Unstructured Facility Navigation by Applying the NIST 4D/RCS Architecture

Roger BOSTELMAN, Tsai HONG, Tommy CHANG,
William SHACKLEFORD, and Michael SHNEIER
Intelligent Systems Division
National Institute of Standards and Technology,
100 Bureau Drive, Gaithersburg, MD 20899, USA

ABSTRACT

The National Institute of Standards and Technology's (NIST) Intelligent Systems Division (ISD) is working with the material handling industry, specifically on automated guided vehicles, to develop next generation vehicles. ISD is also a participant in the Defense Advanced Research Project Agency (DARPA) Learning Applied to Ground Robots (LAGR) Project embedding learning algorithms into the modules that make up the Four Dimensional/Real-Time Control System (4D/RCS). 4D/RCS is the standard reference model architecture which ISD has applied to control many intelligent systems. Technology from LAGR is being transferred to the material handling industry through the NIST Industrial Autonomous Vehicles Project. This paper describes the 4D/RCS structure and control applied to LAGR and the transfer of this technology through a demonstration to the automated guided vehicles industry.

Keywords: LAGR, 4D/RCS, automated guided vehicles, hierarchical control, reference model architecture, industrial autonomous vehicles

1. INTRODUCTION

The National Institute of Standards and Technology's (NIST) Intelligent Systems Division (ISD) has been studying industrial vehicles, namely automated guided vehicles (AGVs) like the ones shown in figure 1, and their application to manufacturing and distribution for several years.

Figure 1 – Typical AGVs used in the material handling industry: unit load (left), forklift (right)

This effort, called the Industrial Autonomous Vehicles Project, aims to provide industries with standards, performance measurements, and infrastructure technology needs for the material handling industry. NIST recently sponsored a survey of AGV manufacturers in the US, conducted by Bishop[1], to help determine their "generation-after-next" technology needs. Recognizing that basic engineering issues to enhance current AGV systems and reduce costs are being addressed by AGV vendors, the study looks beyond today's issues to identify needed technology breakthroughs that could open new markets and improve US manufacturing productivity. Results of this study are described in [1].

Within the survey and high on the list, AGV vendors look to the future for: reduced vehicle costs, navigation in unstructured environments, onboard vehicle processing, 3D imaging sensors, and transfer of advanced technology developed for Department of Defense to this industry. Current AGVs are "guided" by wire, laser or other means, operate in structured environments tailored to the vehicle, have virtually no 3D sensing and operate from a host computer with limited onboard-vehicle control.

The availability of AGVs that can operate in unstructured environments expands the market to include new customers in printing and distribution, for instance. Operations in these industries can be quite dynamic such that a next-generation AGV must adapt quickly and adroitly to change. AGVs are now capable of loading truck trailers at a loading dock, but lack the robust 3D imaging systems and perception algorithms for such a task. The same is true for detecting and avoiding obstacles in unstructured indoor environments where AGVs are typically used.

For over 30 years, NIST has been developing a standard control architecture, called the Real-time Control System (RCS), that allows for distributed intelligent control and plug-and-play control algorithms and advanced 3D sensors and vehicles [2, 3]. 4D/RCS is the most recent version of RCS developed for the Army Research Lab Experimental Unmanned Ground Vehicle program. ISD has studied and used 4D/RCS in defense mobility [4], transportation [5], robot cranes [6], manufacturing [7, 8] and several other applications.

In the past year, ISD has been applying 4D/RCS to the DARPA LAGR program [9] which aims to develop algorithms that enable a robotic vehicle to travel through complex outdoor terrain. The goal is to enable the control system of the vehicle to learn which areas are traversable and how to avoid areas that are impassable or that limit the mobility of the vehicle. To accomplish this goal, the program provided small robotic vehicles to the participants.

---

[1] Commercial equipment and materials are identified in this paper in order to adequately specify certain procedures. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

The vehicles are equipped with four computer processors (right and left cameras, control, and the planner); wireless data and emergency stop radios; GPS receiver; inertial navigation unit; dual stereo cameras; infrared sensors; switch-sensed bumper; front wheel encoders; and other sensors listed later in the paper. Towards fulfilling the US AGV vendors request of advancing the AGV industry, NIST also equipped the vehicle with active and passive RFID (radio frequency identification) and a 2D laser scanner as shown in Figure 10.

