



ANRCP-1999-18
April 1999

Amarillo National Resource Center for Plutonium

A Higher Education Consortium of The Texas A&M University System,
Texas Tech University, and The University of Texas System

Remote Inspection System for Hazardous Sites

Jarrett Redd, Christoph Borst, Richard A. Volz, and Louis J. Everett
Computer Science Department
Texas A&M University

This report was prepared with the support of the U.S. Department of Energy (DOE) Cooperative Agreement No. DE-FC04-95AL85832.

However, any opinions, findings, conclusions, or recommendations expressed herein are those of the author(s) and do not necessarily reflect the views of DOE. This work was conducted through the Amarillo National Resource Center for Plutonium.

Edited by

Angela L. Woods
Technical Editor

600 South Tyler • Suite 800 • Amarillo, TX 79101
(806) 376-5533 • Fax: (806) 376-5561
<http://www.pu.org>

This page intentionally left blank.

AMARILLO NATIONAL RESOURCE CENTER FOR PLUTONIUM/
A HIGHER EDUCATION CONSORTIUM

A Report on

Remote Inspection System for Hazardous Sites

Jarrett Redd, Christoph Borst, Richard A. Volz, and Louis J. Everett
Department of Computer Science
Texas A&M University
College Station, Texas 77843

Submitted for publication to

ANRC Nuclear Program

April 1999

This page intentionally left blank.

Remote Inspection System for Hazardous Sites

Jarrett Redd, Christoph Borst, Richard A. Volz, and Louis J. Everett
Department of Computer Science

Abstract

Long term storage of special nuclear materials poses a number of problems. One of these is a need to inspect the items being stored from time to time. Yet the environment is hostile to man, with significant radiation exposure resulting from prolonged presence in the storage facility. This paper describes research to provide a remote inspection capability, which could lead to eliminating the need for humans to enter a nuclear storage facility.

In general, RI systems can be useful in three types of situations. First, they are very applicable for use in hazardous environments in that they can reduce or eliminate the need for a human to enter such an environment. Second, RI systems can be effectively utilized for areas in which it is necessary to maintain a high level of security. Thus, if the system reduces the need for human presence, it also reduces the security risk associated with that presence. Again, as an example, nuclear storage facilities containing weapons grade plutonium require a very high level of security. Keeping workers out of such an environment reduces the chance of an event in which material is lost due to theft. Finally, RI systems have the capability of allowing humans to explore areas, which would otherwise be impossible to explore. For

example, in other kinds of applications, an endoscopic RI system can allow a doctor to see and travel around inside the body of his patient. The unmanned Viking spacecraft, which flew to Mars, is another example.

While there are many ways in which an RI system might be created, this paper describes the development of a prototype remote inspection system, which utilizes virtual reality technology along with robotics. The purpose of this system is to allow the operator to establish a safe and realistic telepresence in a remote environment. In addition, it was desired that the user interface for the system be as intuitive to use as possible, thus eliminating the need for extensive training. The goal of this system is to provide a robotic platform with two cameras, which are capable of providing accurate and reliable stereographic images of the remote environment. One application for the system is that it might be driven down the corridors of a nuclear storage facility and utilized to inspect the drums inside, all without the need for physical human presence. Thus, it is not a true virtual reality system providing simulated graphics, but rather an augmented reality system, which performs remote inspection of an existing, real environment.

This page intentionally left blank.

TABLE OF CONTENTS

1. SYSTEM DESCRIPTION	1
1.1 <i>The Remote Portion</i>	1
1.2 <i>The Operation Portion</i>	1
2. BACKGROUND	3
2.1 <i>The Human Eye and Stereographic Imaging</i>	3
2.2 <i>The SGI Onyx, Sirius Video, Multi-Channel Option, and Reality Engine</i>	3
2.3 <i>Distributed Environments and InterAgent</i>	4
2.4 <i>Modularized Communications via InterAgent</i>	5
2.5 <i>Background – A Prior System for Telerobotic Control</i>	5
2.6 <i>System Architecture</i>	6
3. SYSTEM DEVELOPMENT	9
3.1 <i>The Flock of Birds</i>	9
3.2 <i>Flock Study</i>	11
3.3 <i>The VR4 Helmet and Multi-Channel Option</i>	12
3.4 <i>Stereo Graphics</i>	14
3.5 <i>The VC-C1 Communications Camera</i>	15
3.6 <i>The Sirius Video Device</i>	18
3.7 <i>LASER-Based Convergence</i>	20
3.8 <i>The Ludlum Geiger Meter</i>	23
3.9 <i>Overlay Graphics</i>	23
3.10 <i>Digital Video and the Sirius Video Device</i>	25
3.11 <i>Eye to Camera Transformations</i>	27
4. SYSTEM INSTALLATION AND REPAIR	31
4.1 <i>System Installation</i>	31
4.2 <i>System Operation</i>	33
Appendix A	A-1
Appendix B	B-1

This page intentionally left blank.

LIST OF FIGURES

Figure 1: Remote Inspection System Architecture.....	8
Figure 2: Camera Mounting Configurations.....	17
Figure 3: Current Sirius Video Pipeline Configuration	20
Figure 4: LASER Convergence System Configuration	21
Figure 5: Sample Remote Inspection Image with Overlay Graphics	24
Figure 6: New Sirius Video Pipeline Configuration.....	26
Figure 7: Fully Installed System in Operation	33

This page intentionally left blank.

1. SYSTEM DESCRIPTION

This prototype remote inspection system consists of two main portions. The operator portion includes the controls and displays for relaying visual information to the user. It is located in the local, or safe, environment. The remote portion includes the robotic manipulator, cameras, and sensors for executing the remote inspection task. It is located in the remote, or hazardous, environment. The system has been designed such that these two environments can be in close proximity to one another or many miles apart depending upon the setup necessary for a given task and/or type of environment. In this prototype, all data between the two sites is transmitted across a network, except for the video images, which are directly transferred by cables. It is possible, however, to compress the images and send them across the network also, but a significant performance loss might occur, depending upon the speed of the network connection and the distance between the two sites.

1.1 The Remote Portion

The remote part of the system is located within the hazardous or secure environment, which is to be inspected. Thus, it is vulnerable to effects from the environment and, in actual operation, would have to be appropriately protected. For example, in radioactive areas, the equipment should be rad-hardened to protect it from damage. Similarly, in secure environments, the equipment might have to be protected against malicious tampering, etc.

For this prototype, the remote portion consists of a Merlin® robotic arm manufactured by the American Robot Corporation. It is a six-degree of freedom arm, which can be controlled by a computer across an RS-232C serial connection. The Merlin is not a mobile robot, nor can it easily be converted into one. It was designed for mounted industrial applications, but is

acceptable for the research environment for which this prototype was created, as the basic principles can be developed and demonstrated. Thus, the actions of this system are constrained to the motion and navigation through the environment of which the Merlin is capable. The robot, in this case, is only required to manipulate the cameras to inspect the surrounding area. This is sufficient for the prototype demonstration.

On the robotic arm are mounted two Canon VC-C1™ communication cameras. These cameras are capable of providing broadcast quality image frames at 30 Hz, in either S-Video or Composite format. The cameras are auto-focus and have pan/tilt capabilities. They can also be computers controlled via RS-232C serial connections. The purpose of the cameras is to provide stereographic, or 3D, images to the operator.

1.2 The Operator Portion

The local part of the system is located within the control room or operator's safe environment. Here, a Silicon Graphics Onyx® computer is utilized for processing the images from the VC-C1 cameras and presenting them to the user, who wears a VR4 head-mounted display (HMD) helmet made by Virtual Research Systems, Inc. The motions of the user's head are then tracked by an Ascension Flock of Birds™ six degrees of freedom (DOF) tracker. The tracker is utilized to determine the position of the user's head and thus determine where the user is looking. This position and orientation information is then transformed and sent to the robotic arm. When the robotic arm moves the cameras to the new configuration, the user sees images that are essentially the same as she/he would have seen had they actually been present in the remote environment. This method of control is also very intuitive to learn and requires little training.

This page intentionally left blank.

2. BACKGROUND

2.1 *The Human Eye and Stereographic Imaging*

The most difficult challenge to meet during the design of this system was providing the necessary actions and adjustments in order to obtain good quality stereographic imaging without the distortion and disorientation normally associated with other 3D systems. The human eye is one of the most sensitive instruments known to man. It is capable of delivering images of incredible sharpness, clarity, and accuracy in even the most demanding environments. To this date, no camera system can come close to duplicating what the eye does naturally. As such, there are certain required things that a RI system must provide in order to satisfy the human eye. For example, the images provided to the user must not flicker nor drop below an approximate minimum frame rate of around 30 Hz. The camera must also not jump around nor move faster than the human eye can follow. It also should not lag behind the eye. Adverse effects such as these, if present, result in eye-strain, disorientation, and sometimes dizziness and nausea for the user. Although it is possible for the human eye to adapt to some of these time-delay effects, it is better if they can be removed from the system to begin with.

The head-mounted display worn by the user introduces additional problems to be overcome. For example, while wearing the HMD, the user's eyes maintain a fixed focus on the projectors of the helmet. These projectors provide the left and right image to each eye, and sit approximately 2 inches in front of each eye. Thus, the user's focal plane is continuously fixed at 2 inches, regardless of how far away the images in the helmet appear to be. This effect can be startling at first, but is quickly accepted by most HMD users. However, this effect imposes a constraint on the cameras used in this system. Since the

users do not change their depth of focus, the cameras must provide the focus for them. That is, the cameras must have auto-focusing capabilities.

Another problem introduced by the HMD is that of convergence. During normal conditions, the left and right eyes perform synchronized convergence to point directly at the object being viewed. As an object gets closer or further away, the convergence angle changes. When an object gets very close to the person, their eyes converge inward sharply, producing the commonly known cross-eyed appearance. However, when wearing an HMD, the user's eyes remain in a fixed, straight-ahead orientation as they observe the projectors. Since the user cannot converge, the cameras must provide convergence for them. That is to say, as objects move toward and away from the cameras, they must synchronously converge to point at the objects. Thus the cameras must have panning capabilities and be able to detect objects and ranges in some fashion.

Lastly, the distance between the left and right eye varies from person to person. This distance, known as the inter-pupillary distance (IPD), must be accounted for by the cameras and the helmet. The projectors inside the helmet must be adjustable such that they can be moved closer together or further apart, as required, in order to match exactly with the user's IPD. Similarly, the cameras providing the images should be placed the correct distance apart. Providing images from an incorrect IPD can cause significant eyestrain and dizziness.

2.2 *The SGI Onyx, Sirius Video, Multi-Channel Option, and Reality Engine*

The time-delay effects discussed in the previous section can generally be prevented by the use of fast cameras and fast video capturing and processing hardware. The VC-C1 camera's frame rate of 30 Hz is sufficiently fast for it to keep up one end of

this speed requirement. The other end is kept up by the Sirius™ Video device, available as an option with SGI Onyx computer systems. It is capable of receiving and transmitting multiple channels of video among various source and destination devices. Specifically, it is capable of capturing the two images from the VC-C1 cameras and processing those images while maintaining the frame rate.

However, the images must also be output to the user and the HMD in parallel, synchronized, and without a loss of frame rate. This task is accomplished by the Multi-Channel Option, another add-on component to the SGI Onyx. This device has six channels with dedicated VRAM buffers that are capable of simultaneously outputting six different images in various formats.

Lastly, since the user is wearing a HMD and cannot see outside of it, interaction with typical computer monitors is not practical. Thus, some form of heads up display must be generated, placed over the video images, and output to the helmet in real-time and without affecting the frame rate. This is accomplished by the Reality Engine™ graphics sub-system inside the SGI Onyx. With two R4000 CPUs, dedicated graphics pipelines, and a five-fold rasterizing system, the Reality Engine easily meets this requirement.

