Unmanned Systems, Vol. 14, No. 1, Winter 1996

Virtual Systems:

Aspects of the Air-Mobile Ground Security and Surveillance System Prototype

Hoa G. Nguyen, William C. Marsh and W. Dale Bryan

Naval Command, Control and Ocean Surveillance Center
Research, Development, Test and Evaluation Division (NRaD)
San Diego, CA, USA
P.O.C.: nguyenh@nosc.mil, (619) 553-1871

Abstract

The Air-Mobile Ground Security and Surveillance System (AMGSSS) is being developed to provide a rapidly deployable, highly mobile, extended-range surveillance capability for force protection and tactical security. In order to rapidly develop a sensors package prototype, we used modular embedded PCs and the Internet TCP/IP protocol as the communication backbone between subsystems. This allowed all subsystems to be developed in parallel from geographically distant locations. The first-cut prototype was a virtual system residing on various desktop computers connected by the Internet. As embedded hardware became available, the desktop modules were gradually replaced. The result is an extremely flexible system, where modules can be added or replaced, swapped with desktop processors for debugging, or physically relocated to distant locations, with only a few changes in a text hosts-table file. Updated software can be downloaded to the embedded systems from their virtual counterpart via FTP. Demonstrations can be set up where only the control/display software needed to be electronically sent to the viewers, to be run on office desktop PCs, controlling the remote sensors payload through the Internet. The TCP/IP protocol also allowed the late addition, with no change to existing hardware or software, of a payload simulator which presents a virtual operational environment to the control/display unit. This paper first discusses the design of the AMGSSS payload prototype including the virtual system concept, then describes the control/display module and payload simulator as another virtual system.

Introduction

The Air-Mobile Ground Security and Surveillance System (AMGSSS) is being developed to provide a rapidly deployable, highly mobile, extended-range surveillance capability for force protection and tactical security [Murphy and Bott, 1995; Nguyen, 1995]. In final form, it will consist of a High-Mobility-Multipurpose-Wheeled-Vehicle (HMMWV) -mounted base station and three air-mobile platforms capable of transporting remote sensing packages to observation points within a 10 km radius. Figure 1 depicts the system being deployed on the battlefield.

AMGSSS system objectives include: high mobility, insensitive to intervening terrain, remote operations over existing low-bandwidth tactical radio links, long-endurance surveillance capabilities, easily scaleable functionality, and the ability for one operator to supervise several remote systems.

A prototype mission payload and control station has been developed over a period of six months. To allow fast prototyping and to achieve a small, portable demonstration package, the following design methodologies were employed:

- Use of miniature embedded Personal Computer (PC) components, with DOS and Windows-based software. This allowed us to exploit in-house expertise in PC programming, and a simple transfer of the finished code to the embedded system hardware.

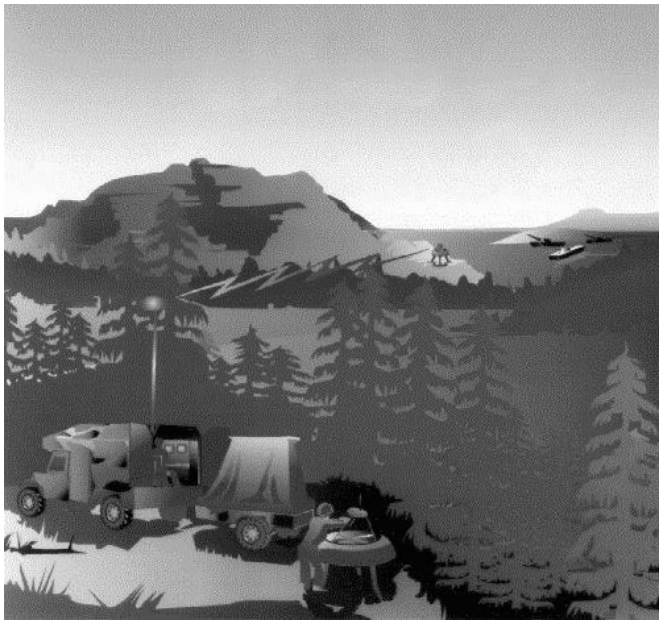

Figure 1. Artist's concept of AMGSSS deployment on the battlefield.