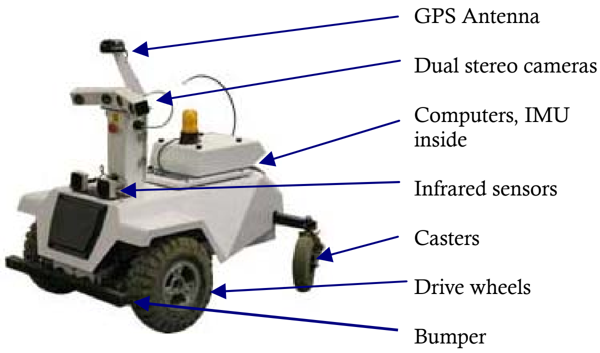


Figure 2. The DARPA LAGR vehicle

Section 2 of this paper describes the 4D/RCS Reference Model Architecture as applied to the (outdoor) LAGR program and detailed in [10]. Section 3 follows with a description of the 4D/RCS LAGR controller applied to an indoor, unstructured environment. Section 4 is a conclusion followed by acknowledgments and references sections.

2. 4D/RCS Applied to LAGR

The 4D/RCS architecture is characterized by a generic control node at all the hierarchical control levels. This node has three principal components: Sensor Processing, World Modeling, and Behavior Generation. The 4D/RCS hierarchical levels are scalable to facilitate systems of any degree of complexity. Each node within the hierarchy functions as a goal-driven, model-based, closed-loop controller. Each node is capable of accepting and decomposing task commands with goals into actions that accomplish task goals despite unexpected conditions and dynamic perturbations in the world.

At the heart of the control loop through each node is the world model, which provides the node with an internal model of the external world (Figure 3).

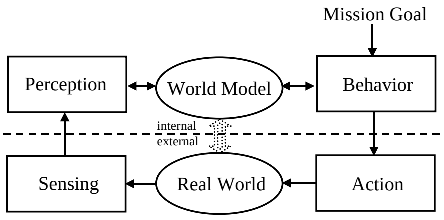


Figure 3. The fundamental structure of a 4D/RCS control loop.

The 4D/RCS architecture for LAGR (Figure 4) consists of two levels. This is because the size of the LAGR test areas is small (typically about 100 m on a side and the test missions are short in duration - typically less than 4 minutes.)

The following sub-sections describe the type of algorithms implemented in sensor processing, world modeling, and behavior generation, as well as a section that describes the learning algorithms that have been implemented.

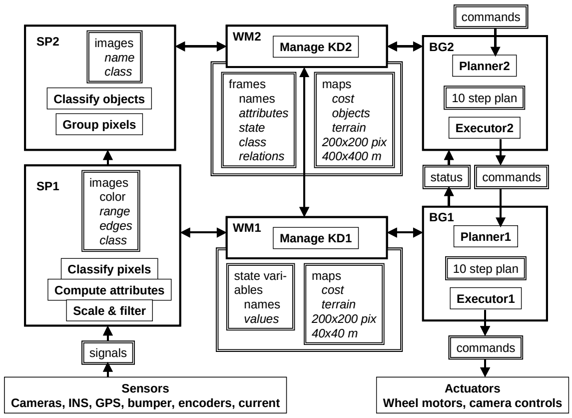


Figure 4. Two-level instantiation of the 4D/RCS hierarchy for LAGR.

Sensory Processing

The sensor processing column in the 4D/RCS hierarchy for LAGR (Figure 4) starts with the sensors on board the LAGR vehicle. Sensors used in the sensory processing module include the two pairs of stereo color cameras, the physical bumper and infrared bumper sensors, the motor current sensor (for terrain resistance), and the navigation sensors (GPS, wheel encoder, and inertial navigation system). Sensory processing modules include a stereo obstacle detection module, a bumper obstacle detection module, an infrared obstacle detection module, an image classification module, and a terrain slipperiness detection module.

