An Intelligent World Model for Autonomous Off−Road Driving

Tsai Hong Hong, Marilyn Abrams, Tommy Chang, Michael Shneier
Intelligent Systems Division
National Institute of Standards and Technology
100 Bureau Drive Stop 8230
Gaithersburg, MD 20899−8230

Abstract

This paper describes a world model designed to act as a bridge between multiple sensory inputs and a behavior generation (path planning) subsystem for off−road autonomous driving. It describes how the world model map is built and how the objects and features of the world are represented. The functions used to maintain the model are explained and the sensors and sensory processing used to provide data for this application are discussed. The paper includes examples of integrating and fusing sensory data from multiple sources into the world model map. The representation is being developed for the Army's Demo III autonomous driving experiment, which is an on−going research project. The paper concludes with a discussion of future research directions.

1 Introduction

The ability to drive autonomously cross country, over rugged terrain is critical to the success of the Demo III program [16]. The requirements for the Experimental Unmanned Vehicle (XUV) developed for Demo III include the ability to drive autonomously at speeds of up to 60 kilometers per hour (km/h) on−road, 35 km/h off−road in daylight, and 15 km/h off−road at night or under bad weather conditions. The control system for the vehicle is designed in accordance with the 4D−Real−time Control System (RCS) [1]architecture which divides the system into perception, world modeling and behavior generation subsystems. This paper's focus is on the design and functionality of the world model. Section 2 provides an overview of the world model module. Section 3 discusses the sensors and algorithms used to provide information to the world model. Section 4 describes the map and object representations used by the world model and the functions used to maintain the model. Section 5 describes future work and discusses issues related to the world model representation, and conclusions are presented in Section 6 .

2 World Model

The world model is the system's internal representation of the external world. It acts as a bridge between sensory processing and behavior generation by providing a central repository for storing sensory data in a unified representation, and decouples the real−time sensory updates from the rest of the system. The world model process has two primary functions:

1. To create a knowledge database (map) and keep it current and consistent. In this role, it updates existing data in accordance with inputs from the sensors, and deletes information no longer believed to be representative of the world. It also assigns (multiple) confidence factors to all map data and adjusts these factors as new data

are sensed. The types of information included in the map are state variables (e.g.,. time, position, orientation), system parameters (e.g., coordinate transforms, sensor to vehicle offsets, etc.), and lists or classes of sensed objects. The world model process also provides functions to update and fuse data and to manage the map (e.g. scrolling and grouping objects.)

2. To generate predictions of expected sensory input based on the current state of the world and estimated future states of the world. For the Demo III off−road autonomous driving application, very little *a priori* information is available to support path planning between the vehicle's position and a final goal position. The world model therefore constructs and maintains all the information necessary for intelligent path planning[8].

The world model implementation fuses information from multiple sensors, including navigation sensors, Ladar, and stereo vision. The navigation system provides information about the vehicle's current position, orientation, speed, velocity, etc. Section 4.1.1 describes the navigation system's units and coordinate systems. Data from the Ladar sensor (Section 3.1) includes a range image (32 rows x 180 columns) processed to provide and array in which each element contains the data described in Table 1. The vehicle is equipped with two pairs of stereo cameras. One provides color imagery, while the other provides infra−red (FLIR) data. The information obtained by processing the stereo range information is shown in Table 2.

Datum	***Description***
Range value	Range (m) read from the sensor
Position	Elevation (m), North/East position and orientation in NIU[1] coords.
Obstacle label	Valid/Invalid; Obstacle; No obstacle
Terrain class label	Tall grass; ground; cover

Table 1 *Processed Ladar Outputs*

Datum	***Description***
Time	System timestamp
Cell position	North/East in NIU coord.
Elevation	Elevation measure (m) and confidence
Roughness	Terrain roughness measure and confidence
Obstacle label	Obstacle; No obstacle; Positive/Negative Obstacle
Terrain class label	Tall grass; bush; tree; rut; soil; rock; outlier

Table 2 *Processed Stereo Range Outputs*

1 The Navigation Interface Unit (NIU) coordinate system is based on a fixed translation from UTM coordinates and the location of the NIU on the vehicle. It is different for each system start−up.

The world model map fuses all sensory information into its map representation (Section 4). The primary use of the model data is to plan safe and efficient paths. The path planning module uses the map to select a locally optimal path from the current position to the commanded goal. The planner heuristically selects a path by starting with a web of potential path segments that extend out 20 m [8] [11]. The path is updated (replanned) approximately ten times per second.

There are two types of path segments: straight and curved. Curved segments extend 20 m from the vehicle. Each is a series of clothoid segments which are kinematically feasible based on the turn rate of the steering wheel. These paths are generated offline for different initial speeds and steering wheel positions. Straight path segments are used from 20 m to 50 m. Although not kinematically feasible, they are computationally simpler. Given that the path is recomputed frequently, an exact solution for more distant segments is not necessary. The planner selects the best combination of segments leading to the goal, using a cost function depending on its task. It does this by first pruning all segments blocked by obstacles. Then it searches through the remaining segments to find the path with lowest cost. Among the factors used in the cost function are terrain elevation, the presence or absence of obstacles, tree cover, the presence of tall grass areas, the relative distance between different path options, and commanded speed.

