Toward Domain-Independent Navigation: Dynamic Vision and Control*

G.D. Hager D.J. Kriegman A.S. Georghiades O. Ben-Shahar

Center for Computational Vision and Control
Yale University
New Haven, CT 06520-8285

Abstract

This paper outlines a set of problems associated with constructing a robust, domain-independent vision-based navigation system suitable for both structured and unstructured environments. The system utilizes visual tracking to monitor a set of automatically selected image features (*markers*), and employs vision-based control to guide the motion of the robot from the image trajectory of a set of markers. An environment is represented as a graph which may be constructed either under human control (e.g. by giving the system a tour) or autonomously as the system explores. In this paper, we review the system architecture and present two image-based mobile robot controllers for following visually-defined trajectories.

1 Introduction

Since early work in the 1970's, such as SRI's Shakey[23] and Moravec's Cart[22], there have been great strides in the development of vision-based navigation methods for mobile robots operating both indoors and outdoors. Much of the efficiency and robustness of the recent systems can be attributed to the use of special purpose architectures and algorithms that are tailored to exploit domain specific image cues. For example, road followers rely on finding the road boundary and lane markers [3, 12] or landmarks [6, 18, 19, 20] whereas mobile robots navigating in hallways have exploited uniform texture of the floor [13], floor/wall features [17, 15], and overhead lights [7]. However, although these domain specializations lead to impressive performance, they do so by imposing particular sensor cues and representations on low-level navigation. As a result, a system that works in one domain may require substantial redesign before it can be used in another.

One path toward achieving domain-independence would be to utilize geometric reconstruction. In particular, dense surface descriptions produced by either range-finders or vision-based reconstruction techniques can be used to determine the free-space or traversable regions independent of the domain. However, these methods require a tremendous amount of computation, have limited resolution, and are difficult to maintain over large spatial extent due to integral odometric error.

Instead, our aim is to develop a vision-based navigation system capable of performing tasks in environments ranging from the usual "corridor and room" building to large open areas such as auditoriums, warehouses, parking lots, or open terrain. Although vision provides a huge amount of data, we quickly focus attention on small portions of the image which are easily distinguished from their local (in the image) surroundings, and track these patches through image sequences. Visual tracking of this type has proven to be simple to perform [2, 10], yet it is robust and it reduces image information to a time history of a small set of feature locations. Consequently, the set of nominal robot paths in our system is represented in terms of the image trajectories of tracked features. During subsequent navigation, the image motion of observed features in comparison to stored feature trajectories provides direct feedback for robot motion control. As features leave the robot's field of view, new features in the map are acquired by predicting their expected image location; once acquired, these features are tracked and used to control robot motion. In this way, the robot operates in a controlled, closed-loop manner during all operations.

The remainder of this paper expands on these ideas and presents preliminary results on two motion control algorithms for implementing tour following.

2 Navigation Using Tracked Markers

Throughout this article, we assume a non-holonomic mobile system with kinematics equivalent to a unicycle. The system travels in the $x-z$ plane and rotates about the y (gravity) axis. It is equipped with a unit focal length camera whose optical center defines the origin of the robot coordinate system. The state of the system

*This research was supported by the National Science Foundation, the Army Research Office, and by DARPA.