Stereo vision is primarily used for detecting obstacles. The SRI Stereo Vision Engine [11] is used to process the pairs of images from the two stereo camera pairs. For each newly acquired stereo image pair, the obstacle detection algorithm processes each vertical scan line in the reference image independently and classifies each pixel as GROUND, OBSTACLE, SHORT_OBSTACLE, COVER or INVALID. Figure 5 illustrates the basic obstacle detection algorithm [12].

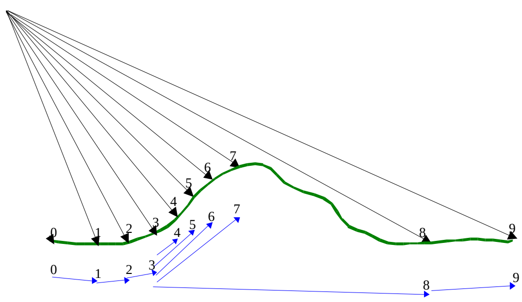


Figure 5. A single vertical scan-line detecting the ground. *Pixel 0 is altered to correspond to the bottom of the vehicle wheel. Pixels 1, 2, 3, 8 and 9 are ground pixels due to shallow slopes. Pixel 4, 5, 6 and 7 are obstacles due to steeper slopes. The slopes are shown by the direction vectors on the bottom of the figure.*

Pixels that are not in the 3D point cloud are marked INVALID. Pixels corresponding to obstacles that are shorter than 5 cm high are marked as SHORT_OBSTACLE. The obstacle height threshold value of 5cm was chosen such that the LAGR vehicle can ignore and drive over small pebbles and rocks. Similarly, COVER corresponds to obstacles that are taller than 1.5 m, a safe clearance height for the LAGR vehicle.

Within each reference image, the corresponding 3D points are accumulated onto a 2D cost map of 20 cm by 20 cm cell resolution. Each cell has a cost value representing the percentage of OBSTACLE pixels in the cell. In addition to cost value, color and elevation statistics are also kept and updated in each cell. This map is sent to the world model at the current level and to the sensory processing module at the level above in the 4D/RCS hierarchy. Figure 6 shows a view of obstacle detection from the operator control unit (OCU). The vehicle is shown driving on a dirt road lined with trees with an orange fence in the background.

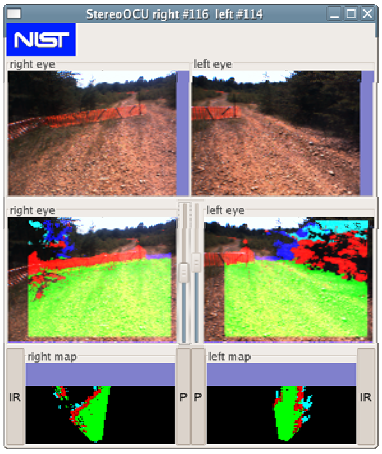


Figure 6. OCU display showing original images (top), results of obstacle detection (middle), and cost maps (bottom). *Red represents obstacles, green is ground, and blue represents obstacles too far away to classify.*

World Modeling

The world model is the system's internal representation of the external world. The world model provides a site for data fusion, acts as a buffer between perception and behavior, and supports both sensory processing and behavior generation. It acts as a bridge between sensory processing and behavior generation in the 4D/RCS hierarchy by providing a central repository for storing sensory data in a unified representation. It decouples the real-time sensory updates from the rest of the system. The world model process has two primary functions: to create a knowledge database and keep it current and consistent, and to generate predictions of expected sensory input.

For the LAGR project, two world model levels have been built (WM1 and WM2). Each world model process builds a two dimensional (200 x 200 cells) map, but at different resolutions. These are used to temporally fuse information from sensory processing. Currently the lower level (SP1) is fused into both WM1 and WM2 as the learning module in SP2 does not yet send its models to WM. Figure 7 shows the WM1 and WM2 maps constructed from the stereo obstacle detection module in SP1. The maps contain traversal costs for each cell in the map. The position of the vehicle is shown as an overlay on the map. The red, yellow, blue, light blue, and green are cost values ranging from high to low cost, and black represents unknown areas. Each map cell represents an area on the ground of a fixed size and is marked with the time it was last updated. The total length and width of the map is 40 m for WM1 and 120 m for WM2. The information stored in each cell includes the average ground and obstacle elevation height, the variance, minimum and maximum height, and a confidence measure reflecting the certainty of the elevation data. In addition, a data structure describes the terrain traversability cost and the cost confidence as updated by the stereo obstacle detection module, image classification module, bumper module, infrared sensor module, etc. The map updating algorithm is based on confidence-based mapping as described in [13]. The costs and the confidences are combined to determine the relative safety of traversing the grid with cost equations detailed in [10].

