

EXPERIENCE WITH VISUAL ROBOT NAVIGATION

Larry H. Matthies and Charles E. Thorpe

Department of Computer Science
Carnegie-Mellon University

Abstract The CMU Mobile Robot Lab is studying issues in the development of autonomous vehicles, including path planning, motion determination, and obstacle detection from video and sonar data. We have built a simple testbed vehicle and a visual navigation system designed to maneuver to a pre-defined location in a static environment. The visual system is based on algorithms developed by Moravec for the Stanford Cart [10]. At each Cart position, these algorithms used stereo correspondence in nine camera images to triangulate the distance to potential obstacles. Motion of the vehicle was determined by tracking these obstacles over time. This paper discusses several issues in the on-going evolution from the Cart to our present system. These issues have led to the use of fewer images per step, to the use of more constraint in the correspondence process, and toward the use of a different motion solving algorithm that better embodies the rigidity property inherent in the problem.

Introduction

The CMU Mobile Robot Lab is committed to the long-term development of robots capable of intelligent interaction with a dynamic environment. Such robots will be able to recognize objects around them, move to and manipulate these objects, and respond to motions in their field of view. Sensing for such robots will be provided by combinations of cameras, sonar and laser range finders, tactile sensors, and other technologies.

As a first step in this direction, we are experimenting with navigation in a static environment. The task we address is to maneuver a vehicle to a pre-defined location in an obstacle-filled room. The navigation system must be able to detect obstacles, plan a path to the destination, and track the position of the vehicle as it follows the path. To date our efforts have been largely limited to the use of video cameras as input.

Problems related to the one just described have received extensive attention in the computer vision literature and elsewhere. In the literature, obstacle ranging is normally done by triangulating from correspondences between a pair of stereo images (other possibilities include the use of cameras and structured light projectors in known configurations [1]. These correspondences have been obtained by matching several kinds of primitive elements. Small patches of texture have been matched by Moravec [10]. Edges have been matched by Baker [2]. Horn and Schunck [7], Nagel [11], and many others have taken the approach of optical flow, which attempts to assess differential motions at every point in the image to obtain correspondence. Algorithms for obtaining sensor motion from matches of discrete

features been given by Gennerv [6], Moravec [10], Tsai [14], and others; from dense information as resulting from optical flow by Bruss and Horn [3]. Complete systems for mobile vehicles, containing modules for ranging, world modelling, and motion determination, have been described by Gennerv [6], Moravec [10], and Crowley [4]. The recent thesis by Lawton [8] gives an extensive survey of work on the processing of dynamic image sequences and reinforces the conclusions arrived at here.

All of this effort has produced very few systems that work convincingly in real environments. The amount of image processing to be performed and the lack of special hardware made the systems quite slow; incorrect stereo matches and instabilities in the algorithms made systems that dealt with motion unreliable. In short, the problem is not easy.

For our approach, we have begun by building a simple testbed vehicle, driving it with a system with which we are familiar (that of Moravec), and gaining experience with the problems the system encounters in operation. Our experience has led to several improvements and has given us useful insight into the nature of the motion analysis problem. This paper discusses a number of issues that arose in our work with the obstacle detection and motion determination components of the system. The issues fall into three broad categories: the number of images used at each step of the vehicle's travel, the constraint applied to stereo correspondence, and the nature of the algorithm used to solve for motion.

Testbed Vehicle and Stanford Cart Algorithms

Our testbed vehicle is a motorized tricycle equipped with two video cameras and a 68000 microprocessor (figure 1) [12]. The onboard processor is used only for motor control and camera switching operations; vision and navigation functions are performed remotely on a Vax-11/780. Images are digitized one at a time into a Grinnell frame buffer interfaced to the Vax. A tether to the vehicle carries AC power to the motors, coaxial cable from the Grinnell to the cameras, and serial communication lines from the Vax to the 68000.

The vision system has its base in the algorithms developed by Moravec for the Stanford Cart [10]. These algorithms ran the Cart in a stop-go-stop loop. At the outset, the system digitized nine images and built a model of its local environment. The model was very simple, consisting only of three-dimensional points ("features") triangulated from stereo correspondences in the nine images. The features were initially obtained by picking small, high-variance patches in the center image and using a coarse-to-fine correlation technique to find the same patch in the remaining images. Once triangulation was complete, a path was planned to the destination and the vehicle was moved along the path, typically by a distance of about one meter.

From the second Cart position a new local model was constructed. This was done by finding the old features in the new set of images and performing another triangulation. Vehicle travel was then described by a transformation that mapped the new feature locations into the old. If x_{old} is an old feature location, x_{new} is its new location, and $T(x_{new})$ is the motion transformation applied to x_{new} , then the unknown motion was the transformation T yielding the least squared error in the weighted sum of terms

$$(x_{old} - T(x_{new}))^2.$$

Weights were assigned in inverse relation to a feature's distance from the vehicle. The locations of features near the Cart were known to higher precision than the locations of those that were distant; hence, close features received more weight than those that were far away.