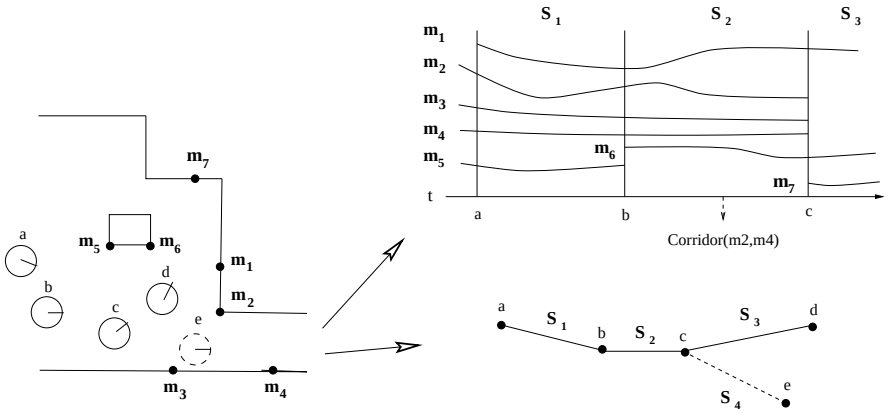

Figure 1: A navigation example showing the environment (left), the sequence-based representation (upper right), and the graph (lower right). The robot starts at point a observing the markers $\mathbf{m}_1$ through $\mathbf{m}_5$. As it moves to position b, $\mathbf{m}_5$ falls out of view, but the robot is able to acquire $\mathbf{m}_6$. As it continues its motion from b to c, it notes that there is a corridor opening between $\mathbf{m}_2$ and $\mathbf{m}_4$; this information is added as an annotation to the sequence. As it reaches c it is forced to drop $\mathbf{m}_3$ and $\mathbf{m}_4$, but acquires $\mathbf{m}_7$. At this point, it could have chosen instead to move down the corridor to the right. A later traversal moving to position e would add a branch point to the graph as shown by the dotted line.

in the plane is therefore given by $\mathbf{r} = (x, z, \theta)^t \in \mathbb{R}^3$ where (x, z) denotes the position of the robot in the plane and θ denotes its orientation. The kinematics of the system are

$$(\dot{x}, \dot{z}, \dot{\theta}) = (-s\sin(\theta), s\cos(\theta), \omega) \tag{1}$$

where s and ω are the linear and angular velocity of the robot body. Note that the camera is fixed and points in the "forward" (z axis) direction. Also, for the sake of simplicity we assume that the camera has an unlimited field of view. In practice, providing the system with an independent pan axis for the camera would achieve nearly the same results.

We assume that the robot has already acquired data about a set of nominal paths through the environment. Here we briefly summarize the important aspects of that representation and refer the reader to [26, 8, 10] for more details on the techniques used to acquire it.

The set of nominal paths is represented as a directed graph (the *map*) based on the recorded visual trajectories of tracked features which we call *markers*. We represent the trajectory of marker i as a function $\mathbf{m}_i(t), b_i \leq t \leq e_i$, where $t = b_i$ is the time of marker acquisition, and $t = e_i$ is the time at which the marker is lost. In general, the range of the functions $\mathbf{m}_i(t)$ depends on the marker type. For the purposes of this article we assume each marker is a point feature characterized by an image location and thus $\mathbf{m}_i : \mathbb{R}^+ \to \mathbb{R}^2$.

Arcs of the graph correspond to the trajectories of collections of markers called *sequences,* and the nodes of the graph correspond to the initiation or termination of a sequence. More formally, a sequence, $\mathbf{S}_j$, is defined by a set of markers $\mathbf{S}_j = \{\mathbf{m}_{j_1}, \ldots, \mathbf{m}_{j_n}\}$ which are simultaneously visible over some non-empty interval $\mathrm{dom}(\mathbf{S}_j) = [\max(b_{j_1}, \ldots, b_{j_n}), \min(e_{j_1}, \ldots, e_{j_n})]$. We write $\mathbf{S}_j(t) = \langle \mathbf{m}_{j_1}(t), \ldots, \mathbf{m}_{j_n}(t) \rangle$, $t \in \mathrm{dom}(\mathbf{S}_j)$ to denote the feature trajectory of the sequence. We assume that every marker belongs to a sequence at every time point, and that sequences are maximal — that is, if $\mathbf{S}_j$ is a sequence, there is no $\mathbf{S}_k, k \neq j$ such that $\mathbf{S}_j \subset \mathbf{S}_k$. Note that with this particular set of definitions, it follows that one sequence ends and another begins if and only if a marker is acquired or lost. Also note that the arcs are directed since markers that are visible while going in one direction, may not be visible in the other direction.

Finally, we assume that the time history of the control inputs to the system during sequence generation is stored with the sequence.

