Proc. of 1997 Int'l Conference on Robotics and Automation, Albuquerque, New Mexico, April 20-25, 1997, pp. 2559-2564.

An Interactive System for Creating Object Models from Range Data based on Simulated Annealing[1]

William A. Hoff, Frederick W. Hood, Robert H. King
Center for Robotics and Intelligent Systems
Colorado School of Mines Golden, CO 80401

Abstract

Sensing to recognize and locate objects is a critical need for robotic operations in unstructured environments. An accurate 3-D model of objects in the scene is necessary for efficient high level control of robots. Drawing upon concepts from supervisory control, we have developed an interactive system for creating object models from range data, based on simulated annealing. Site modeling is a task that is typically performed using purely manual or autonomous techniques, each of which has inherent strengths and weaknesses. However, an interactive modeling system combines the advantages of both manual and autonomous methods, to create a system that has high operator productivity as well as high flexibility and robustness. Our system is unique in that it can work with very sparse range data, tolerate occlusions, and tolerate cluttered scenes. We have performed an informal evaluation with four operators on 16 different scenes, and have shown that the interactive system is superior to either manual or automatic methods in terms of task time and accuracy.

I. Introduction

In hazardous applications such as remediation of buried waste and dismantlement of radioactive facilities, robots are an attractive solution. Such environments are typically unstructured, in the sense that the types and locations of objects are not known in advance.

Control of robots and machinery for use in such operations ranges from full manual control (direct teleoperation by a human operator) to full automatic control (no input from a human operator). Between these two extremes lies a paradigm called supervisory control [1], which allows the system to perform low level tasks automatically under the supervision of a human operator. Supervisory control is a promising technique for near term operations: it retains the flexibility of human intelligence to respond to unforeseen events and combines it with the speed and accuracy of the computer for performing low-level tasks [2].

In order for the system to perform low level tasks under supervisory control, three-dimensional (3-D) graphical models are needed which accurately represent the location, type, shape, *etc.*, of objects in the scene [3]. For certain environments, blueprints and architectural drawings may exist, but these cannot be relied upon to be accurate. In unstructured environments, the 3-D site model must be created from sensor data.

The usual practice in creating object models is to physically measure their locations and then manually create them in software or with an interactive user interface. However, this is a time consuming task, and one that should be avoided in a hazardous environment. Some systems allow the locations of objects to be measured remotely, using a range finder or other sensor. However, the human operator still must manually create the object model, using the location information as a guide. Also, it is not sufficient to specify just the location of the object — the model requires the object orientation, as well as any other parameters describing its size, shape, *etc.*

Two dimensional (2-D) video images and/or three dimensional (3-D) range images can provide the necessary data to create object models. Past work on object modeling has been either primarily manual or primarily automatic. Manual techniques have the disadvantage of requiring a substantial amount of effort from the human operator. Automatic techniques have been limited in performance and reliability. Most automatic systems are limited to recognizing a small number of simple object models, in clean, uncluttered scenes. The generality and flexibility of current systems is very limited, especially for those that use image data (as opposed to range data). Therefore, there is a need for techniques which are more automated, but are robust and flexible in dealing with new environments.

In this paper, we describe supervisory techniques that we have developed for creating object models, that have been shown to be highly robust and flexible. These

[1] This project has been funded by the INEL University Research Consortium. The INEL is managed by Lockheed Martin Idaho Technologies Company for the U.S. Department of Energy, Idaho Operations Office, under Contract No. DE-AC07-94ID13223

techniques are interactive; that is, they use the human to guide the sensing system in creating and updating models of objects. The motivation for using interactive techniques is that purely manual and purely automatic techniques both have disadvantages. However, a hybrid system has the potential to combine the best attributes of each to create a system that has high operator productivity as well as high flexibility and robustness. The strategy is similar to the use of the human in supervisory control, in which human intelligence and machine intelligence are integrated to create a more powerful system that uses the best elements from each component. The human provides high level reasoning and overall guidance; the computer provides quantitative analysis and repeatable operation. Thus, we have extended the supervisory control paradigm to sensing.