3 Sensors and Sensor Processing

Sensor processing algorithms use sensor data to compute vehicle position, range, obstacle lists, obstacle positions, and terrain information. The suite of sensors used in the mobility system include a Schwartz Electro−Optics (SEO) Scanning Laser Rangefinder (Ladar), a pair of color cameras for stereo vision, a stereo pair of infrared (FLIR) cameras, a stereo pair of monochrome cameras, a pan−tilt platform, a Global Positioning System (GPS) sensor, a force bumper that alerts the system to obstacles in the vehicle's immediate path, and an Inertial Navigation System (INS) sensor[2]. The Ladar and stereo camera sensors are described in Sections 3.1 and 3.2. All sensors are mounted on the vehicle, which is equipped with electric actuators on the steering, brake, transmission, transfer case, and parking brake. Feedback from the sensors provides the controller with engine rotations per minute, speed, temperature, fuel level, etc. Multiple navigation sensors are used. A Kalman filter [7] computes vehicle position and orientation using data from the inertial dead reckoning system and the carrier phase differential GPS unit.

3.1 Ladar Sensor

The Ladar sensor is mounted on a pan/tilt platform to increase its field of view. The range of the tilt motion is ~~+/−~~± 30° resulting in an effective Ladar field of view of about 80°. Range data are read into an image array containing 32 rows and 180 columns. Using *a priori* knowledge about the location and orientation of the Ladar mounting on the vehicle, calibration factors, and vehicle position data, we transform the range information into position and orientation values in a world coordinate frame. Section 3.1.1 describes the sensor processing algorithms used to detect and label obstacles using Ladar data. Section 3.1.2 describes the algorithm we use to classify the detected obstacles. Table 3 shows the specifications of the SEO Ladar [14].

2 Certain commercial equipment, instruments, or materials are identified in this paper in order to adequately specify the experimental procedure. Such identification does not imply recommendation or endorsement by NIST, nor does it imply that the materials or equipment identified are necessarily best for the purpose.

Property	*Specification*
Scan resolution	32 lines × 180 pixels
Scan coverage	±10.2° × ±43°
Angular resolution	0.658° × 0.5°
Maximum frame rate	60 scans/s
Range	5 m to 45 m (20 % Target Reflectivity)
Range resolution/accuracy	±7.6 cm
Data measurement rate	Range: 345,600 measurements/s.
Data measurement rate	Intensity: 345,600 measurements/s.
Day/Night Operation	Range Independent of ambient light
Intensity	256 grey level values

Table 3 *SEO Ladar Specifications*

3.1.1 Obstacle Detection

Obstacles are defined as objects that project more than some distance *d* above or below the ground plane. Positive obstacles are detected in the range images, while negative obstacles are detected in the world model map.

The positive obstacle detection algorithm works column by column in the Ladar range image [3]. The algorithm starts with a point, *g*, known to be on the ground. An initial ground value is assigned at the location where the front wheels of the vehicle touch the ground, known from INS and GPS sensors. Given point *g*, the algorithm processes upwards from the bottom pixel in the column to the top pixel, as follows:

1. Let p_i be the i^{th} pixel in the column, where pixel 0 is at the bottom of column. Let $x_{i,} y_{i,} z_i$ be the Cartesian coordinates of p_i. Let *g* be the last known ground pixel in the column, initially obtained from the vehicle's position sensors.
   Compute the slope between the ground point, *g*, and the next pixel p_k. Pixel p_k is labeled a positive obstacle if

$$\frac{\left(z_k - z_g\right)^2}{\left(x_k - x_g\right)^2 + \left(y_k - y_g\right)^2 + \left(z_k - z_g\right)^2} \geq \sin^2\left(\alpha\right)$$

   where α is a predefined constant representing the maximum allowed slope. The value of $\sin^2\left(\alpha\right)$ is constant, and is pre−computed for efficiency.

2. Pixel p_k may fail the above test but still be a positive obstacle. This is because the slope test is a function of distance. The obstacle can be far from the current ground point due either to occlusion or to the resolution of the sensor which degrades as a function of distance. To resolve this ambiguity, the height of the obstacle is required to be greater than a constant, *H*.

i.e., $|z_k - z_g| < H$

3. If p_k is not an obstacle, it is assumed to be ground and replaces g as the current ground pixel. The process iterates up the column with each pixel being compared to the closest ground pixel.
4. If p_k is an obstacle, g is unchanged, and is compared with pixels $p_k, p_{k+1}, \ldots$ as above, until another ground pixel is found. When this occurs, point g is set to the new ground pixel value and the process repeats.

