Vision for Mobile Robot Navigation: A Survey

Guilherme N. DeSouza and Avinash C. Kak

Abstract—This paper surveys the developments of the last 20 years in the area of vision for mobile robot navigation. Two major components of the paper deal with indoor navigation and outdoor navigation. For each component, we have further subdivided our treatment of the subject on the basis of structured and unstructured environments. For indoor robots in structured environments, we have dealt separately with the cases of geometrical and topological models of space. For unstructured environments, we have discussed the cases of navigation using optical flows, using methods from the appearance-based paradigm, and by recognition of specific objects in the environment.

Index Terms—Mobile robotics, navigation, computer vision, indoor navigation, outdoor navigation.

1 Introduction

Writing a survey paper on computer vision for mobile robot navigation—a subject that was in high prominence in the 1980s and the first half of the 1990s—is daunting, if not actually hazardous to one's career. A survey paper cannot merely be a catalog of all the articles published on a subject—the list would be much too long and nothing substantive would be said about each contribution. The alternative is to include in a survey only those contributions that the authors deem interesting—which is what we have done in this paper. We do realize that the latter approach is asking for trouble, especially in the light of the fact that the members of the research community hold widely differing opinions on what contributions were genuinely original and what merely derivative.

Another challenge in writing a survey on computer vision for mobile robot navigation is that even the perception of what constitutes progress varies widely in the research community. To us, for a mobile robot to engage in vision-based hallway navigation in the kinds of environments shown in Fig. 1 represents significant progress for the entire research community. But, others would pooh-pooh such an accomplishment on various grounds. To the extent that successful attempts at vision-based navigation in the cluttered environments of the kind shown in the figure use underlying models of space that are highly geometrical, the "nonbelievers" could question the "nonhuman" nature of these models. The nonbelievers would add—and do so rightly—that such representations would be difficult to extend to other scenarios where a robot may need to seek out certain objects in the environment on the basis of their semantic significance in the presence of competing clutter.

Despite these challenges, we will forge ahead in this article and highlight some of the more interesting (and, hopefully, important) experimental milestones.

• *The authors are with the Robot Vision Laboratory, School of Electrical and Computer Engineering, Purdue University, 1285 EE Building, West Lafayette, IN 47907-1285. E-mail: {gdesouza, kak}@ecn.purdue.edu.*

Manuscript received 23 July 1999; revised 27 June 2000; accepted 10 May 2001.
Recommended for acceptance by K. Bowyer.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 110291.

The progress made in the last two decades has been on two separate fronts: vision-based navigation for indoor robots and vision-based navigation for outdoor robots. We believe that the strides made in both these areas have been significant. For example, 20 years ago it would have been impossible for an indoor mobile robot to find its way in a hallway as cluttered as the one shown in Fig. 1, but now it is not much of a challenge. In the past, complex camera images such as the one shown in the figure used to be formidable to deal with because of an inordinate number of features that would be output by any feature detector. This difficulty was compounded by the subsequent need to determine whether or not any subset of these features matched robot expectations. But now, using a model-based framework such as that of FINALE [75], a robot need only examine those portions of the camera image that contain low-level features in the "vicinity" of model features, the extent of the "vicinity" being determined by the uncertainty in the position of the robot. A system such as FINALE requires a geometrical representation of space. One can now also design a vision-based navigation system that uses a topological representation of space and an ensemble of neural networks to guide a robot through interior space, the topological representation helping the robot figure out which neural networks to use in what part of the space. This can be done along the lines of NEURO-NAV [102], [103], and FUZZY-NAV [121]. In its latest incarnation, FINALE can navigate at an average speed of 17 m/min using an ordinary PC-based architecture (Pentium II 450Mhz, with no special signal processing hardware) and its self-localization routine, which helps the robot figure out where it is using vision, runs in less than 400 ms per image.

Equally impressive progress has been achieved in computer vision for outdoor robotics, as represented by the NAVLAB system [142], [143], [124], [146], [126], [128], the work on vision-guided road-following for "Autobahns" [36], [37], [35], [34], and the Prometheus system [46], [47], [48], [50], [49], [129]. One of the first systems developed for NAVLAB [143] could analyze intensity images, such as the one shown in Fig. 2, for ascertaining the position and the boundaries of a roadway. Starting in 1986, when a Chevy van was converted into NAVLAB 1, until today's converted metro buses known as Navlab 9 and 10, CMU and its partners have developed a family of systems for automated

Fig. 1. Indoor robots can now localize themselves using computer vision in scenes of the complexity depicted here. Reprinted with permission from [75].

Fig. 2. NAVLAB 1 can analyze intensity images such as this and determine the boundaries of a roadway. Reprinted with permission from [143].

navigation on highways. Four of these systems are: RALPH (Rapidly Adapting Lateral Position Handler) [128], ALVINN (Autonomous land vehicle in a neural network) [124], [125], [126], [127], AURORA (Automotive Run-Off-Road Avoidance) [21], and ALVINN-VC (for Virtual Camera) [61], [63], etc. A measure of the success of Navlab-based systems was the "No hands across America" test drive that consisted of a 2,849 mile trip from Pittsburgh, Pennsylvania, to San Diego, California. In this trip, a Navlab 5 vehicle was able to steer completely autonomously for more than 98 percent of the distance (a human driver handled the throttle and brake). Equally noteworthy has been the work carried out under the framework of the EUREKA-project Prometheus that has focused on exploring the potential of robot vision technology for improving traffic safety.

In the rest of this paper, our goal is to survey what we believe are the more important aspects of vision for mobile robot navigation. Since there exist large differences in how vision is used for indoor and outdoor robots, we have divided the rest of this paper accordingly. Each of these two categories is further subdivided on the basis of the mode in which vision is used.

2 Indoor Navigation

From the pioneering robotic vehicle work by Giralt et al. in 1979, [44], and later by Moravec in 1980 and 1983, [107], [108] and Nilsson in 1984, [114], it became clear that, implicitly or explicitly, it was imperative for a vision system meant for navigation to incorporate within it some knowledge of what the computer was supposed to see. Some of the first vision systems developed for mobile robot navigation relied heavily on the geometry of space and other metrical information for driving the vision processes and performing self-localization. In particular, interior space was represented by CAD models of varying complexity. In some of the reported work [20], the CAD models were replaced by simpler models, such as occupancy maps, topological maps or even sequences of images. When sequences of images were used to represent space, the images taken during navigation were submitted to some kind of appearance-based matching between the perception (actual image) and expectation (goal image or goal images stored in the database).

We believe all of these and subsequent efforts fall into three broad groups:

- *Map-Based Navigation*. These are systems that depend on user-created geometric models or topological maps of the environment.
- *Map-Building-Based Navigation*. These are systems that use sensors to construct their own geometric or topological models of the environment and then use these models for navigation.
- *Mapless Navigation*. These are systems that use no explicit representation at all about the space in which navigation is to take place, but rather resort to recognizing objects found in the environment or to tracking those objects by generating motions based on visual observations.

2.1 Map-Based Approaches

Map-Based Navigation consists of providing the robot with a model of the environment. These models may contain different degrees of detail, varying from a complete CAD model of the environment to a simple graph of interconnections or interrelationships between the elements in the environment. In some of the very first vision systems, the knowledge of the environment consisted of a grid representation in which each object in the environment was represented by a 2D projection of its volume onto the horizontal plane. Such a representation is usually referred to as "occupancy map" and it was formally introduced in [109]. Later, the idea of occupancy maps was improved by incorporating "Virtual Force Fields" (VFF) [14]. The VFF is an occupancy map where each occupied cell exerts a repulsive force to the robot and the goal exerts an attracting force. All forces are then combined using vector addition/subtraction and this resultant force on used to indicate the new heading for the robot. Occupancy-based representations are still used today in many research navigation systems.

A more elaborate version of the occupancy map idea is the S-Map [71], [72]—for "Squeezing 3D space into 2D Map." It requires a spatial analysis of the three-dimensional landmarks and the projection of their surfaces into a

2D map. Another elaboration consists of incorporating uncertainty in an occupancy map to account for errors in the measurement of the position coordinates associated with the objects in space and in the quantization of those coordinates into "occupied/vacant" cells. Some of the techniques that have been proposed for dealing with uncertainties in occupancy maps are: 1) Histogram grids or Vector Field Histogram [15], [16], where the detection of an object or the detection of a vacancy leads to the incrementing or decrementing, respectively, of the value of the corresponding cell in the grid. The new heading of the robot depends on the width of the "valleys" found in the histogram grid and the paths leading to the goal position. 2) Fuzzy operators [119] where each cell holds a fuzzy value representing the occupancy or the vacancy of the corresponding space. Also, sensor-fusion-based approaches [20] have been proposed in which readings from cameras and range finders are combined to produce occupancy-based models of the environment, etc.

Since the central idea in any map-based navigation is to provide to the robot, directly or indirectly, a sequence of landmarks expected to be found during navigation, the task of the vision system is then to search and identify the landmarks observed in an image. Once they are identified, the robot can use the provided map to estimate the robot's position (self-localization) by matching the observation (image) against the expectation (landmark description in the database). The computations involved in vision-based localization can be divided into the following four steps [13]:

- **Acquire sensory information.** For vision-based navigation, this means acquiring and digitizing camera images.
- **Detect landmarks**. Usually this means extracting edges, smoothing, filtering, and segmenting regions on the basis of differences in gray levels, color, depth, or motion.
- **Establish matches between observation and expectation.** In this step, the system tries to identify the observed landmarks by searching in the database for possible matches according to some measurement criteria.
- **Calculate position**. Once a match (or a set of matches) is obtained, the system needs to calculate its position as a function of the observed landmarks and their positions in the database.

As one would expect, the third step above—establishing matches between observation and expectation—is the most difficult. Often, this requires a search that can usually be constrained by prior knowledge about the landmarks and by any bounds that can be placed on the uncertainties in the position of the robot. Various approaches to vision-based localization of robots differ in how these constraints are taken into account.

In 1988, Sugihara [138] presented one of the pioneering studies in robot self-localization using single camera images. In that work, some of the problems of localization were first pointed out and since both the location of the robot and the identity of the features extracted by the vision system were assumed to be unknown, the problem was reduced to a geometric constraint satisfaction problem. Later, in 1989, Krotkov [80] extended this work by proposing a model where the effects of observation uncertainty were also analyzed. A few years later, in 1993, Atiya and Hager, [5], developed a real-time vision-based algorithm for self-localization under the same assumptions. In this paper, they mentioned both the lack of treatment of observation error in [138] and the disregard with false positives—detecting features that do not correspond to a known landmark—in [80]. Atiya and Hager proposed that the sensory error be represented by a tolerance, which led to a set-based algorithm for solving the matching problem and computing the absolute location of a mobile robot for indoor navigation. We will refer to such approaches by *absolute or global localization*.

Absolute localization is to be contrasted with *incremental localization* in which it is assumed that the location of the robot is known approximately at the beginning of a navigation session and that the goal of the vision system is to refine the location coordinates. In such cases, an expectation view can be generated on the basis of the approximately known initial location of the robot. When this expectation view is reconciled with the camera perception, the result is a more precise fix on the location of the robot. Examples of such systems are [100] by Matthies and Shafer, where stereo vision was used for error reduction; a system by Christensen et al. [24], where stereo vision was used in conjunction with a CAD model representation of the space; the PSEIKI system described in [1], [69] which used evidential reasoning for image interpretation; the system presented by Tsubouchi and Yuta [147] that used color images and CAD models; the FINALE system of Kosaka and Kak [75], and Kosaka et al. [76] that used a geometric model and prediction of uncertainties in the Hough space and its extended version [116], [117] which incorporated vision-based obstacle avoidance for stationary objects; the system of Kriegman et al. [79] that used stereo vision for both navigation and map-building; the NEURO-NAV system of Meng and Kak [102], [103] that used a topological representation of space and neural networks to extract features and detect landmarks; the FUZZY-NAV system of Pan et al. [121] that extended NEURO-NAV by incorporating fuzzy logic in a high-level rule-based controller for controlling navigation behavior of the robot; the system of [165], in which landmarks were exit signs, air intakes and loudspeakers on the ceiling and that used a template matching approach to recognize the landmarks; the system of Horn and Schmidt [53], [54] that describes the localization system of the mobile robot MACROBE—Mobile and Autonomous Computer-Controlled Robot Experiment—using a 3D-laser-range-camera, etc.

In yet another approach to localization in a map-based navigation framework presented by Hashima et al. [52], correlations are used to keep track of landmarks in the consecutive images that are recorded as the robot moves. We will refer to this type of localization as localization derived from *landmark tracking*.

The three different approaches to vision-based localization—Absolute (or Global) Localization, Incremental Localization, and Localization Derived from Landmark Tracking—will be visited in greater detail in the next three sections.

2.1.1 Absolute Localization

Since in absolute localization the robot's initial pose is unknown, the navigation system must construct a match between the observations and the expectations as derived from the entire database. On account of the uncertainties associated with the observations, it is possible for the same set of observations to match multiple expectations. The resulting ambiguities in localization may be resolved by methods such as: Markov localization [145], partially observable Markov processes [137], Monte Carlo localization [60], [32]), multiple-hypothesis Kalman filtering based on a mixture of Gaussians [27], using intervals for representing uncertainties [5], [80], and by deterministic triangulation [138], etc.

Here, we will explain further the absolute localization method advanced by Atiya and Hager [5]. The basic idea of their approach is to recognize in the camera image those entities that stay invariant with respect to the position and orientation of the robot as it travels in its environment (Fig. 4). For example, consider a triple of point landmarks on a wall in the environment. If all three of these points could be identified in each image of a stereo pair, then the length of each side of the triangle and the angles between the sides would stay invariant as the robot moves to different positions with respect to these three points. So, the length and the angle attributes associated with a triple of landmark points would be sufficient to identify the triple and to set up correspondences between the landmarks points in the environment and the pixels in the camera images. Once such correspondences are established, finding the absolute position of the robot simply becomes an exercise in triangulation. But, of course, in the real world, the coordinates of the landmark points may not be known exactly. Also, the pixel coordinates of the observed image points may be subject to error. Additionally, given a multiplicity of landmark points on a wall (or walls), there may be ambiguity in establishing correspondences between the landmark triples and the observed pixel triples.

Atiya and Hager have used a set-based approach for dealing with the above mentioned uncertainties and have cast the ambiguity issue in the form of the labeling problem in computer vision. In the set-based approach, the uncertainty is expressed in the form of intervals (sets). A landmark j observed in the image plane is represented by a pair of observation intervals—observation from the left and right cameras, $o_j = (o^l, o^r)$. The observation interval consists of the coordinates of that landmark with respect to the camera coordinate frame, plus or minus a certain tolerance ϵ, which represents the error in sensor measurement. Similarly, a landmark location interval p_j consists of the coordinates of the landmark j with respect to the world frame plus or minus a tolerance δ. If the robot position $\mathbf{p}$ is known, the landmark location interval p_j can be projected into the image plane using a known projection operator $Proj(\mathbf{p}, p_j)$, giving an "expected observation interval" h. Like o, h is a pair of coordinate intervals in the left and the right image planes. In the ideal case, where observation and landmark position are error free, o and h would coincide. This matching between the observation interval o_i and the landmark position interval is a tuple $\lambda =< o_i, p_j >$ and a labeling $\Lambda = \{< o_i, p_j >\}$ is a set of such matchings.

Therefore, in a set-based model of the uncertainty, each new observation results in new intervals. The observations are then combined by taking intersection of all such intervals; this intersection should get smaller as more observations are combined, giving a better estimate of the uncertainty. If the error is not reduced to within the robot's mechanical tolerance, additional sensing (new observations) must be added to reduce the localization error.

Atiya and Hager solved the overall localization problem—given a list of known landmarks and a list of visual observations of some of those landmarks, find $\mathbf{p} = (\mathbf{x}, \mathbf{y}, \mathbf{\Theta})$, the position and orientation of the robot—by decomposing it into the following two subproblems:

Problem 1. *Given n point location intervals $p_1, p_2, \ldots, p_n$ in a world coordinate frame that represent the landmarks and m pairs (stereo) of observation intervals $o_1, o_2, \ldots, o_m$ in the robot camera coordinate frame, determine a consistent labeling, $\Lambda = \{< o_i, p_j >\}$, such that*

$$Proj(\mathbf{p}, p_j) \bigcap o_i \neq \emptyset, \qquad \forall \ < o_i, p_j > \in \ \Lambda.$$

Problem 2. *Given the above labeling, find the set of robot positions, P, consistent with that labeling. That is, find the set*

$$P = \{\mathbf{p} \mid Proj(\mathbf{p}, p_j) \bigcap o_i \neq \emptyset\}, \qquad \forall \ < o_i, p_j > \in \ \Lambda.$$

In practice, the first problem, the labeling problem, consists of finding not one, but all possible labelings between p_js and o_is and choosing the best labeling according to some criterion. The authors suggest using that labeling which contains the largest number of matches. Another suggested criterion is to use all labelings with more than four matches. A labeling is called "consistent" when every o_i participates only once in the labeling.

The labeling problem is solved by comparing all possible triangles subject to heuristics discussed in [5], that can be extracted from the image with all possible triangles that can be formed by the landmark points. The two lists of three lengths and three angles for each triangle, one describing the observations and the other describing the landmark positions, can now be searched and cross compared to form the labelings $\{< o_i, p_j >\}$.

As we mentioned before, both the landmark position and the observation are expressed by intervals. When converting to the triangular representation, the uncertainties around p_j and o_i must also be transformed and expressed in terms of intervals for the lengths and angles of the triangles. Fig. 3 shows the differences between the maximum and the minimum lengths and the maximum and the minimum angles associated with a triple of points in the presence of uncertainties. We can see the largest and smallest side lengths in Fig. 3a and the largest and smallest angles in Fig. 3b that can be formed using the uncertainty regions around p_j or o_i.

Once a consistent labeling Λ is found, the second subproblem can be attacked. For that, the authors proposed

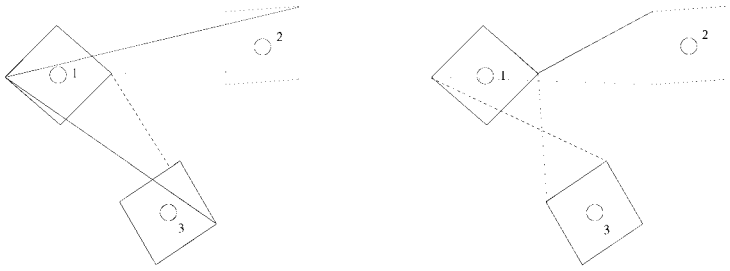


Fig. 3. (a) Largest (full line) and smallest (dotted line) sides. (b) Largest (full line) and smallest (dotted line) angles.

an approach based on the analysis of geometrical constraints involving landmarks in the form of triangles, the uncertainty regions around the landmarks, and the need to rotate and translate the observations in order to make the sides of the two triangles given by $< o_i, p_j >$ parallel. If the uncertainty regions o_1 and o_2 represent two observations matched against two landmark positions p_1 and p_2, the position of the robot $\mathbf{p} = (\mathbf{x}, \mathbf{y}, \mathbf{\Theta})$ would be given by:

$$\Theta_o = atan\left(\frac{t_y + \lambda s_y}{t_x + \lambda s_x}\right), \quad \Theta_d = atan\left(\frac{r_y}{r_x}\right)$$

$$\Theta = \Theta_o - \Theta_d,$$

where t represents segments connecting two uncertainty regions o_1 and o_2 $(t = o_1 - o_2)$, s represents segments inside the regions $(s = o_{2,k} - o_{2,j}$, where k and j are two corners of $o_2)$, and r is the segments connecting the landmark locations p_1 and p_2 $(r = p_1 - p_2)$. As happened when transforming from location and observation intervals to triangular representation, the uncertainty regions can yield many possible values for Θ. These values are combined to form an interval (set). The intersection of all computed intervals over the matching $< o_i, p_j >$ gives Θ^*. Having computed Θ, the x and the y coordinates of the robot's position are given by

$$x_i = p_{i,x} + sin(\Theta^*)t_i - cos(\Theta^*)s_i$$
$$y_i = p_{i,y} - sin(\Theta^*)s_i - cos(\Theta^*)t_i.$$

Again, the intersection of sets x_i and y_i computed for all regions yields x^* and y^*. As mentioned at the beginning of this section, assuming independent observations, when combining the observations through intersections, the size of the intervals Θ^*, x^*, and y^* must tend to zero in the limit. In practice, if the estimation is not accurate enough, new observations would be needed. Fig. 5 should convey a sense of how the computed positional uncertainty tends to zero as the number of observations is increased.

2.1.2 Incremental Localization

Using Geometrical Representation of Space. In a large number of practical situations, the initial position of the robot is known at least approximately. In such cases, the localization algorithm must simply keep track of the uncertainties in the robot's position as it executes motion commands and, when the uncertainties exceed a bound, use its sensors for a new fix on its position. By and large, probabilistic techniques have evolved as the preferred approach for the representation and the updating of the positional uncertainties as the robot moves. One of these approaches, which we will use to explain incremental localization, is the FINALE system [75].