We have developed a system which uses sparse, noisy range data (about 50 - 100 points per object) obtained from a stereo vision sensor. The operator can interact with the system through a combination of *traded* and *shared* control, to create models of objects in the scene. We have evaluated our system with a combination of synthetic and real scenes. In informal tests with four operators, we have shown that the supervisory system is superior (in terms of task time and accuracy) over purely manual and purely automatic modeling. Our effort is unique by virtue of the use of the interactive techniques, sparse range data collected from stereo vision and other sources, and an optimization technique called simulated annealing for fitting primitive 3-D object models to the range data.

The rest of our paper is organized as follows: Section II covers background information related to object modeling; Section III gives an overview of our system; Section IV describes implementation details and experimental results; and Section V provides conclusions.

II. Background

Past work on creating object models has generally fallen into the categories of primarily manual or primarily automatic. With primarily manual systems, the operator creates, sizes, and places a graphical object model to correspond to a physical object that he or she observes in the scene. In Sandia's Graphical Programming System [3], the operator can control the robot to touch an object with a probe in order to determine its position. Other systems use the operator to examine imagery obtained from sensors and create object models to fit the observed objects [4, 5]. Other related work includes photogrammetric reconstruction from still photographs, developed by TRW and Vexcel Corp. In all these systems, the operator rather than the computer, performs the bulk of the work in creating and specifying the attributes of the model.

In the category of primarily automatic systems, much work has been done in the computer vision field on automatically recognizing and creating models from range data. Here, we wish to draw a distinction between techniques which just construct a surface map [6] or occupancy map [7] of the scene, and those (like ours) which fit geometric models to discrete objects in the scene. In the latter class of techniques, researchers have developed systems to fit superquadric volumetric primitives [8], generalized cylinders [9], and parametric geons [10], to range data. Hebert, *et al* [11] use the operator to select an initial region of interest in the image, then automatically fits a cylinder to the surface data.

Other work matches a specific object model to range data. For example, Grimson, *et al* matches a model derived from MRI data to laser range data [12]. Besl and McKay [13] register two 3-D shapes using the iterative closest point algorithm. Most techniques use an iterative algorithm to find a solution, either based on least squares or Kalman filtering [14].

Most previous work does not address the segmentation of the data – *i.e.*, how to automatically distinguish points on the object from points in the background. Also, the objects must be relatively unoccluded for the model fitting to converge correctly. In typical unstructured environments, such as a drum half buried in a landfill, these techniques could not correctly create the object model. Finally, most algorithms require fairly dense range data with many points on the object of interest.

Recently, researchers have argued that fully automated systems for object recognition and modeling are currently incapable of matching the human's ability to employ background knowledge, common sense, and reasoning. One way to increase the flexibility of a computer vision system is to allow an outside entity, such as a human, to provide context and constraints to the vision system. This approach has been used successfully in the domain of overhead image interpretation. Recently, under DARPA's RADIUS project, much work has gone toward automating portions of the interpretation process, using image understanding techniques [15]. Much success has been achieved by developing interactive techniques to extract building and road models [16]. For example, an image analyst can provide an initial approximation for a building outline and let the vision system perform local optimization of the variables to fit the data [17]. Although these techniques are promising, they have been applied only to the overhead image interpretation domain (primarily 2-D). They must be modified to apply them to

the robotic domain (primarily 3-D), and in which viewpoints are much more unconstrained.

III. Detailed Description

Our system consists of three main functional elements: (1) a source of range data points, (2) a model fitting algorithm, and (3) interactive (supervisory) techniques. These are discussed in the sections below.