- Decomposition of the complex system into functional tasks operating on dedicated PC components, connected by Ethernet and TCP/IP (Transmission Control Protocol/Internet Protocol) networks. This allowed parallel development of subsystems, easy incremental testing, and a simple final integration.

These decisions turned out to be auspicious. The resulting system exceeded original expectations in flexibility, usefulness, and ease of modification. The flexibility provided by TCP/IP also allowed the late addition of a payload simulator residing on the same computer as the control/display station. This paper describes the overall system design including a virtual system developed as a first-cut prototype, then touches on the control/display and payload simulator, an extremely portable demonstration package--so portable it can be electronically mailed to you.

The AMGSSS Mission Payload Prototype

Figure 2 is the AMGSSS Mission Payload Prototype system functional diagram, showing connections between each subsystem computers. There are two computers at the control end (a laptop running the control/display and optional payload simulator, and a tactical radio/Ethernet modem), and three computers on the remote payload side (payload processor, image processor, and tactical radio/Ethernet modem). Interprocessor communications is via TCP/IP using Ethernet cables and tactical-radio-frequency modems.

All subsystems except the control/display laptop use industrial PC/104-form-factor (~10 cm x 10 cm footprint) computers. These are PC-compatible DOS-based products using highly integrated components produced for notebook computers, and thus are both low-power and economical. Communications hardware include credit-card sized PCMCIA Ethernet PC Cards and Tactical Communications Interface Modules (TCIMs, one for each radio controller computer) acting as modems interfacing the radio controller computers to the handheld tactical radios [Martin and Bryan, 1995].

Figure 3 shows the laptop computer being used as the control/display (C/D) station, together with a radio-frequency shielded package containing the radio controller and handheld radio. The two component packages are connected via an Ethernet coaxial cable.

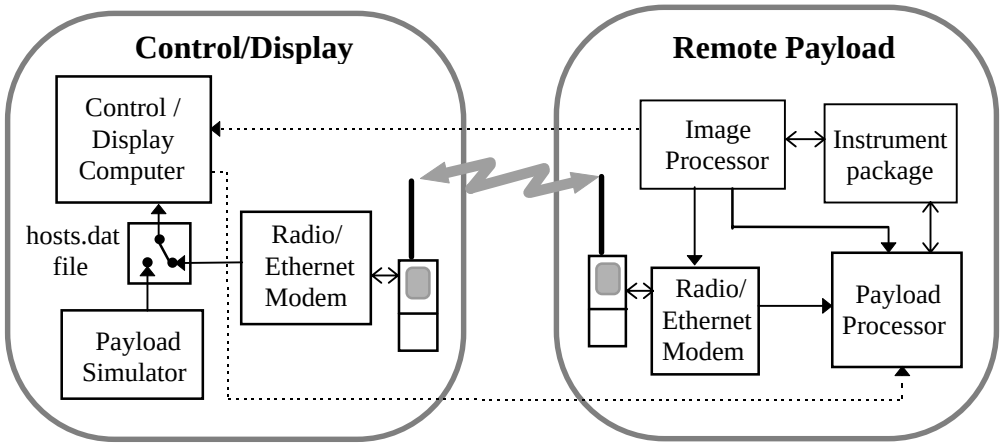

Figure 2. System functional diagram.

Figure 3. The prototype control/display computer and radio.

Figure 4 shows the prototype sensor payload packaged in a portable configuration. The package includes a visible-light video camera, an infrared video camera, and a laser rangefinder, all mounted on a pan-and-tilt unit. Additionally, there is a serial port connection to an optional portable Northrop-Grumman Acoustic Unattended Ground Sensor. The PC/104 computers are housed in the compartment below the sensors.

The system employs a message-passing distributed processing architecture. Messages and commands are passed between the subsystems via the Ethernet cables and via the radio link between the control and payload ends. Each computer has an Internet (IP) address, and uses a hosts data file ("hosts.dat") to find the Internet addresses of the other modules. The network protocol is set up to automatically switch to an all-direct Ethernet configuration, by-passing the radios, upon detecting the absence of the radio modems. This was intended as an early developmental configuration, but proved very useful throughout the developmental cycle and later as a valuable demonstration tool.

Figure 4. The prototype sensors payload package.

