Vision-based robot localization without explicit object models

Gregory Dudek *   Chi Zhang

Centre for Intelligent Machines & School of Computer Science
McGill University, 3480 University Street
Montreal, QC, Canada H3A 2A7

Abstract

We consider the problem of locating a robot in an initially-unfamiliar environment from visual input. The robot is not given a map of the environment, but it does have access to a collection of training examples, each of which specifies the video image observed when the robot is at a particular location and orientation.

We address two variants of this problem: how to estimate translation of a moving robot assuming the orientation is known, and how to estimate translation and orientation for a mobile robot.

Performing scene reconstruction to construct a metric map of the environment using only video images is difficult. We avoid this by using an approach in which the robot learns to convert a set of image measurements into a representation of its pose (position and orientation). This provides a metric estimate of the robot's location within a region covered by the statistical map we build. Localization can be performed on-line without *a prior location estimate. The conversion from visual data to camera pose is implemented using a multi-layer neural network that is trained using backpropagation. An aspect of the approach is the use of an* inconsistency *measure to eliminate incorrect data and estimate components of the pose vector. The experimental data reported in this paper suggests that the accuracy and flexibility of the technique is good, while the on-line computational cost is very low.*

1 Introduction

The problem of locating an observer within a (partially) known environment has recently received attention and is closely related to object pose estimation. Position estimation (or localization) is significant particularly in the context of mobile robot localization and navigation. Typical approaches involve the detection or tracking of *a priori* landmarks or beacons and an associated viewpoint estimation computation[LDW92, BR93].

Landmark-based localization is difficult not only because it implies solving at least a weak instance of the inverse imaging transformation, but also because selecting appropriate general-purpose landmarks that combine visibility, detectability and stability is a challenging problem. In this paper we propose an approach to viewpoint estimation that avoids the selection of explicit landmarks and uses instead the statistical variations of low-level features across the environment.

We consider two variants of the localization problem: (1) the problem of estimating the position of a mobile robot in the plane given its orientation (i.e. having a compass), (2) the problem of estimating a mobile robot's position and orientation in the plane without prior pose information. Unlike many existing positioning schemes, the method we have developed allows a robot to both

- autonomously *extract* a representation of a generic environment (i.e. without depending on a specific class of geometric structure) and
- perform localization without requiring a prior input position estimate.

Note that the emphasis in this paper is on position estimation without requiring prior information, unlike sensor fusion approaches such as Kalman filtering which we see as a subsequent processing option, or methods such as shape-from-motion. Also, unlike existing work on robot control using image data (for steering [JPT93] or navigation [Bro85]), we are interesting on computing numerically accurate position information as opposed to simply generating control information.

In this paper we assume that a robot will carry out an exploratory phase, collecting data to construct a representation of some or all of an environment. This representation is based on the variations in the sensor data rather than an reconstruction of the 3-D environment. The approach described here deals with position estimation within a limited region of the environment

*This article appears in the *Proceedings of the 1996 International Conference on Robotics and Automation, IEEE Press, April 1996, Minneapolis, MN.* The authors gratefully acknowledge the support of the National Sciences and Engineering Research Council.

(for example, a room). A large-scale map can then be constructed from a collection of such local maps. The focus here, however, is on constructing an image-domain representation of the environment and converting the current visual input into an estimate of the local pose (position and/or orientation). The on-line portion of our technique is based on three main steps: encoding the input image using a non-linear feature set, associating the representation with a manifold generated by interpolating between previously seen examples, and validating the position estimate that is produced.

2 Images and pose: formalism

Correctly recovering environmental structure using only image data is known to be a difficult and computationally costly problem. In general, it entails a solution to the inverse problem defined by the surface geometry, reflectance and imaging arrangement. Instead, our approach to localization from image data extracts the camera position from image measurements without using any explicit model of surfaces in the environment. The *"perceptual structure"* (as opposed to physical 3-D structure) of a local region of the environment is recorded by statistically encoding image properties as a function of camera position. For a camera mounted on a mobile robot, we describe the position of the camera with a fixed orientation by

$$\mathbf{q} = (x, y).$$

Our method is based on relating statistical variations in image properties directly to pose $\mathbf{q}$.