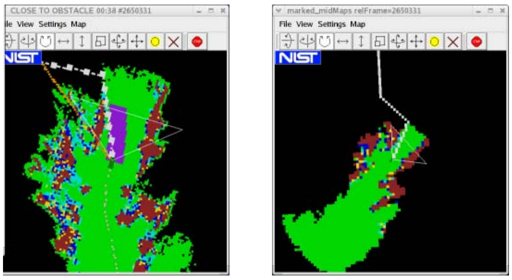


Figure 7. OCU display of the World Model cost maps built from sensor processing data. *WM1 builds a 0.2 m resolution cost map (left) and WM2 builds a 0.6 m resolution cost map (right).*

The final cost placed in each map cell represents the best estimate of terrain traversability in the region represented by that cell, based on information fused over time. Each cost has a confidence associated with it and the map grid selects the label with the highest confidence. The final cost maps are constructed by taking the fused cost from all the sensory processing modules.

Efficient functions have been developed to scroll the maps as the vehicle moves, to update map data, and to fuse data from the sensory processing modules. A map is updated with new sensor data and scrolled to keep the vehicle centered. When the vehicle moves out of the center grid cell of the map, the scrolling function is enabled. The map is vehicle-centered, so only the borders need to be initialized. Initialization information may be obtained from remembered maps saved from previous test runs.

The cost and elevation confidence of each grid cell is updated every sensor cycle: 5 Hz for the stereo obstacle detection module, 3 Hz for the learning module, 5 Hz for the classification module, and 10 Hz to 20 Hz for the bumper module. The confidence values are used as a cost factor in determining the traversability of a cell.

Behavior Generation

Top level input to Behavior Generation (BG) (Figure 8) is a file containing the final goal point in UTM (Universal Transverse Mercator) coordinates. At the bottom level in the 4D/RCS hierarchy, BG produces a 1.3 m/s max. speed for each of the two drive wheels updated every 20 ms, which is input to the low-level controller included with the government-provided vehicle. The low-level system returns status to BG, including motor currents, position estimate, physical bumper switch state, raw GPS and encoder feedback, etc These are used directly by BG rather than passing them through sensor processing and world modeling since they are time-critical and relatively simple to process.

Two position estimates are used in the system. Global position is strongly affected by the GPS antenna output and is more accurate over long ranges, but can be noisy. Local position uses only the wheel encoders and inertial measurement unit (IMU). It is less noisy than GPS but drifts significantly as the vehicle moves, and even more if the wheels slip.

The system consists of five separate executables.

Each sleeps until the beginning of its cycle, reads its inputs, does some planning, writes its outputs and starts the cycle again. Processes communicate using the Neutral Message Language (NML) in a non-blocking mode, which wraps the shared-memory interface [14]. Each module also posts a status message that can be used by both the supervising process and by developers via a diagnostics tool to monitor the process.

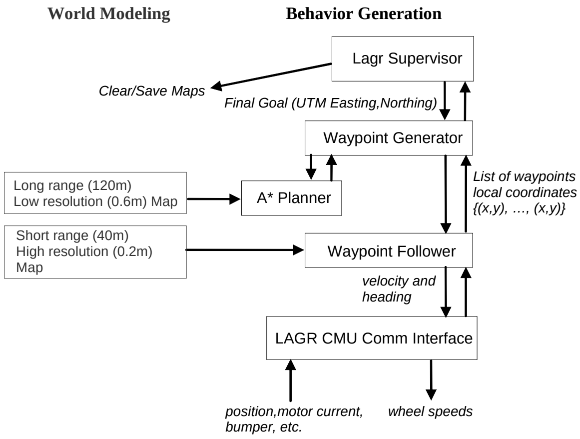


Figure 8. Behavior generation high level data flow diagram.

