the robot's cognitive system. That is, occlusion provides additional evidence for grouping together the edge-points of a crater (besides proximity, etc.), and should lead to an object-concept which is in some sense stronger than the "object" which consists of a cluster of rocks.

We wish to emphasize again that none of these perceptual results are provided explicitly in the sensory data that the robot receives; rather, it is one of our major research goals to precisely specify the processes by which the perceptual notion of object-hood is extracted from sensory data in which it is merely implicit. It will be a considerable triumph when our robot looks around its environment and conceptualizes the same nine terrain objects that we do.

B.3. Visual Occlusion

By allowing opaque objects to occlude each other in our new world simulation, we have of course made the simulation a closer model of physical reality. But the important contribution of this addition is that it adds to the psychological reality that can be included in the robot's problem-solving system. In the previous subsection, we discussed one way in which occlusion interacts with the formation of object-concepts (namely, an object signals its presence by blocking out a segment of the background). There are a number of other relationships between occlusion and perceptual processes. For instance, sometimes an object may appear to be truncated or even split up by an occluding object which lies in front of it. Occlusion greatly affects the process of tracking, in which the robot attempts to keep sight of an object while moving. Indeed, the whole process of navigating around obstacles toward goals is affected by the relationship between the robot's ability to see an object and its ability to move towards it. We will not go into these psychological consequences further at this point, because none of them have been implemented in the robot simulation as of yet.

We had not included occlusion in the old environment, and we had not planned to include it in the new one, primarily because we supposed that it would be difficult to simulate. It turned out to be even more difficult than we expected! Two different algorithms for computing occlusion were developed (i.e., they compute which point objects are visible to the robot given the robot's position, and which are hidden behind some terrain object), with the expenditure of about two man-months of effort, plus much computer time. In view of the important improvements that occlusion brings to the model, we feel justified in the effort that we have expended.

We will very briefly describe the two algorithms that we have developed. In the first, a 360 degree scan is made of the horizon, while bookkeeping procedures keep track of the frontmost face of each object as it is scanned (this can be quite tricky, as the terrain objects have many concavities). During this scan, all point objects visible from the robot's position are determined. Then a second computation is made to see which of those objects happen to fall within the robot's visual field at the moment.

In the second algorithm, a determination is first made of which point objects might fall within the robot's visual field. Then a second computation determines for each object whether it is in fact visible, or whether it is hidden behind some terrain object. This involves bookkeeping different from that used in the first algorithm, and is somewhat less tricky.

Because of this simplification, and because fewer points are examined, the second algorithm runs faster than the first (we are considering a combination of the two which might run faster yet). In both cases, the computation is much speeded by the fact that it is coded in FORTRAN rather than in LISP, and is invoked via the LISP/FORTRAN interface discussed in BBN Report No. 2543.

In general, for a typical visual field, the computation of what the robot can see requires on the order of Ø.6 seconds of CPU time, which is much faster than the corresponding computation ran in the old simulation (coded in LISP), where there were only a fourth as many objects, and occlusion was not taken into account!

B.4. Similarities Between Old and New Environments

So far we have been stressing the beneficial differences between the new world simulation and the old. In closing this section, we should remark on the beneficial <u>similarities</u> between the two. Aside from implementation details and the new role of terrain objects, the spirit and the mechanics of the new simulation are retained from the old: the use of point objects bearing feature lists, the emphasis on visual perception, and indeed the entire functioning of the eye are carried over from the old model. In fact, once the occlusion algorithms had been developed, the old LISP code for the vision routines was adapted to the new model in less than an hour's time, most of which was occupied in trying to find and understand old listings! Thus, in converting from the old to the new world simulations, we have come to appreciate the generality of the old model, while still finding ways to extend and improve it.

C. The Robot

The robot in our new simulation does not differ appreciably from the one which was used in the city environment. In its display representation, the robot at present looks something like a car with tailfins or a speedboat, as is seen in the computer drawn plot - Figure 3. Of course, this external form is created only for display purposes, and it can be changed at will without affecting the behavioral properties of the robot in the slightest. Two main factors went into the design of the robot graphic representation: the ability to distinguish its heading direction from either end (since certain settings of the eye can obscure portions of the robot); and the ability to rotate the figure without introducing excess quantization distortion.