A. Stereo Vision Sensor

We have developed a stereo vision sensor for obtaining range data; that is, a computer vision system which accepts images from multiple cameras, automatically matches points between the images and computes the range via triangulation. Measuring range via stereo vision is widely considered to be a difficult problem due to the difficulty in correctly matching points between the two images. The resulting range values may be sparse and ambiguous (because of non-unique matches). We chose stereo because (a) it is a more challenging sensor (if our techniques work well with stereo, it is highly probable that they will work with active sensors), and (b) stereo has potential advantages of cost, size, and mechanical reliability over active sensors. However, our system can work with any source of range data, and we have in fact used it with data from a structured light sensor.

Our system uses three cameras in a trinocular arrangement. The purpose of the third camera is to eliminate matching errors between the first two cameras [18]. An example of a trinocular set of images is shown in Figure 1.

Figure 1 Top, left, and right images from stereo vision system.

The stereo vision system detects "interest" points in the left and right images, and matches them using a cross-correlation technique. Each candidate match determines a point in 3-D. These points are checked by verifying their presence in the predicted location in the top image. Points which are not verified are eliminated. Figure 2 shows the final range points as cross-hairs overlaid on the left image.

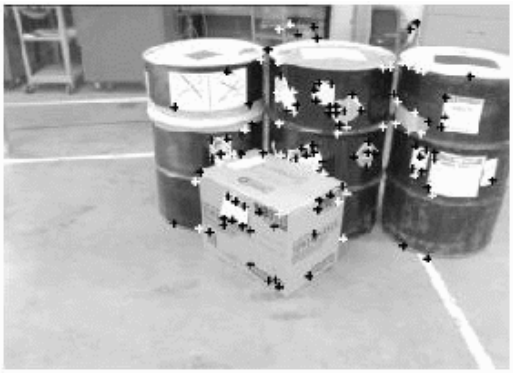

Figure 2 Final range data points (crosshairs).

We have tested the system on 13 real scenes and numerous synthetic scenes. On the average, the system produced 88 3-D points per scene. We manually examined these and found an average of 4 faulty matches remaining per scene (these points typically have a large range error).

B. Model Fitting Algorithm

An algorithm has been developed to fit geometric primitive models to range data. The algorithm iteratively adjusts a state vector representing the parameters of the model in order to minimize the error distances between the range data points and the nearest point on the surface of the model. Currently, the state vector represents only the location and orientation (6 DOF) of the model, although we are extending it to include parameters describing size and shape. The algorithm uses the "downhill simplex" method to search for the minimum [19]. However, there is a strong possibility that the algorithm will get stuck in a local minima, which may be far from the true solution.

To avoid this, a simulated annealing algorithm [20] is built on top of the downhill simplex algorithm. The simulated annealing algorithm allows the solution to occasionally move to a point with higher (worse) error. This usually allows the algorithm to escape local minima and eventually find the global minimum. A "temperature" parameter controls the probability of moving to a higher error state: the higher the temperature, the greater the likelihood of moving to a higher error state. The algorithm is analogous to the slow cooling of a metal,

which allows the atoms to arrange themselves into a crystal that is the minimum energy state (hence the name "simulated annealing"). We reduced the temperature at a constant rate; other reduction schedules are possible [21].

An important issue is the definition of the error function. One possibility is to define the error as the sum-of-squared distances between the range data points and the nearest point on the surface of the model. With this approach, points with large distances dominate the total error score. This has the problem that outliers (points which do not belong to the model) greatly affect the resulting fit. A error function is needed in which outliers do not affect the fit. This error function is defined as

$$errorscore = \sum_{i=1}^{\#pts} -G/r_i$$

where r_i = distance from the i^{th} point to the model and G is a constant. As r_i approaches infinity, the contribution of the point to the error score goes to zero. As r approaches zero, the contribution of the point to the error score would approach negative infinity; however, we limit the minimum score to $-G/D_{MIN}$, where D_{MIN} is the estimated uncertainty in position of our data points (1 cm).