2.3 Distributed Environments and InterAgent

Since this RI system was developed as a research tool, it is designed to be easily extendible and modifiable. For example, it is built completely within a distributed telerobotic environment in which computers at many sites can connect and disconnect at will, interacting with the system or merely observing. Since all communications with the remote system take place over the network, it follows that any one of a number of operator environments could be in control while the others observe.

The task of coordinating all this network activity falls to InterAgent, a relatively comprehensive tool kit written by Modulus Technologies, Inc. to assist in building distributed systems. The InterAgent library provides the platform-dependent socket level communications necessary to exchange information among the connected systems on the network. Thus, the integration of various sites becomes platform-independent. Interagent achieves this by establishing a common set of objects among sites. An InterAgent object consists of a name, an ID number, and an associated user-defined structure into which data is inserted. Once an object type has been created, instances can then be generated by a process, filled with useful data that is to be shared, and sent to the network. Other processes that are interested in receiving this data would then register an interest in this particular object ID. Objects sent to the network are then directed to, and only to, interested processes.

InterAgent also provides a router for directing the flow of information from the processes generating the data to the processes interested in the data. This is utilized by two or more processes on separate machines, which need to communicate with one another. For example, suppose a site needs to utilize sensor readings taken at another site. Data readings from the sensor at one site are taken by a computer at that site. They are then placed into InterAgent objects and transmitted by the router over the network to the other site. The data is then extracted from the objects and utilized.

Lastly, InterAgent is capable of establishing several virtual network topologies such as a fully connected network, star network, ring network, etc. This is accomplished at the socket level by the InterAgent router upon startup. Connections can be established by the operator as needed for the task at hand. For example, machines at three different locations can be

interconnected to form a virtual ring network for InterAgent object distribution. The network is virtual in the sense that it does not physically exist. Instead, it is built upon the existing physical Ethernet connections between the systems by allocating time slices on various routers. That is to say, the physical network connections between machines may differ from the virtual connections established by InterAgent. Communications can also be established between systems that are not directly connected on the virtual network. It does this by relaying data through one or many intermediate connections until it reaches the destination.

2.4 Modularized Communications via InterAgent

The prototype in this paper utilizes the benefits of a distributed, modular architecture. For example, many distributed architectures have the ability to task share between systems, thus allowing more to be done within the same amount of time by more effectively utilizing idle processor cycles. In this case, simple tasks such as sending RS-232C robot and camera control commands are deferred to a PC, while the Onyx is heavily loaded down with image processing and graphics tasks, as well as with the task of integrating the data from the various portions of this system.

Distributed systems also have the ability to take advantage of positive aspects of various platforms and operating systems while side-stepping certain shortcomings and weaknesses of other systems. In this case, the Onyx computer is fully utilized for its ability to rapidly process video image and graphics, whereas the PC would not be able to accomplish this task at a speed sufficient for this system. However, the Onyx runs the IRIX operating system which cannot guarantee task completion within a given time period. For time critical tasks, such as

sending robot and camera control commands, the PC takes advantage of the real-time Lynx operating system which can guarantee task completion by a certain deadline.

2.5 Background: A Prior System for Telerobotic Control

To provide telerobotic control and communications between any number of various sites for prior funded work, a modular and distributed system has been created which heavily utilizes the capabilities of InterAgent. This system makes an ideal foundation for the prototype remote inspection system since it provides the telerobotics and communications capabilities necessary for operation over the distance between the remote and operator portions.

The telerobotic system utilizes graphical display software with a graphical user interface to allow telerobotic 6 DOF control of multiple devices across a network. Any active device that has been brought on-line and connected to the system can be controlled from any connected site. The desired device can simply be selected from a pull-down menu and then controlled via any number of controlling devices. Currently, a 6 DOF track ball and 6 DOF flock of birds trackers are available.

To handle the inevitable time delays associated with operating at a distance and across a busy network, a graphical picture of the device is generated on the screen and would be used as a monitoring tool while controlling the device. The graphical picture consists of two superimposed images. First, a solid frame image is drawn, indicating the desired position of the device that the operator wants to obtain. Then, a wireframe image is drawn, representing the current physical location of the device. This position of this image is independent from the previous one and is determined from where the robot actually reports that it is located. Thus, an operator at any location and subjected to any

amount of delay can still accurately control the device by simply driving the solid frame image to the desired location while watching the wire frame image as it updates the position of the actual device as it responds to the operator's commands. The solid frame updates are always immediate while the wireframe updates are delayed by the network transmission time, possibly several seconds. For sites with little to no network delay, the updates are frequent enough to show the motion of the device in real-time.

2.6 System Architecture

To provide the additional communications and control systems necessary for the remote inspection task, the system described in the previous section has been significantly modified and extended. The additional communications and control architecture necessary to achieve RI capability is described below, and a diagram of the system architecture is available in Figure 1. The description is presented as a walk-through of one complete control loop of the system shown in Figure 1.

Operation begins with the robot in a fixed position and the user wearing the 3-D helmet, with a 6 DOF bird sensor mounted to the helmet. First, the user moves his/her head to look in a new direction. An updated 6 DOF reading is then taken from the head tracker that is connected to the Onyx computer by an RS232-C serial port. An InterAgent object called *flock_data* is created, and the 6 DOF reading is transformed into the robot's coordinate frame and inserted into this object. Here it is checked to see if the user-requested position and orientation lie within the bounds of the safe robot workspace. If all six parameters are found to be safe, the object is accepted as is. If any parameter value falls outside of the safe workspace, the value of that particular parameter is changed to the previous safe value that was already sent to the robot. Thus the system continues to

follow the user as closely as it can while still remaining within the safe workspace.

After the safety checks have been performed, the object is transmitted across the network via InterAgent and received by a PC controlling the robot. The local name of this PC is Kiwi. RCCL_Remote, the program directly responsible for robot control, then transforms the reading into a set of robot joint angles via inverse kinematics. A second safety check is then performed to verify that this set of joint angles lies within the valid robot workspace. If the position is not valid, it is ignored. Otherwise, the data is sent to the robot via RS232-C by one of two available serial ports on Kiwi, and an Interagent Object called *site_tree_update* object is created and supplied with the new robot position and orientation data. This *site_tree_update* object is echoed to all interested parties so that graphical models depicting the robot, if any, may be updated to show the current requested position of the robot. This update affects only the position and orientation of the solid frame image (user-requested robot position) discussed in section S.5 above.

Simultaneously, the robot then begins to update its position to match the transform data, generating *site_tree_update* objects as it goes. These objects are also received by the graphical models and update only the wireframe model (actual robot position) as discussed in Section 2.5.

Asynchronously with the above process, the VC-C1 cameras on the robot send images to the Onyx computer at the rate of 30 Hz. These images are then translated into digital format and captured in video memory by the Sirius Video device, described in Section 2.2. The images are then processed. If this processing indicates that the robot has moved, the distance from the cameras to the object being viewed is measured using a LASER range-finding system which is described in Section 3.7. Once the distance to the object is determined, the necessary

convergence angle and focal length from which to correctly stereoscopically view the object is calculated. Another Interagent object called *vcc1_cam_data* is then created and filled with the necessary convergence and focal data. This object is also sent to Kiwi via InterAgent. The other available serial port on Kiwi is then utilized to communicate with both cameras via RS232-C, driving them to the proper convergence angle and setting the proper focal length.

Also asynchronous to the other two processes, radiation exposure rate information is measured from a Ludlum Model 3 Geiger meter. Since this prototype system was created for research involving radioactive environments, the meter is used to provide dosage information that indicates the amount of radiation that the remote equipment is receiving. This tool integration shows the flexibility of this design. A number of tools and instruments could be carried by the robot into the hazardous environment. To obtain the reading, the analog voltage signal from the Geiger meter is sampled and converted to a

raw digital value by a Motorola HC6811 microcontroller. This value is then sent via RS232-C to another PC (whose local name is Grape) which also runs the Lynx operating system. Another object called *geiger_data* is then created, filled with the data, and sent via InterAgent to the Onyx.

The next step in the control loop for this system involves the generation of overlay graphics that function as a heads up display (HUD) for the user wearing the HMD helmet. Since the helmet effectively eliminates vision and traditional interaction with the outside world, a HUD is necessary in order to communicate information to the user. Here, radiation dosage rate information from the Geiger meter, robot position data, and elapsed time are utilized to build text displays and graphical meters to indicate system status. Warning messages can also be displayed to alert the user of significant events.

In the last step, the camera images with their overlay graphics are sent to the HMD helmet through the Multi-Channel Option that was discussed in Section 2.2.

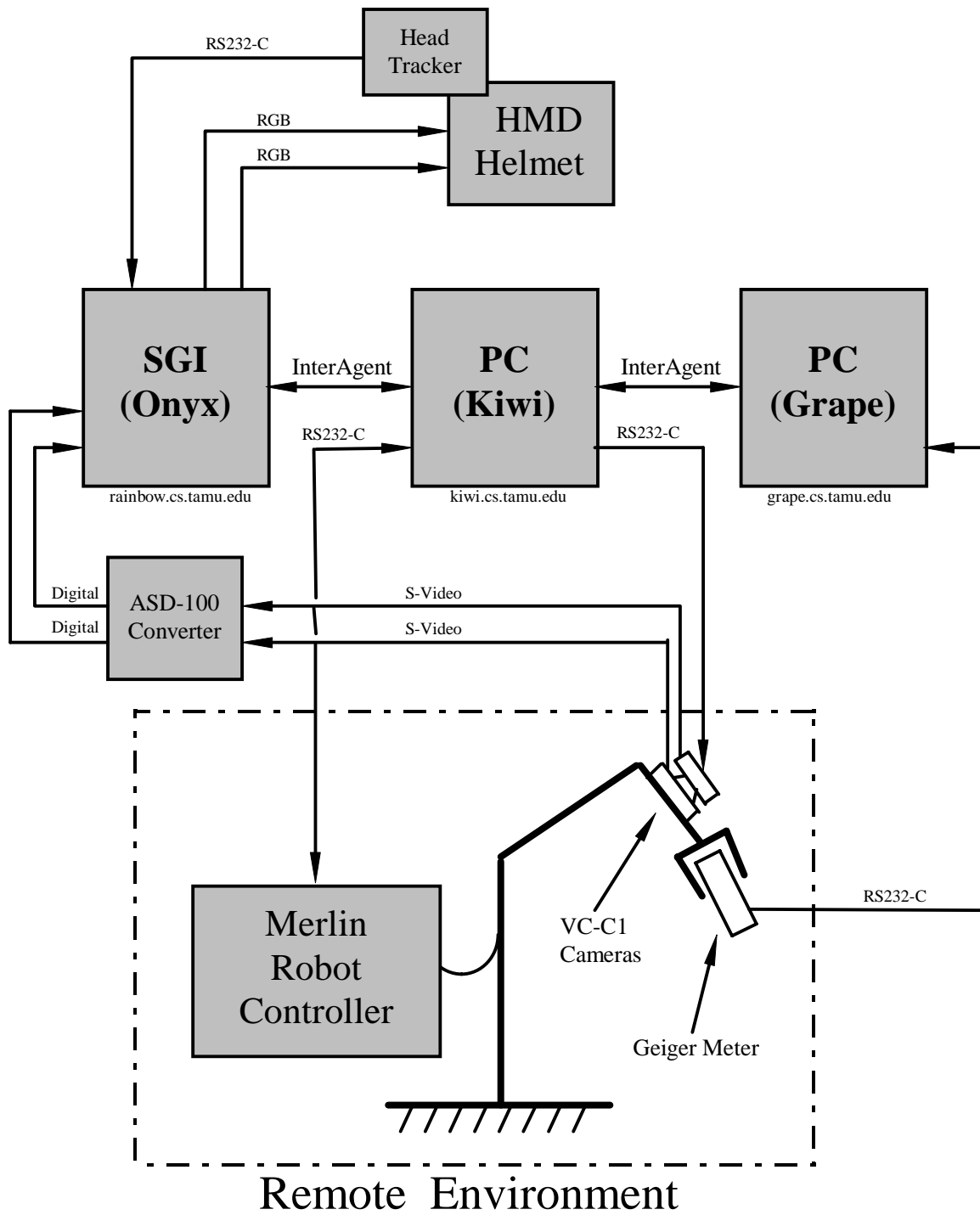


Figure 1: Remote Inspection System Architecture

3. SYSTEM DEVELOPMENT

The first subsystem to be integrated was the flock of birds trackers. An in depth study was done on these trackers to determine the amount of interference and learn how to stabilize the data coming from these sensors. It was found that if the birds are kept at least three feet from other sizeable metal objects, the interference is minimal and the accuracy of the devices acceptable. Next came the HMD helmet, followed by the Multi-Channel Option. At this point, a simple 3D graphics system was created, simulating the flight of a bird through a forest of trees. The user had the capability to look around within the environment along with using head motions to dive and bank. The knowledge gained from building this system provided the platform upon which the 3D viewing system of the current prototype was built.