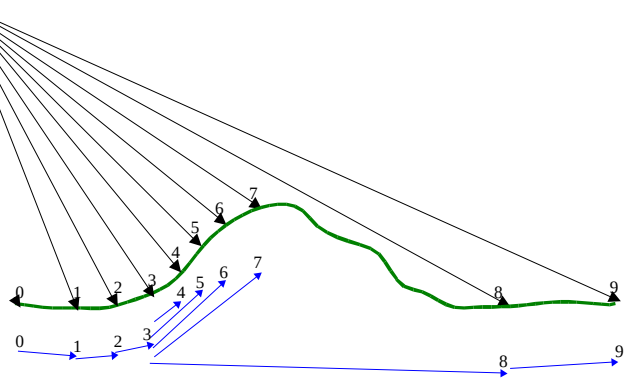


Figure 1 Positive obstacle detection

In Figure 1, pixel 0 corresponds to the bottom of the vehicle wheel. Pixels 1, 2, 3, 8 and 9 are ground pixels. Pixel 4, 5, 6 and 7 are positive obstacles because they fail step 1 and satisfy step 2. The direction vectors shown on the bottom of Figure 1 indicates the vectors in which the slopes are determined. In a way, the algorithm is analogous to flooding; pixels 1, 2, 3, 8 and 9 are flooded because they have shallow slopes.

The results of the positive obstacle detection are shown in Figure 2. The figure on the left is a Ladar scene of a wall obstructed by a truck on the far right. The objects in the foreground are low poles. The figure on the right shows the objects detected as positive obstacles in this scene.

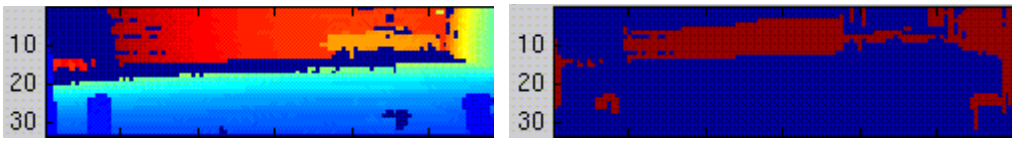


Figure 2 Raw ladar image and detected positive obstacles

The negative obstacle detection algorithm maintains its own high−resolution ground map centered on the vehicle. This ground map contains all the projected ground pixels detected by the positive obstacle detection module. The algorithm first identifies the pixels in the range image that potentially correspond to a negative obstacle (see algorithm details below). Based on the accumulated ground information in the ground map, the algorithm determines more precisely the dimension of the negative obstacle. Thresholds for depth and width are used to reject pixels that correspond to negative obstacles that are too small. For efficiency, the algorithm detects only the borders of negative obstacles. Steps 1 thru 4 describe the algorithm in detail.

1. Let p_k be a ground point, and let w and d be the approximate width and depth of the negative obstacle. i.e., $w = x_k - x_g$ and $d = z_g - z_k$. Let $proj_{p_k}$ be the map location corresponding to the

projection of onto the ground map. Let be the minimum depth for an obstacle, and the minimum width smaller than the vehicle wheel diameter.

2. If both $d < d_{min}$ and $w < w_{min}$ are true, then $proj_{p_k}$ is marked as a ground location and no further work is done.
3. If $proj_{p_k}$ is within a neighborhood corresponding to the area along the line of sight from the closest ground point on the map, then p_k is not labeled as a negative obstacle, and $proj_{p_k}$ is marked as a ground cell.
4. When both $w \geq w_{min}$ and $d \geq d_{min}$ are true and no ground map cell exists within the neighborhood, p_k is labeled a negative obstacle, and $proj_{p_k}$ is not marked as a ground cell.

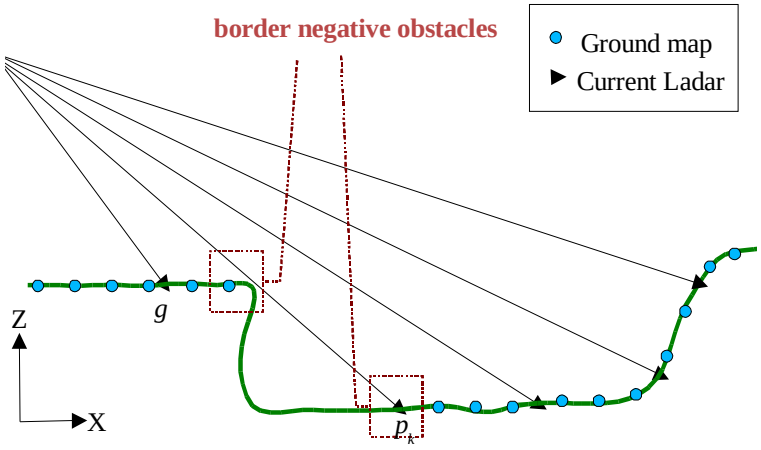


Figure 3 Negative obstacle border detection

Figure 3 graphically describes this process. The points marked as circles represent points in the ground map. The triangular points represent current Ladar hits. The points enclosed in squares fulfill the requirements described in Step 4 above and are labeled as negative obstacle borders.

