Submitted to: International Journal of Robotics Research
Special Issue on Field and Service Robotics

# From Laboratory to Warehouse: Security Robots Meet the Real World

H. R. Everett
Douglas W. Gage

Space and Naval Warfare Systems Center
San Diego, CA 92152-7383

**ABSTRACT**

The MDARS robotic security program has successfully demonstrated simultaneous control of multiple robots navigating autonomously within an operational warehouse environment. This real-world warehouse installation required adapting a navigational paradigm designed for highly structured environments such as office corridors (with smooth walls and regularly spaced doorways) to a semi-structured warehouse environment (with no exposed walls and within which odd-shaped objects unpredictably move about from day to day). A number of challenges, some expected and others unexpected, were encountered during the transfer of the system first to a beta-test/demonstration site and then to an operational warehouse.

This paper examines these problems (and others previously encountered) in a historical context of the evolution of navigation and other needed technologies, and the transition of these technologies from the research lab to an operational warehouse environment. A key lesson is that system robustness can only be ensured by exhaustively exercising its operational capabilities in a number of diverse environments. This approach helps uncover latent system hardware deficiencies and software implementation errors not manifested in the initial system hardware or initial development environment, and identify sensor modes or processing algorithms tuned too tightly to the specific characteristics of the initial development environment.

## 1. BACKGROUND: SITE SECURITY ROBOTS AND THE MDARS PROJECT

The site security (sentry) application has a number of features which exploit the strengths and avoid the weaknesses of mobile robots:

- The operating environment is known in advance, is under friendly control, and can be tailored to some degree to support robotic operations -- this is decidedly not the case with battlefield applications such as reconnaissance, surveillance and target acquisition (RSTA).
- Human operators to backup the robotic system when it encounters problems are readily available in the existing on-site human guard force (which will be augmented versus replaced by the robots).
- Experience-based costs of inventory shrinkage and non-robotic security measures provide a sound and credible basis for robotic sentry system cost/benefit tradeoffs.
- Robots do not get bored during long hours of surveillance, leading to reduced vigilance.
- Robots don't participate in "inside jobs."

The Mobile Detection Assessment and Response System (MDARS) is a joint Army-Navy development effort to provide an automated intrusion detection and inventory assessment capability for use in Department of Defense (DoD) warehouses and storage sites. The program is managed by the Product Manager - Physical Security Equipment (PM-PSE) at Ft. Belvoir, VA, with overall technical direction provided by the Space and Naval Warfare Systems Center, San Diego (SSC San Diego), formerly the Naval Command Control and Ocean Surveillance Center, Research Development Test and Evaluation Division (NCCOSC RDTE DIV, or NRaD).

The MDARS goal is to provide multiple mobile platforms that perform random patrols within assigned areas in order to: 1) detect anomalous conditions such as flooding or fires; 2) detect intruders; and 3) verify the presence and determine the location of inventoried items through the use of specialized RF transponder tags. A separate exterior development effort (Heath-Pastore and Everett 1994) addresses the requirements of outdoor DoD storage sites; this paper is concerned strictly with the MDARS Interior (MDARS-I) system and its operation in typical industrial warehouse environments (Figure 1).


**Figure 1. Early prototype of MDARS Interior robot undergoing beta testing in the Camp Elliott warehouse facility in San Diego, CA.**

The MDARS design is driven by a number of characteristics of the application domain:

- MDARS must function as a key component of a complete security system that also includes fixed detection capabilities and human security guards.
- The patrol coverage of a large number (expandable up to 32) of mobile robotic platforms must be controlled and coordinated to minimize opportunities for undetected intrusion, even by insiders.
- Both the interior warehouse and exterior storage site environments require navigational capabilities intermediate between those of an unknown and dynamic environment (e.g., battlefield) on the one hand, and a completely structured and static environment (e.g., hospital corridors) on the other.

At the highest level of system description, the two areas of the design that particularly reflect the requirements of the MDARS application are: (1) the distribution of processing functionality, especially navigational planning; and (2) the choice of sensors and processing techniques to support vehicle navigation in semi-structured environments.

The mappable nature of the operating environment, the relatively low frequency of exception conditions, the need to achieve coordinated coverage of multiple platforms, and the requirement to simultaneously support up to 32 platforms suggested a navigation control strategy based on centrally planned routine patrol routes, with any deviations handled on an exception basis. This feature is implemented in MDARS via the Multiple Resource Host Architecture (MRHA), in which a Supervisor computer, a pool of centrally located Planner/Dispatcher computers, one or more (human guard) Operator Stations, and a Link Server (to route messages to and from the robots via RF modems) are all interconnected via an Ethernet LAN (Figure 2).