The second stage in development involved integrating the VC-C1 cameras and the Sirius Video device into the system. To help develop knowledge of this technology, a web-controllable pan/tilt camera system was created. This system was made available to the outside world and can also be utilized to locally display and control video images of a remote location, for example the robotics lab at Texas A&M University.

The third stage involved developing a range-finding system to allow the two cameras to focus and converge upon a specific object within the field of view. A sonar-based device was first examined and rejected. This type of device is not appropriate for this application because the signal detects too broad a range of the environment. A more focused technology was required. Thus, the second and final system attempted was LASER-based. This method utilizes the parallax and scaling affects of the cameras to determine distance by processing the camera images and detecting the position of the LASER beam (described below in Section 3.7).

The fourth stage in development involved integrating the Merlin robot into the system. The flock of birds tracker on the HMD helmet was given control of the robot, and a set of protective position and velocity limits, as described in Section 2.6, was installed.

The fifth stage involved producing the overlay graphics software and integrating the Geiger meter into the system. The graphics system was designed to be a transparent heads-up-display to be overlaid upon the camera images without obstructing the user's view of the remote environment. The graphics consists of six panels that contain graphics meters and text messages relaying radiation dosage information along with robot position and time and date.

3.1 *The Flock of Birds*

The 6 DOF position and orientation "Flock of Birds" sensors are based upon the generation of 3 orthogonal EMF fields, one projected along each X, Y, and Z axis. The transmission of these fields is accomplished by an extended range transmitter (ERT) and controlled by an extended range controller (ERC). The ERT is a large black box that is static in the environment and serves as the origin of the sensor coordinate frame. All sensor measurements are relative to the location of the ERT. The ERC periodically commands the ERT to send out EMF pulses along the three projected fields.

The actual flock of birds sensor consists of three wire coils housed inside a small, lightweight plastic cube which can easily be mounted on an HMD helmet, a person's hand, etc. The pulsing EMF fields sent out by the ERT induce a small current in each of the three coils. These currents are then used to calculate the 6 DOF position and orientation of the sensor.

Multiple sensors can be connected together in daisy-chain fashion to a single RS232-C serial port. However, if multiple

ports are available, then they can also be setup with a dedicated port for each sensor.

Currently, four birds are connected utilizing the daisy-chain method on COM3 (/dev/ttyd3) of the Onyx at 9600 baud, no parity, 8 data bits, and 1 stop bit. Data from each sensor is delivered to the port by relaying it down the chain and through the other sensors until it reaches the ERC which, in this configuration, acts as the master device on the daisy-chain. Although the measurements are being continuously taken by the sensors, the data is only sent to the port upon receipt of a request command.

Serial port communications are established via a typical byte-string command format. The driving application will send one or more bytes to the birds indicating the desired setting or action. The birds will then respond with one or more bytes of status or data. There are various types of available commands, including those which place the birds into various output modes and request data to be sent. For example, there are four data output formats available: position, orientation, transform, and quaternion. Depending upon the data output format, the length of the byte-string response from the birds can vary. For example, in position output mode, only 9 bytes are sent. However, in transform mode, 27 bytes are sent.

Device status is indicated at all times by a red LED light on the outside of each unit. This light remains off while the birds are idle and becomes solid red when active. It will also flash codes to indicate any errors that may have occurred. More detailed information on these sensors is available in the Flock of Birds Reference Manual, along with sample code available in ~/src/test_bird on the Onyx USARC account.

As a convenience, a flock of birds library was written as part of this project. The source and object files for this library are located with the sample code as indicated above. As an example of how one might

utilize the sensor system, a typical session with the birds would be conducted as follows:

1. Connect each bird to the daisy chain and adjust jumpers and dip switches to set a unique ID for each bird. Also terminate the ends of the daisy chain bus. Detailed information on this step should be obtained from the reference manual.
2. Turn on the ERC. Red LED light flashes 5 times, indicating good status.
3. Turn on each bird, each time watching for 5 flashes of the red LED light.
4. Utilize the flock of birds library (~/src/test_bird/fob_lib.o) as indicated in step 5 below.
5. A typical code example is shown below.

```
#define FLOCK_SIZE n          /*
number of birds */
go_flock();                  /*
activate ERT */
flock_read_matrix();        /*
get one record */
sleep_flock();              /*
deactivate ERT */
```
6. Turn off each bird.
7. Turn off the ERC.

The library of routines also includes code to reduce jitter by averaging some number of incoming values. Filtering is also implemented by giving each new value some percentage of effect upon the new commanded position. Successively previous values are also given successively lesser weights, thus stabilizing the data significantly without affecting the accuracy. This method was borrowed from the filtering method utilized in the JR3 force/torque sensor

manufactured by JR3, Inc. The amount of jitter control and filtering is adjustable in the code.

3.2 *Flock Study*

During initial integration of the flock of birds sensors, significant data reliability problems were noticed upon analyzing the data obtained from the sensors. There was a very large level of steady state error as well as an apparent shearing effect that produced readings in which the reference coordinate frame was not orthogonal.

In an effort to combat these issues, a detailed study was conducted on the sensors to determine what was causing the problems. Three types of tests were performed.

1. **Steady-State Error Testing** - Each sensor was placed in a static location in a static environment. Readings were taken over a long period of time and no drift in reading was noted. Subsequently, various objects which are typically found in a virtual reality lab or computer lab environment were introduced into the environment and the effect observed. Not surprisingly, it was discovered that large metal objects were capable of significantly distorting the EMF fields emitted by the device, thus causing large and unpredictable errors and noise. It was also discovered that CRT monitors and other powerful electronics devices also produced interference when their own EMF fields entered the sensor environment. In summary, it was found that, typically, metal objects such as chairs and tables must be kept at least 3 feet away from the sensors and at least 5 feet away from the transmitter in order to obtain reliable data. In addition, the metal objects must remain out of the direct line of sight between the ERT and sensor. CRT monitors produce the worst levels of interference among the various objects

tested. The sensors are negligibly affected by the presence of a CRT until they approach within approximately 16 inches of the monitor. At this point, a severe amount of random fluctuation in readings is observed, rendering the sensors absolutely useless. As a general rule, all monitors and electronic devices should be kept approximately 2 feet away from the sensors and at least 5 feet away from the transmitter.

Interestingly, neither the small amount of metal in the VR4 head mounted display helmet, nor the small EMF emitted from the two small monitors inside the helmet affected the sensor readings for translational motions, but significant distortion arose during rotations. This problem was reduced to tolerable levels by placing the bird on a standoff to move it away from the immediate vicinity of the helmet. This approach complicated slightly the calibration technique to obtain the constant transformation between the user's eyes and the bird, but the transformations described in Sec. 3.11 still provide proper correlation between the user's head motions and the camera motion.

2. **Linear and Orthogonal Shear Testing** - Each sensor was placed at various points along a sequence of points which were previously determined to form a straight line and/or an orthogonal box. The shear effect was then examined by comparing the angle between segments of the straight line or between sides of the box as determined by the flock of birds sensors to the known angles. During this test, a severe amount of shearing was observed near the floor, walls, and ceiling of the test environment. This is due to metal reinforcement rods in the floor and

ceiling. Notably, this effect was reduced when the same test was performed at a higher floor in the same building. This is due to the fact that less metal reinforcement is utilized in the structure of higher floors that do not have to support as much weight as lower floors in the same building. Also, a significant interference effect was noted when the same test was performed outside and/or near any windows which faced the outside of the building. The reason for this interference is unknown, although there was a direct line of sight to a strong radio station transmission tower during these tests. Similar interference with electronic equipment is known to have been caused by this radio tower, but how radio transmissions might effect an EMF field device is unknown. Regardless, once a wall was introduced between the outside and the sensor, the unknown interference disappeared. In summary, keeping the sensors at least 1 foot away from the floor, walls, windows, and ceiling of the environment reduces interference to an acceptably small level. Transmitter location next to these objects does not seem to significantly affect sensor readings.

3.3 The VR4 Helmet and Multi-Channel Option

The VR4 head mounted display helmet is a passive device that is capable of receiving various types of video signals on two parallel channels and displaying them on two separate projector monitors. It requires no software support. The helmet is fully adjustable to match the various shapes and sizes of the human head. Careful adjustment of the helmet is crucial to the viewing quality obtained from the helmet. There are four adjustments available.

1. **Rear Head Strap** - This adjustment adjusts the helmet size to fit the circumference of the head. It is primarily responsible for holding the helmet in a fixed position during motion. It also supports the majority of the weight of the helmet.
2. **Top Head Strap** - This adjustment adjusts the height of the helmet. By adjusting this strap, the user can adjust the height of the monitors with respect to their eyes.
3. **Front Monitor Shell** - This adjustment is performed by pushing and pulling on the front shell which surrounds and protects the monitors at the front of the helmet. This effectively brings the monitors closer to or further away from the user's eyes.
4. **IPD** - The IPD knobs, or Inter-Pupillary Distance adjustment knobs can be used to set the distance between the two monitors such that it matches the IPD of the user.

A typical donning procedure, as determined from frequent use, is as follows:

1. Adjust rear strap, top strap, and front shell to their largest settings.
2. Remove glasses that correct for far-sightedness, if necessary. Although you can wear glasses while using the helmet, most people find it uncomfortable. The images inside the helmet are already focused and will be located only several inches from your eyes. Glasses that correct for near-sightedness and astigmatism should probably be worn.
3. Place the helmet down fully onto the head and tighten the rear strap until the helmet is secure and will not shift during head motions. Note that the majority of the helmet weight is due to the relatively heavy projector monitors at the front of

the helmet. Thus, a forward moment is produced. To fully counteract this moment, make sure that the helmet's cable harness is hung freely down the back of the helmet and behind the user.

4. Utilizing the cameras on the robot, display a good stereographic image on the projectors in the helmet. The user will use this image to determine how to adjust the helmet during the following steps.
5. Tighten the top strap to bring the projectors up to the level of the user's eyes.
6. Adjust the IPD knobs until the image in both eyes appears to be in good focus. The image in each eye will appear to shift colors during this procedure. Adjust until the image appears to be the same color and brightness in both eyes. NOTE: Since the IPD knobs move both monitors in and out at the same time, it may be necessary during this procedure to twist the helmet a small amount one direction or the other.
7. Lastly, adjust the front shell to bring the monitors in towards the eyes until the user achieves a comfortable viewing range. Most people are comfortable with the monitors fully away from their eyes. Others prefer the monitors up close.

The helmet can also be adjusted to accept various image formats. There are two 25 pin D-shell RGB ports for computer graphic inputs, and two 4 pin round Y/C (S-video) ports for video inputs. A switch on the front selects between RGB and Y/C. The helmet can also be adjusted by another switch for use with one image (mono) or two images (stereo), and adjusted for "sync on green" or "separate sync."