The FINALE system [75] achieves incremental localization by using a geometrical representation of space and a statistical model of uncertainty in the location of the robot. The FINALE system consists of the following key elements:

- Representing the uncertainty in the position $\mathbf{p} = (x, y, \phi)$ of the robot (position and orientation on the plane) by a Gaussian distribution and, thus,

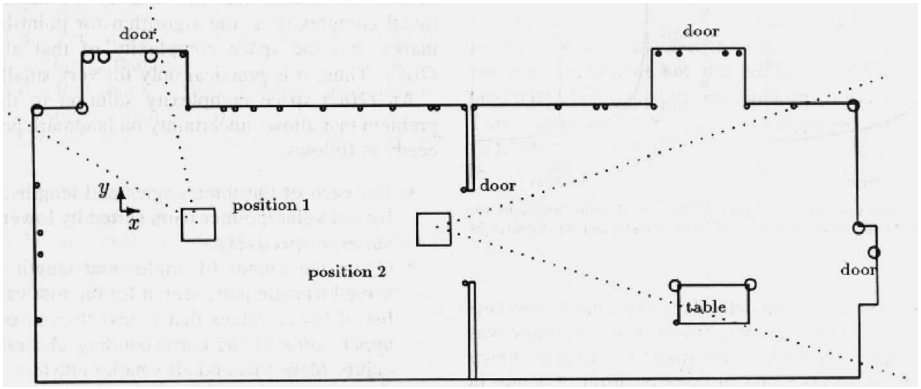


Fig. 4. A map of the environment with landmarks indicated by circles. Reprinted with permission from [5].

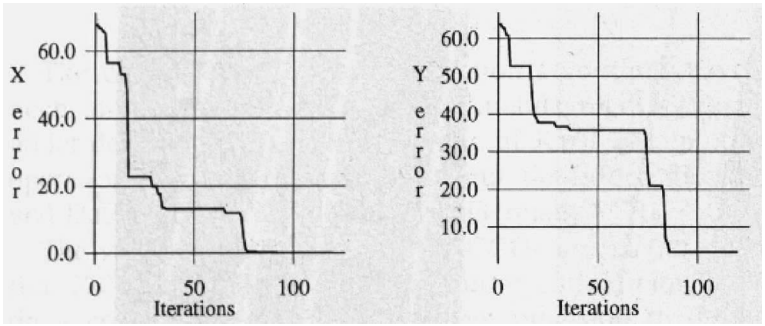


Fig. 5. The evolution of the width of the x and y parameters intervals. Reprinted with permission from [5].

characterizing the uncertainty at each location of the robot by the mean $\bar{\mathbf{p}}$ and the covariance $\Sigma_{\mathbf{p}}$.

- Characterization and parameterization of the motions executed in response to translational and rotational motion commands. In general, when a robot is asked to execute a pure translation, it may actually rotate a little on account of differential slippage in the wheels. Since these rotations occur continuously as the robot is translating, the rotations will be larger the longer the commanded translation. At the end of a commanded pure translation, the robot may end up at a location not on the axis along which it was asked to translate. By the same token, when the robot is commanded to execute a pure rotation, it may also scoot a little on account of the differential slippage in the wheels.
- A determination of how the uncertainty in the position of the robot changes when translational and rotational commands are executed. This involves figuring out the dependency of the changes in $\bar{\mathbf{p}}$ and $\Sigma_{\mathbf{p}}$ on the parameters of translational and rotational commanded motions.
- Design of a model-based Kalman filter consisting of the following components:

  1. Derivation of a constraint equation that, taking into account the camera calibration matrix, relates the positional parameters of a landmark in the environment to either the image space parameters or the Hough space parameters.
  2. Linearization of the constraint equation for the derivation of the Kalman filter equations.
  3. Projection of robot's positional uncertainty into the camera image (and into the Hough space) to find the set of candidate image features for each landmark in the environment.
  4. Using the Kalman filter to update the mean and the covariance matrix of the positional parameters of the robot as a landmark is matched with an image feature.

Fig. 6 depicts pictorially the various steps of the FINALE's self-localization algorithm.

Propagation of Positional Uncertainty through Commanded Motions. The process of modeling the uncertainty in FINALE starts by seeking a functional relationship between the position $\mathbf{p} = (x, y, \phi)$ of the robot before a commanded motion and its position $\mathbf{p}' = (x', y', \phi')$ after the commanded motion has been completed. It is known that if this relationship,

$$\mathbf{p}' = h(\mathbf{p}) \tag{1}$$

is linear, the uncertainty parameters $(\bar{\mathbf{p}}, \mathbf{\Sigma_p})$ and $(\bar{\mathbf{p}}', \mathbf{\Sigma}'_{\mathbf{p}})$ would be related by

$$\begin{aligned} \bar{\mathbf{p}}' &= h(\bar{\mathbf{p}}) \\ \Sigma'_p &= \left(\frac{\delta h}{\delta p}\right) \Sigma_p \left(\frac{\delta h}{\delta p}\right)^T, \end{aligned} \tag{2}$$

where $\frac{\delta h}{\delta p}$ is the Jacobian of the transformation function $\mathbf{h}$. These two equations tell us how to transform the mean and the covariance associated with the position of the robot from just before a commanded motion to just after it, provided we know the functional relationship given by $\mathbf{h}(\mathbf{p})$. To discover the form of the $\mathbf{h}(\mathbf{p})$ function for the case when the robot executes a translational command, the motion executed in response to such a command is parameterized by d, α, and β, as shown in Fig. 7. In that figure, the robot was commanded to execute a translational motion through a distance d_0 along its y_r axis. But, because of differential slippage in the wheels, the robot ends up at a location given by the distance parameter d and the angle parameters α and β. By using simple trigonometry, one

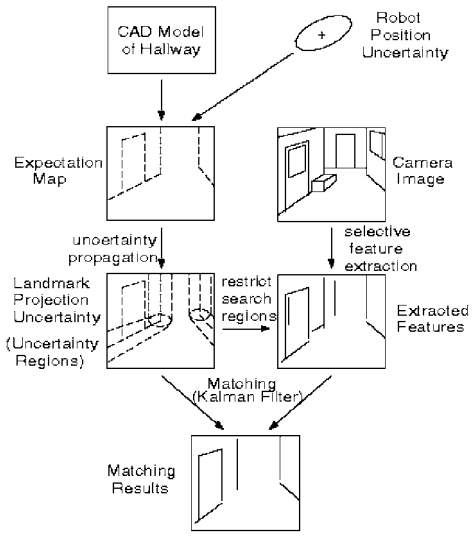


Fig. 6. Processing steps in FINALE's self-locomotion algorithm.

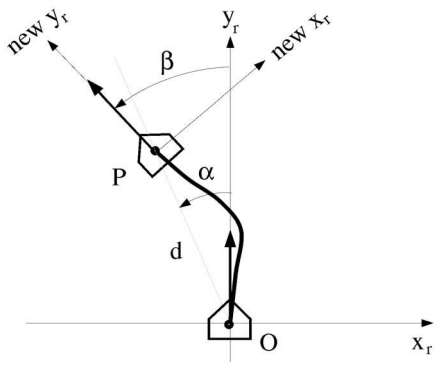


Fig. 7. Initial and final position after a translation command. Reprinted with permission from [75].

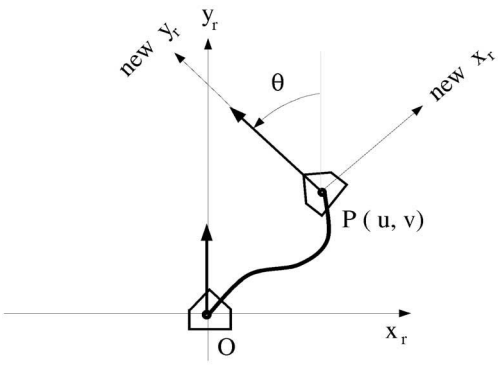


Fig. 8. Initial and final position after a rotation command. Reprinted with permission from [75].

arrives at the following functional relationship between $\mathbf{p}$ and $\mathbf{p}'$:

$$\begin{bmatrix} p'_x \\ p'_y \\ \phi' \end{bmatrix} = \begin{bmatrix} p_x \\ p_y \\ \phi \end{bmatrix} + \begin{bmatrix} -dsin(\phi+\alpha) \\ dcos(\phi+\alpha) \\ \beta \end{bmatrix}. \quad (3)$$

One may think of d, α, and β as random variables with mean values, $\bar{d}$, $\bar{\alpha}$, and $\bar{\beta}$ and covariance matrix

$$\Sigma_T(d,\alpha,\beta) = \begin{bmatrix} \sigma_d^2 & \rho_{d\alpha}\sigma_d\sigma_\alpha & \rho_{d\beta}\sigma_d\sigma_\beta \\ \rho_{d\alpha}\sigma_d\sigma_\alpha & \sigma_\alpha^2 & \rho_{\alpha\beta}\sigma_\alpha\sigma_\beta \\ \rho_{d\beta}\sigma_d\sigma_\beta & \rho_{\alpha\beta}\sigma_\alpha\sigma_\beta & \sigma_\beta^2 \end{bmatrix},$$

where $\rho_{d\alpha}$, $\rho_{d\beta}$, and $\rho_{\alpha\beta}$ are the correlation coefficients. The work reported in [75] shows experimentally obtained results on the d_0 dependence of the mean values $\bar{d}$, $\bar{\alpha}$, $\bar{\beta}$, the standard deviations σ_d, σ_α, σ_β, and the correlations $\rho_{d\alpha}$, $\rho_{d\beta}$, and $\rho_{\alpha\beta}$.

Similarly, when the robot is commanded to execute a pure rotation through angle θ_0 in its own coordinate frame, it ends up at a location given by the parameters θ, u, and v shown in Fig. 8. In this case, the relationship between $\mathbf{p}$ and $\mathbf{p}'$ is given by

$$\begin{bmatrix} p'_x \\ p'_y \\ \phi' \end{bmatrix} = \begin{bmatrix} p_x \\ p_y \\ \phi \end{bmatrix} + \begin{bmatrix} -ucos\phi - vsin\phi) \\ usin\phi + vcos\phi) \\ \theta \end{bmatrix}. \quad (4)$$

Also, as before, if the robot is commanded to make a rotational motion of θ_0, one may think of u, v, and θ as random variables with mean values, $\bar{u}$, $\bar{v}$, and $\bar{\theta}$ and covariance matrix

$$\Sigma_R(u,v,\theta) = \begin{bmatrix} \sigma_u^2 & \rho_{uv}\sigma_u\sigma_v & \rho_{u\theta}\sigma_u\sigma_\theta \\ \rho_{uv}\sigma_u\sigma_v & \sigma_v^2 & \rho_{v\theta}\sigma_v\sigma_\theta \\ \rho_{u\theta}\sigma_u\sigma_\theta & \rho_{v\theta}\sigma_v\sigma_\theta & \sigma_\theta^2 \end{bmatrix},$$

where ρ_{uv}, $\rho_{u\theta}$, and $\rho_{v\theta}$ are the correlation coefficients. The work reported in [75] shows experimentally obtained results on the θ_0 dependence of the mean values $\bar{\theta}$, $\bar{u}$, $\bar{v}$, the standard deviations σ_θ, σ_u, σ_v, and the correlations $\rho_u v$, $\rho_{u\theta}$, and $\rho_{v\theta}$.

The above equations show that the position of the robot at the end of a commanded motion depends nonlinearly on the previous position. The difficulty posed by the nonlinear nature of the transformation equations is overcome by using linear approximations to the equations and making sure that the command motions are small in size so as not to invalidate the approximations. Note that, as is clear from (2), in the linear regime of operation, we only need to know the Jacobian of the transformation from $\bar{\mathbf{p}}$ to $\bar{\mathbf{p}}'$. This Jacobian may be found by taking differentials of both sides of (3) for the case of translational commands and of both sides of (4) for the case of rotational commands to give us

$$\begin{bmatrix} \delta p'_x \\ \delta p'_y \\ \delta\phi' \end{bmatrix} = J_{T_1}\begin{bmatrix} \delta p_x \\ \delta p_y \\ \delta\phi \end{bmatrix} + J_{T_2}\begin{bmatrix} \delta d \\ \delta\alpha \\ \delta\beta \end{bmatrix}$$

$$\begin{bmatrix} \delta p'_x \\ \delta p'_y \\ \delta\phi' \end{bmatrix} = J_{R_1}\begin{bmatrix} \delta p_x \\ \delta p_y \\ \delta\phi \end{bmatrix} + J_{R_2}\begin{bmatrix} \delta u \\ \delta v \\ \delta\theta \end{bmatrix},$$

where J_{T_1}, J_{T_2}, J_{R_1}, and J_{R_2} are the Jacobians for the translational and rotational cases as indicated by the indices used. The detailed expressions for the Jacobians are provided in [75]. We can now write down the following equations for propagating the mean and the covariance of the position vector $\mathbf{p}$ through translational commands:

$$\begin{bmatrix} \bar{p'_x} \\ \bar{p'_y} \\ \bar{\phi'} \end{bmatrix} = \begin{bmatrix} \bar{p_x} \\ \bar{p_y} \\ \bar{\phi} \end{bmatrix} + \begin{bmatrix} -\bar{d}sin(\bar{\phi}+\bar{\alpha}) \\ \bar{d}cos(\bar{\phi}+\bar{\alpha}) \\ \bar{\beta} \end{bmatrix}$$

$$\Sigma'_p(p'_x,p'_y,\phi') = J_{T_1}\Sigma_p(p_x,p_y,\phi)J_{T_1}^T + J_{T_2}\Sigma_T(d,\alpha,\beta)J_{T_2}^T \quad (5)$$

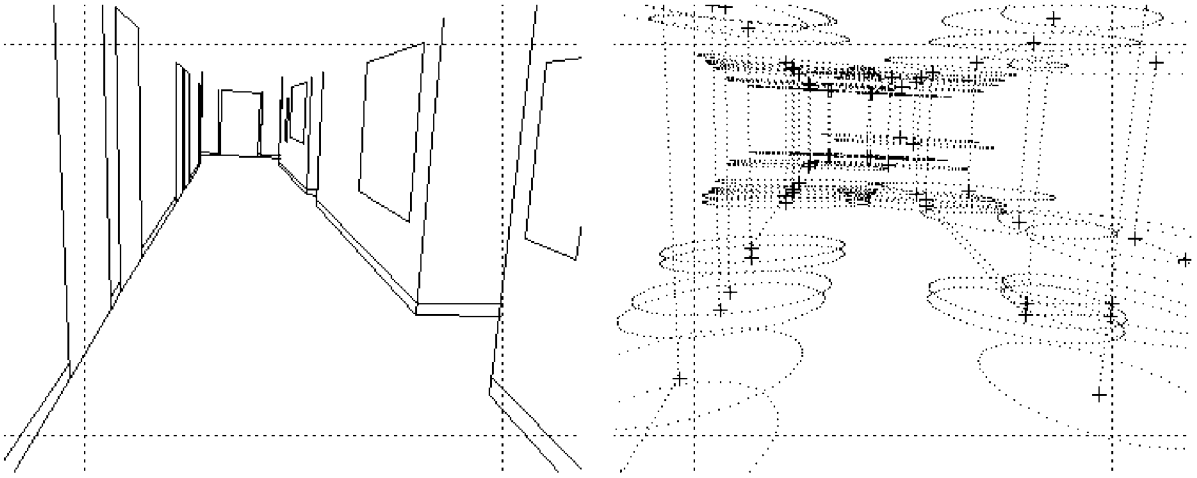
and rotational motion commands:

$$\begin{bmatrix} \bar{p'_x} \\ \bar{p'_y} \\ \bar{\phi'} \end{bmatrix} = \begin{bmatrix} \bar{p_x} \\ \bar{p_y} \\ \bar{\phi} \end{bmatrix} + \begin{bmatrix} -\bar{u}cos\bar{\phi} - \bar{v}sin\bar{\phi}) \\ \bar{u}sin\bar{\phi} + \bar{v}cos\bar{\phi}) \\ \bar{\theta} \end{bmatrix}$$

$$\Sigma'_p(p'_x,p'_y,\phi') = J_{R_1}\Sigma_p(p_x,p_y,\phi)J_{R_1}^T + J_{R_2}\Sigma_R(u,v,\theta)J_{R_2}^T. \quad (6)$$

Equations (5) and (6) tell us how to update the mean and the covariance of the robot's position after it has executed a translational or a rotational motion command.

Projecting Robot's Positional Uncertainty into Camera Image. In order to determine where in the camera image one should look for a given landmark in the environment, we need to be able to project the uncertainty in the position of the robot into the camera image. This can be done with the help of the camera calibration matrix. To explain,



(a) (b)

Fig. 9. Uncertainties around point landmarks. The Mahalanobis distance d = 1 was used to construct the uncertainty ellipses. Reprinted with permission from [75].

consider a single point at coordinates (x, y, z) in the environment of the robot. The image of this world point will be at the pixel coordinates (X, Y), as given by

$$\begin{bmatrix} XW \\ YW \\ W \end{bmatrix} = TH \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix},$$

where we have used homogeneous coordinate representations for world point and the pixel, W being the perspective factor. T is the camera calibration matrix and H the transformation matrix that takes a point from the world frame to the robot-centered frame.

The above vector-matrix equation can be reworked to express directly the relationship between the pixel coordinates (X, Y) and the position vector $\mathbf{p} = (\mathbf{p_x}, \mathbf{p_y}, \phi)$ for a given world point. What we end up with is again a nonlinear relationship. As before, we can choose to work with a linearized version of this equation, the Jacobian yielded by the linearized version being what we need to project robot position covariance into the image. To obtain this Jacobian, we differentiate both sides of the reworked equations to get

$$\delta I \equiv \begin{bmatrix} \delta X \\ \delta Y \end{bmatrix} = J(\bar{I}, \bar{p}) \begin{bmatrix} \delta p_x \\ \delta p_y \\ \delta \phi \end{bmatrix}, \tag{7}$$

where $J(\bar{I}, \bar{p})$ is the Jacobian of I (the pixel coordinates (X, Y)) with respect to $\mathbf{p}$ in the vicinity of the mean values. The mean of I and its covariance matrix—the covariance matrix associated with the pixel coordinates of a single point landmark in the scene—may now be written as

$$\begin{bmatrix} \bar{I}\bar{W} \\ \bar{W} \end{bmatrix} = T\bar{H} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\Sigma_I = J(\bar{I}, \bar{p}) \Sigma_p J(\bar{I}, \bar{p})^T. \tag{8}$$

Fig. 9 illustrates the utility of the covariance matrix Σ_I as a measure of the uncertainty associated with the location of a pixel for an isolated point in the environment. In the left panel of this figure, a wire-frame rendering of the hallway is shown. This is an expectation map made based on the current value of $\bar{\mathbf{p}}$, in the sense that this is what the robot would see if its actual position was at the center of the uncertainty region associated with its position. For each end point of a vertical edge in the expectation map, we have in the right panel an uncertainty ellipse which should contain the pixel corresponding to the end point with a probability of 86 percent. The ellipse corresponds to one unit of Mahalanobis distance as calculated from the covariance matrix Σ_I using (8) and the following statistical parameters of the robot's position:

$$\bar{\mathbf{p}} = \begin{bmatrix} 0.0\ m \\ 0.0\ m \\ 0.0^o \end{bmatrix}$$

$$\Sigma_p = \begin{bmatrix} (0.5\ m)^2 & 0 & 0 \\ 0 & (0.5\ m)^2 & 0 \\ 0 & 0 & (10^o)^2 \end{bmatrix}.$$

An isolated point in the environment is obviously not a perceptually interesting landmark. But, all of the above discussion on projecting robot's positional uncertainties into the camera image can be extended easily to midlevel features such as straight edges formed by the junctions of walls, borders of doors, windows, bulletin boards, etc. As shown in [75], this is best done in Hough space where each straight line feature can be represented by a single point. So, given a straight line feature in the environment, FINALE would know where in the Hough space it should seek the image of that straight line.

Kalman Filtering. So far, we have shown how the mean value and covariance of the robot's position can be updated after each translational or rotational command is executed by the robot. We have also shown how the uncertainty in the robot's position can be projected into the camera image so as to obtain a better localization of a landmark feature in

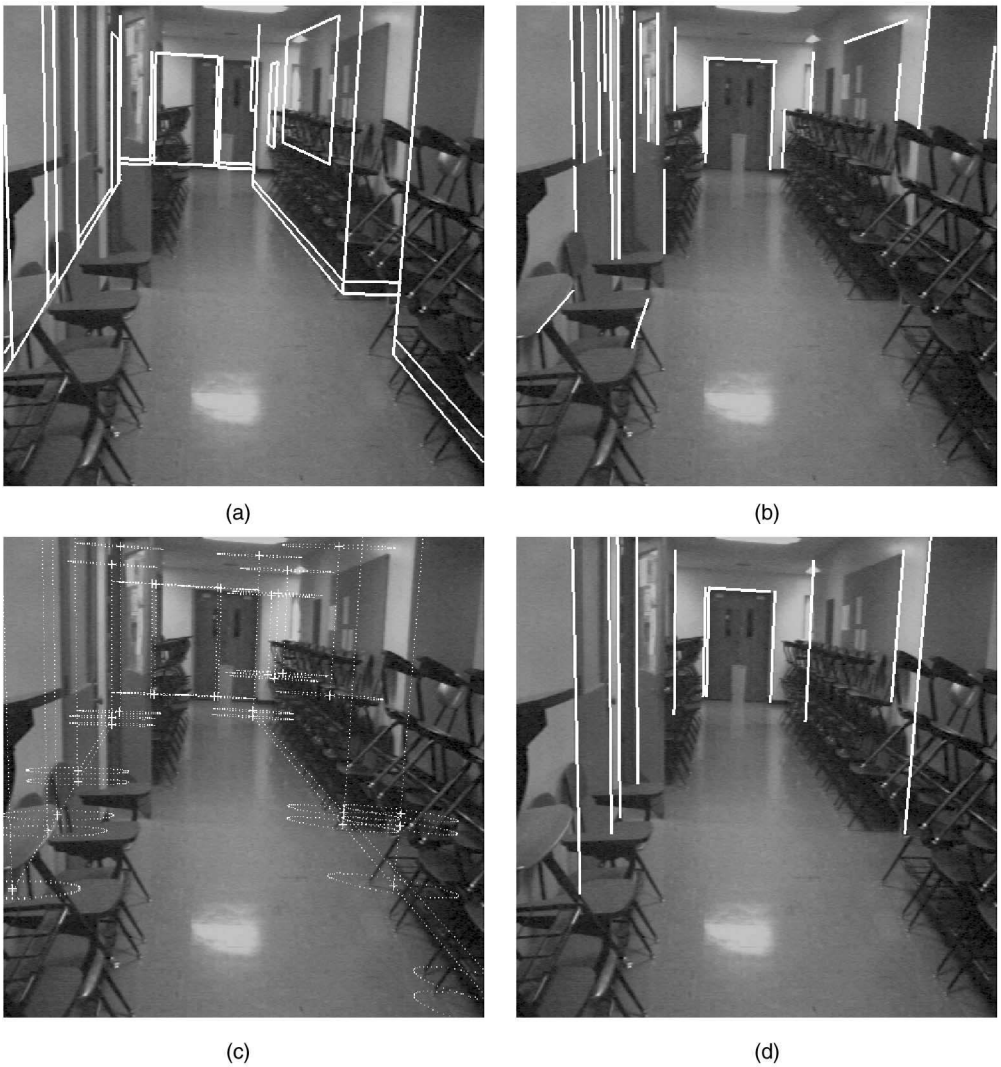


Fig. 10. Some of the intermediate results in FINALE's localization algorithm. (a) The camera image and the superimposed expectation map. (b) Output of the model-guided edge detector. (c) Uncertainty regions associated with the ends of the model line features. (d) Matching after Kalman filtering. Reprinted with permission from [75].

the camera image. The next step consists of FINALE updating the statistical parameters of the position of the robot after a landmark (in the form of a straight line feature) is matched with an image feature. This is done with the help of a model-based Kalman filter that is derived from a linearized version of a constraint equation that must be satisfied by the parameters of a straight line in the environment and the Hough space parameters of the corresponding line in the camera image.