We can describe the dependency of the image on the camera position by the relationship

$$\mathbf{i} = \Phi(\mathbf{q}). \tag{1}$$

This is, in essence, an N-dimensional sensor measurement, where N is the number of pixels in the image (320x240, in our experiments). In order to solve the problem of computing camera position from image data, we wish to invert (1):

$$\mathbf{q} = \Phi^{-1}(\mathbf{i}). \tag{2}$$

In general, computing this inverse mapping directly on images is impractical. We introduce a projection operator on images that re-represents them in a lower M-dimensional subspace that is computationally tractable:

$$\mathbf{G}(\mathbf{i}) = (g_1(\mathbf{i}), g_2(\mathbf{i}), ..., g_M(\mathbf{i})). \tag{3}$$

We can further describe the mapping from camera pose to the image features that are observed by

$$\mathbf{G}(\mathbf{i}) = \mathbf{f}(\mathbf{q}). \tag{4}$$

Camera pose is thus given by $\mathbf{q} = \mathbf{f}^{-1}(\mathbf{G}(\mathbf{i}))$. Assuming that images *usually* vary gradually as a function of camera pose (an assumption we relax later), the images $\mathbf{i}_1$ and $\mathbf{i}_2$ from two neighboring known positions $\mathbf{q}_1$ and $\mathbf{q}_2$ can be used to compute the position of an unknown intermediate position $\hat{\mathbf{q}}$ by interpolation, for example linear interpolation in the simplest case yields:

$$\hat{\mathbf{q}} = \frac{|\mathbf{G}(\mathbf{i}) - \mathbf{G}(\mathbf{i}_1)|(\mathbf{q}_2 - \mathbf{q}_1)}{|\mathbf{G}(\mathbf{i}_2) - \mathbf{G}(\mathbf{i}_1)|} + \mathbf{q}_1 \tag{5}$$

This type of relationship has been examined in a highly constrained context for visual servoing of a robot arm by allowing a camera to follow a path in space using global image measurements, in particular using the principal eigenvectors of the image set [NMN94]. That work differs from ours in several imporant ways, in particular by requiring the invertibility of $\mathbf{f}$.

In the remaining portion of this paper, we describe an approach to selecting the measurement features $\mathbf{G}$. We then go on to propose an interpolation scheme that uses radial basis functions. Next, we propose a measure of *inconsistency* that eliminates the estimates from the portions of the measurement space that give rise to non-uniqueness. Finally, we describe a technique for estimating the orientation of an observer exploiting the fact that an instance of the function $\Phi^{-1}(\mathbf{i})$ we construct assumes measurements taken at a orientation and is *not* trained using data from multiple orientations.

2.1 Converting images directly into camera positions

As presented here, we approximate the mapping $\mathbf{f}^{-1}$ from images $\mathbf{i}_j$ to pose $\mathbf{q}(\mathbf{i}_j)$ by using a collection $\mathbf{I} = \{\mathbf{i}_j\}$ of input images acquired at known camera positions (in practice, $\mathbf{f}$ may not be invertible). These images are then re-expressed via a collection of statistical descriptors from images i.e. global features

$$\mathbf{G}(\mathbf{i}) = (g_1(\mathbf{i}), g_2(\mathbf{i}), ..., g_M(\mathbf{i})). \tag{6}$$

This collection of features is used to construct a non-linear interpolator implemented by an artificial neural network specialized to the current region in space[1].

[1] Alternatives to a neural network implementation are feasible and have also been considered, but space does not permit their elaboration.

This network is, in turn, used to determine the position of the camera from an input image *given that the image is obtained within the correct region of the environment – the region for which the network was trained.* Thus for an input feature vector $\mathbf{G}$ a network output

$$\mathbf{N}(\mathbf{G}) = \mathbf{q} \tag{7}$$

can be determined (in what follows we abuse notation and let the argument $\mathbf{i}$ be implicit). The consequence is that camera position information can be recovered without the use of explicit scene models that imply the solution of the scene reconstruction problem. In addition, very limited on-board computation is needed to estimate the position of the camera using a previously trained system. (Although training itself may involve substantial computational cost, this can be performed off-line if desired.)

2.2 Subspace encoding G

Selecting a suitable set of features by which to encode the ensemble of images is an important consideration for the method. In particular, an important issue is that there should be sufficient features to encode the range of *significant* image variations, while keeping their number small for reasons of coding efficiency (discussed later). Further, senstitivity to lighting variations (for example) must be avoided. Note, in particular, that an encoding such as that produced by a principal components analysis may be optimal for encoding image *content* while being unsuitable for the localization task, because irrelevant aspects of the image may be encoded (such as illumination changes). As such, the subspace of the measurement space that is associated with a given pose estimate is large and the VC-dimension of the problem is larger than desired, leading to various difficulties [VC71].

The difficulties in using principal components of the image ensemble have been verified in a series of experiments we have conducted and we have found they they are not as well suited to position estimation as the non-linear features we have selected. Furthermore, computation of principal components implies the *a priori* availability of the entire image ensemble $\mathbf{I}$ and precludes on-line encoding strategies (there exist on-line strategies for computing the principal components, but this is, itself, a complex issue outside the scope of this paper). In the absence of a robust, efficient, on-line mechanism for selecting coding features, we propose a criterion for evaluating features: they should be sufficient for encoding pose information over a restricted region of the environment using polynomial (linear or quadratic) interpolation.