Although the helmet requires no software support, it does need special hardware since it requires two simultaneously generated images. These simultaneously images can be provided by two Y/C cameras, or by a computer that is capable of generating two parallel streams of RGB graphics output. Currently, this remote inspection system utilizes two cameras to provide two Y/C images that are directly provided to the helmet. However, to achieve graphic overlays in the future, both methods will be needed. First, the two cameras would provide the two Y/C images, which would then be simultaneously captured into memory by the Onyx computer. Graphic overlays would then be applied to each image and the output sent simultaneously to the helmet in RGB format. This simultaneous output would pass through the Multi-Channel Option (MCO) on the Onyx computer.

The MCO provides up to 6 simultaneous output channels, each with a dedicated buffer of video memory. Various settings are available and are listed in the file `/usr/gfx/ucode/RE/dg2/vof` (a binary configuration file included with the code) and described in the file `/usr/gfx/ucode/RE/dg2/vof/README` (see Appendix A.). Two scripts for controlling the VR4 helmet, MCO, and the Onyx console graphics system are available in `~/src/vr4` (See Appendix B.). The first script (`vr4_on`) disables console graphics and places the MCO into a mode enabling two interlaced 640x486 resolution channels at 30Hz. This is the exact format required by the VR4 helmet. The first two channels are enabled, with channel 0 going to the left eye of the VR4 and channel 1 going to the right eye. The second script (`vr4_off`) enables console graphics and disables the MCO, placing the Onyx back into the normal full screen graphics mode. For the exact command sequences and options utilized, refer to the two script files. For more detailed information on the commands

themselves, refer to the man pages for: setmon, gfxinfo, stopgfx, and startgfx.

3.4 Stereo Graphics

The concept behind stereo graphics is based upon the way that the human eye senses depth. Individually, a human eye can only determine two dimensions. While it is possible for the brain to use the image from a single eye to sense depth, this is done simply by making a guess based upon the observed size of known objects. Placed into an environment of unfamiliar or incorrectly sized objects would easily fool the depth sensing abilities of only one eye. However, with two eyes separated by some small distance, two images are obtained from a slightly different viewing angle and integrated by the brain to provide a true sense of depth and immersion in the environment.

This sense of immersion is critical in remote inspection systems in which the goal is a sense of telepresence. However, it is not easy to achieve mechanically what the eye does naturally. First, the human eye selects an object to observe. This object then provides depth cues based upon the perceived shift in lateral location between the left and right images. The eyes then converge inward to match up the two images, thus providing a stereo integration into one 3D image that is analyzed by the brain. Finally, based upon the amount of convergence, the eye receives focus cues that hint at the approximate level of focus necessary to obtain a clean image. The brain then makes the final adjustments to get perfect focus.

To imitate such an amazing system with two cameras and a computer, one must provide two images upon which the brain can accurately perform the stereo integration process. This must be done very careful. The human eye is one of the most finely tuned optical instruments known to man. It will not be fooled by approximated images from incorrectly aligned cameras. The human eye

will detect these minute flaws, resulting in eyestrain. After even a short time in an inaccurate system, significant discomfort, dizziness, headaches, and nausea can develop.

The first step in the process is therefore to accurately determine the object that is currently being observed. While it may be possible to utilize object recognition techniques to achieve this goal, it is unlikely that fast and accurate processing could be achieved in all situations. Adding complexity to the algorithm to handle unknown objects and situations would make the technique more accurate but, at the same time, less and less capable of being achieved in real time. This method also adds unnecessary complexity and CPU task loading to the system that can be avoided by other methods such as the use of range finders. In this prototype, a LASER is used to determine which object is being observed and the distance to that object. Distances with an accuracy of a tenth of an inch can be achieved.

Having accurately calculated the distance to the observed object, the cameras can then be converged upon that object in real time and with virtually no CPU load. Also based upon the distance measurement, an estimate or hint of focal length can be made and sent to the camera. This is done to shorten the amount of time that it takes the cameras to perform their auto-focusing procedure. Although it would be very difficult to accurately determine the proper focal setting, an estimate can be sent to the cameras to place them at a position somewhere near the true focal length. Since the cameras are left in auto-focusing mode during this process, they will then take over and complete the focusing process. Using this hinting process, a correctly focused image can be achieved in approximately one-third of the time it takes the cameras to focus on the image without a hint. This process was patterned after the way the human eye focuses

after getting a hint from the amount of eye convergence, as mentioned above.

However, beyond the intricacies of the eye mechanism, additional difficulties exist. For example, the cameras must be precisely aligned and pointed at the same point in physical space. Without straining, the eye will not be capable of integrating two images that are not precisely converged. They must also be mounted at the same height. Unlike the previous problem, the human eye is physically incapable of converging two images that are offset vertically from each other. Double vision, headaches, and disorientation will quickly result.

Even after eliminating the above difficulties, still others exist. The position of the helmet that provides the stereo images to the user is important. A misalignment of the helmet is equivalent to a misalignment of the cameras. Most important with regard to the helmet, however, is the IPD adjustment of the monitors. A mismatch between the user's IPD and the helmet's IPD will cause the images to be of a different color, brightness, and focal quality. This quickly causes headaches, similar to having a smudge on only one side of a pair of glasses.

3.5 The VC-C1 Communications Camera

The human eye can adapt well to poor or changing image quality, resolution, flicker, color, and aspect. However, forcing the eye to adapt to these conditions causes eyestrain over time. Thus, the high quality VC-C1 optical cameras from Canon were chosen for this system. These cameras provide various advantages to this system, but also introduce some drawbacks that must be overcome.

In addition to providing high-resolution, stable, accurate images, the VC-C1 has a built in RS232-C computer control mechanism. The various commands have been integrated into a library that can be used to create camera control software that communicates with the cameras at 9600 baud,

no parity, 8 data bits, and 2 stop bits. Each command is represented in a byte-string format consisting of a 4-byte header (0xFF, 0x30, 0x30, 0x00), followed by a 1-byte command opcode, followed by up to 3-bytes of optional data, followed by a 3-byte trailer (0xEF, 0x0A, 0x0D). Example software can be found on the Kiwi PC in ~/src/vcc1 (See Appendix B.). For a complete listing of available commands, refer to the VC-C1 owner's manual.

Another benefit provided by the VC-C1 camera is pan/tilt/focus functionality. The pan and focus capabilities are utilized to perform the stereo convergence and focus hinting referred to in section 3.4. Since exact vertical alignment between the two stereo images is crucial, the tilt capability is also utilized to compensate for small amounts of vertical shift between the images. In addition, the lens position on the VC-C1 as well as the way in which the pan and tilt mechanisms move is important. The lens is offset to the left of the center of pan rotation. Thus, as the camera rotates, the lens shifts slightly while it rotates, just like the lens of the human eye does. For example, when the left eye rotates to the right, the lens shifts to the right and drops back slightly. Although this is a very slight shift, eyestrain will result if the images from the cameras in a stereographic remote inspection system do not also shift slightly in the appropriate directions.

A final benefit is the relatively lightweight and compact size that the VC-C1 offers. It is light enough to allow two of the cameras to be mounted on the end of the Merlin robot arm without exceeding the robot's 25-pound maximum payload.

However, for all these benefits, some drawbacks must be accepted and overcome. First, the lens offset mentioned above causes a major problem since, if the cameras are oriented side by side, the lens can never be placed closer together than 6 inches. Since the inter-pupillary distance of the human eye

typically varies between 2 and 4 inches, this 6-inch minimum separation is much too large to accurately provide good stereo vision. Refer to Figure 2a for a graphical representation of this problem. Thus, one or both of the cameras must be reoriented in some other way for the lens to assume this range of separation. One possibility is to mount each camera on its side as shown in Figure 2b. However, this results in two images that must then be rotated back 90 degrees before they are presented to the user. While this rotation can be achieved, it adds additional and unnecessary complexity to the system. A better solution, in which one camera is completely inverted, is shown in Figure 2c. With this solution, one image must be rotated 180 degrees. This rotation is equivalent to one horizontal image inversion and one vertical image inversion. This

process can be easily performed quickly on the computer, introducing very little CPU load. However, an easier solution is to extract the charge-coupled device (CCD) chip from one of the cameras, rotate it, and reinsert it upside down. This is a standard camera image inversion procedure that is easily allowed for by many video camera models. It takes about 15 minutes with the VC-C1.

Another problem introduced by inverting one of the cameras is that the camera mount used to attach the cameras to the robot must be precisely machined to eliminate vertical offset between the cameras. Also, this type of camera mounting requires the mass of the cameras to be placed off to one side of the robot, generating a rather large moment (Refer to Figure 2d.)

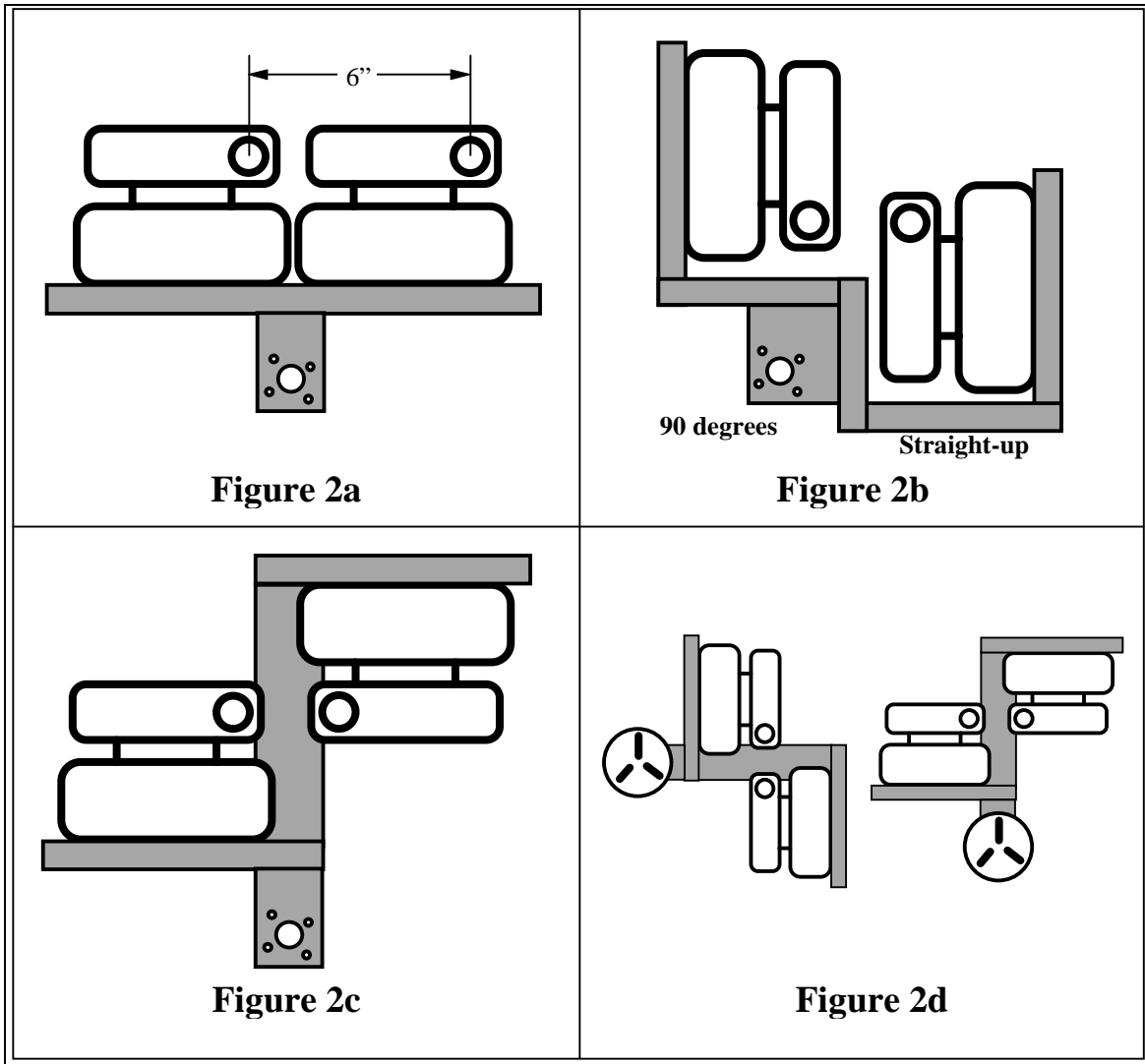


Figure 2: Camera Mounting Configurations

This moment is easily supported by the Merlin robot while the robot is active and motor power is on. However, when the robot is parked and motor power is off, the moment of the cameras and mount are too much for the wrist to support in the normal straight-up mounting orientation utilized by previous cameras mounted on the robot. Without motor power to support this moment, the wrist will sag and the cameras will swing down towards the ground. Although this does

not cause any damage to the system, it disrupts the calibration of the robot. However, the design of the Merlin robot wrist allows a quick solution to this problem. With the motor power off, the wrist has very little strength against rotation in the clockwise direction. However, it has quite a bit of strength against rotation in the counter-clockwise direction. This is due to a built-in 45-degree rotation in the wrist joint. Thus, a 90-degree rotation of the mount shifts the

moment into the counter-clockwise direction. Despite, the intuitively stable appearance of the straight-up mounting and the weak appearance of the 90-degree mounting, the robot is easily capable of holding the cameras in the 90-degree mount configuration.