This error function has the same form as the potential energy resulting from a gravitational or electrostatic force. In fact, each point may be considered to exert a force proportional to 1/r on the model, where r is the distance to the model surface. Very close points exert a large force; and very distant points exert a negligible force. Thus, outliers have a negligible effect on the overall fit.

Determining the distance from a range point to the closest visible point on the primitive object model is a three step process. First of all, a transformation is made from the camera-centered world coordinates to a coordinate system centered at the model's center and aligned along its major axis. This transformation simplifies the distance calculations dramatically. Secondly, the visible surfaces of the model are calculated using translation and orientation information from the current state vector. Finally, the location of the point, the model dimensions, and a record of the visible faces of the model are used to calculate the distance from the point to the closest visible surface. The simplicity of these operations is also due in part to the use of *primitive* objects with known dimensions.

In experimentation to date, we have found that the model fitting algorithm converges to a minimum in several thousand iterations, which takes about 10-15 seconds on a Silicon Graphics Indigo2 workstation (this also includes the time required to draw 3-D graphics). In some cases, this is the true solution (global minimum), but in other cases it is an incorrect solution (local minimum). In the latter cases, the user must intervene (see next section) in order to reach the correct solution.

C. User Interaction Techniques

We have developed software to allow the user to visualize and interactively fit 3-D graphical models on the Silicon Graphics workstation. The software displays range data points and geometric primitives (spheres, cones, cylinders, and parallelepipeds) as overlays on top of background images from the stereo cameras. These objects can be displayed in normal 2-D mode or in 3-D mode, using stereo viewing glasses. For an input device, the operator can use the usual 2-D mouse, or a 6 degree-of-freedom (DOF) mouse.

The user first selects a geometric primitive (model) to fit to the data points. The user can manipulate the size and shape of the model with a wire-frame "handle-box" surrounding the model, or by entering data into a text box. At this point, the user can manipulate the model in manual mode using the 2-D or 6 DOF mouse.

When the simulated annealing algorithm is running, the model is continuously drawn in the latest estimated position, the effect of which is that the model is drawn as an animated figure which initially jitters around and then jockeys into final position. This animation allows the user to immediately see whether the fitting algorithm is converging to the correct solution, and if not, to take action to correct the problem. In the meantime, the simulated annealing algorithm automatically reduces the temperature to a fixed percentage of the previous value. The temperature is defined to be a number between zero and 100. After the temperature has fallen to a sufficiently low level, the model stabilizes around its final position.

There are two techniques with which the user can interact with the system to provide guidance and constraints. These techniques, called *traded* control and *shared* control, draw again upon the paradigm of supervisory control [1]. In *traded* control, the user and the system each take turns controlling the pose of the model. We have found this to be particularly useful when the model fitting algorithm gets stuck in a local minimum. The user can immediately see that the model (shown as a graphical overlay on the image of the scene) has settled into an incorrect pose. By moving the mouse, the operator takes control of the model from the system, and can move it towards the correct pose. Releasing the mouse allows the system resume the model fitting algorithm. A small movement toward the correct pose is often enough to push the model out of the local minimum and allow it to automatically find the correct pose.

In *shared* control, the user controls some of the degrees of freedom, while simultaneously the system

controls the other degrees of freedom. The operator can choose any of three different modes of shared control, as shown in Table 1.

Table 1 Techniques in shared control.

Mode	*Constraints*
Operator designates a 2-D point in the image	Object center must lie along ray emanating from the current viewpoint out in the direction of the selected image point
Operator designates a 3-D point	Object center must be located at the designated point
Operator controls a set of orthogonal axes	Object orientation must match the specified axes

IV. Implementation

This section describes the implementation of the model fitting system and informal evaluations performed with volunteer operators. The system was developed using the Silicon Graphics software package called "Open Inventor", which provides high level C++ class libraries to create, display, and manipulate 3-D models.

Figure 3 User interface for model fitting application, showing cylinder model in initial pose.

Figure 4 Final pose for a sample fitting run.