3 Visual Tracking and Motion Control

3.1 Epipolar Geometry for Mobile Systems

Consider first the perspective projection of an arbitrary point feature i. If the feature's homogeneous coordinates in two distinct images are denoted by $\mathbf{m}_i^1 = (u_i^1, v_i^1, 1)$ and $\mathbf{m}_i^2 = (u_i^2, v_i^2, 1)$, it is well-known that the two measurements must satisfy the following bilinear form known as the epipolar constraint

$$(\mathbf{m}_i^1)^t F_{12} \mathbf{m}_i^2 = 0 \tag{2}$$

where F_{12} is a 3×3 matrix of rank 2 [5]. When the camera's internal parameters are known, F_{12} can be expressed as $F_{12} = skew(\mathbf{t}_{12})R_{12}$ where $R_{12} \in SO(3)$ and $\mathbf{t}_{12} \in \mathbb{R}^3$ denote the rotation and translation between the camera locations at which the correspond-

ing images were acquired, and $skew(\mathbf{t}_{12})$ is the skew-symmetric matrix whose elements are given by $\mathbf{t}_{12}$ [21]. For the constrained motions of a mobile system operating in the $x-z$ plane it follows that F_{12} takes on the simplified form

$$F_{12} = \begin{bmatrix} 0 & -t_z & 0 \\ t_z C + t_x S & 0 & t_z S - t_x C \\ 0 & t_x & 0 \end{bmatrix} \quad (3)$$

$$= \begin{bmatrix} 0 & f_1 & 0 \\ f_2 & 0 & f_3 \\ 0 & f_4 & 0 \end{bmatrix}. \quad (4)$$

where $S = \sin\theta_{12}$, $C = \cos\theta_{12}$, θ_{12} is the relative angle of rotation about the vertical axis which parameterizes R_{12}, and $\mathbf{t}_{12} = (t_x, 0, t_z)^t$.

Given three or more corresponding points, F_{12} can be estimated as follows. First, the epipolar constraint can be expressed as $\mathbf{a}_i^t \mathbf{f} = 0$ where $\mathbf{f} = [f_1, f_2, f_3, f_4]^t$ and $\mathbf{a}_i^t = \begin{bmatrix} u_i^1 v_i^2 & v_i^1 u_i^2 & v_i^1 & v_i^2 \end{bmatrix}$. We then construct the positive semi-definite matrix

$$A = \sum_{i=1}^{n} \mathbf{a}_i \, \mathbf{a}_i^t. \quad (5)$$

The best estimate in a least squares sense for $\mathbf{f}$ is the eigenvector of the matrix A associated with its smallest eigenvalue.

We can relate this to the geometry of the system as follows. Given F_{12}, we see that $\mathbf{t}_{12} = \kappa[f_4, 0, -f_1]$ for some real value κ. Since κ is unknown, we can think of $\mathbf{t}_{12}$ as defining the line joining the two robot locations. Henceforth, we will denote the direction of this line by ψ_{12} which is ambiguous modulo π. We can solve for θ_{12} by first solving the linear system

$$\begin{bmatrix} -f_1 & f_4 \\ -f_4 & -f_1 \end{bmatrix} \begin{pmatrix} C \\ S \end{pmatrix} = \begin{pmatrix} f_2 \\ f_3 \end{pmatrix} \quad (6)$$

and then computing $\theta_{12} = \tan^{-1} S/C$. It is possible to show that estimation of θ_{12} is always well-defined (provided all points do not lie on the horizon line), however $\mathbf{t}_{12}$ is not well-defined when the centers of the two cameras are coincident.

3.1.1 Feed-Forward Marker Acquisition: To transition between sequences as the robot moves along a trajectory it is necessary to acquire landmarks as they come into view. This problem, which is closely related to the *image transfer problem* [1, 11, 27], can be solved as follows [8]. For simplicity, suppose that $\mathbf{S}^1$ and $\mathbf{S}^2$ are two "snapshots" (*e.g.* the first and last images) from a fixed sequence $\mathbf{S}_j$ which contains four or more points. Let $\mathbf{S}^3$ denote a snapshot from a sequence $\mathbf{S}_k$, $k \neq j$ which shares at least three points with $\mathbf{S}_j$. It follows that, from the three shared points, we can compute two fundamental matrices: F_{13} and F_{23} from $\mathbf{S}^1$, $\mathbf{S}^2$, and $\mathbf{S}^3$.