3.1.2 Obstacle Classification

After a pixel has been labeled as an obstacle, additional processing is performed to classify the obstacle type. A number of factors were considered in selecting a classification algorithm for Ladar data. The most important was the need for real−time operation. Also important was the ability to operate on poor quality data. The quality of the range data precludes more than a coarse classification, which currently identifies vegetation and ground (i.e., not vegetation).

The approach is based on the method of Ojala, Pietikännen, and Harwood [12]. It makes use of two texture measures, Local Binary Patterns (LBP), and Contrast. Local Binary Patterns are computed on 3×3 windows, as follows (Figure 4.) First, the center pixel value is used to threshold the other pixels in the window. This results in values of 0 if the pixel is less than or equal to the center pixel, and 1 if it is greater. In order to produce a compact pattern, a weighted sum is computed of the eight surrounding thresholded points. The weights are assigned as powers of 2, so that each location has a unique weight, which indexes a unique bit position in the pattern, associated with that location. Given that there are eight surround pixels, and each has

value 0 or 1 after thresholding, the final value assigned by the operator to the central pixel can be represented by an eight−bit byte, making the implementation very efficient. The LBP values are combined with a contrast measure at each point, computed over the same window. The contrast measure is computed as the difference between the averages of the range values of the pixels greater than the center pixel and those less than the center pixel value. That is, those that have value 1 in Figure 4(b) and those that have value 0. Contrast measures are quantized into eight values. The range contrast can be viewed as a measure of porosity of a volume. If the volume is sparsely filled, the contrast will be large, as in the case of grass. If the foliage is denser, such as for brush or thick shrubs, the contrast will be smaller, since the expected distance the laser will travel before hitting a surface will decrease. If the surface is solid, the contrast should be close to zero.

12	25	13
33	15	17
10	18	5

0	1	0
1		1
0	1	0

2^0	2^1	2^2
2^3		2^4
2^5	2^6	2^8

0	2	0
8	90	16
0	64	0

Figure 4 (a) A 3x3 neighborhood. (b) Result of thresholding by middle value. (c) Weights applied to each thresholded pixel. (d) Resulting value is sum of weighted thresholded values.

The texture measures are computed after the obstacles are computed. First, the connected components of the obstacle image are found, and are surrounded by a bounding box at least 16×16 pixels in size. For each obstacle, a two−dimensional histogram is computed from the texture measures applied to the pixels in the bounding box. The histogram is used to determine the class to which the obstacle belongs. Classes are defined by models, created in a learning phase. A model is simply a two−dimensional histogram computed from the LBP and contrast measures. It is learned by extracting and combining the LPB and contrast measures from a sequence of images containing obstacles that are known to belong predominantly to a single class. In [12], matching the models to the sample data is done using Kullback discrimination. This was too slow for our purposes, and a simple minimum of summed difference of squares was used instead. This works almost as well, especially given the coarseness of the Ladar data. Figure 5 shows a color coded raw Ladar image of a grassy scene and the classification results. Figure 6 shows a scene taken at Ft. Knox with tall grass and obstacles.