The application displays the 3-D data points as small red boxes overlaid on the background image (the "left" image of the stereo set). The user selects a geometric primitive to fit to the data points, which is shown as a translucent model (Figure 3).

The user can manipulate the model in manual mode using the 2-D or 6 DOF mouse, or start the automatic model fitting (simulated annealing) algorithm. In either mode, the pose of the model is continuously displayed in the text fields on the right side of the window. Also displayed is the current "error" score, which represents the total distance of the 3-D points to the model. The color of the model changes from shades of red to green as the error decreases.

When the simulated annealing algorithm is running, it gradually reduces the temperature at each iteration. The temperature is graphically shown as a vertical bar immediately to the right of the image. The user can set the temperature manually by simply moving the bar up or down. We have found this to be useful when the model has appeared to reach the correct pose. In this case, we reduce the temperature to zero immediately to avoid any chance of the model escaping the correct solution. Figure 4 shows the model in its final pose.

An important part of this project was to quantify the benefits of the interactive object modeling techniques in terms of task completion time and model accuracy. We performed two sets of evaluations using human subjects: a preliminary, informal set of experiments using 4 subjects; later, a formal set of experiments with 14 human subjects. In each evaluation, the subjects were asked to fit models to range data using both interactive and purely manual fitting.

To begin with, subjects were trained in the use of the 6DOF mouse. Following the training, the subjects were asked to fit models to data in a variety of scenes. For the informal evaluation, there were 16 scenes (8 with real data and 8 with synthetic data). The formal evaluations used only 8 scenes (all synthetic data). Since the subjects were instructed to favor accuracy over task time, each trial was completed when the subject deemed that the closest fit had been attained.

The cumulative average results are shown in Table 2. In this table, pose error was measured from only the synthetic scenes (since we did not have ground truth pose data for the real scenes). Pose orientation error was determined by measuring the angular deviation of an axis of the model from the known ground truth axis direction. From the results, it is evident that interactive fitting was far superior to manual fitting in task time and slightly better in accuracy.

We also compared the interactive fitting mode to purely automatic mode during our preliminary evaluation.

Four fully automated (*i.e.*, no user interaction) runs from the default starting position were done for each of the 8 synthetic scenes. The system produced a correct result (almost identical to interactive runs for the same scenes) on only 16 out of 32 of these runs, or only 50% of the time. In other words, half the time it was unable to escape local minima.

Table 2: Summary of formal evaluation results.

	Manual	Interactive
Task time (sec)	192.75	40.68
Scaled error score (min 1000.0)	1101.48	1000.62
Orientation error (deg)	1.886	.758

V. Conclusions

We have developed an interactive system for fitting models to range data and have demonstrated its effectiveness (in terms of task time and accuracy) in preliminary evaluations with human operators. Unlike purely manual or purely autonomous systems, our interactive system combines the best attributes of each to create a system that has high operator productivity as well as high flexibility and robustness. Similar to supervisory control, we integrate human intelligence and machine intelligence to create a more powerful system that uses the best elements from each component. The human provides high level reasoning and overall guidance; the computer provides quantitative analysis and repeatable operation.

Our system can use extremely sparse range data (about 50 - 100 points per object), can tolerate occlusions, and work in cluttered scenes. Through a combination of *traded* and *shared* control, the operator supervises the creation of models of objects in the scene. The system uses an optimization technique called simulated annealing for fitting primitive 3-D object models to the range data.

VI. Acknowledgments

The authors wish to acknowledge the contributions of Torsten Lyon, Khoi Nguyen, Doug Swartzendruber, Rex Rideout, and the volunteer test subjects. We also wish to thank to Mark McKay of INEL for his guidance and assistance in this project. Finally, we wish to thank Charles Little of Sandia National Labs for providing us with structured light range data.

VII. References

[1] T. B. Sheridan, *Telerobotics, automation, and human supervisory control*, Cambridge, Massachusetts, MIT Press, 1992.