3.6 *The Sirius Video Device*

At the very heart of this prototype remote inspection system is the need for high speed, high quality video images. In order to analyze the video, provide overlay graphics, and output it to the HMD helmet, the images must be captured by a computer into memory. Without special video processing hardware, this is a rather slow process. However, fast video capture boards exist for a variety of platforms. The Sirius Video device is one of the fast image capture systems available for the SGI Onyx.

The Sirius Video device provides a video library of routines called VL that provide high-level control over the pipelined architecture of the device. This library has the capability of linking various types of video nodes together to achieve the fast capture, processing, and transfer of video images and graphics buffers. The device must first be opened with a call to `vlOpenVideo()`. Next, paths or pipelines over which to transmit video images are setup by connecting two nodes together with a call to `vlCreatePath()`. One node is established as the source node, `VL_SRC`, and the other node is established as the drain node, `VL_DRN`, with two calls to `vlGetNode()`.

Nodes establish connections to physical devices and can be one of six possible kinds. The first type, `VL_VIDEO`, refers to a video imaging device such as a VCR or camcorder. The second type, `VL_GFX`, refers to graphics pipelines available only within the Sirius Video device. These pipelines allow graphic and video image data to be transferred directly into the proprietary SGI graphics subsystem,

bypassing intermediate system RAM. The third type, `VL_MEM`, refers to a system memory buffer. The fourth type, `VL_SCREEN`, refers to a portion of video RAM latched to a bounded window managed by the X-windows display manager. The fifth type, `VL_TEXTURE`, refers to a specially encoded portion of memory within the proprietary SGI texture subsystem. The sixth and final type, `VL_BLENDER`, refers to a graphics node that is to be blended with a `VL_VIDEO` drain node.

Once a path has been established between two nodes, a video image transfer may be conducted between the two nodes, with video image data being drawn automatically from source to drain. The transfer is started with a call to `vlBeginTransfer()` and takes place on a field by field basis, where a field is one half of an interlaced frame of video. Multiple source and drain nodes may be added to a particular path at any time with a call to `vlAddNode()`. This is useful for situations in which multiple sources or drains are needed. Blending two video images together is one example. Finally, various node parameters can be checked and set with calls to `vlGetControl()` and `vlSetControl()`, respectively. This allows the images to be scaled, cropped, and offset, among other things. At the end of every program, the system should be deactivated properly with appropriate calls to `vlDestroyPath()` and `vlCloseVideo()`. More detailed information on these and other VL calls is available in the IRIS Media Libraries Programming Guide.

The VL library and pipelining architecture utilized in the Sirius Video device allow SGI-platform independent video software to be written very quickly, yet still utilizing the special features available on each machine. For example, special features of the Sirius Video device, such as direct video to screen capture, are utilized automatically if

the machine is so equipped. A sample video to screen code segment would be as follows:

```
/* open the Sirius Video device */
    svr = vlOpenVideo("");
/* create the video source node */
    src = vlGetNode(svr, VL_SRC,
    VL_VIDEO, VL_ANY);
/* create the screen drain node */
    drn = vlGetNode(svr, VL_DRN,
    VL_SCREEN, VL_ANY);
/* create the transfer path between the source
and drain nodes */
    path = vlCreatePath(svr, VL_ANY,
    src, drn);
/* begin image data transfer */
    vlBeginTransfer(svr, path, 0, NULL);
/* end image data transfer */
    vlEndTransfer(svr, path);
/* close the path */
    vlDestroyPath(svr, path);
/* close the Sirius Video device */
    vlCloseVideo(svr);
```

The Sirius Video device is contained within a box that sits on the table next to the mainframe computer tower. On the reverse side are where cables connections are made. At the far left side, analog input and output ports are available in two columns of three plugs each. The left column is for the Y/C or S-video format cables while the right column is for composite format. Both NTSC and PAL formats are accepted, but typically the United States uses NTSC equipment exclusively. PAL is utilized overseas. The only difference between the two formats lies

in the width of scan lines in the image and thus, the timing signals contained within the format. In each column, the top two plugs are tied together internally as one input. Thus, one plug can be used to input an analog video image, while the second plug can be used as a direct loop-through to carry the image on to another device. Note that this means that only one analog video source can be captured into the Sirius Video device at a time. The bottom plug is output.

The current configuration of the remote inspection system utilizes the Sirius Video device as indicated in Figure 3. The areas of the drawing in dark print are physically connected to the system, while the gray areas are not utilized. The flow of video images is as follows:

1. Left eye image is sent directly to the helmet, bypassing the Sirius Video completely.
2. Right eye image is sent into the Analog Video Input and echoed back out to the helmet through the Analog Video Loop-Through.
3. The right image is also sent to System Memory through the Memory Pipeline.
4. Once in memory, the image is analyzed as discussed in section 3.7.
5. The image is then sent to the Graphics Subsystem.
6. Once in the subsystem, overlay graphics are generated and applied to the image.
7. Finally, the modified image is sent to the console for viewing.

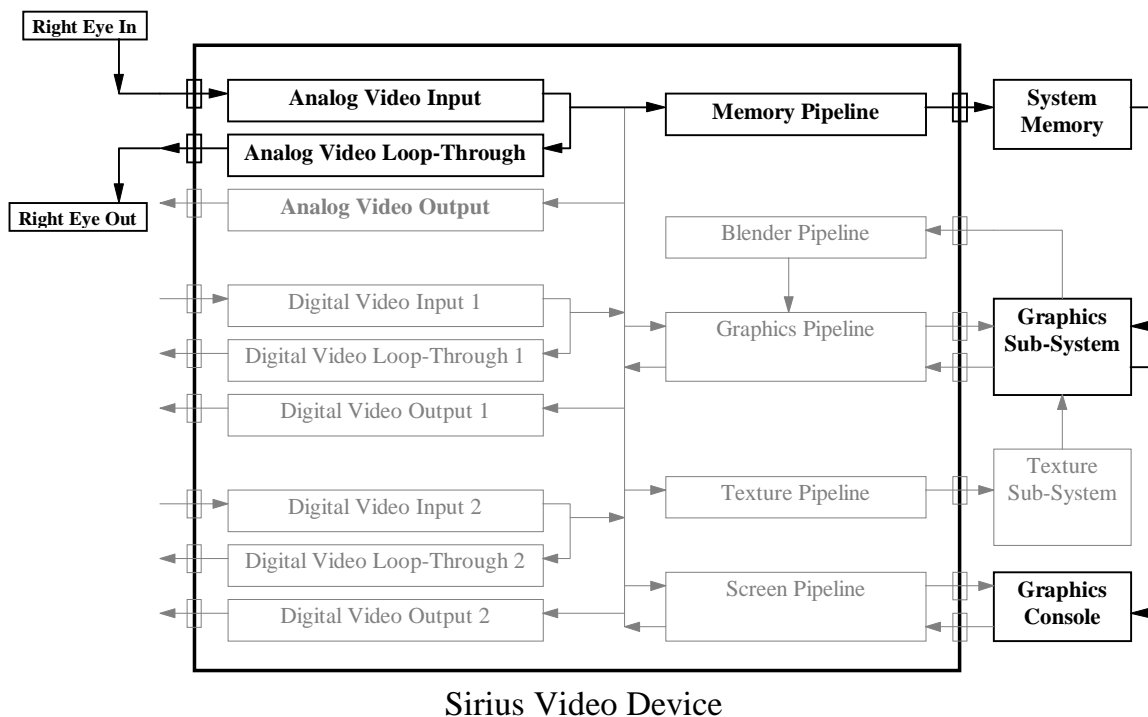


Figure 3: Current Sirius Video Pipeline Configuration

In the future, the system could be configured to more fully utilize the capabilities of the Sirius Video device by taking advantage of the two digital image input lines, thereby circumventing the prior configuration's limit of only one video signal and allowing graphic overlays to be placed inside the helmet. How to accomplish this new configuration will be discussed later in section 3.10.

3.7 LASER-Based Convergence

In order to be able to converge the two VC-C1 cameras on the object being viewed, an accurate distance to that object must be measured. This is accomplished for this system by a LASER mounted on one of the cameras. The LASER is precisely aimed the same direction as the camera and is precisely centered over the top of the camera lens. The LASER placement is crucial to retaining accuracy in the distance calculation. Thus,

the mount for the LASER was designed similar to a scope mount on a rifle. A horizontal screw can be turned to adjust the left and right fall of the dot, while a vertical screw can adjust the up and down drift. Once adjusted the LASER was frozen into position with hot glue. This was chosen since it will hold the LASER very strongly in a fixed orientation but can be removed at a later date if necessary.

Incoming images from the camera are captured by the Onyx computer as discussed in Section 3.6. The images are analyzed to find the red dot produced by the LASER. Based upon the position of this red dot with respect to the center of the image, the distance can be calculated to roughly the nearest tenth of an inch. This level of precision is more than enough for use in the camera convergence process.

At first it might seem strange and even useless that the LASER is pointed exactly the same direction as the camera. It might seem that the position of the red dot in the image would never change. However, although it is true that the actual physical distance between the LASER beam and the center of the camera image never changes, the scaled perceived position of the red dot will change with distance on the image. To see why, refer to Figure 4 which displays a side view of the camera and the LASER, along with two sample images and their respective laser dots. Notice that as the distance from the camera lens increases, the physical area of the environment observed by the camera grows exponentially. However, this area, no matter how large, is still mapped into the same 720x586 charge-coupled device (CCD) chip that receives the image. Thus, the physical area mapped per pixel grows exponentially with distance. This results in a scaling effect that can be utilized to very accurately measure the distance. For example, while a one foot

object in the environment might appear to be quite large when viewed by the camera from only 1-foot away, the same object scales down to appear quite small when viewed 5 feet away. Similarly, the constant distance between the LASER and the center of the camera image also scales with distance.