Fig. 10 shows some of the more important intermediate results when a robot localizes itself using the Kalman-filter based approach. Shown in Fig. 10a is the camera image, superimposed on which is the expectation map constructed from a wire-frame model of the interior space. The white edges in Fig. 10b represent the output of the edge detector. The essential thing to realize here is that the edge detector ignores most of the gray level changes in the camera image; it extracts only those edges that in the Hough space are in the vicinity of the model edges. Fig. 10c shows the unit-Mahalanobis distance uncertainty ellipses for the ends of the model line features. Finally, Fig. 10d shows those edges produced by the edge detector that were matched with the model edges. It is this match that results in the robot able to localize itself.

Our discussion on FINALE has shown the critical role played by an empirically constructed model of the changes in the positional uncertainty of a robot as it executes translational and rotational motions. In general, also important will be sensor uncertainties and the trade-offs that exist between sensor uncertainties and decision making for navigation. For those issues, the reader is referred to the work of Miura and Shirai [104], [105], [106].

Using Topological Representation of Space. An entirely different approach to incremental localization utilizes a mostly topological representation of the environment. For example, in the NEURO-NAV system [102], [103], a graph topologically representing a layout of the hallways is used for driving the vision processes. An example of this graph representation is shown in Fig. 11 where the physical structure of a hallway is shown on the left and its attributed-graph representation on the right. For topological representations of space—in particular, large-scale space— the reader is also referred to the pioneering work of Kuipers and Byun [87].

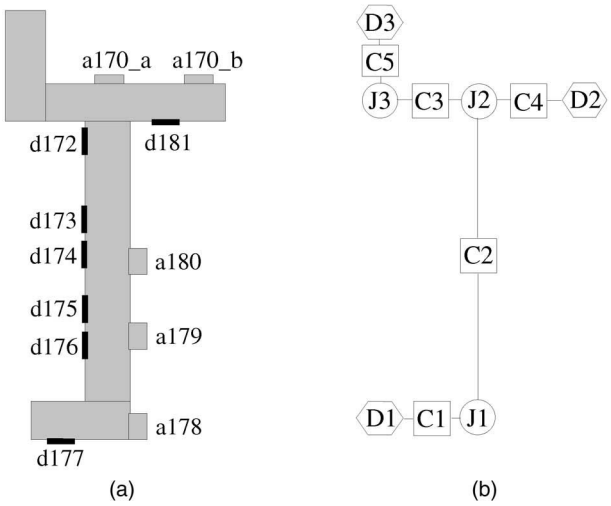


Fig. 11. (a) Physical structure of a hallway segment showing the doors and alcoves. (b) The topological representation of the hallway in (a). Reprinted with permission from [103].

In the example shown in Fig. 11, the graph data structure uses three different kinds of nodes—shown in the form of squares, circles, and diamonds—for representing corridors, junctions, and dead ends, respectively. Each node is attributed. For example, node C2, which represents the main central corridor in the hallway segment shown, has an attribute named *left landmarks* that is a list of pointers to the *door d176, door d175, door d174, door d173, door d172* features of corridor C2. The links of the graph, also attributed, contain information regarding the physical distance between the landmarks represented by the nodes at each end of the link.

The heart of NEURO-NAV consists of two modules, Hallway Follower and Landmark Detector, each implemented using an ensemble of neural networks. Those modules allow the robot to execute motions at the same time as it performs self-localization. Suppose, for example, that the robot is at the lower end of corridor C2 (Fig. 11). If

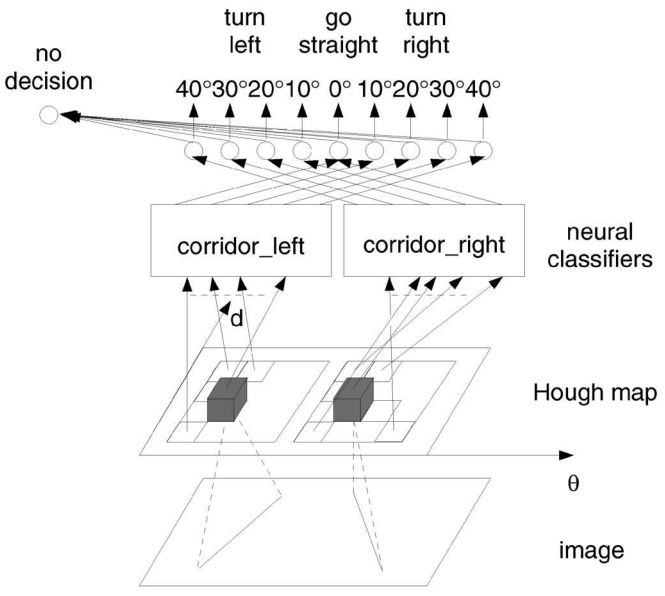


Fig. 12. Hierarchy of neural networks to perform corridor-following. Reprinted with permission from [103].

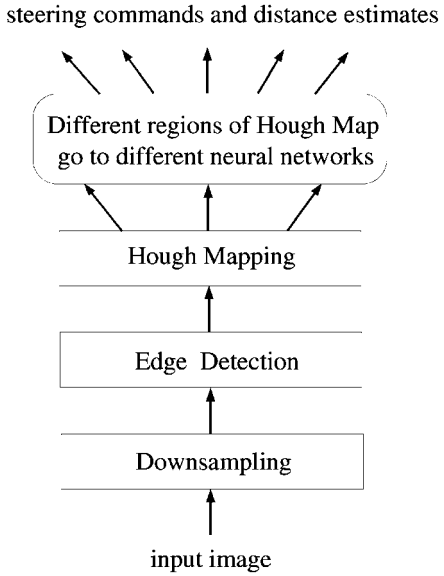


Fig. 13. Image processing steps in NEURO-NAV. Reprinted with permission from [103].

the motion command is for the robot to follow that corridor towards junction J2, turn left, and move down corridor C3, the Hallway Follower module will be invoked with the task of traversing corridor C2 keeping a parallel path with respect to the walls (corridor following). At the same time, Landmark Detector will perform a sequence of searches for landmarks contained in the attributed node of the graph corresponding to corridor C2, namely: door d176, door d175, etc., until it finds a junction (J2). At that point, Hallway Follower will be invoked again, this time to perform a left turn and, later on, to perform a new corridor following over corridor C3.

As was mentioned above, both the Hallway Follower and the Landmark Detector consist of ensembles of neural networks. Each network is trained to perform a specific subtask. For example, inside the Hallway Follower there is a neural network called *corridor_left*, which is capable of detecting, from the camera image, the edge of the floor between the left side wall and the floor. This network outputs the appropriate steering angles that keep the robot approximately parallel with respect to the left wall. The *corridor_right* network inside the Hallway Follower does the same with respect to the right wall. A third network inside the Hallway Follower combines the outputs of the *corridor_left* and the *corridor_right* networks and provides the steering commands needed by the robot (Fig. 12).

The neural networks in NEURO-NAV are driven by the cells of the Hough transform of the edges in the camera image. Different regions of the Hough space are fed into different networks. To speed up the computations (since each steering command must be generated in about a second), the camera image is first down-sampled from a 512x480 matrix to a 64x60 matrix and then a Sobel operator is applied for edge detection (Fig. 13).

Fig. 14a shows a typical example of an image of the hallway as seen by the camera, in Fig. 14b the output of the edge detector, in Fig. 14c the relevant floor edges, and in Fig. 14d the Hough map. The image in Fig. 14a was taken when the robot was pointed somewhat toward the left wall (as opposed

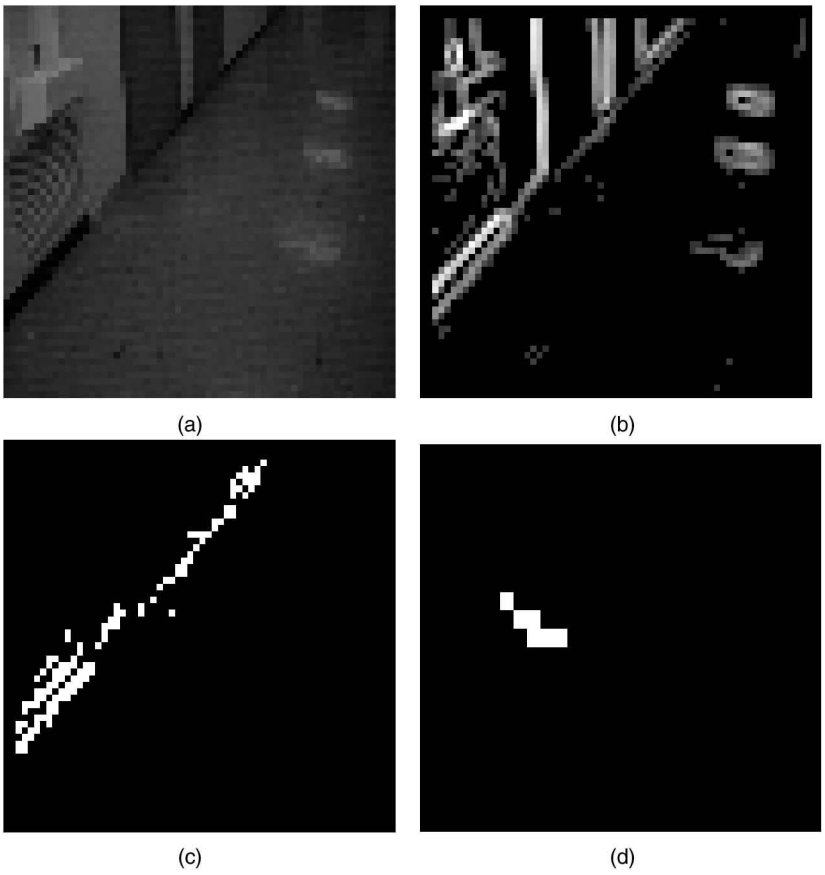


Fig. 14. Results after each step in Fig. 13. Reprinted with permission from [103].

to straight down the center of the hallway). So, the image is rich in edges that correspond to the junction between the left wall and the floor. In the Hough map, these edges occupy cells that are mostly in the left half of the map. It is observations such as these that determine which regions of the Hough space should go to what neural networks.

All neural networks in the different modules of NEURO-NAV are simple three-layered feedforward networks trained using a backpropagation algorithm. During the training phase, a *Human Supervisor* module takes control of the navigation. This module allows a human operator to specify simple command motions while he/she starts digitization routines that grab images that are used subsequently for training. In the work reported in [103], NEURO-NAV was able to generate correct steering commands 86 percent of the time. It generated incorrect commands 10 percent of the time and issued a "no decision" in the remaining 4 percent of the cases. However, even when NEURO-NAV generates an incorrect steering command in a specific instance, it can correct itself at a later instant.

As was mentioned before, there are various neural networks in NEURO-NAV. The invocation of each of these networks is orchestrated by a Supervisory Rule-Based Controller whose details are described in [102]. However, we do want to mention here that most of the outputs from these neural networks are fuzzy in nature. For example, the output of the *corridor_left* network consists of steering angles and they have labels such as $left\_40^o$, $left\_10^o$, $straightahead$, $right\_30^o$, etc., (Fig. 12). Similarly, the output of the *junction_detection* network consists of distances such as: $near$, far, at, etc. Each of these labels corresponds to an output node in a neural network. The value output at each node, typically between 0 and 1, can be taken as the degree of confidence to be placed in the motion directive corresponding to that node. Since these output values can be easily construed as corresponding to fuzzy membership functions, Pan et al. developed a more sophisticated version of NEURO-NAV and called it FUZZY-NAV [121] in which

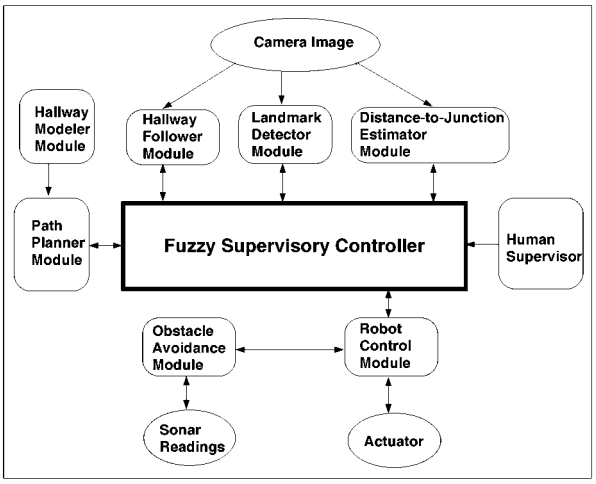


Fig. 15. Software architecture of FUZZY-NAV. Reprinted with permission from [121].

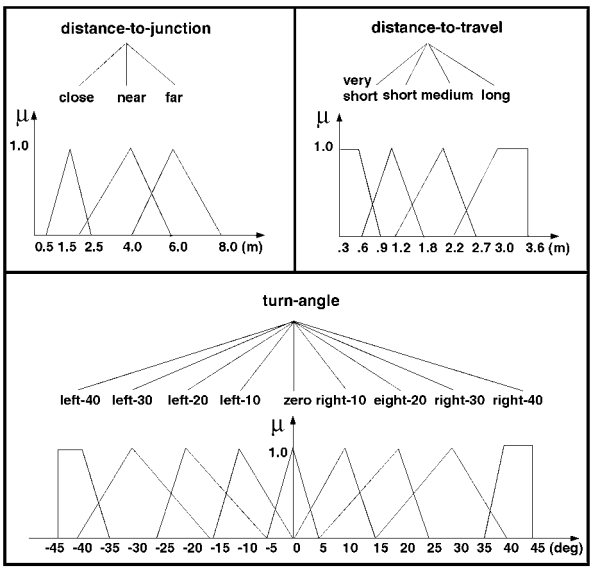


Fig. 16. Linguistic variables and their associated fuzzy terms used in FUZZY-NAV. Reprinted with permission from [121].

the *Rule-Based Supervisory Controller* of NEURO-NAV was replaced by a *Fuzzy Supervisory Controller* (Fig. 15).

The Fuzzy Supervisory Controller of FUZZY-NAV is a real-time fuzzy expert system that takes in the outputs of all the neural networks and decides what commands to issue to the Robot Control Module (see Fig. 15). To accomplish this, the Fuzzy Supervisory Controller uses three linguistic variables, *distance-to-junction*, *turn-angle*, and *distance-to-travel*. Associated with these linguistic variables are a total of sixteen fuzzy terms, as shown in Fig. 16. The semantics of the linguistic variables *distance-to-junction* and *turn-angle* were covered before, when we explained NEURO-NAV, and should be obvious to the reader. The linguistic variable *distance-to-travel* stands for the current best estimate of how far the robot should plan on traveling straight barring any encounters with obstacles. While the value of the linguistic variable *distance-to-junction* is derived from the vision data by one of the neural networks, the linguistic variable *distance-to-travel* is given a value by the firing of one or more rules in the Supervisory Controller. As with all systems using fuzzy logic, the membership functions for the terms displayed in Fig. 16 were arrived at empirically. The Fuzzy Supervisory Controller uses the Fuzzy-Shell [122] framework for fuzzy inference.

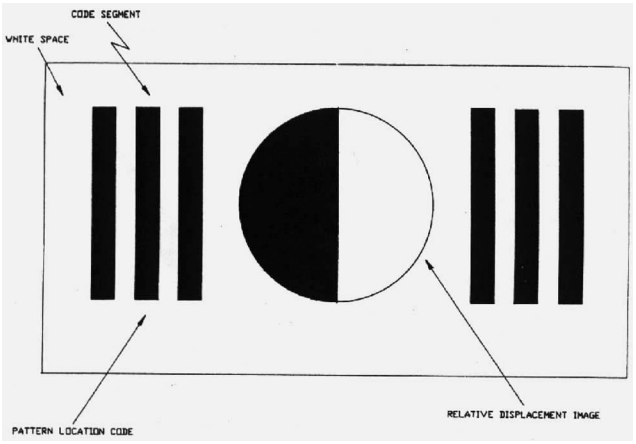
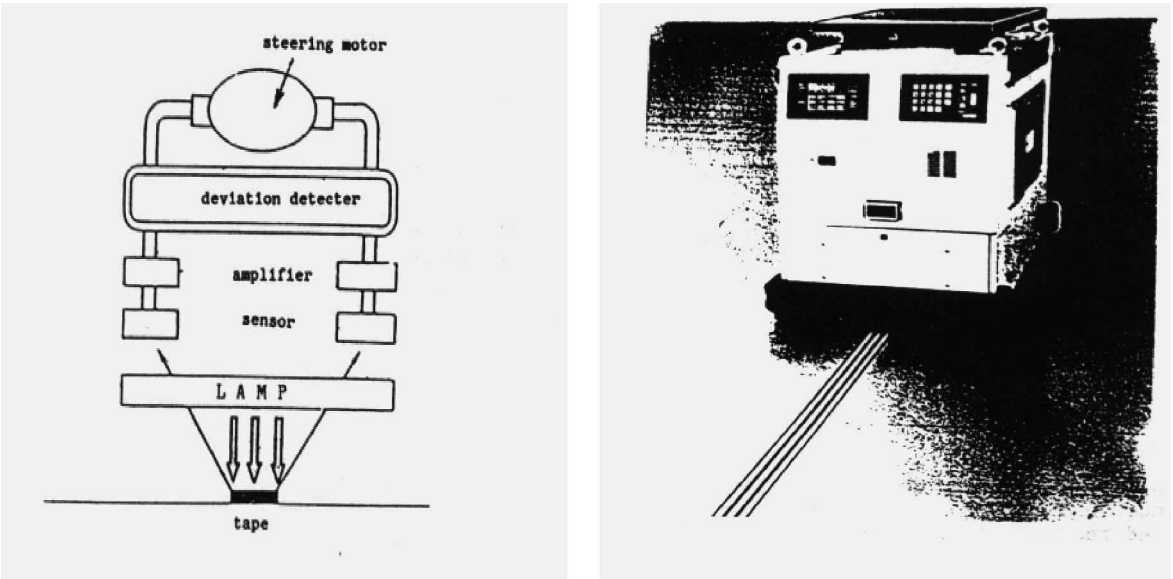


Fig. 17. Artificial landmarks used in [66].

2.1.3 Localization Derived from Landmark Tracking

One can do localization by landmark tracking when both the approximate location of the robot and the identity of the landmarks seen in the camera image are known and can be tracked. The landmarks used may either be artificial ones, such as those shown in Fig. 17, or natural ones, such as doors, windows, etc. The artificial landmarks vary from circles with a unique bar-code for each landmark [66] (same figure) to reflecting tapes stretched along the robot path, as reported by Tsumura in [149] (Fig. 18). In most cases, landmark



(a) (b)

Fig. 18. Reflecting tape guidance system. Reprinted with permission from [149].

tracking in these systems is carried out by using simple template matching. The cameras are usually mounted on the robot so that they look sideways at the walls where the landmarks are mounted or down at the floor where a special tape may be glued. When cameras are not mounted sideways, as in [66], properties of the shapes used for the artificial landmarks allow simple algorithms to be used for the localization of the robot. While sideways-mounted cameras simplify the problem of landmark detection (by eliminating scale and perspective effects), they also constrain the freedom regarding where the robot can move.

Another system that carries out landmark tracking in a map-based approach is by Hashima et al. [52]. In this system, however, the landmarks are natural. The system uses the correlation technique to track consecutive landmarks and to detect obstacles. For both landmark tracking and obstacle detection, the system uses a dedicated piece of hardware called TRV ([132]) that is capable of calculating local correlations of 250 pairs of image regions at 30fps. Localization is achieved by a multiple landmark detection algorithm. This algorithm searches for multiple landmarks, compares the landmarks with their prestored templates, tracks the landmarks as the robot moves, and selects new landmarks from the map to be tracked. As the landmarks are matched with the templates, their 3D pose is calculated using stereo vision. The position of the robot is estimated through a comparison between the calculated landmark position and the landmark position in the map. Obstacle detection is also performed using the TRV. In order to overcome problems with illumination, complementary processing is performed using stereo vision.