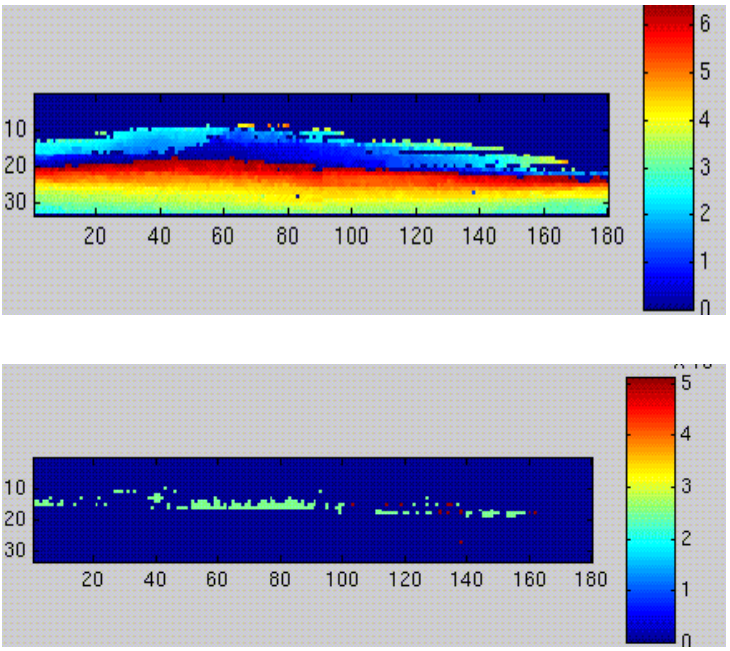


Figure 5 Top: Color coded Ladar image. Bottom:Region classified as tall grass

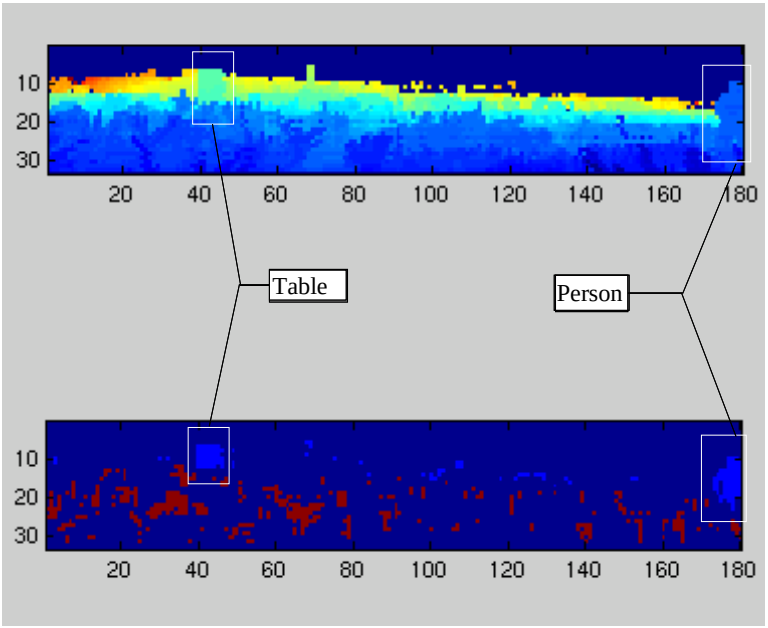


Figure 6 *Top: A range image from Fort Knox, with a table and a person in the image. Bottom: The results of classifying the obstacles in the scene. Red represents vegetation, while Blue represents non−vegetation objects that are still labeled as obstacles.*

3.2 Stereo Vision Sensors

Stereo vision provides another way of computing range information. The system is equipped with a color camera pair with a 60° field of view (FOV) and a FLIR camera pair with a 40° FOV for night vision. The stereo system includes an iris controller; an image acquisition unit; a stereo range algorithm; positive and negative obstacle detection algorithms [9]; and a terrain classification algorithm [2].

A multiresolution, coarse to fine, approach is taken to determining correspondence between the left and right images. First, the two images are rectified to align their scanlines in order to increase processing efficiency. A difference of Gaussian image pyramid is then constructed, and image similarity is computed using a sum of squared difference measure for 7×7 windows over a fixed disparity range. The disparity is estimated, and bad matches are removed using consistency constraints. After smoothing, the pixels are transformed to three−dimensional points by triangulation. Figure 7b is an example of a depth map (320×240 pixels) computed from a stereo FLIR image set. The left image of the set is shown in Figure 7a. The image sets shown in this section are discussed in Reference [10]. Section 3.2.1 discusses obstacle detection using stereo range data and Section 3.2.2 discusses a color image classification algorithm.

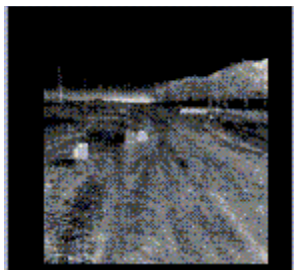
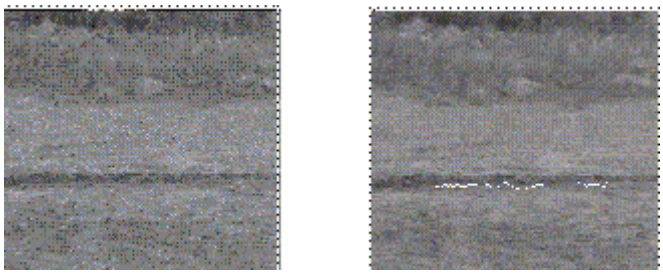


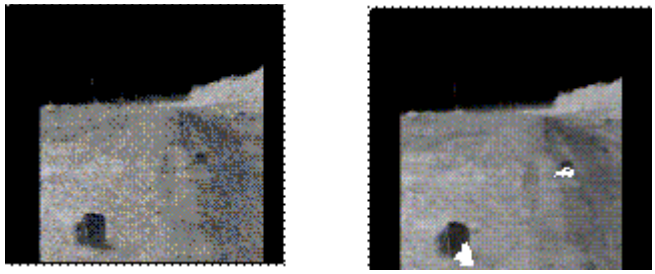
Figure 7 (a) FLIR image (b) Depth (range) map

3.2.1 Obstacle Detection

For each range image column, a set of obstacle detectors is applied to extract gaps and discontinuities in the range data that indicate non−traversable regions. Non−traversable regions are classified into either negative (Figure 8b) or positive obstacles (Figure 8d). Negative obstacles are detected by checking for gaps in the range data followed by a range jump. This usually occurs when a ditch or hole exists. Positive obstacles are detected by checking for upward slanted edges in the range data, i.e., any upward protrusion out of the ground plane steep enough to be non−traversable or to cause a tip− over hazard.