Unfortunately, this was not be the end of the story. The stereo matching process could fail, allowing erroneous feature distances to creep into the least squares algorithm. This occurred primarily in three situations: when features drifted out of the field of view, when formerly "good" features became occluded, and when features were "bad" in the first place. Bad features were those picked on highlights and occluding contours of objects; they tended not to belong to any fixed position in the scene and to drift about arbitrarily in the image. Therefore, several other heuristics were used to improve the reliability of the motion solutions:

- Because all nine images were taken by translating the camera along one line, feature matches from all images should be collinear. This is referred to as the epipolar constraint. A measure of the extent to which this was true was included in the least squares weights.
- When matching a feature in all nine images, distances were computed by taking the matches pairwise. The $(9 \text{ choose } 2) = 36$ distances so resulting were then histogrammed and the best peak was taken as the true feature distance. This ameliorated the effect of a few bad matches among the nine images.
- The three-dimensional distance between two features should remain relatively unchanged as the vehicle moves. This was used in a three-dimensional pruning heuristic that deleted features that exhibited large position change relative to other features.
- A vehicle position prediction was available from the path planning module. This position was added as a heavily weighted point in the least squares equation and acted as a "reasonableness limit" on the motion solution. Without it, bad data could cause the solution to fall well outside the known margin of error.

Number of Images

The Cart used nine-way stereo at each instant in time. These images were digitized by translating one camera along a slider. There is nothing magic about the number nine: it happens that it was a convenient number at the time the Cart software was developed.

There are two advantages to using a large number of images. The first is for blunder detection. As illustrated by the histogram heuristic, the availability of more than two images in a known configuration provides redundant information that can be used to detect incorrect matches. The second advantage comes from averaging. Even without blunders, additional images reduce the uncertainty in feature positions.

The main disadvantage to using a large number of images is obviously the cost involved, both in time and storage. Image processing functions, correlation in particular, constituted a large portion of the time in Cart runs and still consume a large amount of time in our present system. We need at least two images to do stereo and three to do blunder detection, but after that the returns are diminishing even for averaging, since standard deviation shrinks only by the square root of the number of samples.

These arguments convinced us to move to two images in our present system. We have not done a quantitative assessment of the effect of this move. Qualitatively, the speedup was very substantial and, with the addition of the constraints to be discussed below, matching of "good" features appears to be as successful as it was before. However, "bad" features -- occlusions, highlights, and occluding edges -- are not dealt with so effectively.

A second difficulty in using just two images is the effect of baseline. A large distance between cameras increases the accuracy of ranging, but makes the matching problem both more expensive and potentially more error prone due to the larger change in the image. A more subtle problem is that wide-baseline cameras on a fixed mount make it hard for the vehicle to see in front of its nose. This is important if you wish to use the same cameras to see very close objects and at the same time accurately range more distant objects. Using a larger number of images can supply a long baseline while at the same time letting camera-to-camera spacing be relatively small.

We can suggest two attractive variations on the camera theme, neither of which we have tried. One is to capitalize on redundancy in the *time* axis instead of the stereo axis. After solving for motion, there are four images for which the geometry is known, and this could perhaps be used to advantage. This strategy has limitations since the motion solution itself is inaccurate.

The second possibility is to retreat and use three images. A configuration in which two cameras are relatively close together and the third is relatively distant is interesting. The close cameras can be used to obtain an inexpensive match, which in turn strongly constrains the area of match in the remaining camera. The advantages of short baselines, long baselines, and the ability to do blunder detection are all present.

To sum up this section, we have been successful in the use of two-way stereo in the sense that features that one feels "should" be matched correctly usually are. However, we suspect that enough error remains in the matching process to warrant at least the investigation of greater use of time-redundancy in our algorithms.

Constraints on Correspondence

With the correspondence algorithm, we wish to minimize the likelihood of accepting a false target as the correct match for a feature. This has two aspects. Where a correct match is possible, that is the one the algorithm should return; where occlusion or some other phenomenon makes this impossible, the algorithm should inform us that no match was found.

There are at least three strategies to improve the performance of our algorithm:

- constrain the search to a smaller area,
- pick better features to match,
- attempt to match larger semantic units, such as edges, bounded regions, or objects.

So far we have pursued only the first two courses.