Advantages of using an Ethernet TCP/IP interprocessor interface

Some benefits which arose from the selection of the Ethernet TCP/IP communication protocol to connect subsystems are:

1. Distributed, parallel development from many sites over long distances: The AMGSSS subsystems were developed by different groups of engineers at several NRaD sites spaced many miles apart. However, the system did not have to be assembled in one place before testing can take place. The main NRaD Internet backbone was used for subsystem connectivity during early development. This capability can forseeably allow a system to be jointly developed by several companies with much greater ease.

2. Virtual Systems: The TCP/IP connectivity allowed a "virtual" system to be developed and tested even before the PC/104 hardware arrived. Software was developed for each subsystem, which resided on desktop computers, and "remotely integrated" and tested over the Internet. When the actual hardware became available, we simply transfer the software via FTP (File Transfer Protocol) to the various PC/104 single-board computers. Since the final system also uses Ethernet and TCP/IP for connectivity, no changes were needed in the software. In fact, the "virtual system" still existed after the real system was assembled, so that modifications can be readily accomplished and tested before being downloaded into the real system. Figure 5 shows the virtual system used during our system development process.

3. Ease of demonstrations: The same concept above applies to the entire world-wide Internet. Instead of miles apart, the subsystems can be anywhere in the world, connected together via the Internet, with no change to the software. Thus, the control/display program can be executed from any computer with access to the Internet, anywhere in the world, controlling the remote payload half the world away. The hardware need not be shipped anywhere for a live demonstration, only the software program, which can be sent via electronic mail or via FTP.

To prevent possible "hijacking" of the remote payload by an unwanted host, a host/client scheme was employed which forces the host program to be acknowledged by the payload. As shown in figure 2, each TCP/IP link originates at a client and ends at a host. A subsystem can act as a host to some, while as a client to others (e.g., the control/display computer is host to the image processor and radio controller, but a client to the payload processor). A host service need not take any action, except to be open to connections. A client service must actively seek a connection to a host. The client services read a hosts data file

("hosts.dat," a text file available for each computer) to determine the Internet names of the hosts they are supposed to connect to. By making the control/display station a host to the image processor, we effectively lock out any hijacking attempts. The Internet name of the host computer must be in the host data file of the image processor in order for the connection to be made.

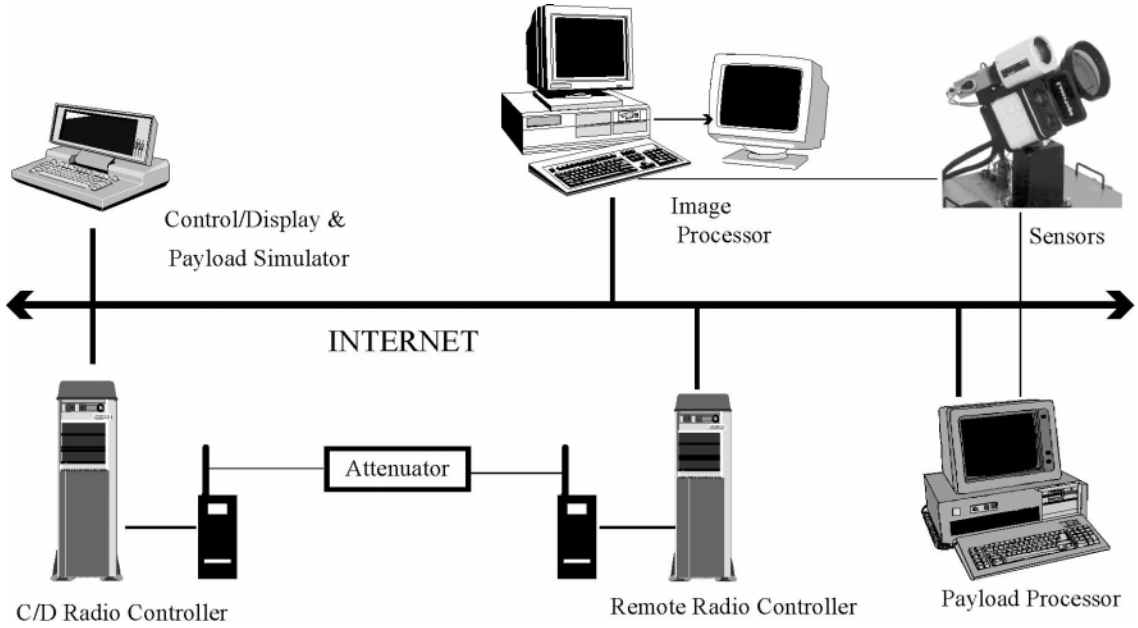

Figure 5. The "virtual system" using the Internet for interconnections

4. Ease of debugging and software modification in the field: If a bug is found in one of the PC-104 - based subsystems, debugging can be accomplished on its counterpart desktop component. A few text changes in the hosts tables allow the substitution of a virtual component for its equivalent embedded computer. More powerful debugging tools can be used on the desktop subsystem, and debugging is accomplished over a hybrid system, consisting of embedded components in the field plus one or a more subsystems on the desktop virtual system at a development facility. The finished code is simply transferred to the embedded computer via FTP.