(a)Left intensity image of ditch (b)Overlay of detected obstacle



(c)Left FLIR image of rocks (d)Overlay of detected obstacles

Figure 8(a) Left intensity image of a ditch. (b) Negative obstacle detected. (c) Left FLIR image (d) Positive Obstacle detected

3.2.2 Terrain Classification

Terrain classification is performed on color images taken from one of the stereo images. Consequently, the classification label is registered with the range image. Classification types currently include green vegetation, dry vegetation, soil/rock, ruts, tall grass, and outliers. The classification algorithm relies on color, and is based on Bayesian assignment. The class likelihoods are represented using a mixture−of−Gaussian model. The parameters of the model are estimated by training data using the Expectation Maximization algorithm. Reference [2] discusses the details of the algorithm.

4 World Model: Maps, Objects and Functions

This section describes the organization of the world model in detail. Section 4.1 describes the actual map structure as well as the terminology used to define and classify objects and terrain in the map. Section 4.2 describes the computational functions performed by the world model to maintain its representation.

4.1 Maps and objects

A modified form of Hebert's [4] grid obstacle map was adopted for representing obstacles in a way suitable for path planning and vehicle control. The map consists of a header and a square, two−dimensional array of cells (Figure 9). Sections 4.1.1 and 4.1.2 describe the contents of each of these elements in detail.

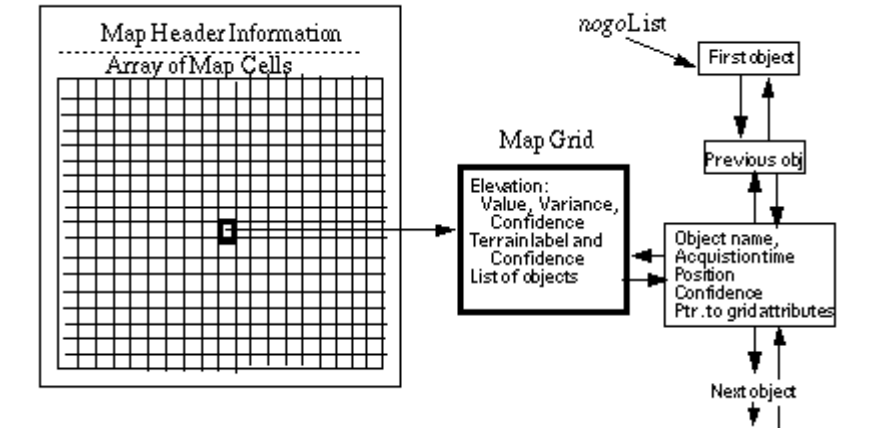


Figure 9 World Model Map

Datum	***Description***
Grid size	The size of each map grid (cell) in meters.
Map size	The number of grids in each dimension of the map.
Map center	Grid center in NIU coordinates (north, east)
Grid center location	Grid center location in the array data structure of the Map
Vehicle position	Current (x,y,z) position of vehicle in NIU coord.
Vehicle orientation	Current (roll, pitch, yaw) angles of vehicle in NIU coord.
Predicted position	Predicted vehicle position at next computational cycle
Predicted rotation	Predicted vehicle orientation at next computational cycle
Predicted curvature	Predicted steering curvature at next computational cycle
Predicted speed	Predicted vehicle speed at the next computational cycle
Offset position	Offset between vehicle position UTM[3] coord. and NIU coord.

Table 4 World Model Map

4.1.1 Header

The map header contains information that is shared by all map grids. Table 4 describes the data fields.

4.1.2 Map Grid Array

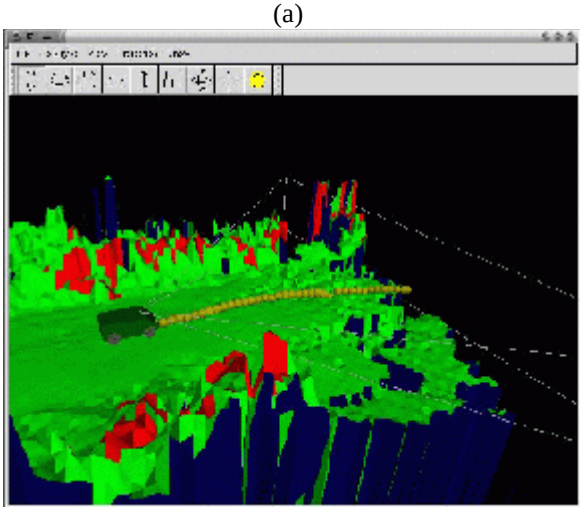
The map is a two dimensional array (301 x 301 cells) containing information extracted from processed sensor data. Figure 10a shows an image of the XUV on an unpaved road at Fort Knox; Figure 10b displays this scene as an elevation map constructed from the information in the world model. The position of the XUV is shown as an overlay on the map; the yellow path in front of the XUV represents the vehicle's planned path, and red areas represent unclassified

3 UTM (Universal Transverse Mercater) coordinates are the earth's coordinates read from a Global Positioning System (GPS).

obstacles.

Each map grid cell represents an area defined by the header grid size (currently 0.4 m × 0.4 m) and is marked with the time it was last updated. The total extent of the map is 120 m x 120 m.