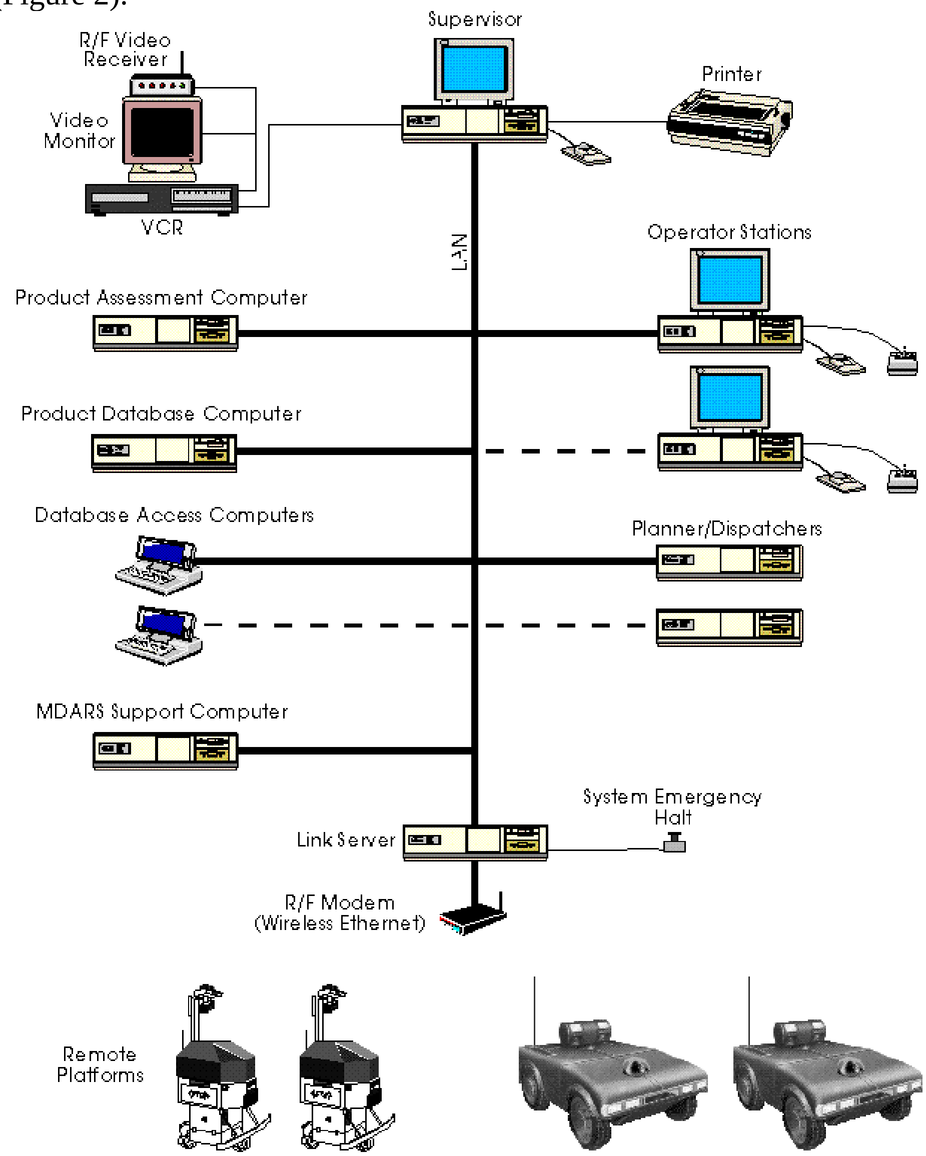


**Figure 2. Block diagram of the MDARS Multiple Host Resource Architecture (MRHA).**

When a robot becomes idle, the Supervisor assigns a Planner/Dispatcher from the resource pool to plan and download a new path. The robot then autonomously executes this set of commands until completion, or until an exception condition is encountered, whereupon the process repeats. Meanwhile, the Planner/Dispatcher is released back to the resource pool for allocation to another platform, so that a small number of Planner/Dispatchers can handle a large number of robots. When an event is deemed to require the attention of a human operator, the Supervisor assigns an Operator Station, which displays the appropriate information to the guard and provides an interface for command entry. Details of the MRHA architecture can be found in previous papers (Heath-Pastore and Everett 1994; Laird, Everett, and Gilbreath 1993; Everett et al. 1993) .

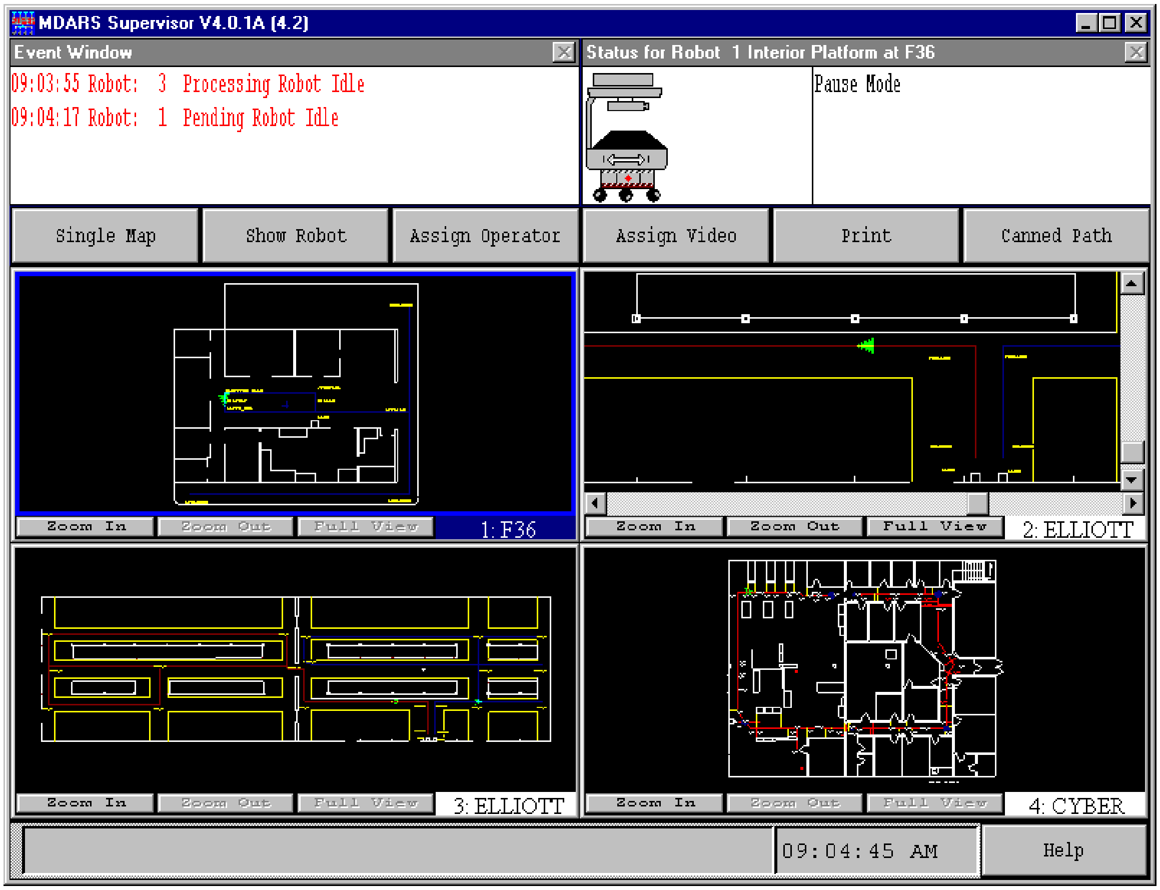


**Figure 3. Supervisor screen display of the Multiple Resource Host Architechture.**

## 2. MDARS NAVIGATIONAL CAPABILITIES

### 2.1 Baseline MDARS-I Navigational Capabilities

The navigational capabilities of the MDARS Interior vehicle build upon those incorporated in the Cybermotion *K2A Navmaster* mobility platform -- preplanned *virtual paths* are downloaded to the platform for execution in various guidance modes. Cybermotion's *virtual path planning* has been supplemented in MDARS-I by SSC-developed *unrestricted path planning.* When an obstacle in the intended path is detected by the robot's array of acoustic sensors, the robot halts and notifies the MDARS Supervisor computer, then uploads its history buffer of recent sensor data to an assigned Planner/Dispatcher, which in turn plans an alternate "unrestricted" path to avoid the obstacle. The details of how this process is supported by the MRHA architecture have been previously documented (Heath-Pastore and Everett 1994; Laird, Everett, and Gilbreath 1993; Everett et al 1993; Holland, Everett, and Gilbreath 1990).

The primary movement command in Cybermotion's virtual path programming is the RUN instruction, which has as its arguments the coordinates of the desired destination and the desired target speed. Given only the RUN instruction, a vehicle will turn toward the destination and accelerate to the prescribed speed. Using a ramped velocity profile, the vehicle will then slow in order to reach a smooth stop at the destination. (The derivative RUNON command operates like RUN except that a smooth transition is made to a following RUN or RUNON command, rather than stopping at the destination.) A *K2A* platform with no sensor will execute a RUN or RUNON solely through odometry: every time the vehicle moves a fraction of an inch, the algorithm reads the drive and steering encoders, calculates the relative rotation and translation, then updates the vehicle's current position and heading estimates.