Let $\mathbf{m}_4^3$ denote the coordinates of a fourth point to be located in $\mathbf{S}^3$ given its known locations $\mathbf{m}_4^1$ in $\mathbf{S}^1$ and $\mathbf{m}_4^2$ in $\mathbf{S}^2$. This location is given by solving the linear system

$$\begin{bmatrix} (\mathbf{m}_4^1)^t F_{1,3} \\ (\mathbf{m}_4^2)^t F_{2,3} \end{bmatrix} \mathbf{m}_4^3 = 0 \quad (7)$$

for the first two components of $\mathbf{m}_4^3$. Figure 2 shows an example of this type of point prediction.

It is important to note that this system does not always have a unique solution: there are certain geometric conditions which lead to a degenerate linear system. More discussion on this point can be found in [8].

3.2 Motion Control

In order to formalize the motion control problem, we now suppose that the robot is traversing an arc in the map corresponding to a stored sequence while tracking a corresponding set of markers. Recall that we assume the following information is available from the previously taken "tour:" (1) a reference sequence $\mathbf{S}_r : \mathbb{R}^+ \to \mathbb{R}^n$ which is of fixed dimension $n/2 >= 3$; and (2) a record $(s_r(t), \omega_r(t),\ t \in [0, \infty)$ of the input control values to the system when $\mathbf{S}_r$ was "recorded." It is important to note that due to wheel slippage, it is not possible, via the kinematic equations, to simply integrate using the recorded sequence of s and ω to compute the position of the robot.

As the robot moves, it observes a sequence of "current" marker values $\mathbf{S}_c : \mathbb{R}^+ \to \mathbb{R}^n$ which correspond to those in $\mathbf{S}_r$ (that is, they arise from the projections of the same points in the world). Our goal is to choose a control strategy such that $\|\mathbf{S}_c(t) - \mathbf{S}_r(t)\|^2 \to 0$ as $t \to \infty$.

3.2.1 Geometry-Based Feedback: Consider a camera which is currently at position and orientation $\mathbf{r}^1$ with snapshot $\mathbf{S}^1 = \mathbf{S}_c(t)$ and a reference position and orientation $\mathbf{r}^2$ with snapshot $\mathbf{S}^2 = \mathbf{S}_r(t)$. From the results of the previous section, one possible strategy to move the robot to the desired location ($\mathbf{r}^2$) would be to compute F_{12} using $\mathbf{S}^1$ and $\mathbf{S}^2$, use this in turn to compute ψ_{12} which parameterizes the line joining $\mathbf{r}^1$ and $\mathbf{r}^2$, and to move along this line until $\mathbf{r}^2$ is reached. Unfortunately, due to the ambiguity modulo π in ψ_{12} we do not know the direction to move on this line. Furthermore, the calculation of ψ_{12} breaks down as the robot nears $\mathbf{r}^2$.

Both of these problems can be solved by using the error $E_{12} = \|\mathbf{S}_1(t) - \widehat{\mathbf{S}_2(t)}\|^2$, where $\widehat{\mathbf{S}_2(t)}$ is the reprojection of $\mathbf{S}_2(t)$ to an image plane rotated about the y axis by θ_{12}. We can disambiguate ψ_{12} by moving in the direction that causes E_{12} to decrease. Furthermore, a decreasing E_{12} implies that the robot moves closer to the desired position $\mathbf{r}^2$ which in turn means that calculation of ψ_{12} becomes less reliable. Therefore, we can