The visual system has been retained exactly as it was in the previous simulation, including the division of the robot's visual field into fifteen subfields. These subfields, or rather their projection onto the landscape, appear as the pie-shaped grille-work to the left of the robot in Figure 3. The point objects which fall within each subfield on the display are just those which are seen within the respective subfields by the robot's sensory system.

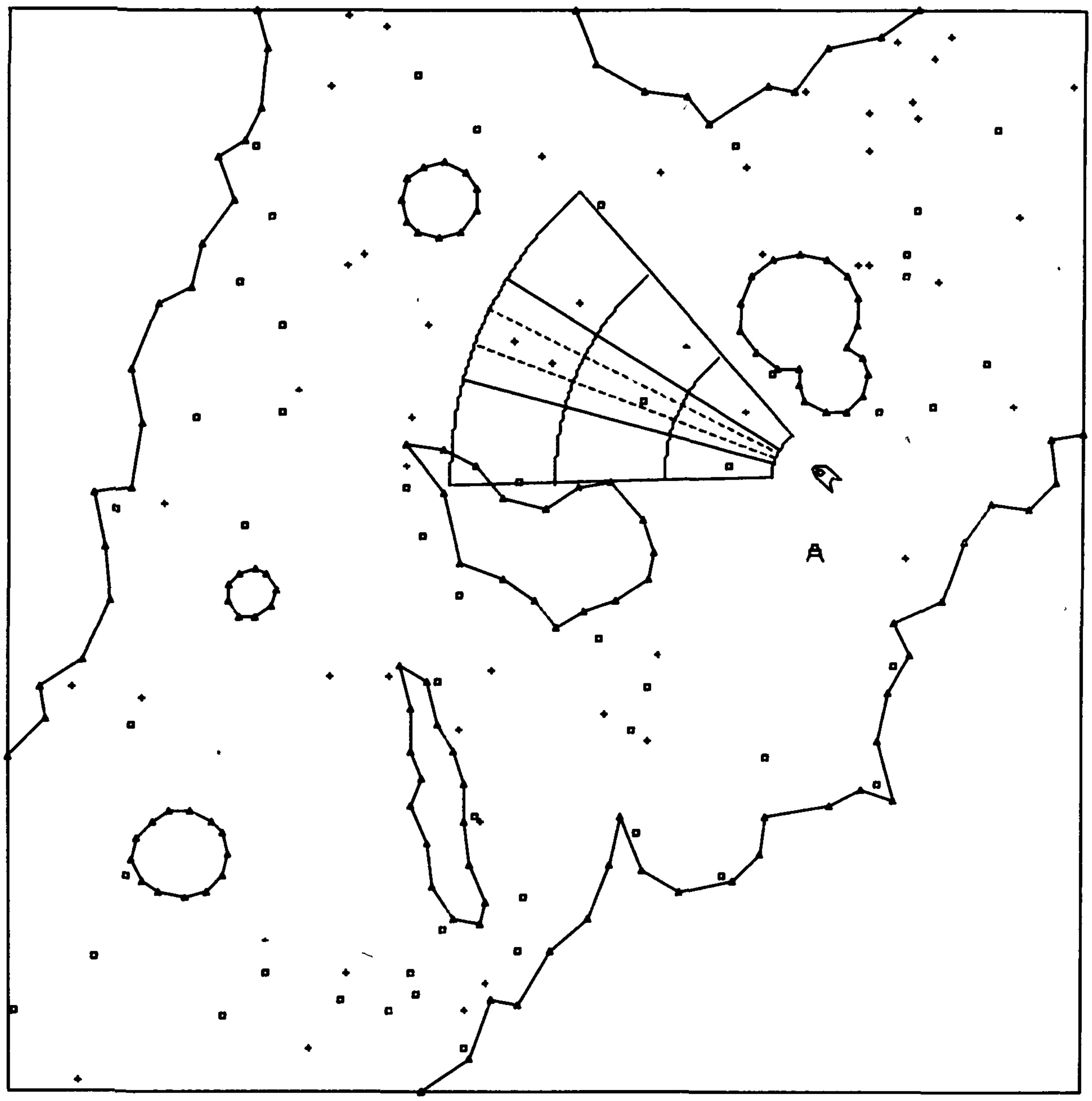

FIGURE 3

At the moment, we have chosen to restrict the robot to a fixed path, consisting of 328 short segments (averaging about seven feet long in scale units) which form an intricate tour of the given terrain. The reason for using a predetermined path at first is simply that the robot does not yet have enough intelligence to set out on its own course; in effect, we are holding it by the hand and guiding it along. Eventually, we do hope to permit the robot to roam freely in its environment, and we are designing our system to allow for that possibility. In the meantime, the fixed path that we have designed contains curves, loops, turn-arounds, double-backs, and so forth, so that it presents a much wider variety of perceptual challenges than did the long linear segments (180 feet long) that were used in the city simulation.

Currently, we have gotten the robot to run all the way around its path while remaining tangent to it, and this activity has been accurately rendered by the display program. The visual system is also working in the new system, and our next task will be to couple the robot's motion to its vision, in the context of the tracking task -- i.e., to keep its eye fixed on a given object while moving along the path. We expect that it will be quite easy to transfer the old tracking routines over into the new simulation.

D. Display

This quarter we redesigned the display program to reflect the change to the Mars-like environment. We wanted the display to be a useful tool in the design and debugging of the robot system. Toward this end, we have implemented the following features:

1) Since there is a large amount of information which is potentially desired, but which is not always needed, options are provided to minimize the clutter on the display screen. For example, the objects in the world can be displayed or turned off by throwing a switch on the IMLAC graphic computer console.

2) Provision is made so that any part of the display can be enlarged.

3) Processing is divided between the IMLAC and TENEX computers so as to attain a balance between maximizing real-time response and minimizing display flicker and blinking. Also, the bandwidth required for communication between the IMLAC and TENEX is held to a minimum for the same reasons.

Our display system is now almost complete and our early design decisions have proved to be good ones. The display is not cluttered except for those brief moments when we wish to display all available information. Response time to local (IMLAC) interactions is real-time and the response time from TENEX interactions is not materially affected by the amount of information transmitted between the machines.