Cybermotion's *Navmaster* configuration adds a sensor turret with RF data link, a sonar system, and a docking beacon to the *K2A* mobility base. In addition to detecting potential obstacles, the ultrasonic sensor suite allows the vehicle to correct its position and heading estimates on-the-fly, triggered by preceding a RUN or RUNON instruction with a WALL, HALL, or APPROACH instruction. The WALL instruction causes the vehicle to monitor the relative offset of a WALL on either side of the vehicle and roughly parallel to its intended path. As the RUN executes, range points are collected along the wall. If these range points fit a straight line within programmable limits, corrections are made both to the heading and to the lateral offset from the wall. HALL simultaneously looks for walls on both sides, while APPROACH corrects the vehicle's dead-reckoned position as it approaches normal to a wall or other fixed reference entity, such as a structural post. Virtual paths may be directly programmed in text, or may be automatically generated by drawing paths on a CAD map of a building (Holland, Everett, and Gilbreath 1990).

## 2.2. Navigational Enhancements for the Warehouse Environment

Cybermotion's wall-following (WALL and HALL) navigational modes were developed initially for structured environments such as office hallways, with many unobstructed walls that allow continuous closed-loop correction of odometric errors. Because the warehouse environments in which MDARS-I will be installed do not in general have such exposed wall surfaces, a supplementary navigational re-referencing scheme is needed.

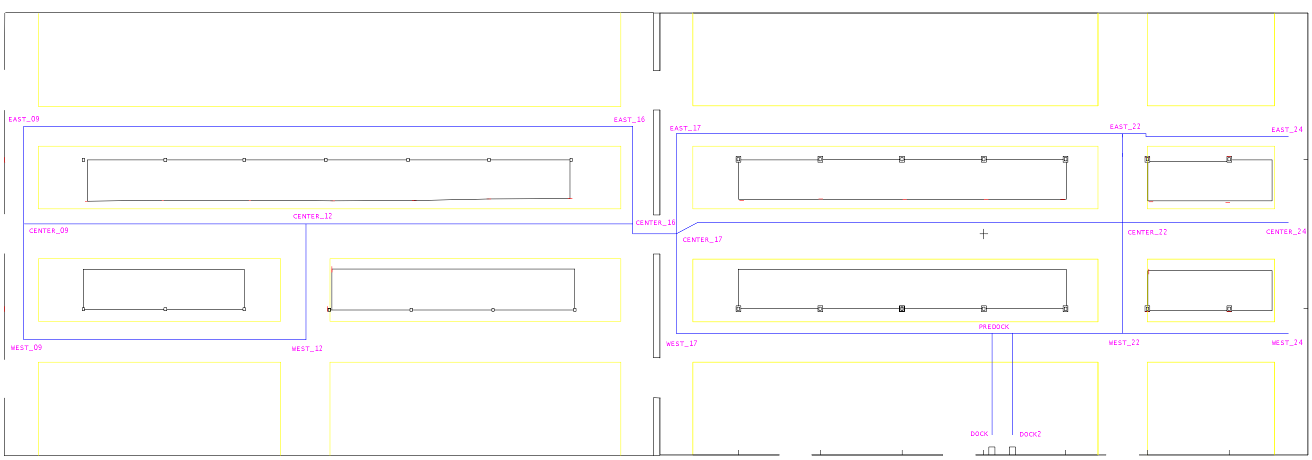


**Figure 4. Map of MDARS-I test/demonstration warehouse site at Camp Elliott in San Diego, CA. Long storage racks supported by posts are surrounded by rectangles that mark the boundary between designated storage and the allowed robot (and forklift) operating area (which includes the network of interconnected virtual paths).**

Many different solutions have been proposed (and a significant number actually implemented) for the problem of determining a robot's actual position within its operating area. The approaches differ greatly in accuracy, operating area size and characteristics, and costs in terms of required sensors, processing, communications, and installation complexity (Borenstein, Everett, and Feng 1996). Typically some sensor is used to measure the distance and/or the bearing to some chosen landmark(s) in the environment. Existing landmark features may be used opportunistically, or reference targets with cooperating characteristics may be deliberately emplaced at surveyed locations. Such targets may participate in the sensing process either actively or passively. The sensing system may involve ultrasonics, visible or IR light, or RF. Everett (1995) provides a comprehensive overview of acoustical, optical, and RF position-location techniques and available systems.

One MDARS-specific navigational re-referencing technique implemented on the Interior robots, *lateral post detection*, is a hybrid scheme combining: (1) IR proximity sensors that determine the presence of a cooperative target of known location at a precisely determined bearing, with (2) ultrasonic range sensors that determine the distance to the target. The scheme is inexpensive in terms of both sensor hardware and processing power requirements, and fits smoothly into the Cybermotion *virtual path planning* structure (Everett 1995).

Short vertical stripes of 1-inch retroreflective tape are mounted on various immobile objects (usually structural support posts) on either side of a virtual path segment. The X-Y locations of these tape markers are encoded into the virtual path program as parameters (distance along the path where the stripe should be detected, and lateral distance expected from path to stripe) for a new STRIPE navigational mode implemented by Cybermotion on the *Navmaster* robot. Installation of the tape takes only seconds, and there is little chance of damage to the unobtrusively flat tape from passing forklifts.

A Banner Engineering *45BB6LLP* retroreflective laser-based proximity sensor is mounted on each side of the *Navmaster's* turret, pointed perpendicular to the robot's direction of travel. As the robot passes the stripe, the sensor triggers a "snapshot" virtual path instruction that records the current side-sonar range values. The longitudinal position of the platform is updated to the known marker coordinate, while lateral position is inferred from the sonar data, assuming that both values fall within specified tolerances. Figure 5 depicts the geometry of the scheme in operation.

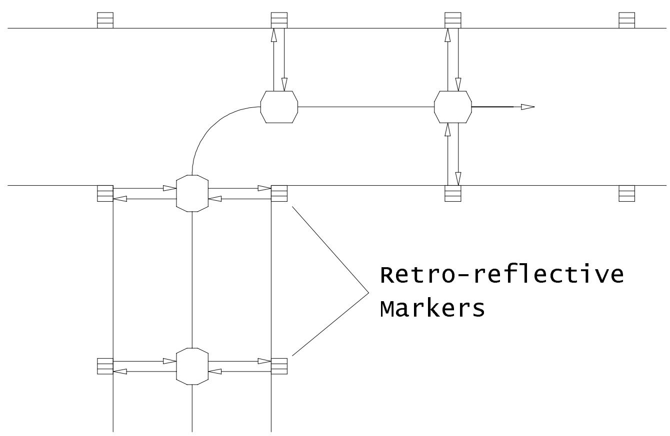


**Figure 5. The geometry of the MDARS-I lateral post detection scheme.**

The accuracy of the longitudinal (marker) correction is much higher than that of the lateral sonar readings, since the polarized Banner sensor responds only to the presence of a retroreflector and ignores reflections from even highly specular surrounding surfaces, while the ultrasonic energy from the sonar will echo back from any reflective surface encountered by its relatively wide beam. Slightly protruding objects in the vicinity of the tape (not unexpected in a warehouse environment) result in a shorter measured lateral range value, but the long-term effect on X-Y bias tends to be averaged out.