The LAGR Supervisor is the highest level BG module. It is responsible for starting and stopping the system. It reads the final goal and sends it to the waypoint generator. The waypoint generator chooses a series of waypoints for the lowest-cost traversable path to the goal using global position and translates the points into local coordinates. It generates a list of waypoints using either the output of the A* Planner [15] or a previously-recorded known route to the goal.

The planner takes a 201 X 201 terrain grid from WM, classifies the grid, and translates it into a grid of costs. In most cases the cost is simply looked up in a small table from the corresponding element of the input grid. However, since costs also depend on neighboring costs, they are automatically adjusted to allow the vehicle to continue motion.

The waypoint follower receives a series of waypoints spaced approximately 0.6 m apart that could be used to drive blindly without a map. However, there are some features of the path that make this less than optimal. When the path contains a turn, it is either at a 0.8 rad (45°) or 1.6 rad (90°) angle with respect to the previous heading. The waypoint follower could smooth the path, but it would at least partially enter cells that were not covered by the path chosen at the higher level. Other features are detailed in [10] along with details of custom behaviors not relevant to the indoor behavior generator. However, one such potential indoor behavior is the “Narrow Corridor/Close to Obstacle” mode that turns on automatically when the vehicle is in tight spaces. In this case the vehicle slows down, builds a detailed world model, and considers a larger number of alternative paths to get around tight corners than in open areas.

The lowest level module, the LAGR Comms Interface, takes a desired heading and direction from the waypoint follower and controls the velocity and acceleration, determines a vehicle-specific set of wheel speeds, and handles all communications between the controller and vehicle hardware.

Learning Algorithms

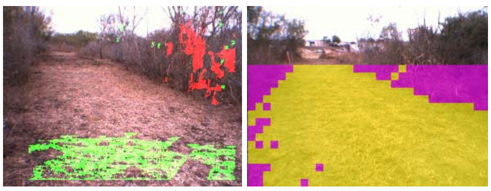
Learning is a basic part of the LAGR program. Learning takes place in all three 4D/RCS architecture columns - sensor processing, world modeling, and behavior generation. There is learning by example, learning from experience, and learning of maps and paths. Most learning relies on sensed information to provide both the learning stimulus and the ground truth for evaluation. In the LAGR program, learning from sensor data has mainly focused on learning the traversability of terrain. This includes learning by seeing examples of the terrain and learning from the experience of driving over (or attempting to drive over) the terrain.

Model-based learning occurs in the SP2 module of the 4D/RCS architecture, taking input from SP1 in the form of labeled pixels with associated *(x, y, z)* positions. Classification is an SP1 process that uses the models to label the traversability of image regions based only on color camera data.

An assumption is made that terrain regions that look similar will have similar traversability. The learning works as follows [16]. The system constructs a map of the terrain surrounding the vehicle, with map cells 0.2 m by 0.2 m. Each pixel passed up from SP1 has an associated red (R), green (G), and blue (B) color value in addition to its (x, y, z) position and label (OBSTACLE or GROUND). Points are projected into the map using the *(x, y, z)* position. Each map cell accumulates descriptions of the color, texture, intensity, and contrast of the points that project into it. When a cell accumulates enough points, it is ready to be considered as a model. To build a model we require that 95% of the points projected into a cell have the same label (OBSTACLE or GROUND).

To classify a scene, only the color image is needed. A window is passed over the image and color, texture, and intensity histograms, and a contrast value are computed as in model building. A comparison is made with the set of models, and the window is classified with the best matching model if a sufficiently good match value is found. Regions that do not find good matches are left unclassified.

Figure 9a shows an image taken during learning. The pixels contributing to the learning are shown in red for obstacle points and green for ground points. Figure 9b shows a scene labeled with traversability values computed from the models built from previous data.