5. Ease of expansion/substitution: TCP/IP and message-passing distributed processing also allows subsystems to be added with ease. A new subsystem component would only require an additional item in the hosts data table, and perhaps additional messages defined for the specific new subsystem. No hardware changes to the existing subsystems are required. Subsystem substitutions are also easily accomplished by changing the TCP/IP addresses in the hosts data table. We added a payload simulator late in the development cycle to help debug the control/display software before the payload was completed. By changing the Internet name for the radio modem in the control/display's hosts data file, we redirect the control/display to a payload simulator instead of the remote payload. Furthermore, since the control/display program runs under Microsoft Windows, a multi-tasking environment, the payload simulator can also be another Windows program running on the same computer. We only have to use the same Internet address for both the control/display and the radio computer in the simulator's hosts data file. The simulator program looks for the address of the control/display computer in the hosts data file, finds the address which happens to be the same computer it is on, and loops back to its own computer looking for the connection to the control/display.
Since the payload simulator replaces the entire payload and provide an appropriate response to each command and query from the control/display, it can act as more than just a debugging tool. Fully loaded with information about the total space surrounding it, including 360-degree panoramic images, preprogrammed motion and associated image streams, it can provide the control/display with a virtual

environment in which to operate. Again, the AMGSSS concept can be demonstrated without physically taking the control/display or the payload anywhere--the control/display and payload simulation software can just be electronically sent to the recipient, to be enjoyed at his/her own pace on his/her own computer.

The rest of this paper will examine the AMGSSS control/display and payload simulator as another self-contained virtual environment.

Control/Display and Payload Simulator software environment

Both the Control/Display program and the Payload Simulator Program operate under the Microsoft Windows Graphic User Interface (developed under Windows 3.1, although they also work under Windows 95). Both were developed using the Microsoft Visual C++ compiler. The Microsoft Foundation Class library was used to provide the basis for object-oriented programming. We used Trumpet Winsock 2.0 to provide the TCP/IP interface under Windows 3.1 (Windows 95 provides its own TCP/IP capabilities).

Control/Display program

A goal of the AMGSSS mission payload prototype is to achieve a compact, portable demonstration system which can easily be transported as accompanied luggage on commercial air flights. Toward this goal, we selected the IBM Thinkpad 755 CD laptop computer (a 100MHz 80486) to host the control/display function, and designed a powerful, full featured operator's console program for it. The Thinkpad was chosen for its color display, PCMCIA ports, availability of a docking station and input video overlay capabilities. The latter two functions were selected for future development (when live video will be added) but are not used for the current configuration.

Figure 6 shows the top-level functional breakdown of the Control/Display program. At the heart of the program is the database ("document" in Visual C++ lingo), containing information about images collected, status of all sensors and subsystems, maps, and alerts received. The database supports three "views": the status view with status of all 3 AMGSSS remote payloads (although only one payload has been developed for this demonstration), the geographic view with sensor information overlaid on top of a map of the operational area, and the situation view which gives the operator a "big picture" of how all the images collected, programmed motion and acoustic scanning programs, and alerts which have come in are interrelated. In addition, there are two independent windows (displaying the panorama scene and any high-resolution snap shot) which can be displayed on top of any view. Communications between the database and the views are through Windows message handling mechanisms.