This lateral post referencing concept was implemented at SSC in May 1994 and later tested in an operational warehouse environment at Camp Elliott in San Diego, CA. The *Navmaster* robot was run continuously back and forth along a 150-foot path, with seven tape markers set on posts 20 feet apart. No other navigational referencing instructions were contained in the path program. Initial heading and location errors were quickly nulled out after the second or third post was detected, and accumulated errors remained essentially insignificant for the remaining length of the path. Each time the robot reversed course at the end of a run, some noticeable heading error was introduced, but this error was then quickly eliminated as lateral-post updates were processed on the return leg. Introduction of objects protruding as much as 16 inches into the aisle immediately adjacent to the retroflective tape caused an expected shift in the actual path followed, but introduced no instabilities in the overall path-following process.

## 3. NEW ENVIRONMENTS PRESENT NEW CHALLENGES

The transfer of the lateral post scheme to the beta-test facility at Camp Elliott was not immediately successful, however. Several completely unanticipated and unrelated problems appeared during initial testing at the warehouse, even after thorough subsystem checkout and integration in the laboratory. In this section, we will place these problems in the historical context of other issues encountered while moving various robots to new operating environments, observed over more than a decade of development of MDARS and its precursor security robots, ROBART I and ROBART II.

### 3.1 Pre-MDARS security robot development

Generally regarded as the world's first autonomous security robot, ROBART I was developed in 1982 at the Naval Postgraduate School (Everett 1982). While rich in collision avoidance sensors, this research platform had no sense of its absolute location, and was thus strictly limited to navigating along reflexive patrol routes defined by the relative locations of individual rooms, while periodically returning to a recharging station by homing on an IR beacon.

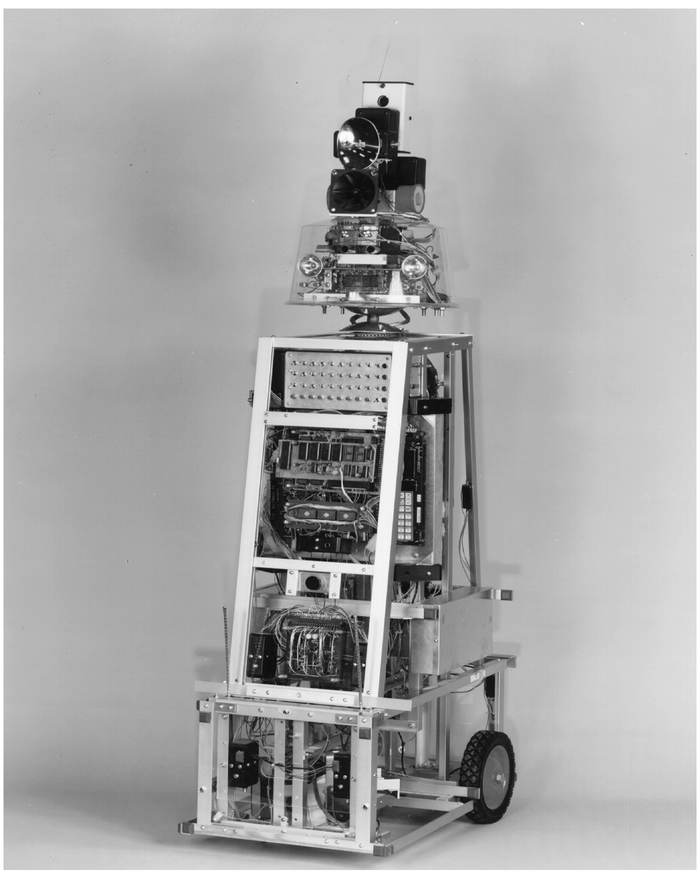

**Figure 6 ROBART I (1980-1982) was developed at the Naval Postgraduate School in Monterey, CA.**

**Figure 7. ROBART II (1982-1992) employed 132 external sensors for navigation and intrusion detection.**

The second-generation follow-on to ROBART I (Figure 6) was ROBART II (Figure 7), which incorporated a multiprocessor architecture and augmented sensor suite to support enhanced navigation and security assessment capabilities. The addition of a world model allowed ROBART II to: (1) determine its location in world coordinates, (2) create a map of detected obstacles and (3) better perform multisensor fusion on the inputs from its suite of security and environmental sensors (Everett et al. 1990). ROBART II was transferred to SSC San Diego (then Naval Ocean systems Center, NOSC) in 1986 and used as a testbed for the development of obstacle mapping and other sensor fusion and navigation capabilities.

**Problem:** When ROBART II was first moved from a small laboratory space to a large test bay in January 1990, a software data conversion rollover error was uncovered that occurred whenever the sonar range exceeded 34 feet. This software bug had lain undetected for over 3 years, even under heavy use, due to the constricted confines of the original lab space. Because of this long period of seemingly reliable performance, the true nature of the bug was not correctly analyzed until after several other suspected causes were systematically eliminated: low battery voltage, electrical supply noise, acoustic interference between sensor transducers, and even acoustical interference from chirping crickets (Everett et al. 1990).

**Problem:** When ROBART II was later relocated to a new laboratory building in 1993, the performance of its reflexive (safeguarded) teleoperation mode (Everett and Laird 1990) (in which a human operator's teleoperation inputs are modulated by inputs from collision avoidance and other sensors) was observed to be substantially degraded: the robot would approach much closer than intended to a wall before turning away. This problem occurred because the surface of the wall in the new laboratory had much less texture than that of the space in which the behavior had been developed and tuned; this reduction in diffuse reflection meant the ultrasonic and near-IR sensors employed did not detect the presence of the wall until it was considerably closer.

### 3.2 MDARS -- Laboratory Experience

Beginning in 1989, the supporting technology base under development on the ROBART series testbeds began an ongoing transition to the MDARS-I engineering development effort. Obstacle detection sensors and *unrestricted path planning* algorithms that permit a robot to avoid collisions and to automatically move around detected obstacles were ported to the Cybermotion *K2A Navmaster* robot.

**Problem:** An array of nine Polaroid ultrasonic sensors aligned 7 degrees below the horizontal was incorporated to detect small obstacles on the floor in front of the robot. When this system was delivered to another government laboratory for test and evaluation, it was discovered that sonar reflections from harmless expansion joints in the concrete floor were being misinterpreted as obstacles; the smoother floor of the original lab space had resulted in completely specular reflection. The solution was to fabricate new mounting plates for the sensor array, eliminating the downward pitch. While this simple reorientation of the transducers effectively eliminated the problem of spurious returns from the floor cracks, it also reduced the robot's capability to detect low-lying obstructions.