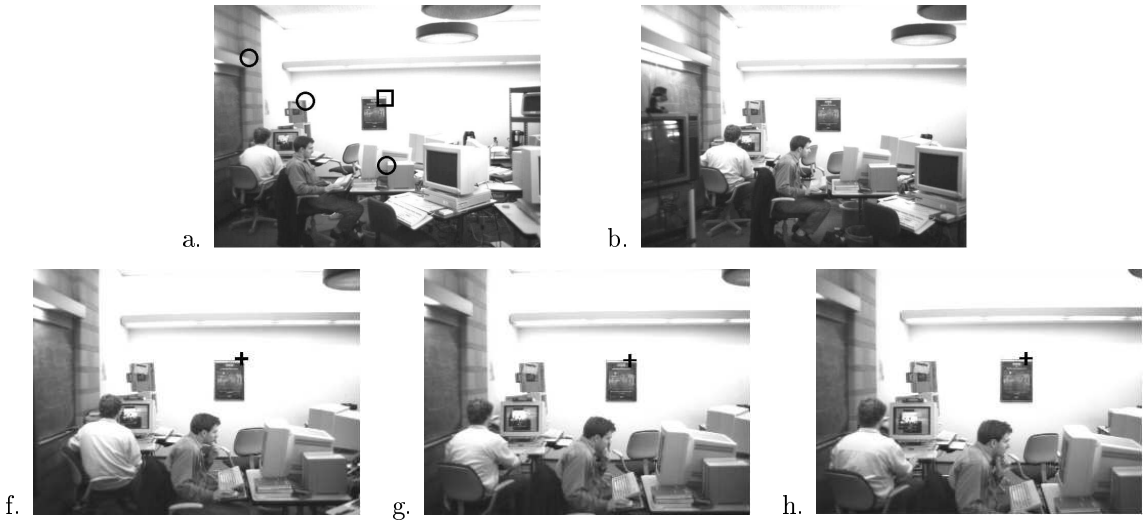

Figure 2: The top row shows two training images taken about one meter apart with the robot looking in the same direction. The remaining images were taken at equal intervals as the robot moved forward approximately ten feet from its position in Figure b. During training the image coordinates of all four features marked in Figure 2.a were known initially, then tracked to their positions in Figure 2.b. The crosses in the subsequent three images indicate the robot's predictions of the feature location.