2.2 Map-Building

The vision-based navigation approaches discussed so far have all required the robot to possess a map (or a model) of the environment. But model descriptions are not always easy to generate, especially if one also has to provide metrical information. Therefore, many researchers have proposed automated or semiautomated robots that could explore their environment and build an internal representation of it. The first attempt at robotic map-building was the Stanford Cart by Moravec [107], [108]. The Stanford Cart used a single camera to take nine images spaced along a 50 cm slider. Next, an interest operator (now known as Moravec's interest operator, which was later improved by Thorpe [140] for FIDO [141]) was applied to extract distinctive features in the images. These features were then correlated to generate their 3D coordinates. All this processing was done remotely and took five hours to traverse 20 meters. The "world" was represented by the 3D coordinates of the features plotted in a grid of two square meter cells. The features were tracked at each iteration of the program and marked in the grid and in the image plane. Although this grid indirectly represented the position of obstacles in the world and was useful for path planning, it did not provide a meaningful model of the environment. For that reason, in 1985, Moravec and Elfes proposed a data structure, known as occupancy grid ([109]), that is used for accumulating data from ultrasonic sensors. Each cell in an occupancy grid has attached with it a probability value that is a measure of the belief that the cell is occupied. The data is projected onto the plane of the floor and each square of the occupancy grid represents the probability of occupancy of the corresponding square in the world by an object. In today's robots, occupancy grids allow measurements from multiple sensors to be incorporated into a single perceived map of the environment and even uncertainties can be embedded in the map, e.g., Histogram grids or Vector Field Histogram [15], [16].

While occupancy-grid-based approaches are capable of generating maps that are rich in geometrical detail, the extent to which the resulting geometry can be relied upon for subsequent navigation depends naturally on the accuracy of robot odometry and sensor uncertainties during map construction. Additionally, for large-scale and complex spaces, the resulting representations may not be computationally efficient for path planning, localization, etc. Such shortcomings also apply to nonoccupancy grid-based approaches to map learning, such as those proposed by Chatila and Laumond [20] and Cox [27].

The occupancy-grid approach to map learning is to be contrasted with the approaches that create topological representations of space [22], [23], [39], [74], [87], [97], [123], [161], [164]. These representations often have local metrical information embedded for node recognition and to facilitate navigational decision making after a map is constructed. The various proposed approaches differ with respect to what constitutes a node in a graph-based description of the space, how a node may be distinguished from other neighboring nodes, the effect of sensor uncertainty, the compactness of the representations achieved, etc. One of the major difficulties of topological approaches is the recognition of nodes previously visited. (In metrical approaches on the other hand, if the odometry and the sensors are sufficiently accurate, the computed distances between the different features of space help establish identify places previously visited.)

In a recent contribution [144], Thrun has proposed an integrated approach that seeks to combine the best of the occupancy-grid-based and the topology-based approaches. The system first learns a grid-based representation using neural networks and Bayesian integration. The grid-based representation is then transformed into a topological representation.

Other notable contributions related to map building are by Ayache and Faugeras [7], [6] using trinocular vision and Kalman filtering, by Zhang and Faugeras [162] using 3D reconstruction from image sequences, by Giralt et al. [44] using sensor fusion techniques, and by Zheng et al. [163], and Yagi et al. [159] using panoramic views.

2.3 Mapless Navigation

In this category, we include all systems in which navigation is achieved without any prior description of the environment. It is, of course, true that in the approaches that build maps automatically, there is no prior description of the environment either; but, before any navigation can be carried out, the system must create a map. In the systems surveyed in this section, no maps are ever created. The needed robot motions are determined by observing and extracting relevant information about the elements in the environment. These elements can be the walls, objects such as desks, doorways, etc. It is not necessary that absolute (or

even relative) positions of these elements of the environment be known. However, navigation can only be carried out with respect to these elements. Of the techniques that have been tried for this, the prominent ones are: optical-flow-based and appearance-based. The reader is also referred to attempts at behavior-based approaches to vision-based navigation in mapless spaces [111], [112], [56].

2.3.1 Navigation Using Optical Flow

Santos-Victor et al. [133] have developed an optical-flow-based system that mimics the visual behavior of bees. It is believed that the predominantly lateral position of the eyes in insects favors a navigation mechanism using motion-derived features rather than using depth information. In insects, the depth information that can be extracted is minimal due to the extremely narrow binocular field they possess. On the other hand, motion parallax can be much more useful especially when the insect is in relative motion with respect to the environment. Also, the accuracy and the range of operation can be altered by changing the relative speed. For example, features such as "time-to-crash" (which is dependent on the speed) are more relevant than distance when it is necessary to, say, jump over an obstacle.

In *robee*, as the robot in [133] is called, a divergent stereo approach was employed to mimic the *centering reflex* of a bee. If the robot is in the center of a corridor, the difference between the velocity of the image seen with the left eye and the velocity of the image seen with the right eye is approximately zero, and the robot stays in the middle of the corridor. However, if the velocities are different, the robot moves toward the side whose image changes with smaller velocity. With regard to the robotic implementation, the basic idea is to measure the difference between image velocities computed over a lateral portion of the left and the right images and use this information to guide the robot. For this, the authors computed the average of the optical flows on each side. From the fundamental optical flow constraint, we have:

$$\frac{\delta I}{\delta x}u+\frac{\delta I}{\delta y}v+\frac{\delta I}{\delta t}=0,$$

where u and v are the horizontal and the vertical flow components. Since the robot moves on a flat ground plane, the flow along the vertical direction is regarded as zero and the expression above reduces to:

$$u=-\frac{I_t}{I_x},$$

where I_t and I_x are the time and x-spatial derivatives of the image, respectively. The images are smoothed, with respect to both space and time prior to the computation of any derivatives. The time derivative is computed simply by subtracting two consecutive smoothed images. At each iteration of the control loop, five 256x256 stereo images are grabbed at video rate and used to compute the time-smoothed images. Then, the last two images on each side are used to compute the average optical-flow vectors. This average is calculated over a subwindow of 32x64 pixels on each side (Fig. 19.)

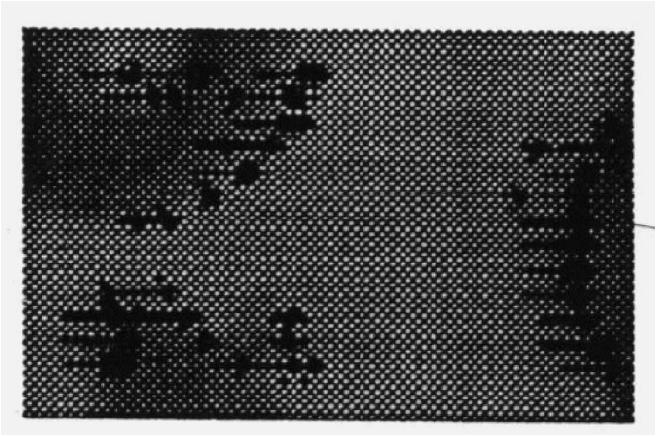


Fig. 19. Sample of the computed optical flows. Reprinted with permission from [133].

Finally, by observing that the left and right flows have opposite directions, the comparison of those two flows can be written as

$$\begin{aligned} e &= u_L + u_R \\ &= T_M\left(\frac{1}{Z_R}-\frac{1}{Z_L}\right), \end{aligned}$$

where T_M is the robot forward motion speed, and Z_R, Z_L provide the horizontal projections of these motions into the right and the left images. To keep the robot centered in a hallway, a PID controller is used (Fig. 20). The input to this controller is the difference between the average optical flows from the left and the right cameras according to the equation above.

The above equations are somewhat oversimplified in the sense that they apply only when the two cameras are pointing in symmetrically divergent directions with respect to the direction of motion. Besides that, strictly speaking, the equations, do not permit the direction of motion to be changed, a condition that cannot be satisfied during rotational motions. However, the authors have shown

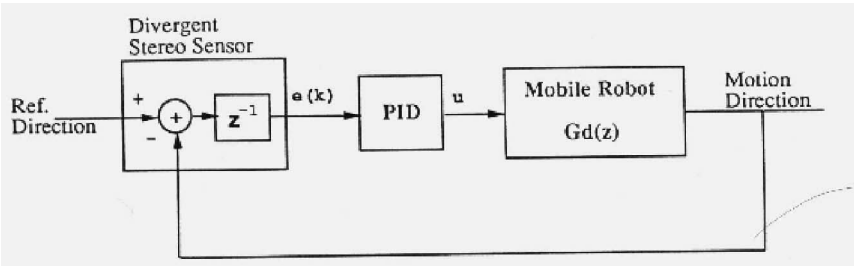


Fig. 20. The PID controller used in [133].

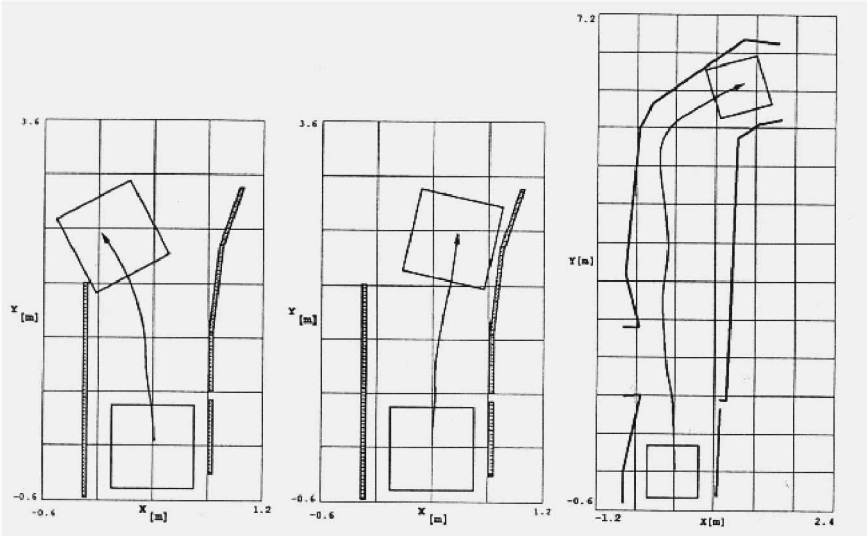


Fig. 21. Examples with (center and right) and without (left) the sustained behavior. Reprinted with permission from [133].

analytically that the equation can be used provided the following conditions are satisfied: 1) The rotational speed is not larger than a certain factor times the translational speed. This factor is a function of the relative position of the cameras in polar representation with respect to the robot coordinate frame. 2) The cameras are mounted as close as possible to the robot rotation center. And, that 3) a calibration procedure is performed in order to compensate for an independent term that appears in the equation for rotational motion.

The technique also runs into difficulties if there is insufficient texture on the walls of a corridor. Since the optical flow calculation depends on the existence of some texture, it is desirable that when the robot runs into a section of the corridor deficit in wall texture, the robot exhibit what the authors have referred to as *sustained behavior*. In the implementation reported in [133], if texture can be extracted from one of the two walls, the robot maintains its velocity and direction as it begins to follow the wall with the texture. If texture disappears from both sides, the robot halts. The sustained behavior was obtained by monitoring the "amount of flow" on both sides. When the flow on one side is not significant, the system uses a reference flow that must be sustained in the other image (Fig. 21).

In a more recent work, [10], Bernardino and Santos-Victor have included two visual behaviors, vergence and pursuit, to design a control strategy for fixating and tracking objects using a stereo head with pan, tilt, and vergence control. Also mimicking bees, Rizzi et al. [131], have developed a homing method based on an affine motion model, which requires no object recognition or 3D information from the scene. The rectified image obtained by applying the affine model to the current image is compared to the goal image and a vector indicating the movement is estimated.

Again, using the optical-flow technique, Dev et al. [33] have implemented wall following navigation by extracting depth information from optical flow.

2.3.2 Navigation Using Appearance-Based Matching

Another way of achieving autonomous navigation in a mapless environment is by "memorizing" this environment. The idea is to store images or templates of the environment and associate those images with commands or controls that will lead the robot to its final destination.

Gaussier et al. [43] and Joulian et al. [65] developed an appearance-based approach using neural networks to map perception into action. The robot camera captures a 270-degree image of the environment by combining a series of images, each with 70 degrees of field of view. This panoramic image is processed and a vector of maximum intensity averages along x (the horizontal axis) is obtained. For each maximum value in this vector, a "local view" is defined. This local view is a 32x32 subwindow extracted from the panoramic window (Fig. 22). The set of all local views for a given panoramic image defines a "place" in the environment. Each place is associated with a direction (azimuth) to the goal. Finally, a neural network (Fig. 23) is used to learn this association and, during actual navigation, it provides the controls that take the robot to its final destination.

Using appearance-based navigation, Matsumoto et al. [98] extended the place recognition idea in Horswill's mobile robot Polly [55] by using a sequence of images and a template matching procedure to guide robot navigation. Subwindows extracted from down-sampled versions of camera images are used to form a sequence of images (Fig. 24) that works as a "memory" of all the images observed during navigation. Each image in this sequence is associated with the motions required to move from the current position to the final destination—this is referred to as VSRR (View-Sequenced Route Representation). After a sequence of images is stored and the robot is required to repeat the same trajectory, the system compares the currently observed image with the images in the sequence database using correlation processing running on a dedicated processor. Once an image is selected as being the image representing that "place," the system computes the displacement in pixels between the view image and the template image. This displacement is then used in a

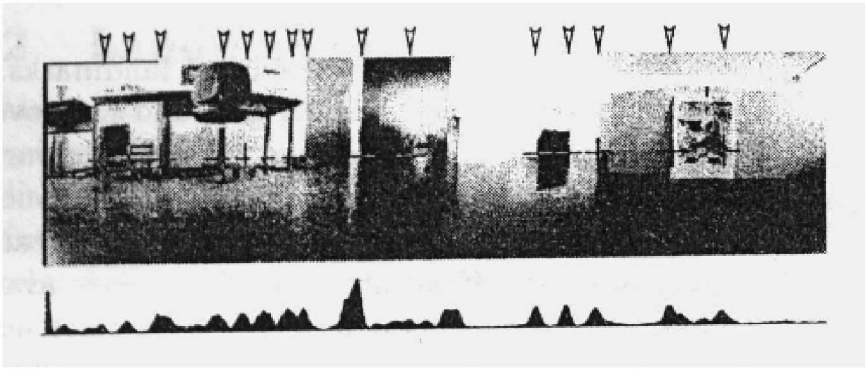


Fig. 22. Panoramic view of a place and the corresponding vector of maximum intensities. Reprinted with permission from [43].

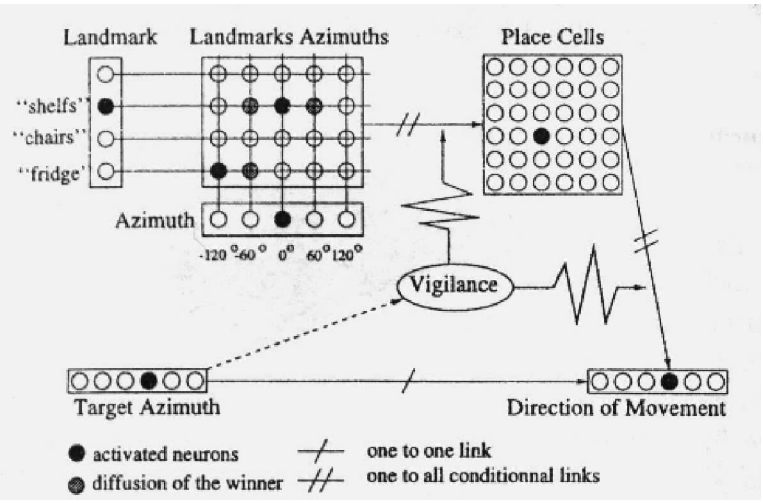
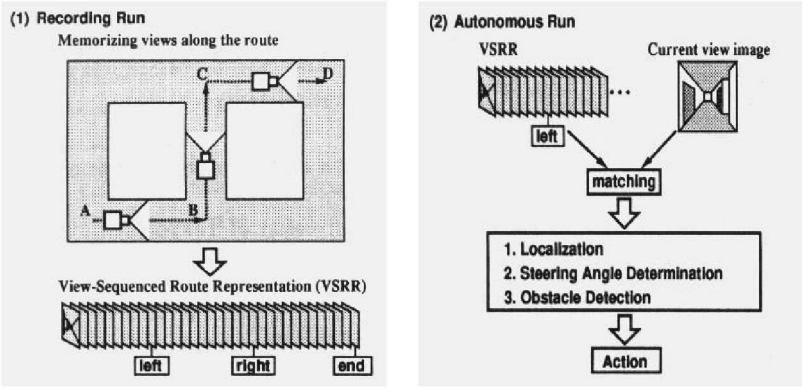


Fig. 23. Navigation neural network. Reprinted with permission from [43].



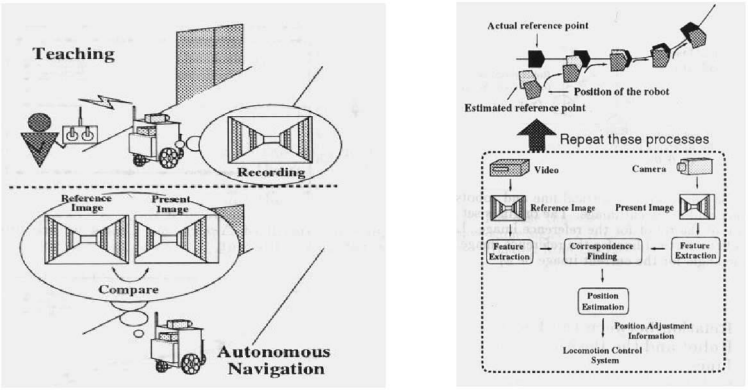
(a) (b)

Fig. 24. Outline of navigation using the view-sequenced route representation. Reprinted with permission from [98].

table to provide real-world displacements and angles to be used in the steering commands.

Another system built around Matsumoto's VSRR concept is reported by Jones et al. [64]. There, the robot is given a sequence of images and associated actions (motion commands). Using zero-energy normalized cross-correlation, the robot retrieves the image from the database that best matches the view image. If the match is above a certain threshold, the robot performs the commands associated with that image, otherwise it halts.

In 1996, Ohno et al. ([115]) proposed a similar but faster implementation also using sequences of images. But since their system (Fig. 25) uses only vertical hallway lines, it needs less memory and can operate faster. The work of Ohno et al. also includes extensive experimentation that shows how to construct association lists of the changes in the position and the orientation of the robot, on the one hand, and the changes in what the robot sees. For another appearance-based approach that uses elastic template matching—a template is a set of features, each extracted from a 7x7 window in the image and their relative positions



(a) (b)

Fig. 25. (a) Recording the image sequence (top) and comparing the current image with the recorded sequence (bottom). (b) The image processing steps for trajectroy correction. Reprinted with permission from [115].

within the entire image—see the work of Balkenius [9]. Martínez and Vitrià [94] have also proposed an appearance-based approach using mixture models learned by the expectation maximization algorithm.

2.3.3 Navigation Using Object Recognition

While in map-based systems it is easy to establish meaningful navigational goals for the robot, most robotic systems are limited to essentially just roaming in mapless systems. The reason for that is that a human operator can use the internal map representation of a structured environment to conveniently specify different destination points for the robot. But, for the mapless case, using the appearance-based approaches mentioned so far, in most cases the robot only has access to a few sequences of images that help it to get to its destination, or a few predefined images of target goals that it can use to track and pursue. Kim and Nevatia [72], [73] have proposed a different approach for mapless navigation. Instead of using appearance-based approaches to memorize and recall locations, a symbolic navigation approach is used. In this case, the robot takes commands such as *"go to the door"* or *"go to the desk in front of you"* etc., and uses the symbolic information contained in these commands to establish the landmarks it needs to recognize and the path it needs to take to reach the goal. For example, a command such as *"go to the desk in front"* tells the robot that the landmark is the desk and the path should point straight ahead. The robot builds what the authors call a "squeezed 3D space into 2D space map" or "s-map" which is a 2D grid that stores the projections of the observed landmarks as they are recognized using a trinocular vision system. Once the target landmark (e.g., desk) is recognized and its location is projected into the s-map, the robot plots a path using a GPS-like path planner and deadreckoning to approach the target.

3 Outdoor Navigation

As with indoor navigation, outdoor navigation usually involves obstacle-avoidance, landmark detection, map building/updating, and position estimation. However, at least in the research reported so far in outdoor navigation, a complete map of the environment is hardly ever known a priori and the system has to cope with the objects as they appear in the scene, without prior information about their expected position. Nevertheless, outdoor navigation can still be divided into two classes according to the level of structure of the environment: outdoor navigation in structured environments and in unstructured environments.

3.1 Outdoor Navigation in Structured Environments

One of the first outdoor navigation systems reported in the literature is by Tsugawa et. al. [148] for a car that could drive autonomously, albeit under a highly constrained set of conditions, at 30 km/hr. This system used a pair of stereo cameras mounted vertically to detect expected obstacles. The navigation relied mostly on obstacle avoidance.