### 3.3 MDARS -- Initial Warehouse Experience at Camp Elliott

**Problem:** Following complete checkout in the MDARS development laboratory at SSC San Diego, the lateral post detection hardware and software were ported to another *Navmaster* platform at the Camp Elliott warehouse site. It turned out that this robot's mobility base was improperly aligned, so that its execution (with no sensor input except odometry) of a RUN command deviated significantly from a straight line, resulting in a lateral displacement of as much as 1.5 - 2 feet over a travel distance of 20 feet. This mechanical alignment problem had not been previously detected because in wall-following (WALL) mode, the robot used a steady stream of wall-distance measurements to continuously correct the error. Because the lateral post detection method used only *intermittent* fixes to update the odometry-derived position estimate, the fact that the robot failed to hold a constant heading resulted in a lateral divergence that eventually became so large the displacement exceeded the effective range of the proximity sensor, and the platform subsequently became "lost." This problem was of course easily solved by employing a properly aligned mobility base.

**Problem:** It soon became apparent that RF communications between the robot and control station was unreliable at several locations in the warehouse. Systematic mapping of RF signal levels with a spectrum analyzer indicated wide variation, including areas of very low signal strength (Figure 8). When the robot entered one of these RF nulls, the Arlan RF units (which provide a 9600-bps serial channel via transparent packetization, error detection, and retransmission) would automatically retransmit packets until they got through. This retransmission was transparent to the higher level processing, except that the higher level protocol's one-second timeout would occasionally be exceeded, resulting in the system retransmitting a complete message. Because the character-oriented data stream was protected by only a simple 8-bit checksum, improperly combined message fragments would occasionally be accepted as a valid message, resulting in the trashing of various status bits (incorrectly indicating a low battery condition, presence of an intruder, a fire alarm, etc.). In one case, such a communications error led the system to initiate a sequential interrogation of 2000 nonexistent inventory tags. This problem has been addressed by moving to a greater bandwidth Arlan RF Ethernet implementation and a more robust checksum in the higher level protocol.

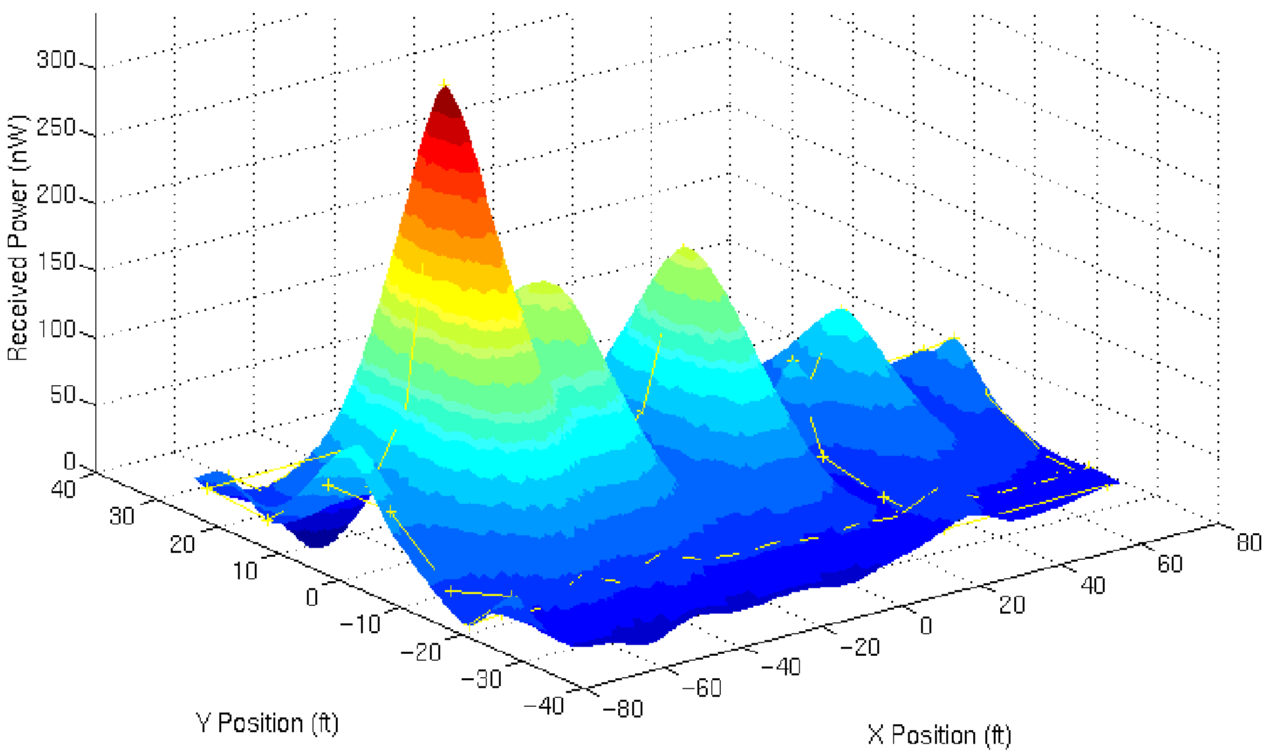


**Figure 8. Survey of RF signal levels measured in Camp Elliott warehouse.**

**Problem:** Improper alignment of individual LEDs in the IR docking beacon on the robot resulted in a small gap in the intended 180-degree beam coverage. Since the docking software was written with the assumption that an interruption of the beam meant a person was temporarily blocking the path, the robot waited for the person to move out of the way -- in this case indefinitely. The LED-alignment problem had not been detected earlier in the laboratory because the robot had never approached the beacon from the "gap" angle (i.e., the robot was always within a smaller ellipsoid of uncertainty when it initiated a docking action). This problem was solved by realigning the beacon LEDs to eliminate the gap in coverage.

**Problem:** The IR docking beacon at Camp Elliott was initially installed near a west-facing exterior doorway. It was soon discovered that the late afternoon sun completely saturated the photodetectors used in the docking process, and the docking station had to be moved. All previous charger locations in the development environment had been along interior walls free of such optical interference.

### 3.4 MDARS -- Early User Appraisal at Anniston Alabama

Following successful testing at the initial Camp Elliott installation, a formal Early User Apprasal (EUA) was conducted at Anniston Army Depot, Anniston Alabama, one of several major Defense Logistics Agency storage facilities in the continental U.S. Installation of the MDARS Interior system was initiated in January 1997. The site was carefully surveyed in advance of installation in order to identify and mitigate environmental factors that might affect navigational performance, and the performance of the RF communications infrastructure was validated before the robots were deployed.