[2] S. Lee, "Intelligent Sensing and Control for Advanced Teleoperation," *IEEE Control System Magazine*, Vol. 13, No. 3, pp. 19-28, 1993.

[3] M. J. McDonald and R. D. Palmquist, "Graphical programming: On-line robot simulation for telerobotic control," *Proc. of International Robots and Vision Automation Conference*, pp. 22-59..22-73, 1993.

[4] J. Wagner, "Interactive computer-enhanced remote viewing system," in *Innovation Investment Area Technology Summary*, vol. DOE/EM-0146P, 1994, pp. 85-86.

[5] D. Cannon, G. Thomas, C. Wang, and T. Kesavadas, "A virtual reality based point-and-direct robotic system with instrumented glove," *Int'l J. of Industrial Engineers - Applications and Practice*, Vol. 1, No. 2, pp. 139-148, .

[6] A. Leonardis, A. Gupta, and R. Bajcsy, "Segmentation of Range Images as the Search for Geometric Parametric Models," *International Journal of Computer Vision*, Vol. 14, No. 3, pp. 253-277, 1995.

[7] Y. Roth-Tabak and R. Jain, "Building an Environment Model Using Depth Information," *Computer Magazine (IEEE)*, Vol. 22, No. 6, pp. 85-90, 1989.

[8] R. Bajcsy and F. Solina, "Three dimensional object representation revisited," *Proc. of First Int'l Conf on Computer Vision*, IEEE Computer Society, London, pp. 231-240, 1987.

[9] T. O'Donnell, T. Boult, X. Fang, and A. Gupta, "The extruded generalized cylinder: A deformable model for object recovery," *Proc. of IEEE Conf. Computer Vision and Pattern Recognition*, pp. 174-181, 1994.

[10] K. Wu and M. Levine, "Recovering parametric geons from multiview range data," *Proc. of IEEE Conf. Computer Vision and Pattern Recognition*, pp. 159-166, 1994.

[11] M. Hebert, R. Hoffman, A. Johnson, and J. Osborn, "Sensor-Based Interior Modeling," *Proc. of Robotics and Remote Systems*, American Nuclear Society, Monterey, CA, pp. 731-737, 1995.

[12] W. E. L. Grimson, et al, "An automatic registration method for frameless stereotaxy, image guided surgery, and enhanced reality visualization," *Proc. of Computer Vision and Pattern Recognition*, IEEE, pp. 430-436, 1994.

[13] P. J. Besl and N. D. McKay, "A Method for Registration of 3-D Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 2, pp. 239-256, 1992.

[14] E. D. Dickmanns and V. Graefe, "Applications of Dynamic Monocular Machine Vision," *Machine Vision and Applications*, Vol. 1, No. pp. 241-261, 1988.

[15] S. J. Gee and A. M. Newman, "RADIUS: Automating Image Analysis Through Model-Supported Exploitation," *Proc. of DARPA Image Understanding Workshop*, Morgan Kaufmann, pp. 185-196, 1993.

[16] S. Heuel and R. Nevatia, "Including Interaction in an Automated Modeling System," *Proc. of Image Understanding Workshop*, ARPA, Palm Springs, pp. 429-434, 1996.

[17] P. V. Fua and Y. G. Leclerc, "Model driven edge detection," *Proc. of DARPA Image Understanding Workshop*, Morgan Kaufmann, pp. 1016-1021, 1988.

[18] N. Ayache and F. Lustman, "Fast and reliable passive trinocular stereo vision," *Proc. of First Intl. Conf. Computer Vision*, IEEE, pp. 422-427, 1987.

[19] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*, 2nd ed., Cambridge University Press, 1992.

[20] S. Kirkpatrick, J. C.D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science*, Vol. 220, No. 4598, pp. 671-680, 1983.

[21] K. H. Hoffman and P. Salamon, "The Optimal Simulated Annealing Schedule for a Simple Model," *Journal of Physics A: Mathematical and General*, Vol. 23, No. 15, pp. 3511-3523, 1990.