In general, outdoor navigation in structured environments requires some sort of road-following. Road-following means an ability to recognize the lines that separate the lanes or separate the road from the berm, the texture of the road surface, and the adjoining surfaces, etc. In systems that carry out road following, the models of the environment are usually simple, containing only information such as vanishing points, road and lane widths, etc. Road-following for outdoor robots can be like hallway-following for indoor robots, except for the problems caused by shadows, changing illumination conditions, changing colors, etc.

One of the most prominent pieces of work in outdoor navigation is the Navlab project and its various incarnations. Navlab, initially developed by Thorpe et al. [142], has now gone through 10 implementations [124], [142], [143], [61], [63] based on vehicles ranging from minivans to metro buses and equipped with computing hardware ranging from supercomputers (Navlab 1) to laptops (Navlab 5 and later).

In its first implementation [142], [143], Navlab used color vision for road following and 3D vision for obstacle detection and avoidance. The R, G, and B images were stored in a pyramid of decreasing resolutions. The images start with 480x512 pixels and end with 30x32 pixels (Fig. 26). The higher resolution images are used for texture classification, while the lower resolution images are used for color classification. The algorithm for road following consists of three stages: pixel classification, best-fit road position vote, and color update.

Pixel classification in Navlab 1 is performed by combining two separate approaches: color classification and texture classification. During color classification, each pixel is

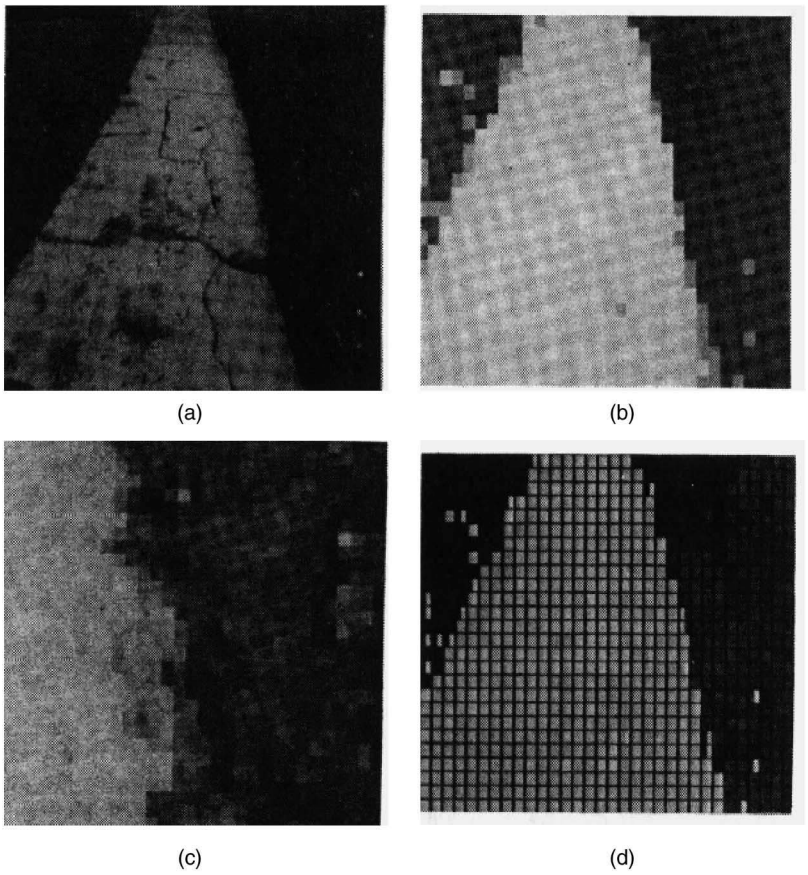


Fig. 26. Pixel classification: "road" and "nonroad" classes. (a) Original image. (b) Color classification. (c) Texture classification. (d) Final classification. Reprinted with permission from [143].

assigned a class depending on the color distributions. The road pixels are represented by four separate Gaussian clusters to account for changes in illumination, road surface properties, etc. Similarly, the nonroad pixels are also represented by four separate Gaussian clusters. Each Gaussian distribution is represented by a three-dimensional mean vector (mean R, G, and B), by a three-by-three covariance matrix, and by the fraction of pixels expected a priori to be represented by that distribution. The probability that a pixel with color vector X belongs to the distribution i, P_i^C, is computed using the Mahalanobis distance: $(X - m_i)^T \Sigma^{-1} (X - m_i)$. The classification is carried out using the standard maximum-likelihood ratio test. The result is shown in Fig. 26b.

Navlab 1 bases texture classification on the observation that road regions tend to appear much smoother in an image compared to nonroad regions. Smoothness is measured via a normalized gradient index that uses a high-resolution gradient (at 240x256 pixels) divided by a weighted sum of a low-resolution gradient (at 60x64 pixels) and the mean pixel value per region. The gradients are calculated using the Robert's operator. The smoothness indices are thresholded to produce a counter of "micro-edges" per 8x8 block of pixels. The result is a 30x32 texture image with values between 0 and 255 representing the number of microedge pixels in the corresponding 8x8 block of the full-resolution image. Finally, the texture classification itself is done using a probability P_i^T that depends on the microedge count (Fig. 26c).

As mentioned before, the final classification of a pixel is carried out by combining the results obtained from color and texture. This combination is simply a weighted sum of the two probabilities:

$$P_i = (1 - \alpha) P_i^T + \alpha P_i^C.$$

Since color classification proved to be more reliable than texture, the value chosen for α reflects that fact. The final result, shown in Fig. 26d, is the classification of the pixels into road and nonroad with a confidence value calculated by:

$$\begin{aligned} C = & max\{P_i,\ i\ is\ a\ road\ class\} \\ & - max\{P_i,\ i\ is\ a\ nonroad\ class\}. \end{aligned} \tag{9}$$

Once the pixels are classified, a Hough-like transform is applied to the road pixels to obtain the two parameters that specify the road vanishing point and orientation: intercept, P, and orientation, θ (Fig. 27). As in a Hough transform, each pixel votes for the points in the (P, θ) space to which it could belong. The point in the (P, θ) space with the most votes is the one that represents the parameters of the road.

Finally, once the position of the road and its edges leading to a vanishing point are determined using the above voting procedure, the pixels are reclassified taking into account the determined road edges. The new parameters (mean and variance) from this new classification are then used to classify

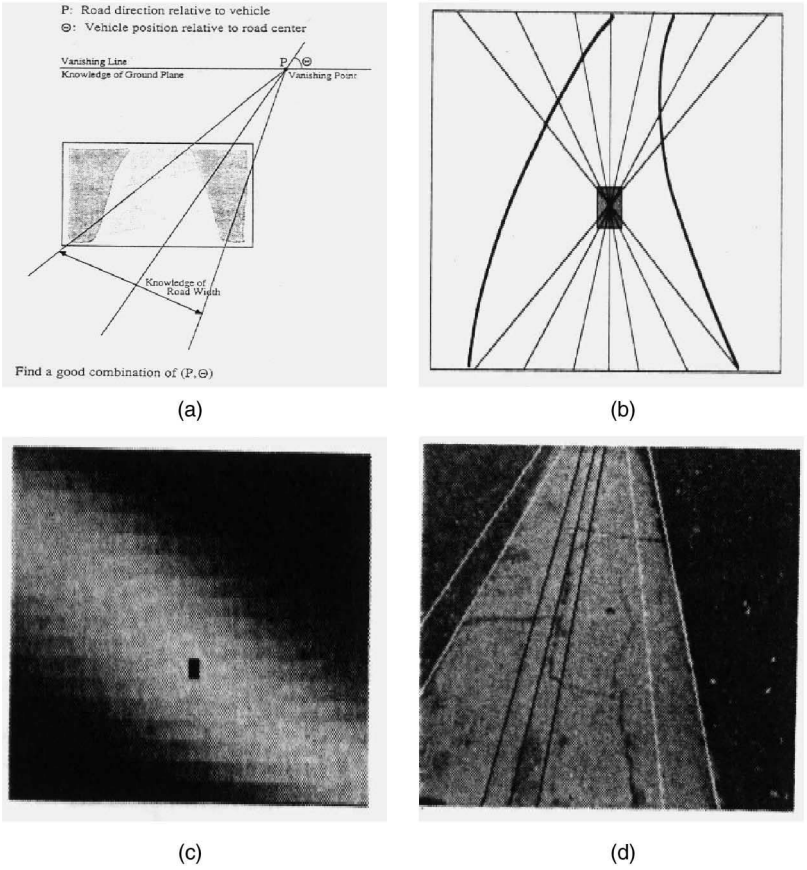


Fig. 27. Road Transform. (a) Road description using P and θ. (b) Possible orientations of the road passing thru one point. (c) Best road orientation and intercept (dark square) in the (P, θ) space. (d) Detected road using dark point in (c). Reprinted with permission from [143].

the pixels in the next image. This allows the system to adapt to gradual changes in the conditions of the road.

The NAVLAB 1 vehicle was also equipped with a neural-network based navigational system, called ALVINN [124], [125], [126], [127]. The ALVINN system (Autonomous Land Vehicle In A Neural Network) was first reported in 1989 [124]. Its later versions came with additional capabilities with regard to robust road detection and confidence estimation. One of the later versions is called ALVINN-VC, where VC stands for Virtual Camera [61], [63]. In the following paragraphs, we will first summarize the key ideas of the core ALVINN system and then briefly mention the added capabilities of ALVINN-VC.

The key idea of ALVINN, as reported in [125], consists of watching a human driver for a few minutes in order to learn his/her reactions when driving on roads of varying properties. The neural network in ALVINN has an input layer consisting of 30x32 nodes, onto which the video camera image is projected (Fig. 28). This input layer is fully connected to a 5-node hidden layer[1] which, in turn, is fully connected to the output layer consisting of 30 nodes that represent each of the possible steering angles that the vehicle must follow in order to keep traveling in the middle of the road.

The neural network is trained using a standard back-propagation algorithm. However, during the learning stage, instead of assuming that only a single output node should be activated for a given orientation of the road, a Gaussian distribution of activations (centered around the

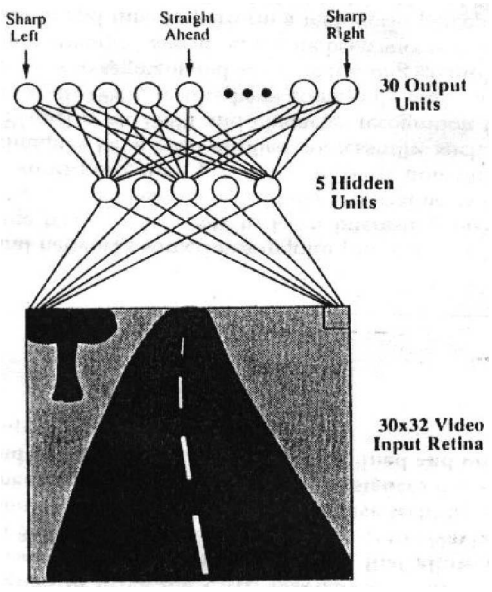


Fig. 28. Architecture of the neural network in ALVINN. Reprinted with permission from [125].

1. In [127], a slightly different neural network with a 4-node hidden layer is employed.

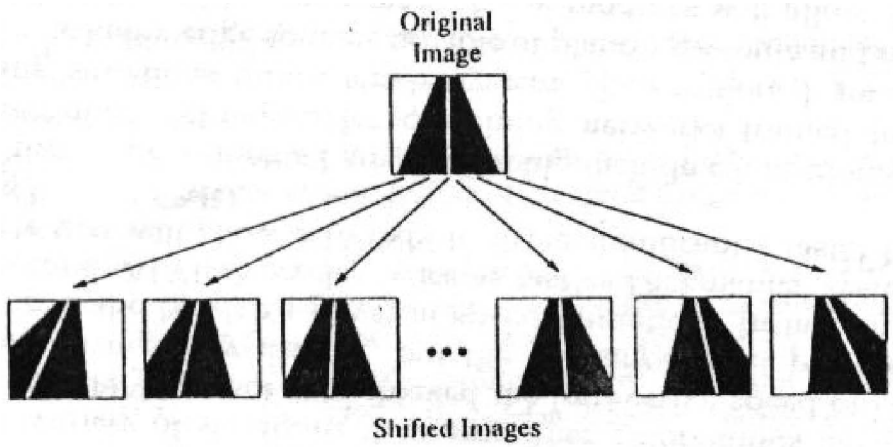


Fig. 29. The original and the 14 "distored" images used to train the NN. Reprinted with permission from [125].

node that corresponds to the correct road orientation) is assumed. The training-phase activation level for each output node is given by

$$x_i = e^{(-{d_i}^2/10)},$$

where x_i represents the activation level for the output node i and d_i represents the distance between the ith node and the node corresponding to the correct steering angle for the road image fed into the input layer. The constant 10 is empirically chosen so that the activation level for the 10th node to the left or right of the correct steering direction is made very small.

Using Gaussian-distributed activations during the training phase produces a finer steering angle during actual navigation because the center of mass of the Gaussian function may lie between two output nodes, instead of corresponding to one or the other node. Moreover, as reported in [125], this results in a kind of "continuous" output function for the neural network, in the sense that small changes in the orientations of the road result in small changes in the activation levels of the outputs. On the other hand, in a more traditional "one of N" classifier output from a neural network, it is required that the network exhibit a nonlinear behavior in its output, the nonlinearity arising from the fact that a slight change in the road orientation could cause an output node to change abruptly from zero to one or the other way around.

The first training scheme for ALVINN used synthetic images. As one would expect, the effort required for generating these images proved too onerous. Besides, when the differences between real images and the synthetic images were significant, the performance of the system in actual driving situations was poor. These problems were circumvented by an "on-the-fly" training scheme in which the neural network is trained for actual driving situations performed by humans. In this scheme, the training is performed online, using data that is collected at the same time as a human performs the steering of the vehicle. A major disadvantage of this approach comes from the fact that since a human driver tends to keep a vehicle consistently in the center of the road, the neural network may not "experience" situations that require correction from a misaligned trajectory. Another related problem is caused by the fact that actual driving situations involve long straight stretches of roads. This causes "on-the-fly" training schemes to overtrain the neural network to straight roads, in the sense that the neural network can forget what it learned earlier about driving on curved roads. Both these problems stem from the following need of a backpropagation algorithm: The training data must adequately represent all possible input conditions. The authors of ALVINN have tried to get around these problems with training by transforming each actual road image into 14 "distorted" images that correspond to the road curving to the left and to the right (Fig. 29). These additional images are then added to the pool used for training.

The overall result of this first implementation of ALVINN was a system capable of driving Navlab 1 at its maximum speed of 20 miles per hour. Later, in [127], ALVINN was reported to perform well at speeds of up to 55 miles per hour using a second version of the testbed vehicle called Navlab 2.

An improved version of ALVINN is the ALVINN-VC system [61], [63]; the latter system is better at detecting roads and intersections. In the VC (for Virtual Camera) version, virtual images are constructed from the actual camera image, the virtual images corresponding to different viewpoints at different distances from the actual camera. The neural networks are trained on these virtual images (Fig. 30). Since the virtual images are obtained by projecting the actual image pixels representing road segments farther away from the vehicle, instead of immediately in front of it (Figs. 31 and 32), this technique allows the system to detect road changes and intersections before they get too close to the vehicle. Another important feature in ALVINN-VC is its capability of measuring its confidence in detecting a road through an idea called Input Reconstruction Reliability Estimation (IRRE). It consists of using the neural network's internal representation to reconstruct the original image and comparing it to the actual original image. Both images, the reconstructed and the observed, are correlated and an index measuring their similarity is calculated and used as the IRRE metric. The justification for the usefulness of this metric comes from the fact that the more closely both images match each other, the more familiar to the neural

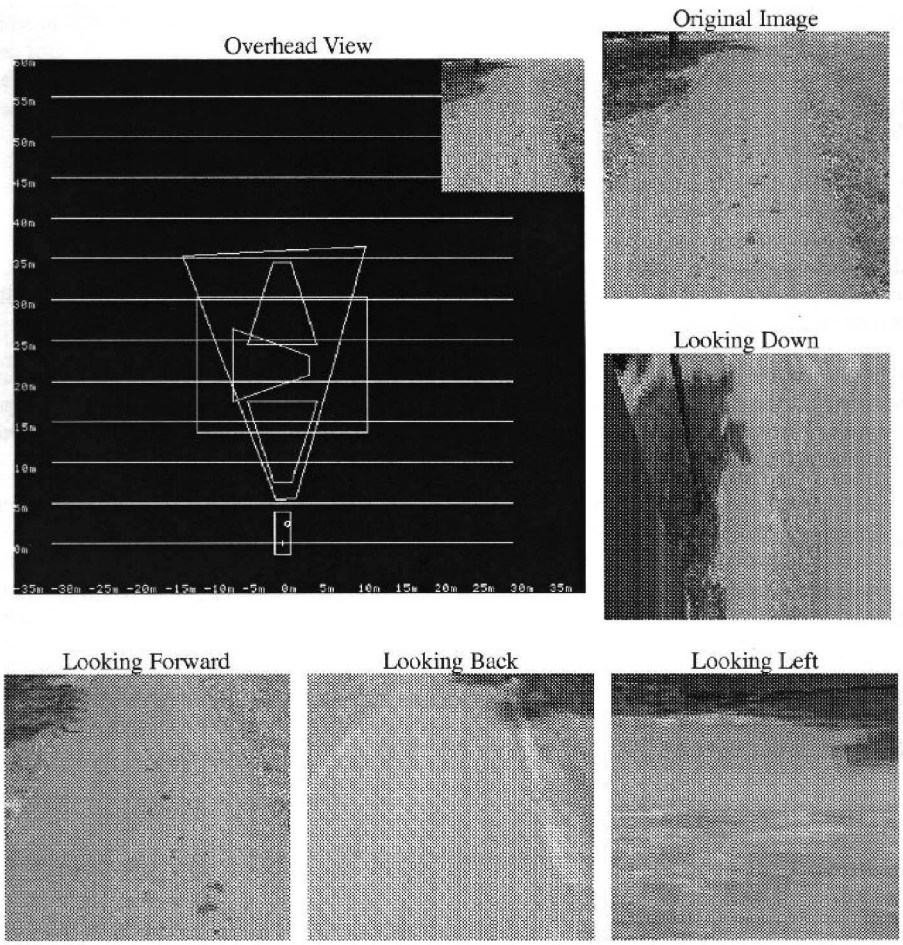


Fig. 30. Virtual images obtained from the actual camera field of view. Reprinted with permission from [61].

network is the input image and, hence, the more reliable is the network's response.

Another pioneering development in road-following was VITS (Vision Task Sequencer) by Turk et al. [151]. VITS was a system developed for Alvin, the eight-wheeled Autonomous Land Vehicle at Martin Marietta Denver Aerospace. VITS is a general framework for vision for outdoor road-following together with obstacle detection and avoidance [38], [150]. The primary vision sensor used by VITS is a fixed-focus CCD color camera that provides 480x512 RGB images, with eight bits of intensity for each color component. The camera is mounted on a pan/tilt unit that is under the direct control of the vision subsystem. The other sensor used by VITS is a laser range scanner, developed by ERIM. The sensor determines range by measuring the phase shift of a reflected modulated laser beam. The laser is continuously scanned over a field that is 30 degrees vertical and 80 degrees horizontal. The output of the scanner is a digital image consisting of a 64x256 array of pixels with eight bits of range resolution.

To fully grasp the functionality of VITS, it has to be seen as a part of the larger reasoning and control framework for Alvin (Fig. 33). The vision system produces a description of

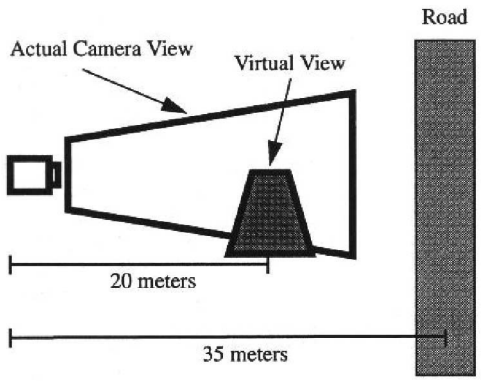


Fig. 31. Virtual camera for detecting roads. Reprinted with permission from [63].

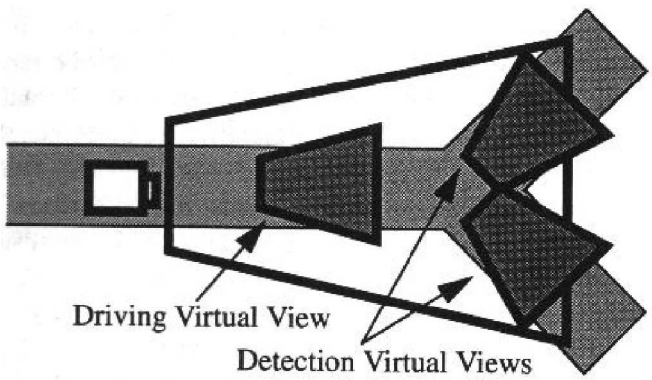


Fig. 32. Virtual camera for detecting intersections. Reprinted with permission from [63].

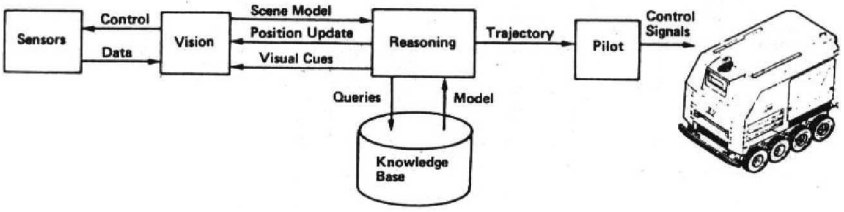


Fig. 33. Alvin system configuration. Reprinted with permission from [151].

the road, in a vehicle-centered coordinate frame, in front of the vehicle from either the video data or the range images or both. To this description is added information such as the current position, speed, heading, etc., (this additional information is supplied by the Reasoning Module to help the Vision Module decide where to sample the image for road pixels) to form a scene model—a data structure where all this information is stored.