(a) **(b)**

Figure 9. Learning by example images *overlaid (a) with red (obstacles) and green (ground) and (b) with traversability information as magenta (obstacles) and yellow (ground).*

3. 4D/RCS Applied to Unstructured Facilities

Changes from the Outdoor System

A number of changes were made to the control system in order to transfer the military outdoor application to an indoor industrial setting. Two RFID sensors were integrated into the vehicle position estimate. An active tag RFID system was used allowing tuned tags to be placed up to a couple of meters away from the onboard vehicle receiver and giving freedom to choose their location. Tag tuning included adding a simple thin metal plate mounted to the tags back and bent to tune detection regions from the original 4 m down to about 2 m. Tuning required initial testing for detection range prior to placement. Also, a passive RFID system was used including tags that provide a more accurate vehicle position to within a few centimeters. RFID systems updates replaced the outdoor GPS positioning system updates in the controller. These two integrated systems are shown in Figure 10. Available passive tags can be procured and integrated into the controller to allow the vehicle to get even more accurate positioning, which may be required near machines or tray stations typically used by AGV's.

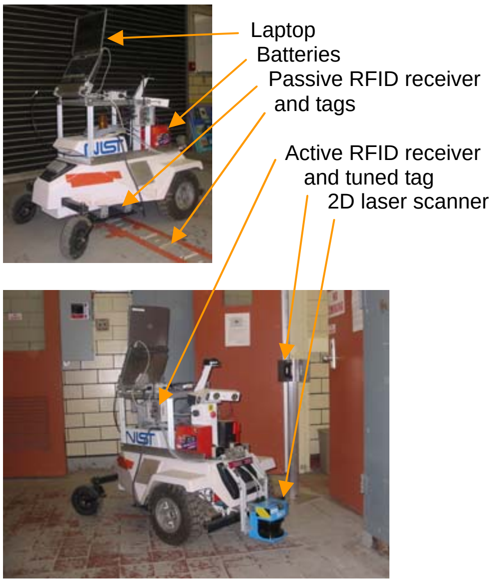


Figure 10. LAGR/IAV vehicle at demonstration start position (top) and after heading through doorway (bottom).

The control system also needed to be less aggressive for safety of people and equipment. The outdoor system is allowed to hit a tree or a bush and never stops to wait for someone or something to get out of the way during a DARPA test. Instead the outdoor system will either turn around completely to find a different path or try to squeeze through whatever space is left. AGVs must detect people or objects in the path of the vehicle according to the International Association of Science and Technology for Development (IASTED) B56.5 Industrial Truck Safety Standard [17], which provided guidelines for our indoor demonstration. Furthermore the indoor setting included tighter corners than are typically encountered outdoors. Unfortunately, the only way to adequately meet the conflicting goals of negotiating tighter corners while maintaining wider standoff margins was to slow the vehicle by about 50% to approximately 0.6 m/s.

The obstacle detection height was also changed for the indoor demonstration. Outdoors, it proved better to avoid even fairly short obstacles such as logs and roots that could sometimes get caught under the wheel and either stop the vehicle or cause significant wheel slip. Indoors, the obstacle height threshold was raised from 5 cm to 30 cm since small obstacles were not common. Also, the uniformity of the floor made the range estimates from stereo vision less accurate than the ground outdoors, causing false obstacles to be detected due to poor stereo correlation. A 2D laser scanner was also added and will soon be integrated into the controller to allow better path planning by removing the false obstacles detected with stereo vision.

Demonstration Results

On April 10, 2006, several vendors and users of automated guided vehicles met at NIST for an IASTED B56.5 Safety Standard meeting. As part of the meeting, attendees watched a demonstration of the LAGR robot navigating through an unstructured environment including a garage and a working machine shop. The robot was started next to its charging station in the autonomous vehicles laboratory (garage). It detected a passive RFID tag placed on the floor and updated its knowledge of its current position at the start point of the route (see Figure 10). A laptop on the robot displayed an overlay of the live video from each of the two stereo color camera pairs, with obstacle detection, color classification, and 2D range information, as well as a high-resolution/short range map and a lower resolution/longer range map of the facility (see Figure 11).