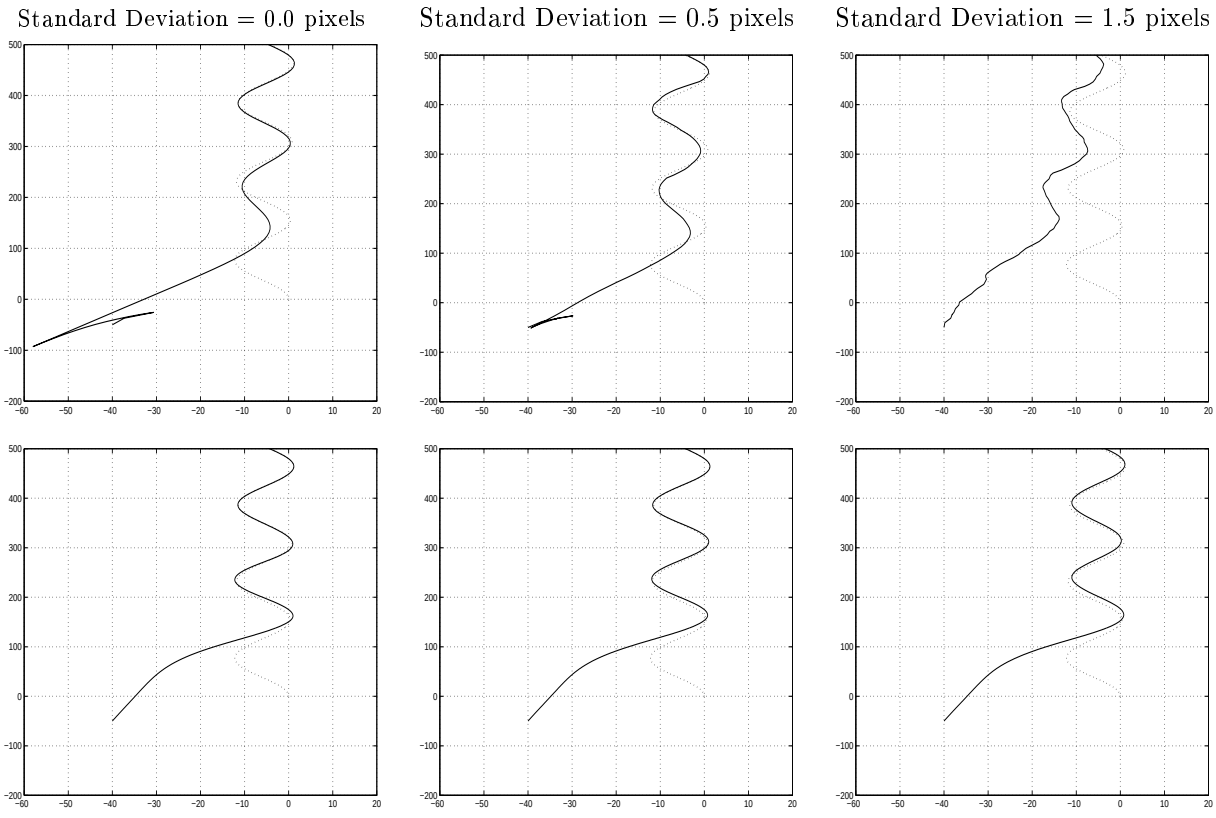

Figure 3: At the top, simulation tests of the geometry-based control algorithm under various noise conditions. At the bottom, simulation tests of the Jacobian-based control algorithms under the same conditions. The figures show the location of the controlled system (solid line) as well as the reference trajectory (dashed line).

use E_{12} to bias the effect of ψ_{12} on the control of the robot. We can then control the robot by a policy of the form,

$$\lambda = \frac{\kappa_1 E_{12}}{1+\kappa_1 E_{12}} \quad (8)$$

$$s(t) = s_r(t) + \kappa_2(E_{12}\cos(\psi_{12}) - \tau) \quad (9)$$

$$\omega(t) = \omega_r(t) + \lambda\psi_{12} \quad (10)$$

where κ_1, κ_2 and τ are design parameters. We can view λ as a continuous switch which determines how close to the reference trajectory ψ_{12} is active in altering the heading of the robot. The value of τ determines how closely the robot "follows" the setpoint on the reference trajectory and κ_2 is a gain.

Figure 3(top row) shows some simulated trajectories of a robot following the same reference trajectory under different noise conditions with $\kappa_1 = \kappa_2 = 1000$ and $\tau = 0.005$. The camera is modeled as having a 30 degree field of view divided into 600 pixels. Noise was zero mean Gaussian noise of the given standard deviation. As is clear from the the figures, the method works well under low noise conditions, but quickly breaks down as more noise is introduced. It also exhibits an odd "switching" behavior at the start of the trajectory. This is due to the fact that speed control (s) is designed for a setpoint "ahead" of the robot; if this is not the case, the robot backs up until the point is sufficiently far ahead to begin chasing it.

3.2.2 Jacobian-based feedback: A second approach to controlling the robot is to consider adaptations of classical "visual servoing" techniques to a non-holonomic system operating in the plane. Methods for holonomic problems of this form have been developed by many authors [14, 9, 4, 24]; several recent articles describing adaptations of these ideas to non-holonomic systems can be found in [16].

Suppose that an observed marker $\mathbf{m}_i$ has image coordinates $\mathbf{m}_i = (u,v)^t \in \mathbb{R}^2$ and external coordinates $\mathbf{P}_i = (X,Y,Z)^t \in \mathbb{R}^3$ expressed in the camera frame of reference. The point $\mathbf{P}_i$ and its projection are related by

$$\mathbf{m}_i = \begin{pmatrix} u \\ v \end{pmatrix} = \frac{1}{Z}\begin{pmatrix} X \\ Y \end{pmatrix}. \quad (11)$$

It follows that the velocity of the projection, $\dot{\mathbf{m}}_i$, due to robot motion $\mathbf{v} = \dot{\mathbf{r}}$ is

$$\dot{\mathbf{m}}_i = \begin{bmatrix} -\frac{1}{Z} & \frac{u}{Z} & -(1+u^2) \\ 0 & \frac{v}{Z} & -uv \end{bmatrix} \dot{\mathbf{r}} = \mathbf{J}_i(u,v,z)\mathbf{v}. \quad (12)$$