Our system matches on both the stereo axis, to find the distance to features, and on the time axis, to "reacquire" features in a new set of images. Thus, there are three constraints that can be applied:

- The epipolar constraint, for use on the stereo axis. This forces the match to lie within a thin band above and below a particular scanline in the image. We have crudely estimated five to ten percent of the image height to be wide enough to encompass the calibration errors of our cameras.
- Near and far distance limits, also for use on the stereo axis. This restricts the match to some horizontal sub-interval of the image. For example, a far limit of infinity simply states that a feature from, say, column 200 in the right image must appear at column 200 or greater in the left image. The near limit governs how much greater than 200 the match can be. With 12.5 millimeter focal length cameras and a 25 centimeter baseline, using a near limit of 1.5 meters and a far limit of infinity reduces the region searched to about one quarter of the image width.
- The motion estimate from the path planner, with margins of error, for use on the time axis. This defines a box in the new set of images in which a feature should appear after the vehicle moves. Since we are currently restricting motion to the plane, this also introduces fairly tight constraint. A useful side effect is that this constraint informs the system when a feature has drifted out of the image.

These constraints define windows that restrict the range of the coarse-to-fine correlator. The best correlation from the window is taken as the candidate match; the no-match situation is assumed if the correlation coefficient is below a (fairly arbitrary) threshold.

It is difficult to quantify the power of these constraints, since "matchability" of features varies highly from scene to scene. However, our experience is that they make a dramatic improvement in the number of features correctly matched. They also have a useful side effect in the context of the coarse-to-fine correlator. This algorithm searches in a pyramid of reduced resolution images, beginning with the 8x8 level and working up to the full 512x512 image. At each resolution it extracts a 4x4 window around the feature in the source image and correlates it with an 8x8 window in the target image. The location of best match defines the center of the search window at the next level. At low levels of resolution, it is difficult to keep features near the edge of the image from being pushed into an incorrect match closer to the center. Intersecting the 8x8 window with the constraints virtually eliminates this problem.

Paradoxically, there are cases in which the constraints introduce problems. This happens when no correct match exists for a feature, but the best correlation within the window passes the match threshold. Without constraint, such features may match far enough out in limbo to be caught as a blunder; with constraint, they are a "near miss" that can be hard to detect.

The second strategy we mentioned above was to pick better points. Picking features is the function of the "interest operator" module. Thorpe has studied a collection of interest operators in this context [13]. He found that while some operators did outperform others, there was more variability between using and not using constraints than there was between interest operators. We have stuck with the Moravec operator, which is cheap and performs fairly well.

We have not tried to match with larger primitives, although we plan to do so in the future.

Motion Solving

The navigation system currently solves for only the three degrees of freedom in the plane. Even with this restriction, the system sometimes exhibits erratic behaviour. Recent test runs led the vehicle to dodge one obstacle in a course that covered about six meters. The travel of the vehicle varied from 20 to 50 centimeters per step. Typically, the system's position model at the end of the run was off by less than half a meter; however, step-to-step errors in the position model could also be this large. Occasionally, drastic errors put the position model off by close to a meter, a situation from which it usually did not recover. Wildly inaccurate obstacle distances sometimes occurred as a result of matching errors, but these were normally caught by the three-dimensional pruning heuristic.

There are several explanations for the erratic behaviour. The simplest is that there is an inherent instability between translation to the side and rotation about the vertical axis. Differential motion of the vehicle in these directions causes very similar relative movement of the features. This problem is most acute when features are not well distributed in space. Because the dominant features in our lab are the brightly colored bookshelves along the walls, this tends to happen in our runs. Our recourse so far has been to push the interest operator to pick an even distribution of points in the image. Because it is impossible to always guarantee good spatial distribution, practical solutions may require other sensors to distinguish these degrees of freedom.

The second cause of inaccuracy may be sensitivity of the least squares motion solution to poorly ranged features. We may need better data editing techniques, perhaps similar to those employed by Gennery [6]. He iteratively examined the residuals in the motion solution and discarded features until the residuals were satisfactory.

Another candidate is the large step size of the vehicle. As mentioned, we have been using steps of up to 50 centimeters between images. In our small workspace this can translate into large changes in the image, particularly in the presence of big foreground objects. One would expect smaller steps to lead to more stability. It is plausible that this would be subject to the same "near miss" problem that we experienced with the matching constraints. We have not experimented with smaller steps because of the slow runtime of the program.

Drastic errors in the position model occasionally happened when the system looked in the wrong place to reacquire features. This can be caused by an inaccurate motion prediction or an overly stringent tolerance in the motion matching constraint. Curing this by slackening the constraint is undesirable; what is needed is an algorithm that copes gracefully with the situation.

So far the algorithmic changes we have discussed that fall within the basic correspondence-followed-by-motion-solution framework of the original system. We have some misgivings about this paradigm. The reason for working in a static environment is so that feature motion can be described by a single rigid transformation. By matching features one at a time, this paradigm lets inconsistent matches creep in from the very start. Moreover, it optimizes differences between inferred three dimensional positions, which seems like an error-prone, secondary measure.