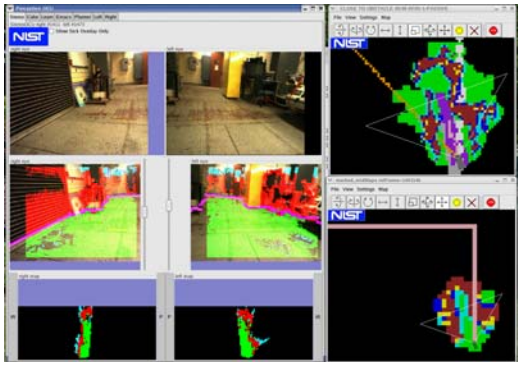


Figure 11. LAGR AGV Graphical Displays – *right and left stereo images (upper left); images overlaid with stereo obstacle (red) and floor (green) detection and 2D scanner obstacle detection (purple) (middle left); right and left cost maps (lower left); low level map (upper right); and high level map (lower right).*

The robot has a programmed stored path shown in pink on the high level map. The measurements taken when building the map were not very accurate, and were rounded to within a meter. Further, straight lines were used as estimates of the true path. Therefore, there is a fair amount of error in its measurement of heading. If it blindly followed the pink path it would run into the yellow cabinet. The disparity between the two camera images in each stereo pair provides an estimate of range to each section of the cabinet. From the range estimate, the slope and height of the cabinet are determined. Since tall and/or vertical surfaces should be avoided, both maps are updated with an obstacle where the cabinet is located.

Obstacles are shown as red overlays in the image and as brown/red marks in the map. The robot then chooses a path slightly to the left of the cabinet. Given its ability to sense the relative position of obstacles, the robot does not need a particularly accurate positioning system. It drives forward updating its position using only the wheel encoders and inertial measurement unit (IMU) for about 8 m until it senses the active RFID tag mounted on a portable stand just before the point where it will need to turn 0.8 rad (90 deg) to exit the room. It can sense this tag from about 2 m away. If the tag could have been detected farther away it could not update the position as accurately. If the detection range were more limited there would be an increased risk of passing the tag without detecting it.

When this tag is detected, both the current position estimate and heading are corrected. The heading is corrected by comparing the difference between the measurements made using the IMU and encoder data at the first passive RFID tag near the charger with the known locations of each tag from a table stored previously. For example if the IMU/encoder output had indicated that the vehicle had gone northwest, but the second tag is known to be north of the first the current IMU/encoder, heading should be corrected 0.4 rad (45°) clockwise. The programmed path indicates a left turn just after the position associated with the active RFID tag. The robot turns left, primarily relying on stereo to find a doorway to pass through and onto the shop floor and on for another ten meters until passing over another passive RFID tag.

The programmed path includes annotations to pause at certain positions, including the position of this tag. If this vehicle had the ability to carry parts, for example, instead of simply pausing, the tag could trigger a docking maneuver to load or unload parts during this pause. After a timer counts down, the robot resumes movement and continues on its approximately planned path. The programmed path down the lane is simply a straight line, but the robot can not drive a straight trajectory since the lane is partially blocked on one side by a set of lockers and on the other at different locations by chairs, trash cans, machines, etc.

The robot is programmed to stop only if necessary, such as if a person is standing in the center of the lane. It will adjust its trajectory if the lane is only partially blocked and it can still find a path wide enough for itself and an additional safety margin. In this case, it continues past the obstacles. The robot continues on its preplanned path, turning at each of three additional active tags before coming to a halt at a final passive tag about 70 m from the start position.

4. Conclusions

The NIST 4D/RCS reference model architecture was implemented on the DARPA LAGR vehicle, which was used to demonstrate learning in this architecture. Sensor processing, world modeling, and behavior generation processes have been described in this paper.

Based on results of a next generation study for the AGV industry, a demonstration was developed at NIST using the LAGR vehicle and controller with additional sensors and enhanced processes. The demonstration was shown to the AGV industry resulting in collaborative activities with the industry expected in the near future.

Future research will include integration of the 2D safety sensor to eliminate false positives on obstacles near ground level caused by low stereo disparity. Demonstration of controlling more than one intelligent vehicle at a time in unstructured environments with other moving obstacles will be studied.

5. Acknowledgements

The authors would like to thank Richard Bishop of Bishop Consulting, Inc. for performing the Next Generation Survey of the AGV industry. We are also grateful for the support of the DARPA LAGR program.