This is a planar version of the so-called Image Jacobian or Interaction Matrix expressed as a function of observed values u and v and the unknown value Z. More generally, if $\mathbf{S}_r$ is comprised of markers with image coordinates $\{\mathbf{m}_i\}$, the evolution of $\mathbf{S}_r$ as a function of the motion of the system can be written by "stacking" the Image Jacobians for the individual markers leading to the general form

$$\dot{\mathbf{S}}_r = \mathbf{J}\mathbf{v} \quad (13)$$

where $\mathbf{J} \in \mathbb{R}^{2n\times 3}$ depends now on the image coordinates and depth of every observed point. Since the motion of the system is already stabilized by encoder feedback, it is usually possible to model system dynamics as a pure time delay and to choose a control input $\mathbf{u} = (\dot{x}, \dot{z}, \dot{\theta})^t$[14]. Under these conditions, given a fixed setpoint $\mathbf{S}^* = \mathbf{S}_r(s)$, feedback systems of the general form

$$\mathbf{u}(t) = -k(\mathbf{J}^t\mathbf{J})^{-1}\mathbf{J}^t\,(\mathbf{S}_c(t) - \mathbf{S}^*), \quad (14)$$

will, in the absence of noise, uncertainty about $\mathbf{J}$ and the given dynamics, be locally asymptotically stable for an appropriate choice of the "gain" k.

There are three issues which arise when implementing a controller of this form. First, since $\mathbf{J}$ is a function of the distance from the robot to the observed feature, we must develop an estimation procedure for this quantity which preserves stability. In our case, since we have the complete tour at our disposal, it is not difficult to compute registered values for Z for every observation for the pre-learned sequence. These values can then be modified online using any of a number of estimation methods [24].

A second issue is to map this control vector to the non-holonomic kinematics. There are several possibilities in this case [25]. We have chosen the following mapping

$$K = \frac{1}{1+|\dot{x}|/\kappa} \quad (15)$$

$$s(t) = s_r(t) + \dot{z} \quad (16)$$

$$\omega(t) = K(\omega_r(t) + \dot{\theta}(t)) - \eta K\dot{x} \quad (17)$$

where η and κ are design parameters chosen to "tune" the system. Note that K, which ranges from 0 when $\dot{x}$ is large to 1 when $\dot{x}$ is small, acts as a continuous switch active about the value $|\dot{x}| = \kappa$. The expression $\dot{x}K$ acts as a "limiter" so that the value of $\dot{x}$ does not become too large and destabilize the system. Thus, intuitively this policy simply heads toward the current reference setpoint when it is far off the path, but then begins to mimic the reference once it is within approximately κ_1 units of the underlying path.

A final issue which arises is the fact that the results of (14) deteriorate rapidly when the orientation difference between the controlled system and the reference are large. However, recall that we can easily compute θ_{cr} using epipolar methods and we can use this value to rotate observed data into the frame of the reference trajectory as described in the previous section. In practice, we do this, apply (14) to the modified values, and adjust the control policy for ω to be

$$\omega(t) = K(\omega_r(t) + \theta_{cr}(t)) - \eta K\dot{x} \quad (18)$$

We then "tune" the controller for a nominal capture region. Figure 3(bottom row) shows several simulations of this controller with varying noise levels. We have chosen $\eta = .012$ and $\kappa = 10$. In general this method appears have much higher accuracy for comparable noise levels than the geometry-based method, although this fact is not surprising given that it makes use of an explicitly calculated value for the depth of each point.

4 Conclusion

We have described a system for domain independent mobile robot navigation in which naturally occurring features of the environment are used as markers or landmarks. During a tour, features are automatically selected, and a representation useful for subsequent navigation is automatically constructed. When navigating, features are acquired as they come into the robot's field of view, tracked over time, and used to control the robot's motion. Two feedback controllers for this purpose have been described. Simulation results suggest that a modification of a traditional image-based control system will work near the desired trajectory. Far from the initial trajectory, a novel approach based on projective geometry can be used to move the system into the capture region of the Jacobian-based control. We are currently working to test these ideas on a real mobile system. We are also working to develop a theoretical basis for the stability of the described system.

References

[1] E. Barett, M. Brill, N. Haag, and P. Payton. Invariant linear methods in photogrammetry and model matching. In J. Mundy and A. Zisserman, editors, *Geometric Invariance in Computer Vision*, pages 277–292. MIT Press, 1992.

[2] A. Blake and M. Isard. *Active Contours*. Springer-Verlag, 1998.

[3] E. D. Dickmanns and V. Graefe. Dynamic monocular machine vision. *Machine Vision and Applications*, v:223–240, 1988.