(a)



(b)

Figure 10(a) XUV at Ft. Knox (b) World Model elevation map of (a)

The information stored in a cell includes:

1. The average ground elevation height; the variance of the height; and a confidence measure reflecting the "goodness" of the elevation data.
2. A data structure describing the terrain covered by the grid cell. This includes a terrain label (e.g., tall grass, water, ruts, etc.), and a cost factor for determining the relative safety of traversing the grid. The terrain label represents the best estimate of the terrain type based on information fused over time. Each terrain type has a confidence associated with it, and the map grid selects the label with the highest confidence. For ease in path planning, the cost of paths is computed based on terrain type.
3. A linked list structure describing the type of object viewed by the sensor. Examples of

different types of objects include roads, buildings, fences, positive or negative obstacles, traversable or non−traversable regions, etc. Each object has a name, a position, a time stamp and a confidence measure. For each cell, objects are stored in a linked list containing pointers to the previous object in the list, the next object in the list, and a pointer back to the map grid describing the object attributes. Certain object labels are explicitly declared, e.g. a non−traversable region is flagged as a "*no go*" area.

4.2 Maintaining and Updating the World Model

The world model map must be maintained and updated in a timely manner. World model functions have been developed to scroll the map as the vehicle moves, to update map data, to fuse data from redundant sensors, and to extract and group information from the object lists. These functions are described in the following sections.

4.2.1 Map Scrolling

An efficient, scrolling, local map is used that updates new sensor data while keeping the vehicle centered on the map. This approach has the advantage of minimizing grid relocation. No copying of data is done; only updating. When the vehicle moves out of the center grid cell of the map, the scrolling function is enabled. The scrolling function includes recentering the map and reinitializing the map borders. Because the map is vehicle−centered, only the borders of the map contain new regions that must be initialized. While the initialization information could be obtained from *a priori* maps, in the current implementation, the information in available *a priori* maps is too coarse (30 m resolution), so the borders are initialized to zero.

4.2.2 Map Updating and Fusion

The map updating algorithm is based on the concept of confidence−based mapping described in Oskard[13]. In this algorithm, confidence measures increase or decrease linearly as the model receives updated information from the sensors. When a map cell receives a vote for a class such as an obstacle, an elevation measurement, a terrain classification, etc., the cell's confidence in that class is incremented by a predefined constant.

Three different types of data must be updated and fused: non−traversable ("*no go*") regions, elevation values, and terrain classifications. The confidence measure of a "*no go*" region is updated based on the obstacles detected from Ladar data [6], stereo range data [9], and the force bumper. The obstacle's confidence increases by empirical predefined constants (i.e. a bumper obstacle constant, a stereo obstacle constant, and a Ladar obstacle constant). When the confidence measure exceeds a pre−defined threshold, the map grid is labeled "*no go*."

The elevation confidence of each grid cell is updated every sensor cycle, e.g., 10 Hz for Ladar data and 1 Hz for stereo data. The new elevation is updated by the weighted average of the current sensed elevation value and the accumulated elevation value:

$$Elevation_t = \frac{(W_i * Elevation_i) + (Conf_{t-1} * Elevation_{t-1})}{Conf_t}$$

$$\textbf{\textit{If}} \quad (Conf_t \leq MaxValue) \quad \textbf{\textit{then}} \quad (Conf_t = Conft_{t-1} + W_i)$$

$$\textbf{\textit{else}} \quad (Conf_t = MaxValue)$$

$Conf_t$ is the confidence measurement of the elevation value at time *t*. W_i is the empirical confidence measurement for sensor$_i$. $Elevation_t$ is the current elevation value read from sensor$_i$ at time *t*, and $Elevation_t$ is the estimated elevation value at time *t*.

Conversely, when a map cell is labeled "ground", the "*no go*" confidence decreases. Decreasing the "*no go*" confidence measure reduces "*no go*" false alarm, and results in a fairly static environment.

A map cell is classified into a terrain type and an object type. Examples of terrain types are tall grass, water, ruts, etc. Examples of object types are rocks, bushes, trees, etc. The confidence of these classes is updated based on the results of both Ladar classification algorithms [5] and stereo classification algorithms [2]. Confidence values increase by a factor determined by sensor characteristics, which are learned off−line by analyzing and testing data collected for this purpose. Confidence values are used as a cost factor in determining the traversablility of a grid.

4.2.3 Object Grouping

The map cell contains a pointer to the object list, and each list node contains a pointer to the previous entry, a pointer to the next entry, and a pointer to the originating (attribute) grid. "*No go*" regions are an example of an object group (Figure 9). The object list also contains the object's position in the grid, the object type, the time stamp, and the confidence measure. The attribute pointer currently points back to the map grid location. This object list is used to compute the object's properties, i.e., velocity, size, and moments, which can be used for object recognition. This data structure is very similar to the one described by Shneier [15] in which objects are indexed spatially by a pointer associated with each node in an octree.