Intuitively, we would prefer a registration technique that rigidly maps all features onto the image at once and optimizes some measure of similarity based on the image. Lucas has described such an algorithm [9]. His objective function is the sum over all features of the squared difference in intensity between a feature and its hypothesized match. An initial motion estimate is used to project all features onto the image, then Gauss-Newton iteration is used to minimize the intensity differences over the motion parameters. The tolerable error in the initial estimate is increased by applying the algorithm first to a blurred version of the

image, then to successively less blurred versions as the parameters converge.

This algorithm addresses our complaints with the correspondence/match paradigm very nicely. Rigidity is incorporated from the outset. By iterating from a coarse motion estimate, it may also avoid the kind of threshold problem that led to the drastic errors when our matching constraints were too tight. Initial results from our implementation suggest good convergence behaviour when most features are well ranged, but we know nothing of its behaviour when some features are ranged poorly. Peter Dew has suggested that it may be advisable to augment the Newton iteration with a safe-guarding technique [5]. Whether or not this is actually necessary should be answered as our work proceeds.

Conclusions

We have presented a largely qualitative assessment of our experience with one particular visual navigation system. Reducing the number of images at each step and increasing the amount of constraint have helped to reduce the runtime of the program roughly ten-fold, to the point where it now uses about 30 CPU seconds per step on a Vax-11/780. The techniques discussed in the previous section for improving the motion solution also show promise, and we expect much improvement in the behaviour of the system when our implementation of these techniques is complete.

Two directions in which our work will continue are toward better algorithms for coping with uncertain data and toward a quantitative analysis of the limits to the accuracy of visual navigation.

Acknowledgements

We wish to thank the members of the Mobile Robot Lab, who have constructed and maintained the vehicle, offered valuable suggestions, and contributed to the good work environment. Hans Moravec, director of the lab, has participated in many fruitful discussions, offered good ideas and pointed out some bad ones, and provided that inestimable quantity, morale support. Thanks to Bruce Lucas for help with his algorithm.

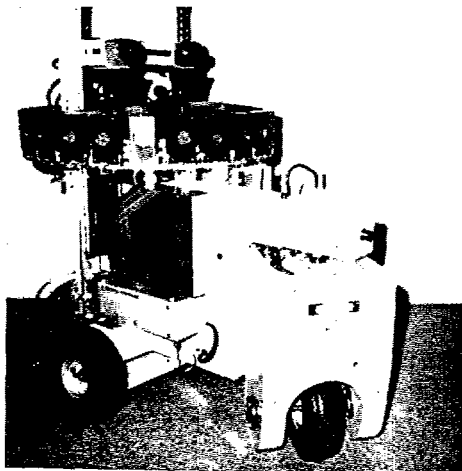


Figure 1: Testbed Vehicle "Neptune"

Shown with stereo cameras and sonar ring

References

1. Gerald J. Agin and Thomas O. Binford. "Computer Description of Curved Objects." *IEEE Transactions on Computers* C-25, 4 (April 1976).
2. H. Harlyn Baker. Edge Based Stereo Correlation. Proceeding of the ARPA Image Understanding Workshop, April, 1980.
3. Anna R. Bruss and Berthold K. P. Horn. "Passive Navigation." *Computer Vision, Graphics, and Image Processing* 21 (April 1983).
4. James L. Crowley. World Modelling and Position Estimation for an Intelligent Mobile Robot. Seventh International Conference on Pattern Recognition, August, 1984.
5. Peter Dew. private communication.
6. Donald B. Gennery. *Modelling the Environment of an Exploring Vehicle by Means of Stereo Vision*. Ph.D. Th., Stanford University, June 1980.
7. Brian G. Schunck and Berthold K. P. Horn. "Determining Optical Flow." *Artificial Intelligence* 17 (April 1981).
8. Daryl T. Lawton. *Processing Dynamic Image Sequences from a Moving Sensor*. Ph.D. Th., University of Massachusetts, February 1984.
9. Bruce D. Lucas. Generalized Image Matching by the Method of Differences: Algorithms and Applications. (thesis in preparation)
10. Hans P. Moravec. Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover. Tech. Rept. CMU-RI-TR-3, Carnegie-Mellon University, September, 1980.
11. Hans-Hellmut Nagel. Constraints for the Estimation of Displacement Vector Fields from Image Sequences. Proceedings of the Eighth International Joint Conference on Artificial Intelligence, August, 1983.
12. Gregg W. Podnar. A Functional Vehicle for Autonomous Mobile Robot Research. (in preparation)
13. Charles E. Thorpe. Analysis of Interest Operators for FIDO. Proceedings of the Workshop on Computer Vision: Representation and Control, IEEE, April, 1984.
14. Roger Y. Tsai. 3-D Inference from the Motion Parallax of a Conic Arc and a Point in Two Perspective Views. Proceedings of the Eighth International Joint Conference on Artificial Intelligence, August, 1983.