6. References

[1] Bishop, R., “Industrial Autonomous Vehicles: Results of a Vendor Survey of Technology Needs,” Bishop Consulting, February 16, 2006.

[2] Albus, J.S., Huang, H.M., Messina, E., Murphy, K.N., Juberts, M., Lacaze, A., Balakirsky, S.B., Shneier, M.O., Hong, T.H., Scott, H.A., Proctor, F.M., Shackleford, W., Michaloski, J.L., Wavering, A.J., Kramer, T., Dagalakis, N.G., Rippey, W.G., Stouffer, K.A., “4D/RCS Version 2.0: A Reference Model Architecture for Unmanned Vehicle Systems,” NISTIR, 2002.

[3] Albus, J.S., Balakirsky, S.B., Messina, E., “Architecting A Simulation and Development Environment for Multi-Robot Teams,” Proceedings of the International Workshop on Multi Robot Systems, Washington, DC, March 18 – 20, 2002.

[4] Balakirsky, S.B., Chang, T., Hong, T.H., Messina, E., Shneier, M.O., “A Hierarchical World Model for an Autonomous Scout Vehicle,” Proceedings of the SPIE 16th Annual International Symposium on Aerospace/Defense Sensing, Simulation, and Controls, Orlando, FL, April 1-5, 2002.

[5] Albus, J.S., Juberts, M., Szabo, S., “RCS: A Reference Model Architecture for Intelligent Vehicle and Highway Systems,” Proceedings of the 25th Silver Jubilee International Symposium on Automotive Technology and Automation, Florence, Italy, June 1-5, 1992.

[6] Bostelman, R.V., Jacoff, A., Dagalakis, N.G., Albus, J.S., “RCS-Based RoboCrane Integration,” Proceedings of the International Conference on Intelligent Systems: A Semiotic Perspective, Gaithersburg, MD, October 20-23, 1996.

[7] Shackleford, W., Stouffer, K.A., “Implementation of VRML/Java Web-based Animation and Communications for the Next Generation Inspection System (NGIS) Real-time Controller,” Proceedings of the ASME International 20th Computers and Information in Engineering (CIE) Conference, Baltimore, MD, September 10 – 13, 2000.

[8] Michalowski, J.L., Warsaw, B.A., “Robot Control System Based on Forth,” Robotics Engineering, Vol. 8, No. 5, pgs 22-26, May 1986.

[9] Jackel, Larry, “LAGR Mission,” http://www.darpa.mil/ipto/programs/lagr/index.htm, DARPA Info. Processing Technology Office, 2005.

[10] Albus, J., Bostelman, R., Hong, T., Change, T., Shackleford, W., Shneier, M., “Learning in a Hierarchical Control System: 4D/RCS in the DARPA LAGR Program” Journal of Field Robotics, Special Issue on Learning in Unstructured Environments (Submitted), 2006.

[11] Konolige K., “SRI Stereo Engine,” http://www.ai.sri.com/~konolige/svs/, 2005.

[12] Chang, T., Hong, T., Legowik, S., Abrams, M., “Concealment and Obstacle Detection for Autonomous Driving,” Proceedings of the Robotics & Applications Conf., Santa Barbara, CA, October 1999.

[13] Oskard, D., Hong, T., Shaffer, C., “Real-time Algorithms and Data Structures for Underwater Mapping,” National Institute of Standards and Technology, 1990.

[14] Shackleford, W., “The NML Programmer's Guide (C++ Version),” http://www.isd.mel.nist.gov/projects/rcslib/NMLcpp.html, 1990.

[15] Heyes-Jones, J., “A* algorithm tutorial,” http://us.geocities.com/jheyesjones/astar.html, 2005.

[16] Shneier, M., Chang, T., Hong, T., and Shackleford, W., “Learning Traversability Models for Autonomous Mobile Vehicles,” Autonomous Robots (submitted), 2006.

[17] Int’l Assoc. of Science and Technology for Development (IASTED) “B56.5 Safety Standard for Guided Industrial Vehicles and Automated Functions of Manned Industrial Vehicles,” http://www.iasted.org/, 2006.