An example of a hypothetical scene model is shown in Fig. 34. The scene model, as determined from each image frame is shipped off to the Reasoning module where the road description is transformed into a fixed world coordinate frame for the calculation of a trajectory for Alvin. This trajectory is then supplied to the Pilot whose job is to actually drive the vehicle by issuing commands to the vehicle controllers. The Vision Module also receives from the Reasoning Module any relevant visual cues, these being features significant for vision processing, features such as intersections, sharp curves, etc. The overall site map and the visual cues are stored in the Knowledge Base module. In addition to computing the trajectory for Alvin to follow, the Reasoning module also carries out an evaluation of the scene models generated from successive camera images for the smoothness and continuity of the road edges.

So far, we have discussed the architectural context for the vision module. Evidently, the most significant duty assigned to the Vision module is the construction of a scene model from the sensor data. We will now describe briefly the algorithmic steps that extract a description of the road from a camera image. Basic to road-description construction is the segmentation of a camera image into road pixels and nonroad pixels. Because environmental conditions, such as the presence or the absence of shadows caused by trees and

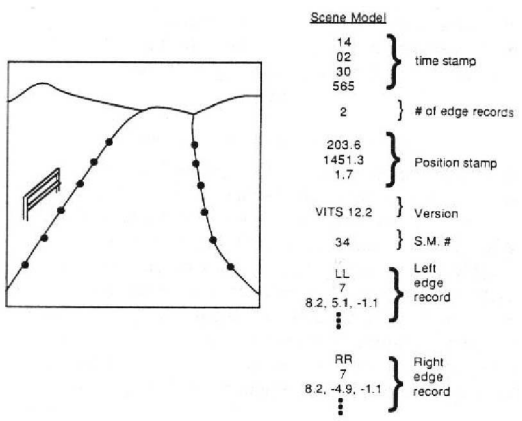


Fig. 34. Example of a road scene and its corresponding model. Reprinted with permission from [151].

other artifacts, fresh tarmac, presence of dirt on the road, brightness of sun, presence of water or snow on the ground, etc., can wreak havoc with fixed thresholding of images, robust extraction of the road pixels from camera images was a major challenge faced by the designers of VITS. VITS uses dynamic thresholding that calculates the decision threshold for separating road pixels from nonroad pixels by projecting into the image a sampling window for road pixels by projecting into the world coordinates the road boundaries found in the previous frame and then reprojecting into the current frame a trapezoid formed by those road boundaries (Fig. 35b).

A predictive sampling window formed in this manner is then used to calculate the color properties of the road pixels and a threshold on these properties used to separate the road pixels from the nonroad pixels in the current frame. This is done by first clustering the pixels in the RGB color space. A cluster is formed for the pixels within the sampling window and the rest of the pixels allowed to group into one or more clusters. Ideally, a projection of these clusters into the important Red-Blue plane will look like what is shown in Fig. 36. (The Red-Blue plane is important for road-following algorithms because the spectral composition of the road pixels tends to be predominantly blue and that of the shoulder pixels predominantly red.) When road and nonpixels cluster as neatly as what is shown in Fig. 36, one can easily construct a linear discriminant function to separate the color space into two halves, one half corresponding to the road pixels and the other half corresponding to the nonroad pixels. But, unfortunately, in actual imagery, the clusters in the RGB can also look like what is shown in Fig. 37. Now, it would not be possible to construct a linear discriminant function to separate the road pixel cluster from the nonroad pixel clusters, the problem being caused by the space occupied by the shaded nonroad pixels. In general, this would call for using nonlinear discriminant functions. The authors of VITS have proposed a solution by constructing bounding rectangles for the road pixel clusters in the color space. The bounding boxes allow them to carry out fast image segmentation by using two global lookup table operations per boxed region. The road segmentations thus obtained are followed by an edge-tracking algorithm for the extraction of road boundaries needed for the scene model.

Another success story in outdoor navigation is the research in road-following carried out for "Autobahns" [36], [37] and all the derived work that came from that [35], [34], including the EUREKA-project "Prometheus" [46], [47], [48], [50], [49], [129]. Investigation in Prometheus centered on the potential of robot vision technology to improve traffic safety and to alleviate fatigue for car and truck drivers. The goal was the development of a technology for an automatic copilot intended to warn the driver of

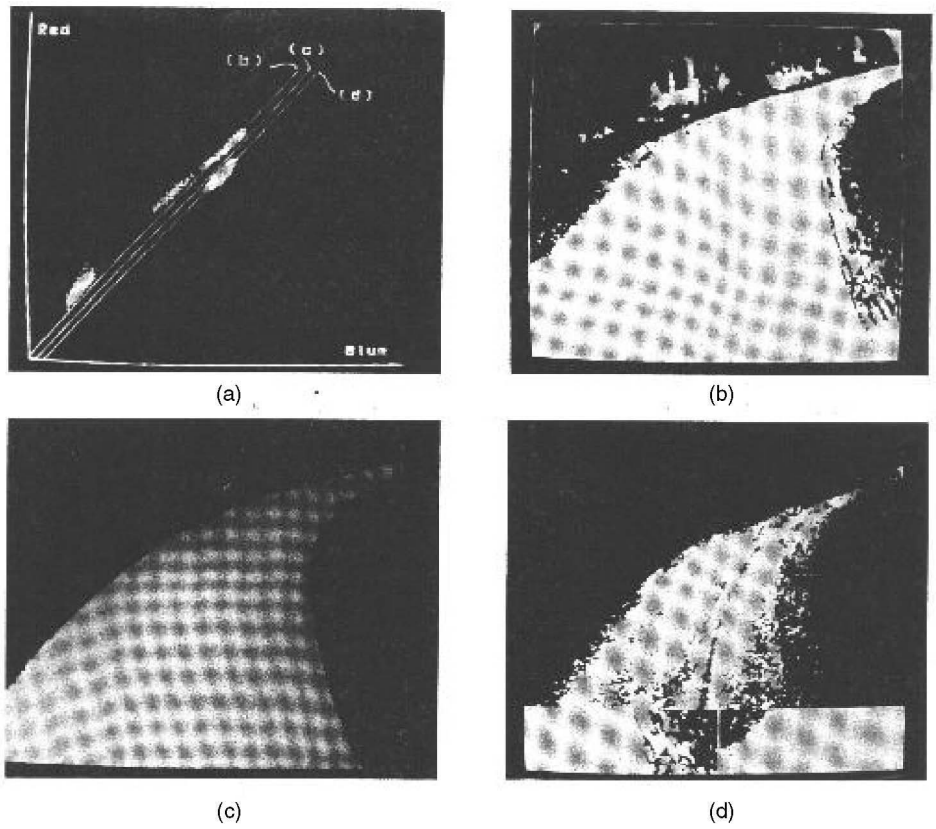


Fig. 35. The segmentation results for different thresholding values. Reprinted with permission from [151].

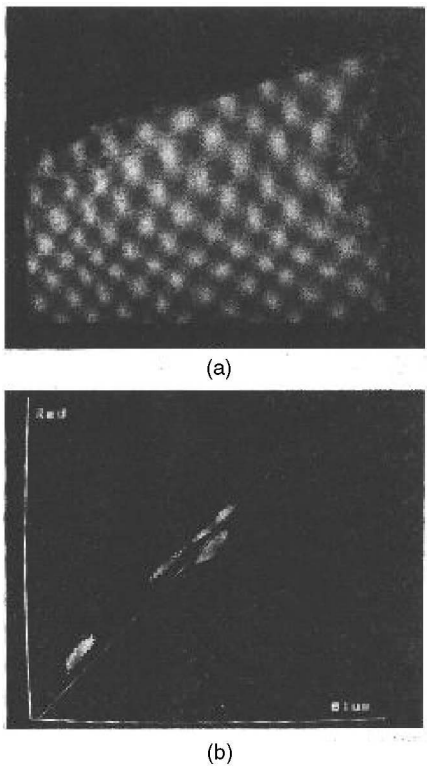


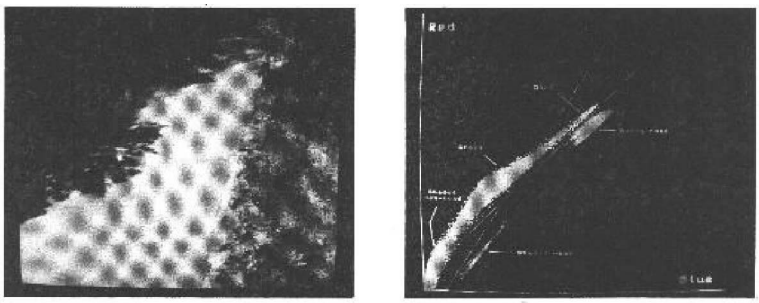
Fig. 36. (a) Original road image. (b) Red/Blue clusters. Reprinted with permission from [151].

imminent danger or even to drive automatically during monotonous periods [47], [48], [49]. This effort resulted in driving speeds of 96 km/h on a highway and 40 km/h on unmarked roads using a method of feature extraction called controlled correlation [85], [86]. In this method, the first step of image processing, the feature extraction, is performed by correlation processing of only selected relevant sections of the image. The methods developed allow dynamic scenes to be analyzed in real time using standard microprocessors.

Other pioneering research contributions on computer vision for road following include those of Waxman's et al. [154], [155], Wallace et al. [152] and Wallace [153], Lawton et al. [88], Goto and Stentz [45], Kuan et al. [83] and Kuan and Sharma [84], and others.

We should also mention that underwater pipe inspection is an example of a class of problems that bears many similarities with vision for road-following. Extracting the contours of a pipe is equivalent to extracting and following the road contours. In [130], for example, Rives and Borrelly took a visual servoing approach and devised a controller that takes as inputs the lines extracted from the image of a pipe and uses this information to generate steering commands for the ROV Vortex vehicle.

Speed is evidently an important issue for outdoor navigation. In order to achieve high speeds, the throughput of the vision system must be maximized. Kelly and Stentz [70] have proposed an adaptive approach based on the active vision paradigm that can be used to increase the throughput of perception and, thus, to increase the maximum speed of a mobile robot. This idea, called



(a) (b)

Fig. 37. (a) Original road image with shadows. (b) Red/Blue clusters. Reprinted with permission from [151].

Adaptive Perception, can be implemented by keeping the focus of attention restricted to only the region of interest. The system is adaptive with respect to three parameters: lookahead, sweep, and scan. These parameters control, respectively, the distance to the region of interest in front of the vehicle, the size of the region of interest, and the scanning resolution.

3.2 Unstructured Outdoor Navigation

An outdoor environment with no regular properties that could be perceived and tracked for navigation may be referred to as unstructured environment. In such cases, the vision system can make use of at most a generic characterization of the possible obstacles in the environment, as in [120] (Fig. 38).

Unstructured environments arise in cross-country navigation as, for example, in planetary (lunar/martian-like) terrain navigation (Fig. 39) [157], [158], [12], [81], [99]. Sometimes in these sorts of applications, the robot is supposed to just wander around, exploring the vicinity of the robot without a clearcut goal [91], [136]. However, in other applications, the task to be performed requires that the robot follow a specific path to a goal position. In those cases, a map of the traversed areas has to be created and a localization algorithm has to be implemented in order to carry out goal-driven navigation. When maps need to be built by a robot, they may be created in either a vehicle-centered coordinate frame vehicle [81] (and updated [82] as the vehicle moves) or with respect to some external reference, such as an external camera attached to a companion device [99] (as in Mars Pathfinder and Lander), or a virtual viewpoint with respect to which range data is projected to form a Cartesian Elevation Map [30] or a global positioning reference such as the sun [29].

The techniques that have been developed for localization in unstructured environments include: external camera observation [99], far-point (mountain peaks) landmark triangulation [139], global positioning [29], etc. For localization by global positioning, presented in [29] is an approach for measuring the position of the sun using a digital inclinometer, a camera with neutral density filters, and an adjustable platform. The system first determines the circles of equal altitude corresponding to each image in a time-indexed sequence of images of the sun and then calculates the final position using a least-squares estimate.

Krotkov and Hebert [81] have presented a mapping and positioning systems for RATLER—Robotic All-Terrain Lunar Explorer Rover. For mapping, they present a correlation matcher that intelligently selects parts of the left and the right images to be processed and subsamples the intensities without subsampling the disparities. By doing so, they save processing time without compromising the accuracy of the depth map or without resorting to special hardware. The positioning system uses traditional sensors such as encoders, inclinometers, a compass, etc. Also using the RATLER vehicle, the work reported in [136] implements a system that uses stereo vision to build a map that represents the terrain up to seven meters in front of the rover.

A highly noted accomplishment in vision-based navigation in unstructured environments is the Mars Pathfinder

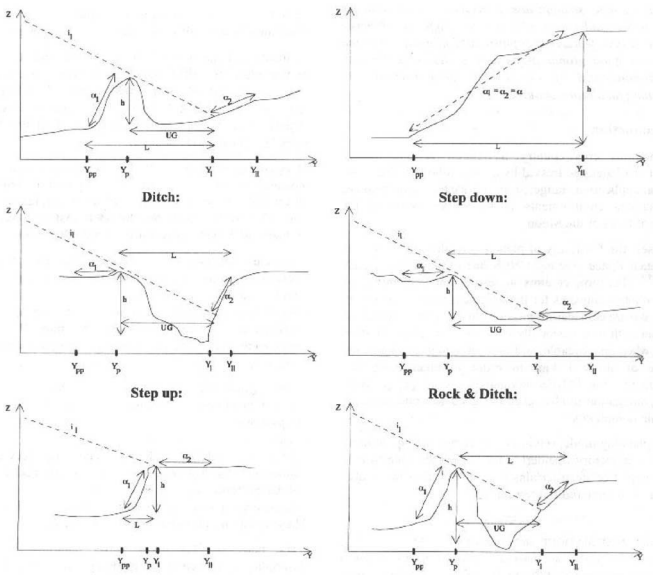
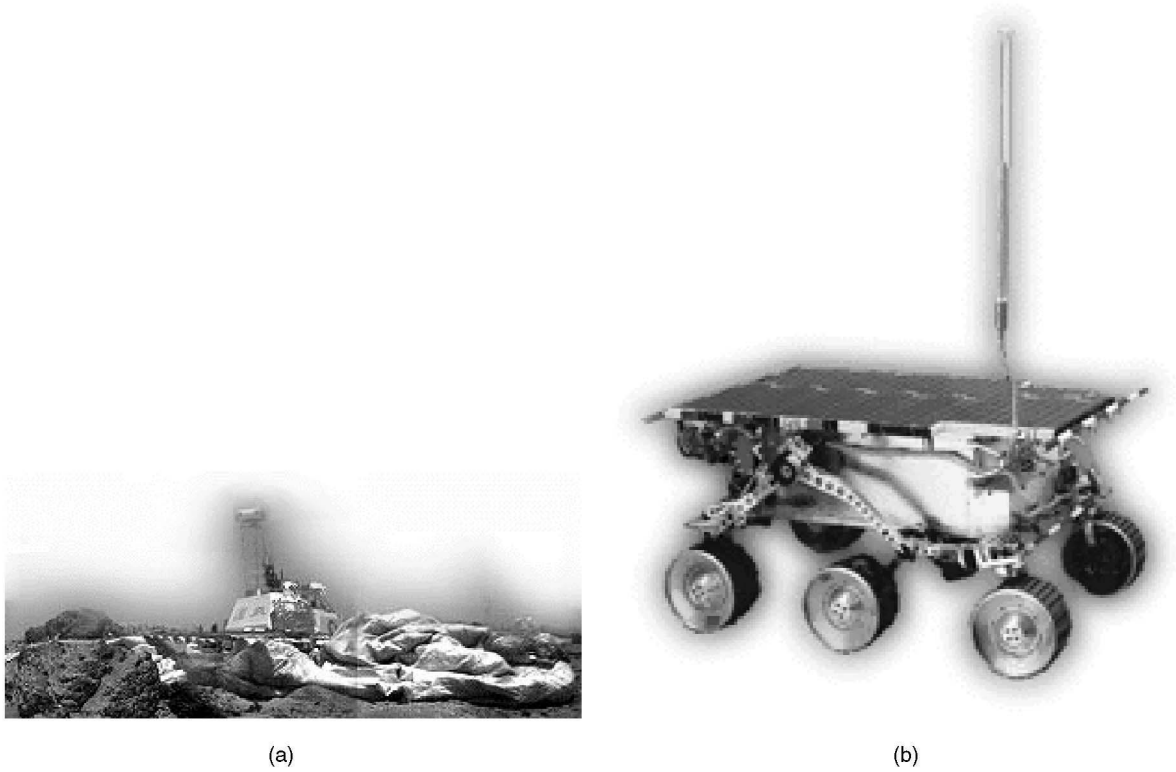


Fig. 38. Characterization of terrains in [120].

Fig. 39. Martian terrain. This can be found at the Jet Propulsion Laboratory (JPL) Web site http://mpfwww.jpl.nasa.gov/MPF/index0.html.



(a) (b)

Fig. 40. (a) Pathfinder lander and (b) rover. This can be found at the JPL Web site http://mpfwww.jpl.nasa.gov/MPF/index0.html.

project [158], [99], [101]. The Mars Pathfinder mission was launched in December 1996 and landed in July 1997. The mission was performed by two devices: the lander and the rover (Fig. 40). The lander is a 1.5 m wide and 1.0 m high tetrahedron with a mass of 264 kg. It is composed of a VME-based, RAD 6000 computer with 128Mb of memory, a multispectral stereo camera pair with a pan/tilt base mounted on a mast 1.5 m above the ground. The cameras have a baseline of 15 cm and resolution of 256x256 pilxels with an angular resolution of 0.001 radians per pixel. The rover, Fig. 40b, is 65 cm long, 48 cm wide, and 30 cm high. It is equipped with an Intel 8085 processor and 500Kb of memory. The navigation by the rover, carried out in cooperation with the lander, consists of four different functions:

1. goal designation,
2. path selection,
3. rover localization, and
4. hazard detection.

In goal designation, human operators at the mission control used a Web-based tool [8] to specify waypoints in 3D views of the landing site that were generated from images obtained using the stereo cameras on the lander. This required that a map of the landing site, centered at the lander, be maintained by the mission control and the images recorded by the lander be used for tracking and locating the rover (Fig. 41). Waypoint coordinates were extracted from the 3D views by pointing to pixels in a stereographic display using a special device called the spaceball. Once the 3D coordinates were extracted, the rover could be commanded to move towards those coordinates using a simple path selection algorithm:

IF there is no hazard,
    move forward and turn toward the goal,
ELSE IF there is a hazard on the left,
    turn in place to the right until no hazard is detected;
ELSE IF there is a hazard on the right,
    turn in place to the left until no hazard is detected.

The mission-controlled supplied localization of the rover was transmitted to the lander once a day. During the rest of the day, the rover would be on its own, using deadreckoning to figure out its position. Because the deadreckoning-based positional errors tend to accumulate, it was determined by simulation of typical navigation scenarios that the rover would be allowed to travel at most 10 m/day [99]. Once the rover was commanded to move, a hazard detection algorithm was started. The rover moved at speeds of 15 cm/s and stopped for hazard detection every 6.5 cm (one wheel radius). The hazard detection algorithm used two cameras and five laser diode-based light stripe emitters (Fig. 42). The stripes extended in front of the vehicle for about 30 cm and to the left or right for 13 cm. The hazard detection algorithm searched for four points along each stripe and assumed the presence of a hazard if: 1) either of the nearest two points for any stripe was not detected or 2) the elevation difference between any two 8-connected neighbors in the 4x5 measurement array exceeded a threshold or 3) the difference between the highest and the lowest elevation of the whole area exceeded another threshold. The shortcomings of the rover navigation system, especially those related to localization,

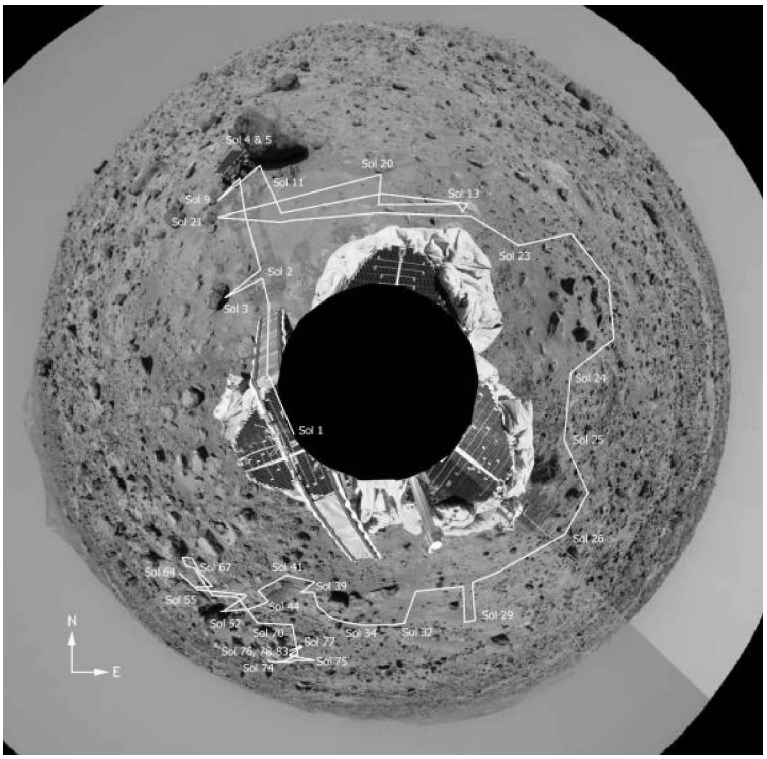


Fig. 41. Tracked positions of the rover. This can be found at the JPL Web site http://mpfwww.jpl.nasa.gov/MPF/index0.html.

are expected to be alleviated in future missions by using new approaches, such as using the maximum-likelihood approach for matching range maps [118].