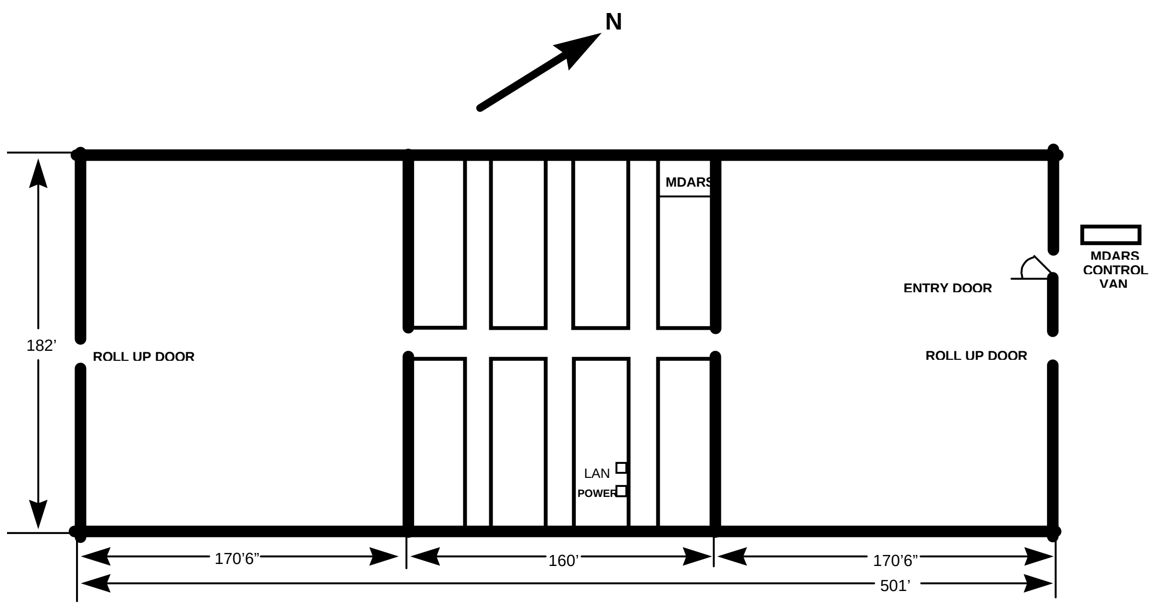


**Figure 9. Map of MDARS Interior EUA warehouse site at Anniston Army Depot in Anniston Alabama This operating environment is roughly the same size and complexity as the earlier beta-test installation site at Camp Elliott.**

**Problem:** After a period of successful operation, the lateral post detection navigation algorithm developed excessive errors at one specific location in the warehouse. Examination of the site revealed that the post to which one of the reflective stripes was attached had been struck by a forklift (Figure 10) and moved approximately 12 cm out of position. The problem was corrected by realigning the reflective stripe on the post to match its position on the map. The lesson: not all non-movable objects are in fact non-movable!

**Figure 10. Navigation reference marker (top center) on structural support post displaced by forklift collision.**

**Figure 11. "Pseudo-wall" of crates in Anniston Depot warehouse showing significant differences in crate offset from the aisle/storage delineator stripe painted on the floor.**

**Problem:** Occasional navigational failures were observed while a robot was following a "pseudo-wall" of stacked crates. Analysis indicated that the wall-following algorithm was deriving the orientation of the "wall" over a very limited space, so that when successive crates were misaligned by more than just a few inches the algorithm would fail (Figure 11). The solution was to loosen the parameters of the algorithm to fit the wall over a larger distance, on the order of two crate widths. This action ensured that the heading update reflected an average across several crates as opposed to the particular orientation of a single crate face.

**Problem:** After an extended period of successful operation, one of the two Anniston robots began drifting off course while following a wall of crates, eventually becoming lost and reporting a blocked condition. Experiments with adjusting the individual crate alignments indicated the navigational algorithm was not exhibiting some pathological sensitivity to the details of the environment, and close examination of the side-sonar plots on the debug screen established that the sensors were working correctly. After additional tests proved that recent changes in the virtual path segments were not the cause, it was finally determined that the mobility base was seriously out of alignment. Careful inspection revealed tar marks on both sides of one of the wheels, suggesting that this wheel had fallen into a crack in the concrete slab that was longitudinally aligned with the path. It had then apparently been wrenched seriously out of alignment when a heading change was ordered and the other two wheels turned but the trapped wheel was unable to pivot. The MDARS-I performance specification had called for crossing a predefined breach (such as an expansion joint), and all preliminary testing had therefore been conducted with cracks running perpendicular to the path of travel. The case of a crack parallel to the path had been overlooked. The only recourse at this point was to fill in the crack and realign the robot

**Problem:** After months of reliable operation, both robots at Anniston began to experience serious communications degradation and apparent "sluggishness" in performance of the PC-104-based Pentium master on-board processor. The problem was traced to RF noise generated by onboard DC-to-DC converters being coupled to the RS-232 cables connecting the onboard processors, generating spurious software interrupts. The RF noise had progressively worsened as the number of onboard power supplies was increased from two to five to support new subsystems installed in the field, and the switching power supplies began to mutually interact (cross modulation). This problem was resolved by adding RF-suppression chokes to all DC-to-DC converters, and by switching from ribbon-type to shielded RS-232 cables. The lesson here is to finish the system integration and testing before you do the field installation, a luxury that regrettably was not possible under the fielding schedule imposed in this case.

### 3.5 Environmental Challenges: Lessons Learned

Our purpose in explicitly discussing the afore-mentioned problems is to point out that the process of moving a mobile robotic system from one operating environment to another (especially when moving from a deliberately benign laboratory development environment to a more complex and less structured environment representative of real-world operations) is one in which unanticipated problems are the rule rather than the exception. The causes of the examples described above fit into a number of different categories:

- Hardware deficiencies and software implementation errors not manifested on the initial system hardware.
- Hardware deficiencies and software implementation errors not manifested in the initial development environment.
- Sensor modes or processing algorithms tuned too tightly to specific characteristics of the initial development environment.
- Subtle interactions between multiple hardware and software components, leading to unexpected

degradation of performance.

A human's perceptual capabilities are powerfully adept at characterizing both the similarities and differences between various features of his/her environment -- at detecting both the general rule and the specific exception to it. A robot's sensory inputs, on the other hand, are far more limited, as it can interpret these inputs only up to the limits of its environmental model.

A robot's software implicitly embodies a world model derived from -- but infinitely simpler than -- the world model present in the mind of the software designer. The designer may deliberately work to capture certain specific aspects of the world in the model, while other aspects may creep in as unintended consequences of various software design decisions. Some problems arise when the robot's implicit world model is not rich enough to support the behaviors required by the application. Many others result when the designer simply fails to understand the limits of the world model instantiated on the robot -- which aspects of his own world model have been implanted in the robot, and which have not.

If a complex robotic system is to operate robustly, the model must take adequate account of the relevant dimensions of variability of the environment as will be perceived by the sensor subsystems. A rough floor is still a floor, and you can drive over it; a low-lying obstacle is still an obstacle, and you likely can't drive over it. The sensors and perceptual software that build a robot's world model have to be able to draw such distinctions. The system designer can not hope to anticipate in advance every condition that will actually be encountered, and how that condition will be perceived via the robot's sensors. The odds of success can be substantially raised, however, by testing the system in as many and varied environments as possible before actually fielding occurs.