Most important is that the display has already proved to be a tremendous asset in the debugging of the robot system. An example occurred while we were hooking up part of the vision algorithm, when the robot turned its eye to an unexpected place. Because the position of the eye relative to the robot was visible on the display and because we knew where the eye should have been, it was immediately obvious that we had a problem. If we had had only printout information, as previously, it would have taken on the order of several hours to diagnose the same problem once we even realized that something was wrong. With the display, diagnosis was completed in about 2 minutes!

As a concluding remark, we would like to mention that we have spent a lot of time and effort creating this display system. We have made a number of false starts and have had to solve some tricky problems such as drawing accurate arcs and rotating the robot so that it did not distort. Now that we are nearing completion of this graphic tool and have been able to use it, we are happy to report that it has already proved to be a great asset and it is obvious that its development will result in a considerable saving in future debugging and development time.

E. Next Steps

In the next period of work on this contract, we intend to consolidate the simulation that has been described in the preceding pages. This will involve picking up some of the loose ends and details, improving some of the code, and documenting the system. A few more features will be added to the display system, such as the ability to display just those point objects which are visible to the robot at a given time, and the ability to identifv objects to the computer with the aid of a cursor that is attached to the display system, rather than having to type in their names or coordinates in order to get information about them.

As we mentioned in Section I.C, we soon will have transferred over the tracking behavior that was already programmed for the robot in the city environment. Having done this, we intend to set out on a slightly different track, rather than proceeding to transfer the remaining behavioral routines that exist for the old simulation. We intend to spend a good deal of effort trying to specify precisely the kinds of interaction that exist among the following three subsystems of our simulation:

(1) The Environment
(2) The Sensori-Motor System
(3) The Problem-Solving System

We feel that the old simulation was inadequate in the manner in which these subsystems were defined and interrelated. We may go so far as to establish each system as an independent job or "fork" within the TENEX executive system, and we are even considering running the three subsystems in real simultaneity, using the distributed computation facilities provided by the ARPA Network.