3.3 Illumination and Outdoor Navigation

A common problem in all outdoor navigation systems is that caused by variations in illumination. The differences in contrast and texture in an image as a function of the time of the day, time of the year, weather conditions, etc., can impose serious limitations on the performance of a navigation system. The use of color to compensate for these differences in illumination conditions has been explored in many outdoor navigation systems. Some of the earliest systems to use color to distinguish shadows from obstacles are by Thorpe et al. [143] and Turk et al. [151]. Another more recent example of this is in [91], where the authors have addressed the problem of vision-guided obstacle avoidance using three redundant vision modules, one for intensity (BW), one for RGB, and one for HSV. The goal is to figure out the position of obstacles in the scene while the robot wanders and explores the environment. Each module output (image) is scanned using a small subwindow (20x10) that is shifted vertically pixel by pixel. That process defines a vertical slice of the original image that is represented (and indexed) by its central value of x. Inside this slice, for each window is computed a histogram of its intensity values, red values, green values, etc., depending on the module (RGB, HSV, or BW). The areas of these histograms are subtracted from the area of the histogram for the "safe window" (lowest window in the slice, which is assumed to have no obstacles Fig. 43). When the difference is above a certain threshold, the height of the window (y-coordinate) is used to estimate the height of the object in the scene. Each module then outputs a value for the height of the obstacle. Those outputs are averaged to produce steering and acceleration commands for the robot. One advantage of the system is its relative adaptability to different environmental conditions. This adaptability is gained by using the "safe window" as a basis for analyzing other windows for the presence of obstacles.

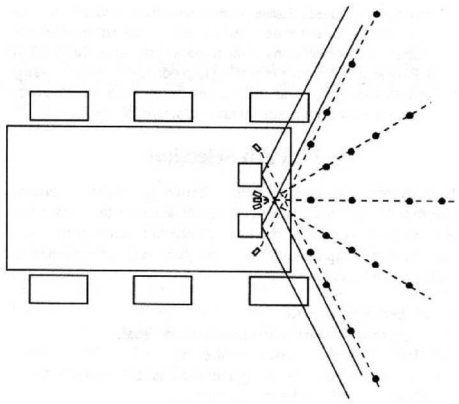


Fig. 42. Rover's hazard sensors: camera fields of view in solid lines and light stripes in dotted lines. Reprinted with permission from [99].

Another system that uses color to overcome lighting problems has been developed by Mori et al. [110]. In a manner similar to [151], Mori uses what is referred to as the color impression factor—Red minus Blue—to compensate for hue shifts due to variations in sunlight conditions, weather, season, view position, etc. However, instead of storing models of "sunny road" and "shaded road," as in Navlab and VITS [143], [151] (which are useful only when the robot is in completely sunny or completely shady

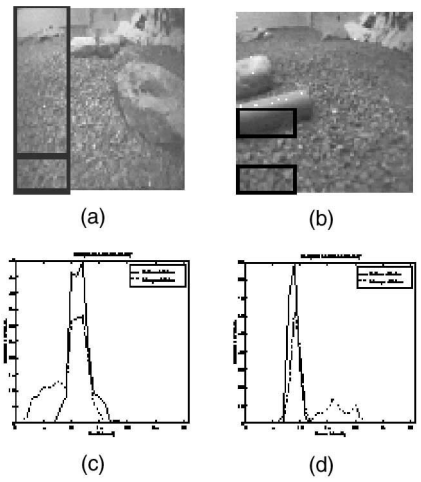


Fig. 43. Vision-guided obstacle avoidance in [91]. (a) The camera image and the superimposed vertical slice with the safe window at the bottom. (b) Highlights a horizontal slice and the safe window. (c) and (d) Shows the histogram of the areas highlighted in (b).

regions), the system presented by Mori uses three windows positioned in the image to correspond to three regions ahead of the robot: far (4 meters), middle (3 meters), and near (2 meters). The windows are analyzed and the computed color impression factor is used to facilitate transitions from/to sunny to/from shaded regions.

We should also mention the work of Fernández and Casals [41]. They set the value of a pixel to H (hue) unless the average value of the RGB components is less than 15 or their differences are less than 10. In those cases, the pixel is represented by its intensity value. The image thus represented is divided into six regions that describe the expected-structure of the road, berms, etc. The obstacles within the "road region" are detected using a simplified form of the structure-from-motion approach, which the authors called height-from-motion. The authors claim that their new H/I (Hue/Intensity) representation is less sensitive to illumination conditions than the normal RGB representation.

4 The Future

As our survey shows, much has already been accomplished in computer vision for mobile robot navigation. If the goal is to send a mobile robot from one coordinate location to another coordinate location, we believe there is sufficient accumulated expertise in the research community today to design a mobile robot that could do that in a typical building (barring for electromechanical abilities needed for tasks such as opening doors, calling elevators, climbing steps, etc.).

But, if the goal is to carry out function-driven navigation —an example being to fetch a fire-extinguisher that is somewhere in a given hallway or to stop at a stop sign under varying illumination and background conditions— we are still eons away. Useful navigation where a robot must be aware of the meaning of the objects encountered in the environment is beset with a harder-to-solve version of the central problem of general computer vision—automatic scene interpretation. Scene interpretation for mobile robots is a harder problem than for stationary arm robots because, in the mobile context, there is much less control over illumination and background scene clutter.

It's probably safe to predict that the near future will witness progress in task and environment specific mobile robots. We should see vision-equipped versions of robots that have already been deployed on experimental basis in patient-care, building security, hazardous-site inspection, etc. With vision, such robots would conceivably engage in smarter navigation and smarter interaction with people and objects in the environment.

Progress will also surely be made, on the one hand, in more efficient ways of representing the metrical and topological properties of the environment and, on the other, in more efficient ways of representing the uncertainties in a robot's knowledge of its environment and its own position relative to the environment. While the incorporation of uncertainties in purely geometrical representations is well understood from related problem domains, less well understood today is the incorporation of uncertainties in representations that are primarily topological but contain locally metric attributes. What is also not well understood today is how precisely a robot should take into account the various uncertainties when it makes navigational or task-oriented decisions. Recent work on partially observable Markov decision processes (POMDP) that allows a robot to maintain a probability distribution over all possible states and to update this distribution using perceptual data gives us a powerful formalism for dealing with this issue [67]. The next few years will surely shed more light on the power of this approach for mobile robots engaged in nontrivial navigational tasks.

5 Conclusions

Compared to 20 years ago, we have a much better sense today of the problems we face as we try to endow mobile robots with sensory intelligence. We are much more cognizant of the importance of prior knowledge and the various forms in which it can be modeled. The models can be geometrical, topological, appearance-based, etc. Different representation schemes vary with respect to the degree of metrical knowledge contained, exhibiting different degrees of robustness with regard to illumination variations, etc. This paper has surveyed these various aspects of the progress made so far in vision for mobile robot navigation.

References

[1] K.M. Andress and A.C. Kak, "Evidence Accumulation and Flow of Control in a Hierarchical Spatial Reasoning System," *AI Magazine,* vol. 9, no. 2, pp. 75-95, 1988.

[2] R.C. Arkin, "Motor Schema-Based Mobile Robot Navigation: An Approach to Programming by Behavior," *Proc. 1987 IEEE Int'l Conf. Robotics and Automation,* pp. 264-271, 1987.

[3] R.C. Arkin, "Motor Schema-Based Mobile Robot Navigation," *Int'l J. Robotics Research,* vol. 8, no. 4, pp. 92-112, 1989.

[4] R. C. Arkin, "Temporal Coordination of Perceptual Algorithms for Mobile Robot Navigation," *IEEE Trans. Robotics and Automation,* vol. 10, no. 3, pp. 276-286, June 1994.

[5] S. Atiya and G.D. Hager, "Real-Time Vision-Based Robot Localization," *IEEE Trans. Robotics and Automation,* vol. 9, no. 6, pp. 785-800, Dec. 1993.

[6] *Artifical Vision for Mobile Robots: Stereo Vision and Multisensory Perception,* N. Ayache and P.T. Sander, eds. MIT Press, 1991.

[7] N. Ayache and O. D. Faugeras, "Maintaining Representations of the Environment of a Mobile Robot," *IEEE Trans. Robotics and Automation,* vol. 5, no. 6, pp. 804-819, 1989.

[8] P. Backes, K. Tso, and K. Tharp, "Mars Pathfinder Mission Internet-Based Operations Using WITS," *Proc. 1998 IEEE Int'l Conf. Robotics and Automation,* vol. 1, pp. 284-291, May 1998.

[9] C. Balkenius, "Spatial Learning with Perceptually Grounded Representations," *Robotics and Autonomous Systems,* vol. 5, no. 3-4, pp. 165-175, Nov. 1998.

[10] A. Bernardino and J. Santos-Victor, "Visual Behaviours for Binocular Tracking," *Robotics and Autonomous Systems,* vol. 25, no. 3-4, pp 137-146, Nov. 1998.

[11] *Active Vision,* A. Blake and A.Yuille, eds. MIT Press, 1992.

[12] L. Boissier, B. Hotz, C. Proy, O. Faugeras, and P. Fua, "Autonomous Planetary Rover: On-Board Perception System Concept and Stereovision by Correlation Approach," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 181-186, May 1992.

[13] *Navigating Mobile Robots: Systems and Techniques,* J. Borenstein, H.R. Everett, and L. Feng, eds. Wellesley, Mass.: AK Peters, 1996.

[14] J. Borenstein and Y. Koren, "Real-Time Obstacle Avoidance for Fast Mobile Robots," *IEEE Trans. Systems, Man, and Cybernetics,* vol. 19, no. 5, pp. 1179-1187, 1989.

[15] J. Borenstein and Y. Koren, "Real-Time Obstacle Avoidance for Fast Mobile Robots in Cluttered Environments," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 572-577, 1990.

[16] J. Borenstein and Y. Koren, "The Vector Field Histogram–Fast Obstacle Avoidance for Mobile Robots," *IEEE Trans. Robotics and Automation,* vol. 7, no. 3, pp. 278-288, June 1991.

[17] D.J. Braunegg, "Marvel: A System that Recognizes World Locations with Stereo Vision," *IEEE Trans. Robotics and Automation,* vol. 9, no. 3, pp. 303-308, June 1993.

[18] R.A. Brooks, "Visual Map Making for a Mobile Robot," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 824-829, 1985.

[19] R.A. Brooks, "A Robust Layered Control System for a Mobile Robot," *IEEE J. Robotics and Automation,* vol. 2, no. 1, pp. 14-23, 1986.

[20] R. Chatila and J.-P. Laumond, "Position Referencing and Consistent World Modeling for Mobile Robots," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 138-145, Mar. 1985.

[21] M. Chen, T.M. Jochem, and D.A. Pomerleau, "AURORA: A Vision-Based Roadway Departure Warning System," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* vol. 1, pp. 243-248, Aug. 1995.

[22] H. Choset, I. Konukseven, and A. Rizzi, "Sensor Based Planning: A Control Law for Generating the Generalized Voronoi Graph," *Proc. Eighth IEEE Int'l Conf. Advanced Robotics,* pp. 333-8, 1997.

[23] E. Chown, S. Kaplan, and D. Kortenkamp, "Prototypes, Location, and Associative Networks (PLAN): Towards a Unified Theory of Cognitive Mapping," *Cognitive Science,* vol. 19, no. 1, pp. 1-51, Jan. 1995.

[24] H.I. Christensen, N.O. Kirkeby, S. Kristensen, and L. Knudsen, "Model-Driven Vision for In-Door Navigation," *Robotics and Autonomous Systems,* vol. 12, pp. 199-207, 1994.

[25] I.J. Cox, "Blanche: Position Estimation for an Sutonomous Robot Vehicle," *Proc. IEEE/RSJ Int'l Workshop Robots and Systems,* pp. 432-439, 1989.

[26] I.J. Cox, "Blanche: An Experiment in Guidance and Navigation of an Autonomous Robot Vehicle," *IEEE Trans. Robotics and Automation,* vol. 7, no. 2, pp. 193-204, Apr. 1991.

[27] I.J. Cox, "Modeling a Dynamic Environment Using a Bayesian Multiple Hypothesis Approach," *Artificial Intelligence,* vol. 66, no. 2, pp. 311-344, Apr. 1994.

[28] *Autonomous Robot Vehicles,* I.J. Coxand G.T. Wilfong, eds. Springer-Verlag, 1990.

[29] F. Cozman and E. Krotkov, "Robot Localization Using a Computer Vision Sextant," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 106-111, May 1995.

[30] M. Daily, J. Harris, D. Keirsey, K. Olim, D. Payton, K. Reiser, J. Rosenblatt, D. Tseng, and V. Wong, "Autonomous Cross-Country Navigation with the ALV," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 718-726, Apr. 1988.

[31] L. Delahoche, C. Pégard, B. Marhic, and P. Vasseur, "A Navigation System Based on an Omnidirectional Vision Sensor," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* pp. 718-724, Sept. 1997.

[32] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte Carlo Localization for Mobile Robots," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 1322-28, May 1999.

[33] A. Dev, B. Kröse, and F. Groen, "Navigation of a Mobile Robot on the Temporal Development of the Optic Flow," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* pp. 558-563, Sept. 1997.

[34] E.D. Dickmanns, "Computer Vision and Highway Automation," *Vehicle System Dynamics,* vol. 31, no. 5, pp. 325-343, 1999.

[35] E.D. Dickmanns and B. Mysliwetz, "Recursive 3D Road and Relative Egostate Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 14, no. 2 Feb. 1992.

[36] E.D. Dickmanns and A. Zapp, "A Curvature-Based Scheme for Improving Road Vehicle Guidance by Computer Vision," *Proc. SPIE-The Int'l Soc. Optical Eng.-Mobile Robots,* vol. 727, pp. 161-168, Oct. 1986.

[37] E.D. Dickmanns and A. Zapp, "Autonomous High Speed Road Vehicle Guidance by Computer Vision," *Proc. 10th World Congress on Automatic Control,* vol. 4, 1987.

[38] R.T. Dunlay and D. G. Morgenthaler, "Obstacle Avoidance on Roadways Using Range Data," *Proc. SPIE-The Int'l Soc. Optical Eng.-Mobile Robots,* vol. 727, pp. 110-116, Oct. 1986.

[39] S. Engelson and D. McDermott, "Error Correction in Mobile Robot Map Learning," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 2555-2560, 1992.

[40] O. Faugeras, *Three-Dimensional Computer Vision: A Geometric Viewpoint.* MIT Press, 1993.

[41] J. Fernández and A. Casals, "Autonomous Navigation in Ill-Structured Outdoor Environments," *Proc. 1997 IEEE Int'l Conf. Intelligent Robots and Systems,* pp. 395-400, Sept. 1997.

[42] J. Ferruz and A. Ollero, "Autonomous Mobile Robot Motion Control in Non-Structured Environments Based on Real-Time Video Processing," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* pp. 725-731, Sept. 1997.

[43] P. Gaussier, C. Joulain, S. Zrehen, and A. Revel, "Visual Navigation in an Open Environment without Map," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* pp. 545-550, Sept. 1997.

[44] G. Giralt, R. Sobek, and R. Chatila, "A Multi-Level Planning and Navigation System for a Mobile Robot; A First Approach to Hilare," *Proc. Sixth Int'l Joint Conf. Artificial Intelligence,* vol. 1, pp. 335-337, 1979.

[45] Y. Goto and A. Stentz, "The CMU System for Mobile Robot Navigation," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 99-105, Mar/Apr. 1987.

[46] V. Graefe, "Dynamic Vision System for Autonomous Mobile Robots," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* pp. 12-23, 1989.

[47] V. Graefe, "Vision for Autonomous Mobile Robots," *Proc. IEEE Workshop Advanced Motion Control,* pp. 57-64, 1992.

[48] V. Graefe, "Driverless Highway Vehicles," *Proc. Int'l Hi-Tech Forum,* pp. 86-95, 1992.

[49] V. Graefe, "Vision for Intelligent Road Vehicles," *Proc. IEEE Symp. Intelligent Vehicles,* pp. 135-140, 1993.

[50] V. Graefe and K. Kuhnert, "Vision-based Autonomous Road Vehicles," *Vision-Based Vehicle Guidance,* I. Masaki, ed. Springer-Verlag, pp. 1-29, 1992.

[51] *Visual Servoing,* K. Hashimoto, ed. World Scientific, 1993.

[52] M. Hashima, F. Hasegawa, S. Kanda, T. Maruyama, and T. Uchiyama, "Localization and Obstacle Detection for a Robot for Carrying Food Trays," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* pp. 345-351, Sept. 1997.

[53] J. Horn and G. Schmidt, "Continuous Localization for Long-Range Indoor Navigation of Mobile Robots," *Proc. IEEE Int'l Conf. Robotics and Automation,* vol. 1, pp. 387-394, May 1995.

[54] J. Horn and G. Schmidt, "Continuous Localization of a Mobile Robot Based on 3D-Laser-Range-Data, Predicted Sensor Images, and Dead-Reckoning," *Robotics and Autonomous Systems,* vol. 14, no. 2-3, pp. 99-118, May 1995.

[55] I. Horswill, "Polly: Vision-Based Artificial Agent," *Proc. Int'l Conf. AAAI,* pp. 824-829, 1993.

[56] E. Huber and D. Kortenkamp, "Using Stereo Vision to Pursue Moving Agent with a Mobile Robot," *Proc. IEEE Int'l Conf. Robotics and Automation,* vol. 3, pp. 2340-2346, May 1995.

[57] Y.K. Hwang and N. Ahuja, "Gross Motion Planning—A Survey," *ACM Computing Survey,* vol. 24, no. 3, pp. 219-291, 1992.

[58] Y. K. Hwang and N. Ahuja, "A Potential Field Approach to Path Planning," *IEEE Trans. Robotics and Automation,* vol. 8, no. 1, pp. 23-32, 1992.

[59] M. Isard and A. Blake, "Contour Tracking by Stochastic Propagation of Conditional Density," *Proc. Fourth European Conf. Computer Vision (ECCV),* vol. 1, pp. 343-356, Apr. 1996.

[60] M. Isard and A. Blake, "Condensation—Conditional Density Propagation for Visual Tracking," *Int'l J. Computer Vision,* vol. 29, no. 1, pp. 5-28, 1998.

[61] T.M. Jochem, "Using Virtual Active Vision Tools to Improve Autonomous Driving Tasks," Technical Report CMU-RI-TR-94-39, Robotics Inst., Carnegie Mellon Univ., Oct. 1994.

[62] T.M. Jochem, D.A. Pomerleau, and C.E. Thorpe, "Vision Guided Lane Transition," *Proc. IEEE Symp. Intelligent Vehicles,* Sept. 1995.

[63] T.M. Jochem, D.A. Pomerleau, and C.E. Thorpe, "Vision-Based Neural Network Road and Intersection Detection and Traversal," *Proc. IEEE Conf. Intelligent Robots and Systems,* vol. 3, pp. 344-349, Aug. 1995.

[64] S.D. Jones, C. Andresen, and J.L. Crowley, "Appearance Based Processes for Visual Navigation," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* pp. 551-557, Sept. 1997.

[65] C. Joulian, P. Gaussier, A. Revel, and B. Gas, "Learning to Build Visual Categories from Perception-Action Associations," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* pp. 857-864, Sept. 1997.

[66] M.R. Kabuka and A.E. Arenas, "Position Verification of a Mobile Robot Using Standard Pattern," *IEEE J. Robotics and Automation,* vol. 3, no. 6, pp. 505-516, Dec. 1987.

[67] L.P. Kaelbling, M.L. Littman, and A.R. Cassandra, "Planning and Acting in Partially Observable Stochastic Domains," *Artificial Intelligence,* vol. 101, no. 1-2, pp. 99-134, May 1998.

[68] H. Kamada and M. Yoshida, "A Visual Control System Using Image Processing and Fuzzy Theory," *Vision-Based Vehicle Guidance,* I. Masaki, ed. pp. 111-128, Springer Verlag, 1991.

[69] A.C. Kak, K.M. Andress, C. Lopez-Abadia, M. S. Carroll, and J.R. Lewis, "Hierarchical Evidence Accumulation in the PSEIKI System and Experiments in Model-Driven Mobile Robot Navigation," *Uncertainty in Artificial Intelligence,* M. Henrion, R. Shachter, L.N. Kanal, and J. Lemmer, eds. pp. 353-369, Elsevier, 1990.

[70] A. Kelly and A. Stentz, "Minimum throughput Adaptive Perception for High Speed Mobility," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* pp. 215-223, Sept. 1997.

[71] D. Kim and R. Nevatia, "Representation and Computation of the Spatial Environment for Indoor Navigation," *Proc. Int'l Conf. Computer Vision and Pattern Recognition,* pp. 475-482, 1994.

[72] D. Kim and R. Nevatia, "Symbolic Navigation with a Generic Map," *Proc. IEEE Workshop Vision for Robots,* pp. 136-145, Aug. 1995.

[73] D. Kim and R. Nevatia, "Recognition and Localization of Generic Objects for Indoor Navigation Using Functionality," *Image and Vision Computing,* vol. 16, no. 11, pp. 729-743, Aug. 1998.