Refer again to Figure 4. Note that the actual position of the red dot in the image follows an exponential curve. This would be difficult and inaccurate to utilize in the distance measurement calculations. Notice also that the distance can neither be easily nor accurately calculated by the following equation:

$$d_x = \frac{\tan \theta}{x_2} - d_f$$

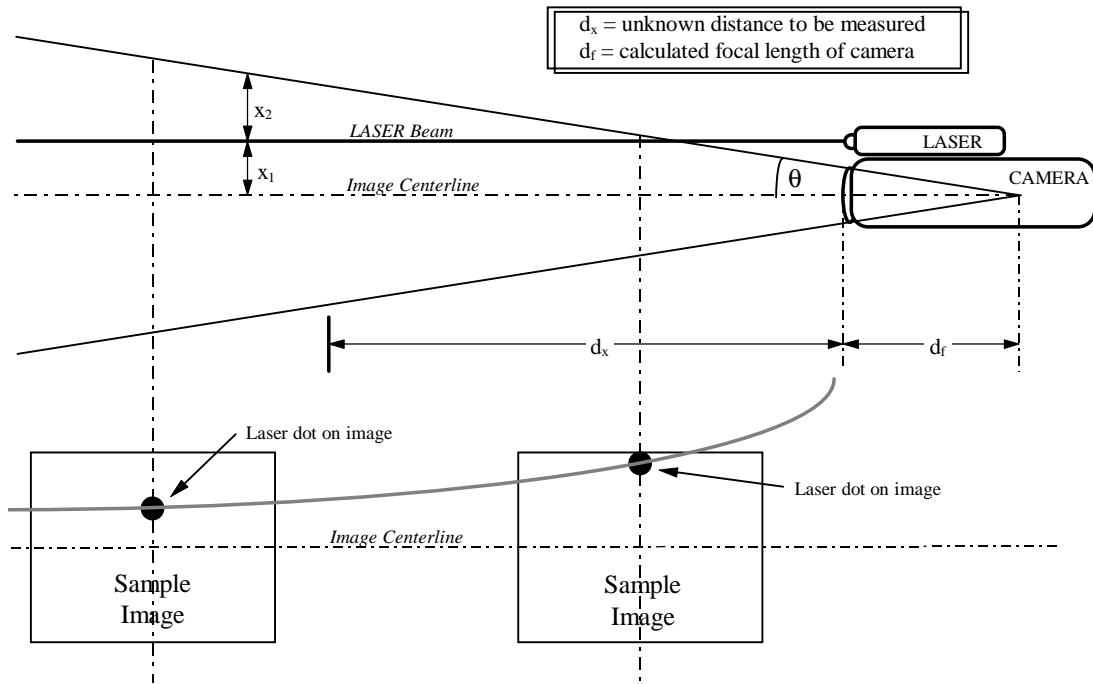


Figure 4: LASER Convergence System Configuration

This equation is perhaps the most straightforward method of distance calculation, based upon the geometry of the LASER setup. Note here that θ is one half of the field of view angle of the camera and can be easily measured. The problem that renders this equation unusable is one of units. The desired value d_x needs to be calculated in terms of physical real-world units. However, the term x_2 in the equation is measured in terms of pixels on the image. Thus, a units conversion term must be introduced which is capable of converting pixel units into real-world distance units. And this conversion term changes as a function of the distance that is being calculated! Although it is theoretically possible to solve this equation, unnecessary complexity and inaccuracy will be introduced.

Therefore, a better solution lies in calculating the ratio of the pixel length of the portion of the top half of the image which lies below the laser, x_1 , and the pixel length of the top half of the image, x_1+x_2 . Because of the ratio, the pixel units cancel out, independent of the distance being measured. And x_1+x_2 is actually just half of the vertical resolution of the camera, or $Y_{res}/2$. The equation can thus be derived to be:

$$d_x = \left(\frac{Y_{res}/2}{x_1} \right) d_f - d_f$$

Note that this equation requires the focal length of the camera, d_f . This can be easily measured by placing an object of known length at two different known distances from the camera lens. The lens focal length can then be calculated with the law of sines since the camera has a fixed field of view angle θ . Note that the focal length and field of view angle will change, however, as a function of zoom. This can be compensated for since the change will be linear with the zoom setting.

It is interesting to note that while the normal process of attempting to detect a red dot somewhere in a high-resolution image is quite CPU intensive, the process of finding the dot in this system imposes virtually no CPU load at all. A careful examination of Figure 4 shows that, if the laser is properly aligned, then the red dot must lie somewhere on the vertical centerline of the image. Even better, it must lie somewhere on the upper half of the vertical centerline of the image since, in worst case at infinity, the dot will tend towards the exact center of the image. And in practice, the search routine can even be constrained to the top quarter of the vertical centerline of the image since, by the time the dot drifts down to the second quarter, the object being viewed is so far away that convergence is no longer an issue. Thus, even a complex dot detection algorithm would not impose much of a load while finding a dot in such a small location.

The method used by this system to find the dot is quite simple. Starting at the top of the image, each pixel is inspected to see if the red value exceeds some threshold. If it does, the top of the laser dot may have been detected. Or alternatively, some bright white or even yellow light may have been detected since colors such as these all contain high levels of red as well. Therefore, the pixel is checked again to see if the corresponding green and blue values have dropped below some threshold. If so, the top of some bright red area has been detected in the image. However, depending upon the objects and/or lights in the surrounding environment, this still may not be the laser dot.

With the location of the top of the possible laser dot area determined, the location of the bottom must now be determined to check for proper size. The search is thus continued from the bottom of the top quarter of the vertical centerline until the lower portion of some bright red area is detected. This may or may not be the same

bright red area as detected in the previous top-down scan. Thus, the locations of the two detected boundaries are compared to estimate the vertical size of the object. If it lies between some minimum and maximum size, it might be the laser dot. Otherwise it is not the laser dot, the previous convergence angle is maintained, and the next cycle of images is examined. Since convergence only needs to occur at a very slow rate of around 5Hz, this is acceptable. At an image rate of 30 Hz, the dot really only needs to be detected, therefore, in about one image out of every six.

If the vertical size looks promising, the horizontal size is then checked in a similar manner. If both sizes are within the correct range, the bright red area is assumed to be the laser dot, and the top, bottom, left, and right sides are averaged to find the center. This location is then utilized to calculate the next convergence angle.

Filtering and jitter reduction are also implemented to stabilize and smooth the incoming data. This is done in a similar way to the flock of birds filtering and jitter control algorithms discussed at the end of Section 3.1.

3.8 The Ludlum Geiger Meter

Small gamma-emitting radiation trace sources are placed into the environment to simulate a radioactive environment. A Ludlum Geiger meter is then used, on the highest sensitivity setting, to measure the radiation dosage that the equipment is absorbing. Although there is not a significant amount of radiation in this research environment, the idea has been shown to be very effective.

A voltage tap line runs out of the meter and is sampled by a Motorola HC811 microcontroller. Before the voltage is sampled, however, it is run through an op-amp that is configured as an analog buffer. This isolates the difference between the meter's battery ground and the microcontroller's power supply ground. It

also prevents the microcontroller from sourcing current into the meter should the batteries die during a run. The value is also amplified and shifted into the microcontroller's acceptable analog to digital conversion range.

A small assembly program running on the HC811 performs the analog to digital conversion only when requested. Since the HC811 is connected to the Grape PC by an RS-232C serial port, this request comes in the form of a single byte transmitted to the microcontroller. Upon receipt of this byte, the microcontroller performs the conversion and sends a raw data byte back across the port, representing the voltage from the meter with respect to the microcontroller's valid conversion range. This raw value is then picked up by a program running on Grape and translated into a floating point voltage value. This value is then transmitted across the network to the Onyx where it is used to generate the Geiger meter panel for the overlay graphics, as discussed in section 3.9 below.

3.9 Overlay Graphics

The overlay graphics in this system consists of a cross-hair sighting reticle and 6 separate panels and meters which are drawn transparently over the camera images such that they do not obstruct the user's vision of the environment. Refer to Figure 5 for a sample camera image and associated overlay graphics. The first two panels are located at the top center of the image. These indicate current time and date, along with the user's name and ID number, if applicable. Thus, a video recording of the RI task can be made and kept as a record for later reference.

The third and fourth panels are located at the bottom of the image. The third is an elapsed time indicator that shows the total amount of time that the user has been conducting the task. Or, for hazardous remote inspection tasks, this panel shows the amount

of time that the remote equipment has been subjected to the harsh environment during the task. Next to this indicator, the fourth panel indicates an accumulated radiation dosage as determined by integrating over time the sampled value from the Geiger meter. This panel can be configured in software to alert the user when certain dosage levels have been reached. By default, the panel is configured to be drawn in a “safe green” color as long as the level stays below 30 micro-rads (μR). Upon exceeding 30 μR , the meter turns to a “warning yellow” color. At 40 μR , the meter turns to an “alert red” color. At 50 μR , the meter begins to flash. Beyond 60 μR , the meter turns darkly opaque and flashes more rapidly. In addition, the sighting reticle also begins to flash, thus hopefully catching the user’s attention whether or not they happen to be looking at the dosage panel. Again, these

levels were chosen arbitrarily and can be changed to suit the application at hand.

The last two panels are graphical meters. The panel in the top right corner indicates the radiation dosage rate as directly measured from the Geiger meter. This meter is in bar graph form and ranges from zero to one milli-rad per hour (mR/hr) as provided by our Ludlum Geiger meter. This panel is also color safety coded. As long as the rate stays below 0.6 mR/hr, the bar graph is “safe green” in color. Upon exceeding 0.6 mR/hr, the bar graph changes to “warning yellow”. Finally, the bar graph turns “alert red” after exceeding 0.8 mR/hr. Similar to the previous panel, the levels at which the color changes take place were chosen arbitrarily and can be easily changed to suit the needs of any given application. The range of the meter can also be adapted to fit the incoming data.



Figure 5: Sample Remote Inspection Image With Overlay Graphics

The last panel has six vertical sliding indicators that inform the operator as to their location within the defined safe workspace for the task. The bars represent the three position and three orientation parameters in a typical 6 DOF space. The lower edge of each bar represents the minimum value that the particular associated degree of freedom can take on. Similarly, the upper edge indicates the maximum value. The line in between the two edges floats up and down to indicate the current value. If the line floats down to within 20% of the minimum value, the bar color will turn to “warning yellow”. If the line drops within 10% of the minimum value, the bar will turn to “alert red”. Otherwise all bars are “safety green”.

3.10 Digital Video and the Sirius Video Device

A proposed future configuration of the Sirius Video device would allow the two camera images to be simultaneously captured by the Onyx. The Sirius box has two digital video inputs but only one analog video input, thus simultaneous capture can only be accomplished by converting the images to Digital format prior to input into the Sirius box. This allows for overlay graphics to be introduced into both images and sent back out to the helmet.

This method will also have the side benefit of speeding up the video capture cycle, since the system would be converting the next set of analog images into digital format while the Sirius is still processing the previous two images. This would reduce the cycle time by the amount of time it takes the Sirius box to convert the analog images to digital itself, and make more efficient use of all hardware involved.

However, this method would also require the introduction of more hardware as shown in Figure 6. Two analog to serial image converters, such as the Meridian ASD-100, linked with two-serial to parallel digital

translators (e.g., Meridian SER-100D) could perform this task. The modified flow of video images would proceed something similar to the following:

1. Left and Right eye images would be received at the analog inputs of the external converters. The analog to digital conversion would take place in parallel while the Sirius Video device processed the images from the previous cycle.
2. When the Sirius is ready, the two digital images would be fed into the two Digital Video Inputs.
3. Since there is only one pipeline to each subsystem, a trick would be used to allow the two images to pass through the Sirius Video device in parallel. The left eye image would be sent directly through the Graphics Pipeline to the Graphics Subsystem, while at the same time, the right eye image would be sent through the Memory Pipeline to the System Memory.
4. Once the left image arrived in the Graphics Subsystem, overlay graphics would be generated and applied to the image. At the same time, the right image in memory would be processed as discussed in Section 3.7.
5. Next the right image would be then sent through the now idle Graphics Pipeline into the Graphics Subsystem.
6. Once the right image arrived in the Graphics Subsystem, the same overlay graphics generated in step 4 would be applied to the image.
7. Finally, both modified images would be sent to the helmet by the Multi-Channel Option’s direct Graphics to Video pipeline, discussed in Section 3.6.