5 Future Work and Discussion

This paper has described the implementation of a sophisticated world modeling system, but much research remains to be done to add to its capabilities and performance. One area of future research involves tracking moving obstacles because confidence−based mapping may not be adequate for this task. Currently, a hypothesis/prediction model for predicting the motion of moving obstacles is not implemented. Such an approach would be useful for maintaining knowledge of local obstacles at the sensory processing level. This knowledge could be used to improve detection accuracy and could also be used to detect moving obstacles.

The use of *a priori* maps would enhance the scope of the current world model. Currently survey maps, GPS maps, and aerial maps contain fine resolution and significant information about existing topology and structures. In order to take advantage of this knowledge, research is needed to register *a priori* maps with our sensor generated map. Also, higher resolution *a priori* maps must be generated; current geological survey maps are too coarse for autonomous driving applications.

Additional research is also needed to broaden the system's terrain and object classification capabilities. The number of terrain and object classes currently used is small; the ability to recognize and label bodies of water, rocky roads, buildings, fences, etc. would enhance the vehicle's autonomous driving performance.

6 Conclusion

A world model has been described that is designed to act as a bridge between multiple sensory inputs and a behavior generation (path planning) subsystem for off−road autonomous driving. The world model map and object representations have been described, as well as the functions used to maintain the model. To more fully understand the application, the sensors and sensor−processing algorithms were presented, and the information extracted from them was described. Also included were examples of integrating and fusing sensory data from multiple sources into the world model map. The representation is currently being used as part of the Demo III autonomous driving experiment at Ft. Knox. The map is used by a path planner [1] module to compute safe and task−appropriate routes [8]. The current world model represents an on−going research project. Possible future research directions were discussed to enhance the world model's capabilities.

REFERENCES

[1] Albus, J., "4−D/RCS Version Reference Model Architecture for Unmanned Ground Vehicles," Proceedings of the 2000 International Conference on Robotics and Automation, San Francisco, CA., April, 2000.

[2] Belluta, P., Manduchi, R., Matthies, L., Owens, K., Rankin, A., "Terrain Perception for Demo III", Proceedings of the Intelligent Vehicles Symposium, Dearborn, Michigan, October 2000.

[3] Chang, T., Hong, T., Legowik, S., Abrams, M., "Concealment and Obstacle Detection for Autonomous Driving," Proceedings of the Robotics & Applications 1999 Conference, Santa Bar−bara, CA, October 1999.

[4] Herbert, M., Thorpe, C., Stentz, A., Intelligent Unmanned Ground Vehicles, Autonomous Navi−gation Research at Carnegie Mellon, Kluwer Academic Researchers, 1997.

[5] Hoffman, R., Jain, A.K., "Segmentation and Classification of Range Images," PAMI(9), No. 5, pp. 608−620., Septemper 1987.

[6] Hong, T., Legowik, S., Nashman, M., "Obstacle Detection and Mapping System," NISTIR 6213, August 1998.

[7] Kalman, R.E., "A new approach to linear filtering and prediction problems," Trans. ASME, Series D,J. Basic Eng., V.82, pp. 35−45, 1960.

[8] Lacaze, A., Albus, J., Meystel, A., "Planning in the Hierarchy of NIST−RCS for Manufacturing," Proceedings of the International Conference on Intelligent Sysyems: A Semiotic Perspective, Gaith−ersburg, MD, October 20−23, 1996.

[9] Matthies, L., Kelly, A., Litwin, T., "Obstacle Detection for Unmanned Ground Vehicle: A Pro−gress Report." Jet Propulsion Lab., April 1995.

[10] Matthies, L., Litwin, T., Owens, K., Murphy, K., Coombs, D., Gilsinn, J., Hong, T., Legowik, S., Nashman, M., Yoshimi, B. "Performance Evaluation of UGV Obstacle Detection with CCD/FLIR Stereo Vision and LADAR", IEEE Workshop on Perception for Mobile Agents, Santa Clara, CA, June 1998.

[11] Murphy, K., Abrams, M., Balakirsky, S., Coombs, D., Hong, T., Legowik, S., Chang, T., La−caze, A., "Intelligent Control for Unmanned Vehicles," Proceedings of the World Automation Con−gress Conference (WAC 2000), June 2000.

[12] Ojala T., Pietikainen M., Harwood D., "A comparative study of texture measures with classifica−tion based on feature distributions," Pattern Recognition 29: 51−59, 1996.

[13] Oskard, D., Hong, T., Shaffer, C., "Real−time Algorithms and Data Structures for Under water Mapping," National Bureau of Standards, 1990.

[14] Schwartz Electro−Optics, Inc. Ladar Specifications.

[15] Shneier, M., Kent, E., Mansbach, P., "Representing workspace and model knowledge for a robot with mobile sensors, National Bureau of Standards, 1994.

[16] Shoemaker, C. M., Bornstein, J. A., "The Demo3 UGV Program: A Testbed for Autonomous Navigation Research," Proceedings of the IEEE International Symposium on Intelligent Control, Gaithersburg, MD September 1998.