[4] B. Espiau, F. Chaumette, and P. Rives. A New Approach to Visual Servoing in Robotics. *IEEE Trans. on Robotics and Automation*, 8:313–326, 1992.

[5] O. Faugeras. *Three-Dimensional Computer Vision*. MIT Press, Cambridge, MA, 1993.

[6] C. Fennema, A. Hanson, E. Riseman, J. Beveridge, and R. Kumar. Model-directed mobile robot navigation. *IEEE Trans. on Robotics and Automation*, 20(6):1352–69, 1990.

[7] T. Fukuda, S. Ito, F. Arai, and Y. Yokoyama. Navigation system based on ceiling landmark recogntion for autonomous mobile robot. In *IEEE Int. Workshop on Intelligent Robots and Systems*, pages 150–155, 1995.

[8] G. Hager, D. Kriegman, E. Yeh, and C. Rasmussen. Image-based prediction of landmark features for mobile robot navigation. In *IEEE Conf. on Robotics and Automation*, pages 1040–1046, 1997.

[9] G. D. Hager. A modular system for robust hand-eye coordination. *IEEE Trans. Robot. Automat*, 13(4):582–595, 1997.

[10] G. D. Hager and K. Toyama. The "XVision" system: A general purpose substrate for real-time vision applications. *Comp. Vision, Image Understanding.*, 69(1):23–27, January 1998.

[11] R. Hartley. A linear method for reconstruction from lines and points. In *Int. Conf. on Computer Vision*, pages 882–887, 1995.

[12] M. Hebert, D. Pomerleau, A. Stentz, and C. Thorpe. Computer vision for navigation; the cmu ugv project. In *Proceedings of the Workshop on Vision for Robots*, pages 87–96. IEEE Computer Society Press, 1995.

[13] I. Horswill. Polly: A vision-based artificial agent. In *AAAI*, pages 824–829, 1992.

[14] S. Hutchinson, G. D. Hager, and P. Corke. A tutorial introduction to visual servo control. *IEEE Trans. Robot. Automat*, 12(5):651–670, 1996.

[15] D. Kim and R. Navatia. Symbolic navigation with a generic map. In *Proceedings of the Workshop on Vision for Robots*, pages 136–145. IEEE Computer Society Press, 1995.

[16] D. Kriegman, G. Hager, and A. Morse, editors. *The Confluence of Vision and Control*. Springer-Verlag, 1998.

[17] D. J. Kriegman, E. Triendl, and T. O. Binford. Stereo vision and navigation in buildings for mobile robots. *IEEE Trans. on Robotics and Automation*, 5(6):792–803, December 1989.

[18] K.-D. Kuhnert. Fusing dynamic vision and landmark navigation for autonomous driving. In *IEEE Int. Workshop on Intelligent Robots and Systems*, pages 113–119, July 1990.

[19] A. Lazanas and J.-C. Latombe. Landmark-based robot navigation. In *Proc. Am. Assoc. Art. Intell.*, 1992.

[20] T. S. Levitt, D. T. Lawton, D. M. Chelberg, and P. C. Nelson. Qualitative landmark-based path planning and following. In *Proceedings of AAAI-87*, pages 689–694, Los Altos, July 1987. Morgan Kaufman.

[21] H. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[22] H. P. Moravec. The stanford cart and the cmu rover. *Proceedings of the IEEE*, 71(7), July 1983.

[23] N. J. Nilsson. Shakey, the robot. Technical Note 323, SRI, April 1984.

[24] N. Papanikolopoulos, P. Khosla, and T. Kanade. Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision. *IEEE Trans. Robot. Automat*, 9(1):14–35, 1993.

[25] D. Popa and J. Wen. Nonholonomic path planning with obstacle avoidance: A path-space approach. In *IEEE Conf. on Robotics and Automation*, pages 2662–2667, 1996.

[26] C. Rasmussen and G. D. Hager. Robot navigation using image sequences. In *Proc. American Association for Artificial Intelligence*, pages 938–943, 1996.

[27] A. Shashua and M. Werman. Trilinearity of three perspective views and its associated tensor. In *ICCV95*, pages 920–925, 1995.