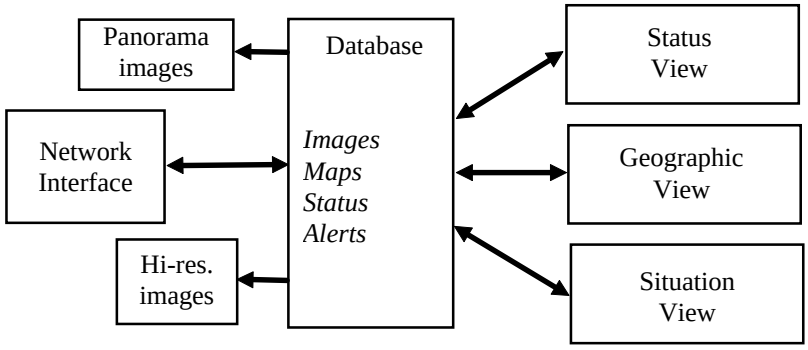

Figure 6. The Control/Display program functional breakdown.

Status view:

The status view is a text screen providing status information on all subsystems, sensors, and radio links. Status messages on the radio link is provided by the radio computers (to the control/display computer on the control side, and to the payload processor on the remote side) every 10 seconds. The payload processor assembles all sensor and remote subsystem status messages and forward them over the radio link to the control/display every 30 seconds. In addition to the information presented by the status view, a green status light in the upper left hand corner of all views acts as the system's heart beat. It blinks every time a message comes over the radio link, and becomes gray if the link is down. This enables the operator to monitor the overall system status even when the display is showing the geographic or situation view.

Geographic view:

The geographic view revolves around a map of the operational area. Initially, we used a vector map to allow display at multiple scales. However, we found that the vector maps we could find (in this case, the Central Intelligence Agency's World Database maps) were of insufficient detail. So we switched to multi-resolution scanned raster topological maps. Figure 7 is a screen capture of the geographic view showing the topological map, overlaid laser ranging results (the lines ending in stars--clicking on the stars brings up a box with range, time, and azimuth and elevation information), motion alert (the triangle) and visual angle of displayed images (the shaded wedge corresponds to the center image in the panorama). The panorama images are not part of the geographic view itself, and can be pulled up on any of the three views. All screens and operational images showed here have been obtained during a system operational test at Mission Trails Regional Park in San Diego, California, in late October, 1995.

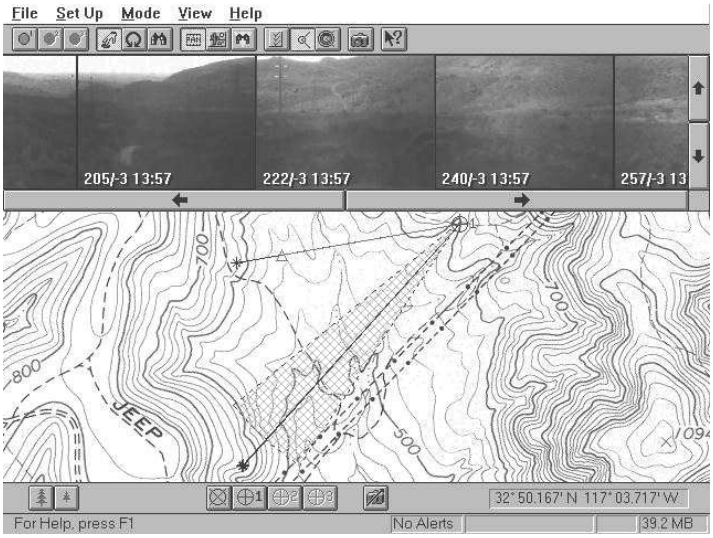

Figure 7. Control/display screen showing the geographic view and panorama images.

Situation view:

The main function of the situation view is to present a clear picture of the relationships between all collected images, planned panorama and motion scans, laser ranges and alerts currently in the system database. The view is laid out as three concentric rings representing +45 degrees, 0, and -45 degrees in elevation. Superimposed on these rings are wedge sections correlating to images collected or scans to be executed. Figure 8 shows two rings of panorama images (overlapping in elevation), the lighter box on the inner ring corresponds to the center pane in the panorama image strip above. The operator can select the amount of information to be displayed on the view by clicking on the items listed at the lower left corner of the screen.