Well-implemented adaptive behavior in a robot (such as Cybermotion's wall-following) can bring along the downside risk of actually *masking* a hardware or software fault that may then manifest itself later at an unfortunate time. One possible solution to this problem is to instrument adaptive behaviors in order to monitor their adaptation modes -- to install "intelligence" analogous to that of the human driver who, while driving straight down the highway, notes the fact when his car's steering constantly "pulls left." In addition, it is necessary to provide ample redundancy in the implementation of critical functionality, particularly with regard to environmental perception.

Behavioral robustness is required if mobile robots are to infiltrate viable markets. Truly practical robots must be mass producible, rather than handcrafted, and they must function acceptably over as wide a range of environments as possible without excessive manual tuning. Thus the designer of mobile robotic hardware and software must accommodate the full range of variability within manufacturing processes and within target operating environments, or face the consequences in the form of unreliable real-world performance.

## 4. ROBOTS WORKING WITH AND FOR HUMANS

Mobile robots deployed in real-world applications must of necessity be capable of successfully interacting with humans, the operators who task them and monitor their performance as well as those who by plan or happenstance share the robots' workspace.

### 4.1 Coexistence of Mobile Robots and Human Workers

The large number of robots performing industrial tasks such as welding, painting, and spraying in factory environments has generated a base of experience in the coexistence of human factory workers and industrial manipulators. The key concern with a powerful manipulator is the safety of workers entering the workspace to perform supporting tasks such as feeding materials or making minor process adjustments. Fortunately, the limits of the typical factory robot's workspace are static and access can be easily controlled. For mobile robots, on the other hand, the workspace is neither static nor in many cases physically constrained.

**Problem:** Because it was recognized from the beginning that MDARS-I robots would have to share the warehouse with forklift operators and other human workers, precautions were taken at Camp Elliott to prevent accidents. Whenever a robot approaches the passageway between the two bays (see figure 3), a flashing light mounted in a highly visible location above the door is activated to alert forklift operators who might be working nearby. Unfortunately, human operators do not necessarily heed such warnings. In October 1994, a speeding forklift operator spilled his entire load in an emergency stop that narrowly avoided a collision with a robot approaching from the "blind" alley to his right. The virtual path structure within the warehouse was then edited to prohibit the robot from approaching this virtual node from the side corridors, thus assuring that the forklift driver could see the robot directly ahead as it makes its way towards him down the central corridor.

While a robotic mail cart equipped with tactile bumpers repeatedly following an installed guidepath at 1 mph does not pose a significant threat to the safety of the people walking the corridor (nor do they pose a threat to it), the initial MDARS experience demonstrates vividly that a larger, heavier security robot autonomously patrolling a warehouse in which potentially inattentive forklift operators are operating poses issues of safety for the robot as well as for the humans involved.

Beyond simple safety, the activities of humans in the warehouse unpredictably reconfigure the robot's environment as cartons and pallets are moved about daily, somewhat complicating the navigation task and making the role of the MDARS operator (security guard) all the more important as the assessment engine of last resort. Finally, the need to cultivate acceptance of robots by human co-workers is clear if they are to operate successfully in the same space, since any seriously adversarial situation between robot and human co-worker will eventually be manifested as a system malfunction.

### 4.2 Early User Appraisal by Real-World Security Guards

When the system was judged ready for the formal EUA process, the Anniston guard force was trained and given control of the MDARS system in March 1998. The guards were generally enthusiastic and willing to learn and use the system, although most of them had had little or no previous contact with computers. Because it was possible to provide only minimal training and documentation, a number of command and control aspects that needed improvement were quickly encountered and identified. This evolutionary feedback process of using the real-world user community to help find system deficiencies has proven very advantageous from the standpoint of minimizing later problems associated with user acceptance of the system.

#### 4.2.1 Problems with Teleoperation

In general, the guards got into trouble whenever they manually operated the robot, even though the obstacle avoidance sensors are coupled to provide "telereflexive" or "safeguarded" operation (Everett and Laird 1990). In a relatively brief period of operations, guards: 1) hooked a robot's bumper around a post; 2) took a robot for a joyride into an adjacent area not intended for robotic operations; 3) tried to manually drive a robot onto the recharging dock; and 4) drove into a roped-off chemical detector storage area. Password protection has been added to restrict the use of manual driving to situations where it is really needed.

**Problem:** A single pan/tilt-mounted camera on the MDARS-I platform is used for both teleoperated driving of the platform and visual assessment of potential threats. Because the system will automatically point this camera in the direction of a suspected threat, it is very possible for the operator to try to drive the robot when the camera is not facing forward. This scenario is not always obvious to the driver, and often causes erratic driving behavior. In one instance, the camera was turned about 90 degrees to the left, resulting in the guard driving in a clockwise circle because he was unable to reconcile what was shown in the video with the command he was giving via the joystick. To alleviate this problem, the guard's ability to control the camera has been tightened, and the camera is now automatically pointed forward in certain situations, including whenever the operator assumes manual control.

This problem of ensuring that sensory view is coordinated with the needs of a teleoperated task is a general one, and the situation is even more difficult when a single operator is attempting to simultaneously drive a vehicle, point a sensor, and aim a weapon. This is an ongoing research challenge being addressed by the ROBART III development (Everett and Gage 1996).

**Figure 12. ROBART III (1992-) is a significantly augmented supervised autonomous response robot intended for urban warfare scenarios inside previously unmapped interior structures.**

### 4.2.2 "Lost" Guards

To an unexpected degree, the security guards had considerable difficulty in understanding where the robot was located based solely on the map displays, and in making correct use of this information.

**Problem:** Since many of the security guards have never actually been inside most of the buildings where robots are or will be located, they can not recognize landmarks based on video feedback, and have difficulty orienting themselves based on a simple 2-D planview map. Additional details and landmarks were subsequently added to the map to assist the guards in matching up the video image with the map image.

**Problem:** Maps which ended at the building boundaries did not provide the guards with enough context to effectively direct a response force to a particular building to assess a situation. The map area displayed was greatly expanded to include the outlines of several other adjacent buildings, as well as a North indicator.

**Problem:** The guards sometimes found it difficult to locate the icon of a stationary robot in a large zoomed-out map. Solutions being explored include setting a minimum icon size constraint and having the system automatically focus on the robot icon.

**Problem:** The guards found it very difficult to determine that the robot was in fact physically located at a different place than that indicated on the map. While this should not happen, of course, it is a real-world reality of the current system, and analogous problems will be intrinsic in any quasi-autonomous mobile robotic system realizable in the foreseeable future. Better landmarks in the map may mitigate this problem, but the long term solution will be more robust navigation to minimize the chances of the robot getting lost in the first place.