In any case, we will be probing into the question of the temporal coordination of several processes which, as we indicated in last year's Final Progress Report, constitutes one of the major categories of problems that are currently hindering progress in the development of robot intelligence.

II. Assistance with the NASA-JPL Robot Project

A. Conversion of SAIL to TENEX

During the past quarter, most of our effort on converting SAIL to run on the TENEX monitor was invested in debugging and tightening up the runtime support package which SAIL user programs map into their address space to complement the compiler output. This package is commonly referred to as "the Segment".

During this quarter, we have also been doing an extensive amount of testing by running several "benchmark" programs which are known to work in DECUS SAIL and at Stanford. Some bugs still remain, but we have nevertheless successfully run a variety of programs: PTRAN and RTRAN, part of the compiler bootstrap; MONKEY, a theorem-proving demonstration program which uses coroutines extensively; IFN, a program for editing out unwanted assembly switches from FAIL (assembly language) source files, which makes heavy use of IOSER, the largest block of TENEX-specific code in the Segment; and most important of all, TEST, a program from Stanford for checking out a SAIL system.

One issue that we have been concerned with is how to make LEAP (an associative data base facility embedded in SAIL) make effective use of the TENEX features. It looked relatively simple to make the Stanford LEAP work on TENEX, but this LEAP was designed for a swapping monitor, not a more sophisticated demand-paged monitor such as TENEX. An important consideration is that the number of LEAP items is limited in Stanford LEAP to 4K by a 12-bit address field in the basic pointer representation. This assumes a core restriction which does not exist in TENEX and 4K is entirely too small a number for realistic applications such as robot "world model" data bases. To raise this limit requires changing the internal pointer representation, which is so fundamental and low-level that it implies writing a whole new LEAP. The SAIL group at Case Western Reserve University has recognized this need and has embarked on a large project to write a TENEX LEAP that will provide a very large data base kept on disk, using TENEX file/fork page mapping to bring data into the user's address space as needed. This will result in a LEAP which can be used to solve problems involving very large data bases, yet the user with a small LEAP data base will not be penalized. Case is planning their LEAP around the Stanford version of SAIL, but they have expressed interest in using TENEX SAIL as soon as our work is complete. We believe this LEAP will adequately serve the needs of the JPL Robot Project, which requires a large data base and reasonable retrieval times.

The past quarter has also included work which will enable the SAIL compiler to run without the 10/50 compatibility package and which will compile calls to new runtime support routines for TENEX. We hope to complete the compiler work during the next quarter. In the meantime, the old compiler, running in compatibility mode, is available.

B. Consultation

During this quarter we continued in our role of consultants on issues relating to the use of the IMLAC, TENEX, and the ARPA Network. Additionally, we consulted on issues relating to the computer simulation of a Mars-like environment and robot. This simulation is to be used by JPL in developing the robot system until robot hardware is available to provide real-world input. As a result of these discussions, it was determined that the simulation under development by BBN (described above) meets the needs of JPL. It was also determined that several minor modifications will be necessary to fit the display program into JPL's IMLAC computer and to simulate a simpler "eye".

C. Demonstrations

A modified version of the IMLAC display code for the Mars-like environment was created and run at JPL via the ARPA Network. By operating in LINKed mode (the IMLAC's at BBN and at JPL tied together via TENEX and the ARPA Network in such a way that Teletype output appearing on one also appears on the other), we were able to control the movement of the simulated robot by means of a simple control program written in LISP and FORTRAN.

This demonstration also provided a means by which JPL could evaluate our robot simulation to determine if it would fit their needs.

As a side benefit of performing this demonstration, we learned how to operate effectively in the LINKed mode and determined some system improvements that will make such interactions simpler in the future. These improvements are currently being implemented in the TENEX monitor.