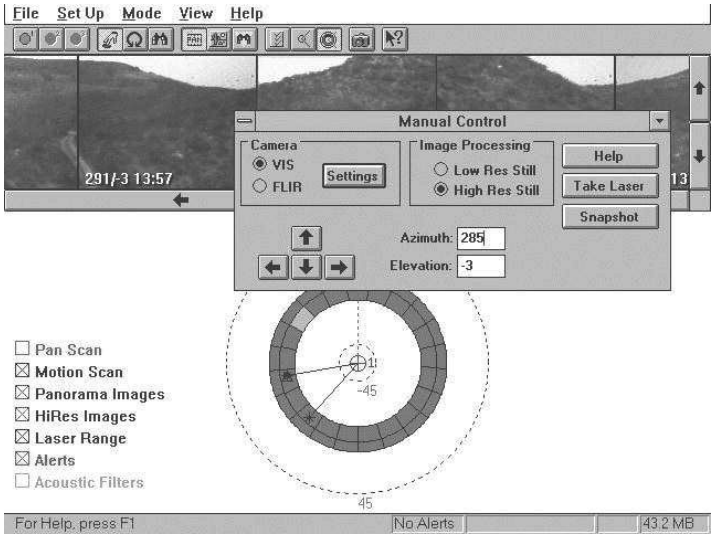

Figure 8. Control/display screen showing the situation view, panorama images, and a manual control dialog box.

Programmed execution and responses:

Besides the capability for manual control of the sensor package (Figure 8 shows a typical manual control dialog box), the AMGSSS control/display program also offers several program modes to simplify the operator's work load. The operator can program the system to perform panoramic sweeps to get a feel for the general landscape (Figure 8 shows two panorama rings, at -3 and +5 degrees in elevation). Likewise, the system can be programmed to look for motion at certain locations and at some sensitivity, using the visible or infrared camera. Acoustic detections can also be filtered. Only certain categories (ground vehicle, jet, propeller planes, etc.) or all can be passed to the operator.

Programmable responses to alerts include automatic capture and transmission of a high-resolution image, automatic range determination via the laser rangefinder, video streaming, or a return to manual control. Video streaming currently uses JPEG compression and frame differencing. Work is in progress to migrate to the H.261/H.263 video-teleconferencing standards. Figure 9 shows a motion alert and associated high-resolution image showing a moving HMMWV on a dirt road.

Payload simulator

The payload simulator is a Windows-based program. It can interact with the control/display (C/D) program in three configurations: (1) direct connection of the C/D computer and simulator computer by an Ethernet cable, (2) both computers connected to the Internet, or (3) both programs residing on the same computer. Only one change in the simulator's "hosts.dat" text file has to be made to switch from one configuration to any other.

The payload simulator provides the control/display program with a virtual environment in which to operate. It contains images covering 360 degrees in azimuth around the virtual payload, with elevations up to +/-30 degrees (the images were obtained using the real payload). Two types of images are required by the control/display, low-resolution images (160 x 120 pixels) are used in panoramas, while high-resolution images (248 x 240 pixels) are returned with alerts. Both can also be requested manually as snapshots. At full field-of-view, each image covers an area of 17 degrees by 12 degrees, and compresses to an approximately 1.5-K byte JPEG file. To provide complete panorama coverage, we need five rings of 21 images each, totaling 157 KB. Full-resolution images are JPEG files of approximately 15 KB each.

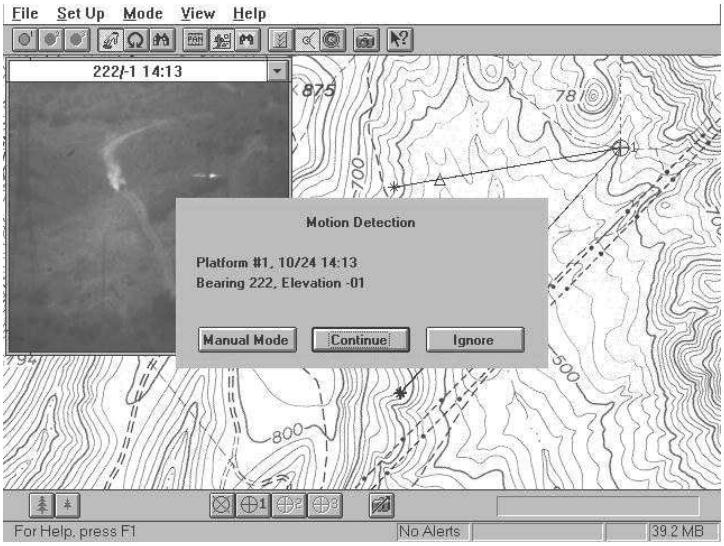

Figure 9. Control/display screen showing motion detected.