### 4.2.3 Complexity of the User Interface

A third set of problems stemmed from the inherent complexity of the user interface. From the outset, the implementation team understood the need to make the user interface acceptable to computer-naive security guards; nevertheless, the end result was still too complex. The overall lesson is familiar, but one which requires repetition: "Simplify and clarify the user interface." Here are a number of specific lessons:

- Display information in terms meaningful for the operator: rather than presenting a number representing battery voltage, show percentage of charge remaining, or operating time remaining, preferably in graphical format. Use color to differentiate "plenty of power" from "power getting low", and add an audio alert for "power dangerously low".
- Make clear to the user what action(s) he or she is expected to perform. If a simple acknowledgment is desired, then the system should display a single big "OK" button, with the cursor already placed on it. Do not display options that are not currently available as grayed-out icons -- make them completely invisible. Provide a brief top-level explanation of what is going on with graphics or large bold text, with supporting details available but not intrusive.
- Automatically monitor system state proactively and defensively.

To illustrate this last point, the following is a representative case of interacting system deficiencies, all fortunately correctable with minimum effort.

Although the *Navmaster* mobility base sends the digitized value of the battery voltage to the onboard Scheduler processor once a second, the detection of a low-battery condition made no use of this value, instead relying on a separate BATLOW K2A status signal, which is only set by the mobility base just before a path begins to execute. Therefore, a robot which is not commanded to move will never report a low-battery condition.

The system initially paid no attention to the consequences of a guard placing a robot offline away from the charging station, or simply leaving it assigned to the Operator indefinitely. In either case, the batteries would discharge, disabling the robot, if the guard failed to release the robot in time.

When a low-battery condition is reported by the robot, an audible alert is generated only if the robot is attached to the Supervisor (but not to the Operator).

**Problem:** Monitoring the MDARS systems is only one of several activities guards typically perform (i.e., they also monitor fixed cameras, acknowledge alarms from other systems). On one occasion, the night shift manually assigned the robot to the Operator Display, experimented for a while, then forgot about it for the rest of the evening. This left the warehouse vulnerable, since the random patrol function wasn't being performed, and as described above caused the batteries to discharge because the camera and transmitter were on and the platform couldn't automatically dock with the charger.

One of the guards actually requested that we put a timer on the Operator that would cause it to note the robot was idle, tell the guard to release it, then if no response was given, force a release after a certain period of time. The software has since been changed to time the user's activities on the Operator station, and if the robot is idle for an extended period of time, it is automatically released back to Supervisor control.

Given the extremely limited perceptual abilities of autonomous mobile robots, it is often convenient and sometimes necessary to rely on a human operator to bail the system out of a tight spot (figuratively or literally). This approach can seem very comfortable during system development, when the helpful operator is the developer himself. But turning over a complex system resulting from five years of intensive effort by a skilled and experienced 10-person development team to a group of security guards without such technical background will reduce that comfort level considerably. Instead of addressing a problem with "assign it to the operator and let him handle it," we now try to resolve as many problems as possible with software. In those cases where the system must indeed rely on operator intervention, one must ensure that the system does its part by providing the user with information and tools that facilitate and encourage a correct response.

This philosophy can be summarized as follows:

- First, understand the users -- understand the job they are assigned to do, how they do that job, and their level of education, training, and experience.
- Second, respect the users -- listen to their concerns and suggestions. And remember that they will have the final say in judging whether the robot application is a success or failure.
- Third, support the users -- make it as easy as possible for them to provide the appropriate inputs to the robotic system in every situation, and for them to be successful in doing their jobs.
- Finally, work very hard to make the system "user-proof" -- make it as difficult as possible for them to make inappropriate inputs to the system.

## 5. SUMMARY

Transplanting a mobile robot system from the benign laboratory environment in which it was initially developed to a semi-structured test and demonstration environment is seen to be a challenging process, but necessary for the successful deployment and validation of robust system capabilities prior to actual operational fielding. The MDARS-I security robot system has successfully made this transition through a series of steps culminating in the performance of security patrols in an operational DoD warehouse. The valuable lessons learned from this Early User Appraisal will be exploited to improve the chance of success as the MDARS system is transfered to successively more demanding environments.

## ACKNOWLEDGEMENT

This work has been performed under the Mobile Detection Assessment and Response System (MDARS) project, managed by Mr. Jerry Edwards (Product Manager - Physical Security Equipment (PM-PSE)) at Ft. Belvoir, VA. The contributions of the entire MDARS implementation team towards this paper are gratefully acknowledged, especially Gary Gilbreath and Kelly Grant for their documentation of the Anniston security guards' interactions with the system.

## REFERENCES

Borenstein, J., Everett, H. R., and Feng, L. 1996, *Navigating Mobile Robots: Systems and Techniques*, AK Peters, Ltd., Wellesley, MA.

Everett, H. R. 1982. "A Microprocessor Controlled Autonomous Sentry Root", Masters thesis, Naval Postgraduate School, Electrical Engineering Department.

Everett, H. R. 1995. *Sensors for Mobile Robots: Theory and Application*, AK Peters, Ltd., Wellesley, MA.

Everett, H. R., and Laird, R. T 1990. "Reflexive Teleoperated Control," July, *Proceedings of AUVS-90*, Dayton, OH, pp. 280-292.

Everett, H. R., Gilbreath, G. A., Tran, T., and Nieusma, J. M. 1990. *Modeling the Environment of a Mobile Security Robot*, TD 1835, Naval Ocean Systems Center, San Diego, CA.

Everett, H. R., Gilbreath, G. A., Heath-Pastore, T. A., and Laird, R. T 1993. "Coordinated Control of Multiple Security Robots," *Proceedings of SPIE Mobile Robots VIII*, Boston, MA, September.

Everett, H. R., and Gage, D. W. 1996. "A Third Generation Security Robot," SPIE Mobile Robot and Automated Vehicle Control Systems, Vol. 2903, Boston, MA, pp. xxx-xxx, November.

Everett, H. R., Gage, D. W., Gilbreath, G. A., Laird, R. T., and Smurlo, R.P. 1994. "Real-world Issues in Warehouse Navigation," SPIE Mobile Robots IX, Vol. 2352, Boston, MA, pp. 249-259, November.

Heath-Pastore, T. A., and Everett, H.R. 1994. "Coordinated Control of Interior and Exterior Autonomous Platforms," *ISRA '94, Fifth International Symposium on Robotics and Manufacturing*, Maui, HI, August.

Holland, J., Everett, H. R. and Gilbreath, G.A. 1990. "Hybrid Navigational Control Scheme for Autonomous Platforms," *Proceedings of SPIE Mobile Robots V*, Boston, MA, November, pp. 291-298.

Laird, R. T., Everett, H. R., and Gilbreath, G.A. 1993. "A Host Architecture for Multiple Robot Control," *Proceedings of the ANS Fifth Topical Meeting on Robotics and Remote Systems*, American Nuclear Society, Chicago, IL.