Since this method has yet to be implemented, it is difficult to estimate the cycle speed. However, discussions with SGI personnel indicate that the same full 30Hz rate

that is achieved by the present configuration should also be achievable by the new configuration.

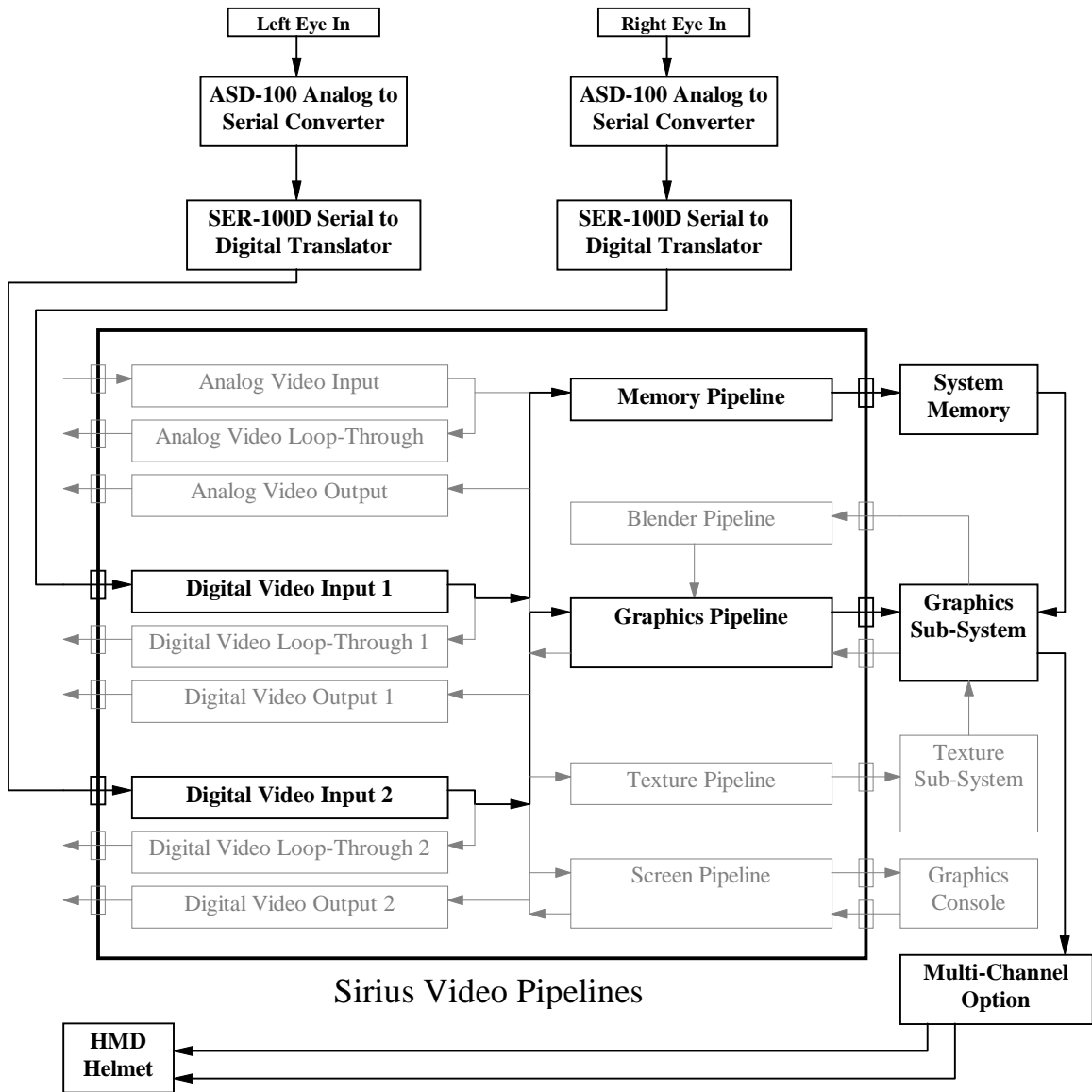


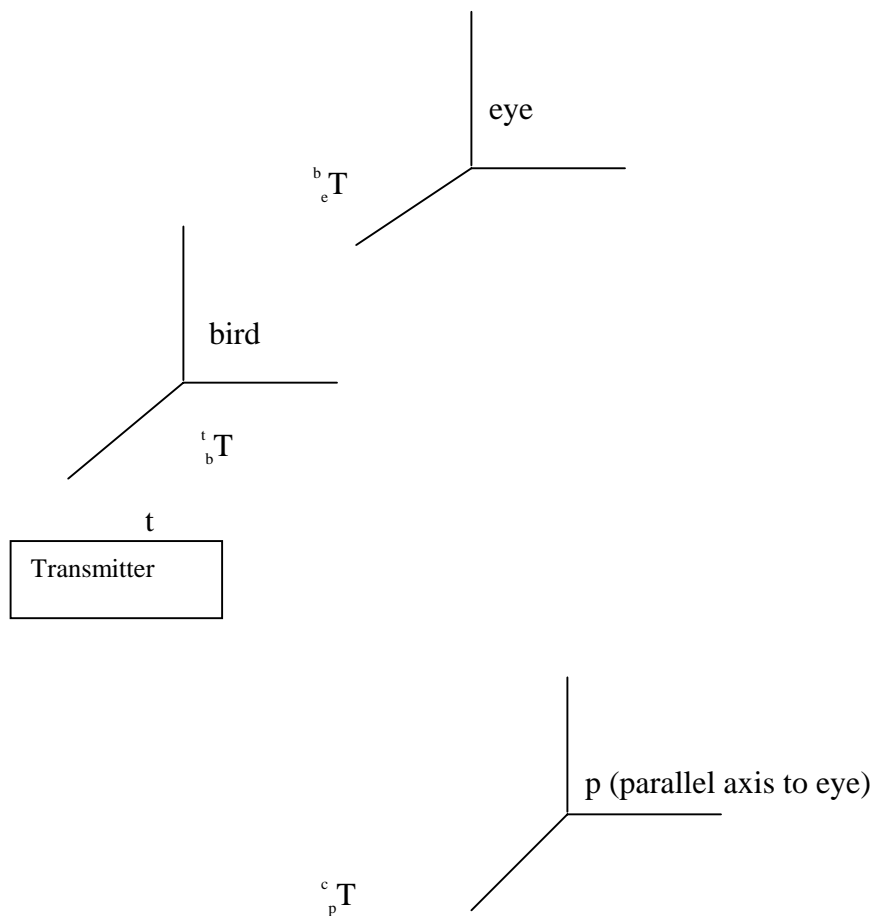
Figure 6: New Sirius Video Pipeline Configuration

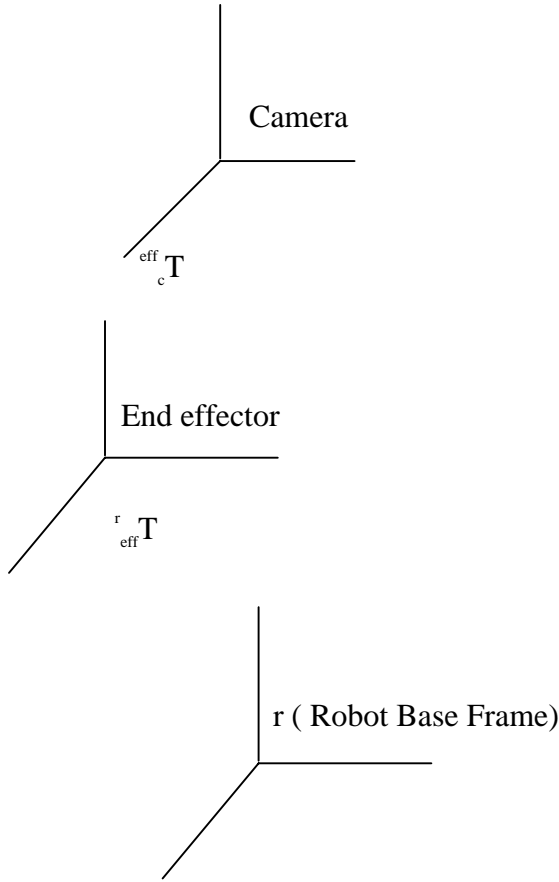
3.11 Eye to Camera Transformations

As described above, the movement of the cameras at the end effector of the robot is not exactly aligned to that of the eye, as the bird is mounted on top of the helmet. Hence there was a need to insert transformations from the sensor on top of the VR helmet to the eye and from the end effector of the robot to the center of the cameras. Then, with the proper commands, it is possible to make the center point of the camera move as the center point between the user's eyes.

More precisely, commands given to the robot cause the end effector to move in accordance with them. However, our desired input command is the center point between the user's eyes. This input is not directly

measured. Rather, it is the position and orientation of the bird that are measured. Moreover, the desired position is expressed in terms of the center point between the camera, the position of the end effector, which is the only thing the robot controller can command. The figures and derivation below show a sequence of transformations that relate all of the relevant coordinate frames, and the calculations to determine the end effector position that would yield the desired camera position. These are based upon taking an initial calibration measurement to determine where the user and robot are initially, and then having the camera position track the user's eyes thereafter.





Transformations between the transmitter and the eye.

Eye w.r.t. transmitter is constant:

$${}^b C_e = {}^b T_e \cdot I \quad (1)$$

Let ${}^t T_b$ be the transmitter w.r.t. transmitter (as measured by the hardware), then

Eye w.r.t. transmitter:

$${}^t C_e = {}^t T_b \cdot {}^b C_e \quad (2)$$

$$\text{Therefore, } {}^t C_e = {}^t T_b \cdot {}^b C_e \quad (3)$$

$$\text{Initial eye w.r.t. t: } ({}^t T_b \cdot {}^b T_e)_0 = {}^t P_0 \quad (4)$$

$$\text{ith eye w.r.t. t: } ({}^t T_b \cdot {}^b T_e)_i = {}^t P_i \quad (5)$$

Let ${}^0 T_i$ be the ith configuration of the eye w.r.t. the initial configuration of the eye, then

$${}^t P_i = {}^t P_0 \cdot {}^0 T_i \quad (6)$$

$$\text{Therefore: } {}^0 T_i = {}^t P_0^{-1} \cdot {}^t P_i \quad (7)$$

Transformations between the robot base frame and the camera.

“Parallel Axis” w.r.t. camera is constant:

$${}^c C_p = {}^c T_p \cdot I \quad (8)$$

Let ${}^{\text{eff}} T_c$ be the camera w.r.t. the end effector (this is constant), then

PA w.r.t. end effector:

$${}^{\text{eff}} C_p = {}^{\text{eff}} T_c \cdot {}^c C_p \quad (9)$$

Let ${}^r T_{\text{eff}}$ be the end effector w.r.t. r (this is the commanded robot position), then

PA w.r.t. robot base frame:

$${}^rC_p = {}^rT_{\text{eff}} \cdot {}^{\text{eff}}C_p \quad (10)$$

$$\text{Therefore: } {}^rC_p = {}^rT_{\text{eff}} \cdot {}^{\text{eff}}T_c \cdot {}^cT_p \quad (11)$$

$$\text{Initial PA w.r.t. r: } ({}^rT_{\text{eff}} \cdot {}^{\text{eff}}T_c \cdot {}^cT_p)_0 = {}^rQ_0 \quad (12)$$

$$\text{Ith PA w.r.t. r: } ({}^rT_{\text{eff}} \cdot {}^{\text{eff}}T_c \cdot {}^cT_p)_i = {}^rQ_i \quad (13)$$

Recall 0T_i is the ith configuration of the eye w.r.t. the initial configuration of the eye, and we wish to match motions of the eye and the PA frame, so that 0T_i is also the ith configuration of PA w.r.t. its initial configuration. So,

$${}^rQ_i = {}^rQ_0 \cdot {}^0T_i \quad (14)$$

Therefore, using (7):

$${}^rQ_i = {}^rQ_0 \cdot {}^tP_0^{-1} \cdot {}^tP_i \quad (15)$$

Rewriting in detail:

$$({}^rT_{\text{eff}} \cdot {}^{\text{eff}}T_c \cdot {}^cT_p)_i = ({}^rT_{\text{eff}} \cdot {}^{\text{eff}}T_c \cdot {}^cT_p)_0 ({}^tT_b \cdot {}^bT_e)_0^{-1} ({}^tT_b \cdot {}^bT_e)_i \quad (16)$$

The transformations in the italics are known or measured at system startup, and tT_b is measured from the hardware at each iteration. Now $({}^tT_b)_i$ can be calculated for each iteration as follows:

$$({}^tT_b)_i = ({}^rT_{\text{eff}} \cdot {}^{\text{eff}}T_c \cdot {}^cT_p)_0 ({}^tT_b \cdot {}^bT_e)_0^{-1} ({}^tT_b \cdot {}^bT_e)_i ({}^{\text{eff}}T_c \cdot {}^cT_p)^{-1} \quad (17)$$

Hence, Equation (17) produces the needed robot command $({}^rT_{\text{eff}})_i$

This page intentionally left blank.