Having a full-resolution image version of each low-resolution image requires a total of approximately 1.7 MB. Providing images taken with both visible and infrared cameras will approximately double this requirement (for an infrared lens with similar field-of-view). This is still well under the capacity of any hard disk.

The payload simulator normally is in a reactive mode, providing the control/display program with data which it asked for, such as individual images, panorama scans, laser ranges, etc. However, if the control/display operator selects motion or acoustic scanning mode, the payload simulator will provide alerts drawn from a predefined alerts list, although at somewhat random times. The user can also manually generate a motion or acoustic alert from the simulator (see Figure 10), as a debugging tool. Another debugging capability provided by the payload simulator is the display of messages passed between the control/display and the payload, such as shown in Figure 10. Responses from the simulator can be set to various speeds using the Timing menu, simulating AMGSSS operations over the Internet or under various radio conditions.

Adding the payload simulator to the system has been simplified by the pre-existence of the TCP/IP interprocessor communications scheme, requiring no change in any existing software or hardware. Besides adding debugging capabilities, the control/display and payload simulator provides a complete virtual system for a software-only demonstration of the AMGSSS operating concepts. The entire AMGSSS demonstration, including image database, is compact enough to be obtained via FTP over the Internet for private demonstrations at any time.

Conclusions

Using a modular system architecture and the TCP/IP communications protocol for inter-processor connectivity, complex systems can be developed much more quickly and efficiently. Subsystems can be developed in parallel by different groups (even by geographically-distant companies) at their separate work sites, and integrated only over the Internet. The final system may not even be a single piece of hardware but scattered all over the world, connected only by the global Internet. Since the communication protocol is not hardware specific, subsystems can often be software programs residing on computers of opportunity. Late additions and substitutions of components are also made much simpler, often not involving any changes to the rest of the system.

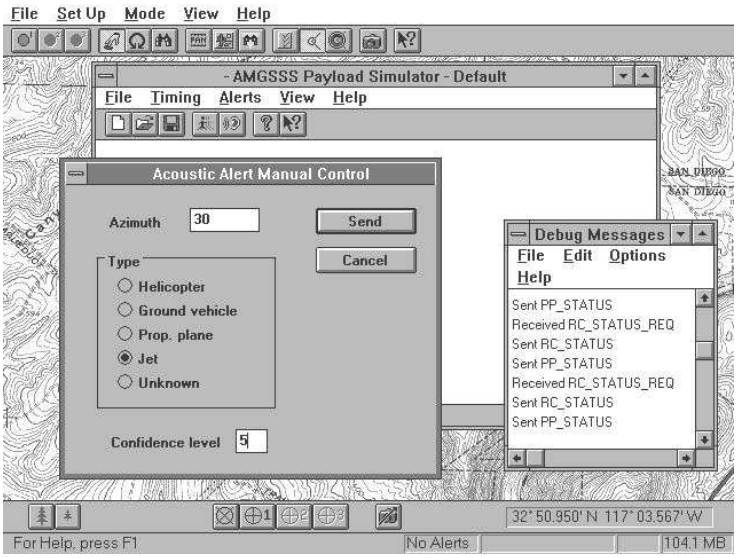

Figure 10. Payload simulator on top of Control/Display program.

We have provided two examples of such systems. The first is the virtual developmental system which led to the AMGSSS Mission Payload Prototype, and the second is a specific outgrowth of the prototype, a virtual demonstration environment using a payload simulator. Both have resulted from modular system architectures and the TCP/IP interprocessor protocol.

Acknowledgments

This work was supported by the U.S. Army's Physical Security Equipment Management Office (PSEMO) and the Office of the Undersecretary of Defense (Acquisition, Tactical Systems/Land Systems).

References

Martin, B and D. Bryan, "Low-cost Miniature Interface and Control Systems for Smart Sensors, Tactical Radios, and Computer Networks," IEEE Military Communications Conference (MILCOM 95), San Diego, CA, Nov. 6-8, 1995.

Murphy, D. and J. Bott, "On the Lookout: The Air Mobile Ground Security and Surveillance System (AMGSSS) Has Arrived," *Unmanned Systems*, Vol. 13, No. 4, Fall 1995.

Nguyen, H. G., "Air Mobile Ground Security and Surveillance System," in *NRaD Robotics,* World-Wide-Web URL: *http://www.nosc.mil/robots/,* 1995.