[74] D. Kortenkamp and T. Weymouth, "Topolgical Mapping for Mobile Robots Using a Combination of Sonar and Vision Sensing," *Proc. 12th Nat'l Conf. Artificial Intelligence,* vol. 2, pp. 979-984, 1994.

[75] A. Kosaka and A.C. Kak, "Fast Vision-Guided Mobile Robot Navigation Using Model-Based Reasoning and Prediction of Uncertainties," *Computer Vision, Graphics, and Image Processing—Image Understanding,* vol. 56, no. 3, pp. 271-329, 1992.

[76] A. Kosaka, M. Meng, and A.C. Kak, "Vision-Guided Mobile Robot Navigation Using Retroactive Updating of Position Uncertainty," *Proc. IEEE Int'l Conf. Robotics and Automation,* vol. 2, pp. 1-7, 1993.

[77] A. Kosaka and G. Nakazawa, "Vision-Based Motion Tracking Using Prediction of Uncertainties," *Proc. IEEE Int'l Conf. Robotics and Automation,* vol. 3, pp. 2637-2644, May 1995.

[78] D.J. Kriegman and T.O. Binford, "Generic Models for Robot Navigation," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 746-751, 1988.

[79] D.J. Kriegman, E. Triendl, and T.O. Binford, "Stereo Vision and Navigation in Buildings for Mobile Robots," *IEEE Trans. Robotics and Automation,* vol. 5, no. 6, pp. 792-803, 1989.

[80] E. Krotkov, "Mobile Robot Localization Using a Single Image," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 978-983, 1989.

[81] E. Krotkov and M. Hebert, "Mapping and Positioning for a Prototype Lunar Rover," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 2913-2919, May 1995.

[82] E. Krotkov and R. Hoffman, "Terrain Mapping for a Walking Planetary Rover," *IEEE Trans. Robotics and Automation,* vol. 10, no. 6, pp. 728-739, Dec. 1994.

[83] D. Kuan, G. Phipps, and A. Hsueh, "A Real-Time Road Following Vision System," *Proc. SPIE Int'l Soc. Optical Eng.-Mobile Robots,* vol. 727, pp. 152-160, Oct. 1986.

[84] D. Kuan and U.K. Sharma, "Model Based Geometric Reasoning for Autonomous Road Following," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 416-423, Mar/Apr. 1987.

[85] K. Kuhnert, "Towards the Objective Evaluation of Low-Level Vision Operators," *Proc. Sixth European Conf. Artificial Intelligence,* pp. 657, 1984.

[86] K. Kuhnert, "A Vision System for Real Time Road and Object Recognition for Vehicle Guidance," *Proc. SPIE-The Int'l Soc. Optical Eng.-Mobile Robots,* vol. 727, pp. 267-272, Oct. 1986.

[87] B. J. Kuipers and Y.-T. Byun, "A Robot Exploration and Mapping Strategy Based on a Semantic Hierarchy of Spatial Representations," *J. Robotics and Autonomous Systems,* vol. 8, pp. 47-63, 1991.

[88] D.T. Lawton, T.S. Levitt, C. McConnell, and J. Glicksman, "Terrain Models for an Autonomous Land Vehicle," *Proc. 1986 IEEE Int'l Conf. Robotics and Automation,* pp. 2043-2051, Apr. 1986.

[89] X. Lebègue and J.K. Aggarwal, "Significant Line Segments for an Indoor Mobile Robot," *IEEE Trans. Robotics and Automation,* vol. 9, no. 6, pp. 801-815, Dec. 1993.

[90] J. Leonard and H. Durrant-Whyte, "Mobile Robot Localization by Tracking Geometric Beacons," *IEEE Trans. Robotics and Automation,* vol. 7, no. 3, pp. 89-97, June 1991.

[91] L.M. Lorigo, R.A. Brooks, and W.E. Grimson, "Visually-Guided Obstacle Avoidance in Unstructured Environments," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* pp. 373-379, Sept. 1997.

[92] T. Lozano-Perez and M.A. Wesley, "An Algorithm for Planning Collision-Free Paths among Polyhedral Obstacles," *Comm. ACM,* vol. 22, no. 10, pp. 560-570, 1979.

[93] M.J. Magee and J.K. Aggarwal, "Determining the Position of a Robot Using a Single CalibrationObject," *Proc. Int'l Conf. Robotics,* pp. 140-149, 1984.

[94] A. Martínez and J. Vitrià, "Clustering in Image Space for Place Recognition and Visual Annotations for Human-Robot Interaction," *IEEE Trans. Systems, Man, and Cybernetics B,* vol. 31, no. 5, Oct. 2001.

[95] *Vision-based Vehicle Guidance,* I. Masaki, ed. Springer-Verlag, 1991.

[96] *Proc. Int'l Symp. "Intelligent Vehicles."* I. Masaki, ed., yearly since 1992.

[97] M.J. Mataric, "A Distributed Model for Mobile Robot Environment Learning and Navigation," Technical Report AITR-1228, Massachusetts Inst. of Technology AI Lab, 1990.

[98] Y. Matsumoto, M. Inaba, and H. Inoue, "Visual Navigation Using View-Sequenced Route Representation," *Proc. IEEE Int'l Conf. Robotics and Automation,* vol. 1, pp. 83-88, Apr. 1996.

[99] L. Matthies, E. Gat, R. Harrison, B. Wilcox, R. Volpe, and T. Litwin, "Mars Microrover Navigation: Performance Evaluation and Enhancement," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* vol. 1, pp. 433-440, Aug. 1995.

[100] L. Matthies and S.A. Shafer, "Error Modeling in Stereo Vision," *IEEE J. Robotics and Automation,* vol. 3, no. 3, pp. 239-248, 1987.

[101] L. Matthies, T. Balch, and B. Wilcox, "Fast Optical Hazard Detection for Planetary Rovers Using Multiple Spot Laser Triangulation," *Proc. IEEE Int'l Conf. Robotics and Automation,* vol. 1 pp. 859-866, Apr. 1997.

[102] M. Meng and A.C. Kak, "NEURO-NAV: A Neural Network Based Architecture for Vision-Guided Mobile Robot Navigation Using Non-Metrical Models of the Environment," *Proc. IEEE Int'l Conf. Robotics and Automation,* vol. 2, pp. 750-757, 1993.

[103] M. Meng and A.C. Kak, "Mobile Robot Navigation Using Neural Networks and Nonmetrical Environment Models," *IEEE Control Systems,* pp. 30-39, Oct. 1993.

[104] J. Miura and Y. Shirai, "Hierachical Vision-Motion Planning with Uncertainty: Local Path Planning and Global Route Selection," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems,* July 1992.

[105] J. Miura and Y. Shirai, "Vision-Motion Planning with Uncertainty," *Proc. IEEE Int'l Conf. Robotics and Automation,* vol. 2, pp. 1772-1777, May 1992.

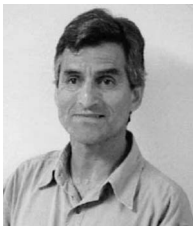
[106] J. Miura and Y. Shirai, "An Uncertainty Model of Stereo Vision and Its Application to Vision-Motion Planning of Robots," *Proc. 13th Int'l Joint Conf. Artificial Intelligence,* Aug. 1993.

[107] H.P. Moravec, "Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover," PhD thesis, Stanford Univ., Sept. 1980. (published as *Robot Rover Visual Navigation.* Ann Arbor, MI: UMI Research Press, 1981.)

[108] H.P. Moravec, "The Stanford Cart and the CMU Rover," *Proc. IEEE*, vol. 71, no. 7, pp. 872-884, July 1983.
[109] H.P. Moravec and A. Elfes, "High Resolution Maps from Wide Angle Sonar," *Proc. IEEE Int'l Conf. Robotics and Automation*, pp. 116-121, 1985.
[110] H. Mori, K. Kobayashi, N. Ohtuki, and S. Kotani, "Color Impression Factor: An Image Understanding Method for Outdoor Mobile Robots," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems*, pp. 380-386, Sept. 1997.
[111] T. Nakamura and M. Asada, "Motion Sketch: Acquisition of Visual Motion Guided Behaviors," *Proc. 14th Int'l Joint Conf. Artificial Intelligence*, vol. 1, pp. 126-132, Aug. 1995.
[112] T. Nakamura and M. Asada, "Stereo Sketch: Stereo Vision-Based Target Reaching Behavior Acquisition with Occlusion Detection and Avoidance," *Proc. IEEE Int'l Conf. Robotics and Automation*, vol. 2, pp. 1314-1319, Apr. 1996.
[113] S. Negahdaripour, B. Hayashi, and Y. Aloimonos, "Direct Motion Stereo for Passive Navigation," *IEEE Trans. Robotics and Automation*, vol. 11, no. 6, pp. 829-843, Dec. 1995.
[114] N.J. Nilsson, "Shakey the Robot," Technical Report 323, SRI Int'l, Apr. 1984.
[115] T. Ohno, A. Ohya, and S. Yuta, "Autonomous Navigation for Mobile Robots Referring Pre-Recorded Image Sequence," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems*, vol. 2, pp. 672-679, Nov. 1996.
[116] A. Ohya, A. Kosaka, and A. Kak, "Vision-Based Navigation of Mobile Robot with Obstacle Avoidance by Single Camera Vision and Ultrasonic Sensing," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems*, pp. 704-711, Sept. 1997.
[117] A. Ohya, A. Kosaka, and A. Kak, "Vision-Based Navigation by Mobile Robots with Obstacle Avoidance Using Single-Camera Vision and Ultrasonic Sensing," *IEEE Trans. Robotics and Automation*, vol. 14, no. 6, pp. 969-978, Dec. 1998.
[118] C. Olson and L. Matthies, "Maximum Likelihood Rover Localization by Matching Range Maps," *Proc. IEEE Int'l Conf. Robotics and Automation*, vol. 1 pp. 272-277, May 1998.
[119] G. Oriolo, G. Ulivi, and M. Vendittelli, "On-Line Map Building and Navigation for Autonomous Mobile Robots," *Proc. IEEE Int'l Conf. Robotics and Automation*, pp. 2900-2906, May 1995.
[120] R. Pagnot and P. Grandjean, "Fast Cross Country Navigation on Fair Terrains," *Proc. IEEE Int'l Conf. Robotics and Automation*, pp. 2593-2598, May 1995.
[121] J. Pan, D.J. Pack, A. Kosaka, and A.C. Kak, "FUZZY-NAV: A Vision-Based Robot Navigation Architecture Using Fuzzy Inference for Uncertainty-Reasoning," *Proc. IEEE World Congress Neural Networks*, vol. 2, pp. 602-607, July 1995.
[122] J. Pan, G.N. DeSouza, and A.C. Kak, "Fuzzy Shell: A Large-Scale Expert System Shell Using Fuzzy Logic for Uncertainty Reasoning," *IEEE Trans. Fuzzy Systems*, vol. 6, no. 4, pp. 563-581, Nov. 1998.
[123] D. Pierce and B. Kuipers, "Learning to Explore and Build Maps," *Proc. 12th Nat'l Conf. Artificial Intelligence*, vol. 2, pp. 1264-1271, 1994.
[124] D.A. Pomerleau, "ALVINN: An Autonomous Land Vehicle in a Neural Network," Technical Report CMU-CS-89-107, Carnegie Mellon Univ., 1989.
[125] D.A. Pomerleau, "Efficient Training of Arificial Neural Networks for Autonomous Navigation," *Neural Computation*, vol. 3, pp. 88-97, 1991.
[126] D.A. Pomerleau, "Reliability Estimation for Neural Network Based Autonomous Driving," *Robotics and Autonomous Systems*, vol. 12, pp. 113-119, 1994.
[127] D.A. Pomerleau, "Neural Network Vision for Robot Driving," *The Handbook of Brain Theory and Neural Networks*, M. Arbib, ed. 1995.
[128] D.A. Pomerleau and T. Jochem, "Rapidly Adapting Machine Vision for Automated Vehicle Steering," *IEEE Expert, Intelligent Systems and Their Applications*, vol. 11, no. 2, Apr. 1996.
[129] U. Regensburger and V. Graefe, "Visual Recognition of Obstacles on Roads," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems*, pp. 980-987, 1994.
[130] P. Rives and J. Borrelly, "Underwater Pipe Inspection Task Using Visual Servoing Techniques," *Proc. IEEE Int'l Conf. Intelligent Robots and Systems*, pp. 63-68, Sept. 1997.
[131] A. Rizzi, G. Bianco, and R. Cassinis, "A Bee-Inpired Visual Homing Using Color Images," *Robotics and Autonomous Systems*, vol. 25, no. 3-4, pp. 159-164, Nov. 1998.
[132] N. Sawasaki, T. Morita, and T. Uchiyama, "Design and Implementation of High-Speed Visual Tracking System for Real-Time Motion Analysis," *Proc. IAPR Int'l Conf. Pattern Recognition*, vol. 3, pp. 478-483, 1996.
[133] J. Santos-Victor, G. Sandini, F. Curotto, and S. Garibaldi, "Divergent Stereo for Robot Navigation: Learning from Bees," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition*, 1993.
[134] J. Santos-Victor, G. Sandini, F. Curotto, and S. Garibaldi, "Divergent Stereo in Autonomous Navigation: from Bees to Robots," *Int'l J. Computer Vision*, vol. 14, no. 2, pp. 159-177, Mar. 1995.
[135] R. Schuster, N. Ansari, and A. Bani-Hashemi, "Steering a Robot with Vanishing Points," *IEEE Trans. Robotics and Automation*, vol. 9, no. 4, pp. 491-498, Aug. 1993.
[136] R. Simmons, E. Krotkov, L. Chrisman, F. Cozman, R. Goodwin, M. Hebert, L. Katragadda, S. Koenig, G. Krishnaswamy, Y. Shinoda, W. Whittaker, and P. Klader, "Experience with Rover Navigation for Lunar-Like Terrains," *Proc. 1995 IEEE Int'l Conf. Intelligent Robots and Systems*, pp. 441-446, Aug. 1995.
[137] R. Simmons and S. Koenig, "Probabilistic Robot Navigation in Partially Observable Enironments," *Proc. Int'l Joint Conf. Artificial Intelligence*, pp. 1080-1087, Aug. 1995.
[138] K. Sugihara, "Some Location Problems for Robot Navigation Using a Single Camera," *Computer Vision, Graphics, and Image Processing*, vol. 42, pp. 112-129, 1988.
[139] K. Sutherland and W. Thompson, "Localizing in Unstructured Environments: Dealing with the Errors," *IEEE Trans. Robotics and Automation*, vol. 10, no. 6, pp. 740-754, Dec. 1994.
[140] C. Thorpe, "An Analysis of Interest Operators for FIDO," *Proc. IEEE Workshop Computer Vision: Representation and Control*, pp. 135-140, Apr./May 1984.
[141] C. Thorpe, "FIDO: Vision and Navigation for a Mobile Robot," PhD dissertation, Dept. Computer Science, Carnegie Mellon Univ., Dec. 1984.
[142] C. Thorpe, T. Kanade, and S.A. Shafer, "Vision and Navigation for the Carnegie-Mellon Navlab," *Proc. Image Understand Workshop*, pp. 143-152, 1987.
[143] C. Thorpe, M.H. Herbert, T. Kanade, and S.A. Shafer, "Vision and Navigation for the Carnegie-Mellon Navlab," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 10, no. 3, pp. 362-372, May 1988.
[144] S. Thrun, "Learning Metric-Topological Maps for Indoor Mobile Robot Navigation," *Artificial Intelligence*, vol. 99, no. 1, pp. 21-71, Feb. 1998.
[145] S. Thrun, "Probabilistic Algorithms in Robotics," Technical Report CMU-CS-00-126, Carnegie Mellon Univ., 2000.
[146] D.C. Tseng and C.H. Chang, "Color Segmentation Using Perceptual Attributes," *Proc. 11th Conf. Pattern Recognition*, vol. 3, pp. 228-231, 1992.
[147] T. Tsubouchi and S. Yuta, "Map-Assisted Vision System of Mobile Robots for Reckoning in a Building Environment," *Proc. IEEE Int'l Conf. Robotics and Automation*, pp. 1978-1984, 1987.
[148] S. Tsugawa, T. Yatabe, T. Hirose, and S. Matsumoto, "An Automobile with Artificial Intelligence," *Proc. Sixth Int'l Joint Conf. Artificial Intelligence*, pp. 893-895, 1979.
[149] T. Tsumura, "Survey of Automated Guided Vehicle in Japanese Factory," *Proc. Int'l Conf. Robotics and Automation*, pp. 1329-1334, Apr. 1986.
[150] M.A. Turk and M. Marra, "Color Road Segmentation and Video Obstacle Detection," *Proc. SPIE-The Int'l Soc. Optical Eng.-Mobile Robots*, vol. 727, pp. 136-142, Oct. 1986.
[151] M.A. Turk, D.G. Morgenthaler, K.D. Gremban, and M. Marra, "VITS—A Vision System for Autonomous Land Vehicle Navigation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 10, no. 3, pp. 342-361, May 1988.
[152] R. Wallace, K. Matsuzaki, Y. Goto, J. Crisman, J. Webb, and T. Kanade, "Progress in Robot Road-Following," *Proc. IEEE Int'l Conf. Robotics and Automation*, pp. 1615-1621, Apr. 1986.
[153] R.S. Wallace, "Robot Road Following by Adaptive Color Classification and Shape Tracking," *Proc. IEEE Int'l Conf. Robotics and Automation* pp. 258-263, Mar/Apr. 1987.
[154] A.M. Waxman, J.J. LeMoigne, L.S. Davis, and B. Srinivasan, "Visual Navigation of Roadways," *Proc. IEEE Int'l Conf. Robotics and Automation*, pp. 862-867, Mar. 1985.
[155] A.M. Waxman, J.J. LeMoigne, L.S. Davis, B. Srinivasan, T.R. Kushner, E. Liang, and T. Siddalingaiah, "A Visual Navigation System for Autonomous Land Vehicles," *IEEE Trans. Robotics and Automation*, vol. 3, no. 2, pp. 124-141, Apr. 1987.

[156] J. Weng and S. Chen, "Vision-Guided Navigation Using SHOSLIF," *Neural Networks,* vol. 11, no. 7-8, pp. 1511-1529, Oct-Nov. 1998.
[157] B. Wilcox and D. Gennery, "A Mars Rover for the 1990's," *J. Brit. Interplanetary Soc.,* vol. 40, pp. 484-488, 1987.
[158] B. Wilcox, L. Matthies, D. Gennery, B. Copper, T. Nguyen, T. Litwin, A. Mishkin, and H. Stone, "Robotic Vehicles for Planetary Exploration," *Proc. 1992 IEEE Int'l Conf. Robotics and Automations,* pp. 175-180, May 1992.
[159] Y. Yagi, S. Kawato, and S. Tsuji, "Real-Time Omnidirectional Image Sensor (COPIS) for Vision-Guided Navigation," *IEEE Trans. Robotics and Automation,* vol. 10, no. 1, pp. 11-22, Feb. 1994.
[160] Y. Yagi, Y. Nishizawa, and M. Yachida, "Map-Based Navigation for a Mobile Robot with Omnidirectional Image Sensor COPIS," *IEEE Trans. Robotics and Automation,* vol. 11, no. 5, pp. 634-648, Oct. 1995.
[161] B. Yamauchi and R. Beer, "Spatial Learning for Navigation in Dynamic Environments," *IEEE Trans. System, Man, and Cybernetics, Part B,* vol. 26, no. 3, pp. 496-505, June 1996.
[162] Z. Zhang and O. Faugeras, "A 3D World Model Builder with a Mobile Robot," *Int'l J. Robotics Research,* vol. 11, no. 4, pp. 269-285, 1992.
[163] J.Y. Zheng, M. Barth, and S. Tsuji, "Autonomous Landmark Selection for Route Recognition by a Mobile Robot," *Proc. IEEE Int'l Conf. Robotics and Automation,* pp. 2004-2009, 1991.
[164] U.R. Zimmer, "Robust World-Modeling and Navigation in a Real World," *Proc. Third Int'l Conf. Fuzzy Logic, Neural Nets and Soft Computing,* vol. 13, no. 2-4, pp. 247-60, Oct. 1996.
[165] P. Zingaretti and A. Carbonaro, "Route Following Based on Adaptive Visual Landmark Matching," *Robotics and Autonomous Systems,* vol. 25, no. 3-4, pp. 177-184, Nov. 1998.

Guilherme N. DeSouza received the BS and MS degrees in computer engineering from the Federal University of Rio de Janeiro. From 1985 to 1994, he worked as a associate researcher for the Brazilian Power Systems Research Center-CEPEL. At CEPEL, he worked in areas such as real-time systems, neural networks, fuzzy logic, and nonmonotonic reasoning expert systems, all in the scope of supervisory control and data acquisition systems (SCADA) for power plants and substations. He is currently working in the Robot Vision Lab, at Purdue University and his research focus is on vision-based architectures for mobile robotics and visual servoing.



Avinash C. Kak is a professor of electrical and computer engineering at Purdue University. He has coauthored the widely used book *Digital Picture Processing*, published by Academic Press, 1982. He is also a coauthor of *Principles of Computerized Tomographic Imaging*, which was recently chosen by SIAM Press for republication in their *Classics in Applied Mathematics* series. His latest book, *Programming with Objects: A Comparative Presentation of C++ and Java*, is expected to be out in the near future. His current research interests are focused on the sensory aspects of robotic intelligence, especially robotic vision. He is the chief editor of the journal *Computer Vision and Image Understanding* published by Academic Press.