4. SYSTEM INSTALLATION AND OPERATION

4.1 System Installation

Since most of the hardware utilized for this system is also utilized for other tasks not related to this project, this remote inspection prototype was designed to allow for repetitive assembly and disassembly. The general procedure is described below:

1. Install the silver, Z-shaped aluminum camera mount on the Merlin robot arm. This should be done such that the mount plate lies up against the robot itself and not against one of the aluminum ring adapters. The mount should point away from the robot to the right side, as seen while looking down the arm of the robot in calibration position.
2. Install the two VC-C1 cameras on the mount. The camera with the LASER should be mounted to the top plate. The cameras should be facing away from the arm and should be aligned carefully. A separation distance of approximately 2.5" should be set between the two camera lens.
3. Run the cable harness up the robot arm and connect the cables to the cameras. There should be one gray, round 8pin RS232-C connector for each camera. Note that these two 8pin connectors are bused together at the other end to allow for simultaneous control of the two cameras. This cable **MUST** be the one that is used when controlling the cameras in this fashion. An ordinary serial cable, or a normal bused serial cable will burn out the cameras! There should also be one small, black, round power connector and one round black, round 4pin S-video connector for each camera as well. Make sure the S-video connector labeled

“RIGHT” is connected to the top camera. Lastly, connect the white, 2pin LASER power line to the plug on the LASER. Align the black marks on the plug and the connector. Make sure the silver LASER switch is in the off (center) position.

4. Connect the opposite harness cable ends. The RS232-C cables are bused together into a 9pin D-shell connector. Connect a long, RS232-C 9pin cable to this connector and plug it into the Kiwi PC serial port, COM1. Plug the power adapters into a multi-plug extension cord and plug the extension cord into a nearby wall outlet. Tape the power adapters and multi-plug end of the extension cord securely to the shoulder of the Merlin. Run the S-video cables into the control room. Under the current configuration, the connector labeled “RIGHT” should be connected to the middle S-video plug on the far left column of S-video plugs on the back of the Sirius Video box. A short S-video cable should then be plugged into the upper S-video plug directly above this plug and connected to the VR4 helmet box right eye input. The connector labeled “LEFT” should be connected directly to the VR4 helmet box left eye input. Make sure the three DIP switches on the VR4 helmet box are set for “Sync on green, Stereo, and Y/C”. Also make sure that both of the RGB D-shell plugs on the back of the box are empty.

In the proposed configuration, both S-video connectors should be plugged into the ASD-100 analog inputs. Note that these cables may have to be swapped out for composite-style cables, depending upon the ASD-100 device input requirements. Also for the new configuration, the two white RGB cables located behind the Multi-Channel Option box will have to be

plugged into the D-shell plugs on the back of the VR4 helmet box. The cable labeled “Left(0)” and connected to channel 0 of the MCO should be connected to the left eye RGB input. The cable labeled “Right(1)” and connected to channel 1 should be connected to the right eye RGB input. Make sure that the VR4 helmet box DIP switches are set for “Sync on green, Stereo, and RGB”. Also make sure that both of the 4 pin S-video plugs are empty. Attach the other end of the white, 2pin LASER power line to an adjustable DC power supply, which is adjusted to approximately +3.2V. The white wire should be hooked to power, and the red wire should be hooked to ground, as labeled. Note that voltage dividing a +5V or similar power supply will NOT work since the LASER needs to draw so much current from the supply.

5. Strap the Geiger meter to the forearm of the robot with tape. Slide the hand held measuring unit into the Merlin gripper and close it. Run the brown 2pin voltage tap wire down the robot arm and into the breadboard, which contains the circuit for modifying the Geiger meter voltage. Locate the 8 pin LM741 op-amp chip at the top of the board. Plug the silver wire into pin 3 of this chip, and plug the copper wire into ground, as labeled. Attach a DC power supply which can provide simultaneous +5V, +12V, and -12V levels to the circuit. Attach each voltage level to the appropriately labeled wires.
6. Turn on the LASER power supply and test to see if the laser is working and is producing a very bright and clear red dot. Turn on the Geiger meter to the battery check setting and observe the remaining battery power. If sufficient, move the dial

to the next to the last setting, marked as “X1”. Hold a gamma source near the hand held measuring unit and see if a voltage is generated in the meter. Check with a multi-meter to see if a corresponding voltage is being produced on pin 6 of the LM741 op-amp chip. If so, connect a wire from this pin to channel 0 of the analog to digital input bus on the HC811 microcontroller on the mini-board. Plug the RJ-11 phone jack end of the mini-board cable into the mini-board serial port and run the cable over to Grape. Attach a 25 to 9 pin adapter and/or cable to the 25 pin end of the mini-board cable and plug the 9 pin end of the cable into Grape’s serial port, COM1. Plug the black wire of the mini-board power connector into the same ground as the Geiger meter’s copper voltage tap wire. Plug the red wire of the mini-board power connector into +5V. Make sure both of the red and green status LEDs on the mini-board are glowing. If the green LED is not lit, there is a problem with the connection to the serial port on Grape, or a problem with the serial port itself. If the red LED is not lit, there is a problem with the mini-board, or the mini-board does not have +5V power connected to it.

7. Check to make sure that all cables have enough travel for the full range of robot motion.
8. Turn on the two VC-C1 cameras and check to see if they are sending images and focusing properly.

At this point, the remote inspection system should be installed and ready for operation. Refer to Figure 7 for a picture of the fully installed and running system.

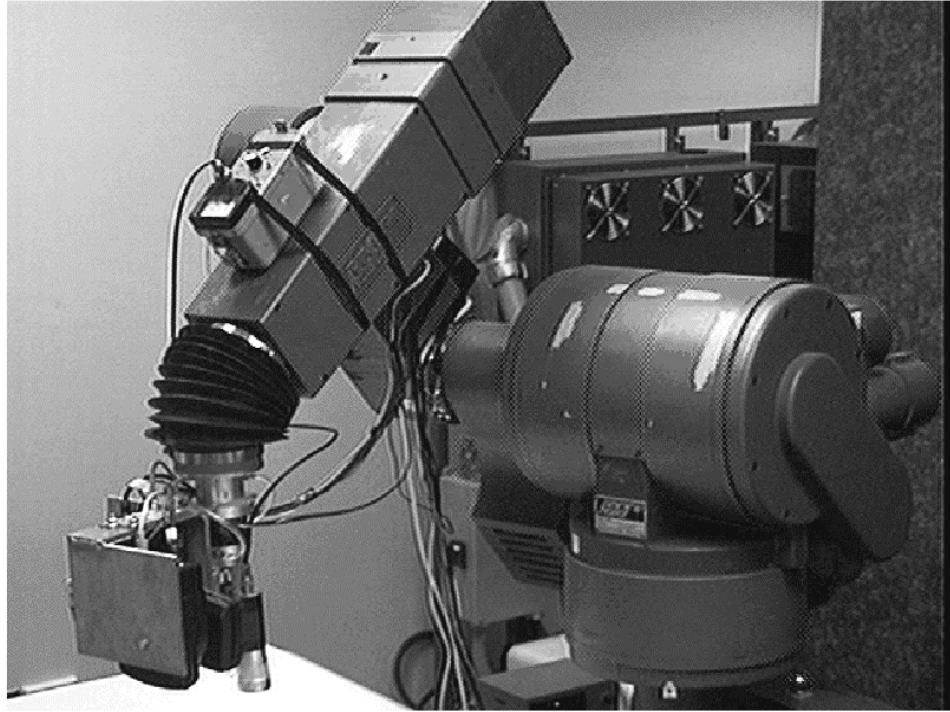


Figure 7: Fully Installed System in Operation

4.2 System Operation

Assuming the system has been properly installed and tested according to the 8 step procedure in Section 4.1 above, the following procedure should be followed in order to bring the system on-line and operate it.

1. Turn on the two power supplies, the LASER, the two VC-C1 cameras, and the Geiger meter to setting "X1". Turn the Geiger meter audio switch to "On".
2. Make sure the Merlin's motor power is off, then turn on the robot controller. Turn on the air supply to the pneumatic gripper.
3. Log into Grape, Kiwi, and Rainbow (the Onyx) as "usarc".
4. Turn on Merlin's motor power at both locations (one next to Rainbow and one next to Kiwi). Make sure the Merlin is calibrated, if necessary, with the "mcal" command on Kiwi. Then move the arm to the home position using the "move to home" command on Kiwi.
5. On Rainbow, execute the following commands to startup the USARC telerobotics software. For more information about this software, consult the documentation.

- irouter &
 - site_server -stamu &
 - vcp &
 - sms &
6. Execute the command “geiger_trx -- -srainbow &” located in the “~/src/geiger” directory on Grape.
 7. Execute the command “vcc1_rcx_dumb -- -srainbow &” located in the “~/src/vcc1” directory on Kiwi.
 8. Execute the command “converge” located in the “~/src/sirius_test” directory on Rainbow.
 9. Turn on the VR4 helmet and place it on the operator and adjust it properly for good quality stereo vision. By this point, an actual stereo image should be available inside the helmet.
 10. Make sure the area around the Merlin robot is clear of obstacles, and execute the command “rccl_remote -stamu -- -srainbow” on Kiwi. WARNING: This command makes the robot active!!

NOTE: At this point, the system is fully operational and ready to operate. The next step will place the system into operation. For your reference, Figure 7 in Section 4.1 contains a picture of the system while completely installed and in operation.

11. Execute the command “watch -stamu” located in the “~/src/test_bird” directory on Rainbow. WARNING: This command will cause the robot to shift left rapidly as it assumes the home position for remote inspection. WARNING: This command will command the robot to follow the motions of the operators head. Move slowly and cautiously. An assistant should ALWAYS be present nearby with their hand on the motor power kill button just in case something goes wrong.
12. When it is time to end the run, press Control-C in both of the windows, which are running the “rccl_remote” and “watch” programs. Execute the “move to calib” command on Kiwi. Then turn off Merlin’s motor power. This will immobilize the robot.
13. Remove the hand-held measuring unit from Merlin’s gripper and turn off the robot controller and the air supply to the gripper.
14. Turn off the silver laser switch, both VC-C1 cameras, and both power supplies. Unplug the RJ-11 phone jack end of the mini-board cable from the serial port on the mini-board. Both the red and green status lights should go dark.
15. Turn off the VR4 helmet box and kill all executing programs.
16. Log out of all computers.

REFERENCES

There were no references cited for this report.

This page intentionally left blank.

Appendix A and Appendix B: Hard copies available upon request.