Measurement features were derived from statistics of edge images (computed using the Canny-Deriche edge operator [Can86]) to minimize the effects of illumination variations. The statistical descriptors used are either global, or based on computations over large receptive fields so as to minimize the dependence of the algorithm on either any specific object in the scene or any specific assumptions about image or scene structure. The perceptual structure associated with a position in space hence consists of the following classes of measurements:

- First and second moments of the edge distributions at 2 scales (global and local).
- Mean edge orientations at 2 scales;
- Densities of parallel lines at four orientations (sampling orientation space).

Note that these features comprise the first central moments of the edge distribution in space and orientation space, and hence are natural choices for efficiently encoding a distribution [DH73]. An important aspect of the encoding of the image is to reduce that complexity of the problem of interpolating between image measurements $\mathbf{G}(\mathbf{i})$ and pose estimates.

2.3 The Interpolator

Several approaches have been considered for performing the interpolation between images. A neural network implementation has proven both robust and efficient, and is described here. It is used to compute the interpolation between training images to compute an approximation to the function $\mathbf{f}^{-1}$ on measurements of new images $\mathbf{G}(\mathbf{i})$. The backpropagation algorithm [RHW86] was used to optimize the weights of a three-layer neural network that converts a visual input feature vector into an estimated camera position. The input layer of the network has a set of input units whose activities represent the current visual measurements. The hidden units in the middle layer allow the network to compute non-linear functions of the inputs, they have sigmoidal activation functions, and by adapting their incoming weights they can learn to extract a set of features that are useful for estimating camera position. The output layer is composed of sigmoidal units that represent the estimated pose parameters of the camera.

Using an excessive number of features (or too many hidden units in the network) has very undesirable consequence for the number of weights within the network and consequently for the number of training examples required. This is caused by an increase in problem complexity that can be described by the Vapnik-Chernovenkis dimensionality ($VC - dimension$) of the problem [VC71, BEHW86].

The output layer is composed of groups of units organized into outputs sets Ω_j each encoding the value of one of the dimensions q_i of the pose (in this case, one set for the x coordinate and one set for the y coordinate). Each "radial basis" unit in an output set represents the likelihood expressed as a one-dimensional Gaussian of a particular value v_i of that dimension. The desired activity of a unit is proportional to the probability density of the true coordinate value under its Gaussian. The estimate of a particular pose component q_i is obtained by computing a weighted sum of the outputs o_j of the units in the associated output set:

$$q_i = \sum_{j \in \Omega_i} o_j v_j. \tag{8}$$

After training, inputs for which the output vector is inconsistent with a Gaussian distribution are rejected, thus eliminating some cases that would otherwise be in error.

Backpropagation was used to compute the derivative of the total error with respect to each weight in the network and a conjugate gradient method was then used to update the weights[2]. The number of weight updates needed in the experiments described later was typically between 1000 and 3000.

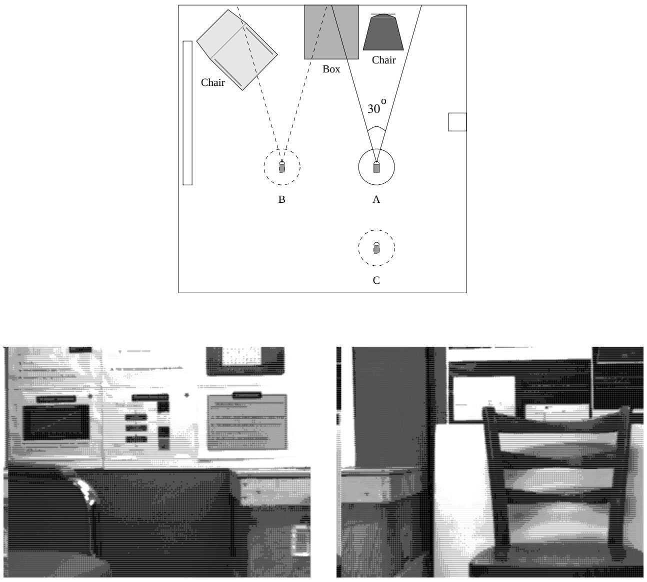
2.4 Data collection

The localization method was evaluated via a series of tests in two basic contexts: a camera head in which the pan and tilt angles had to be estimated, and a mobile robot in which absolute (Cartesian) position had to be estimated assuming that the orientation was known. The pan-tilt estimation problem involves learning from images on various positions on the viewing sphere and being able to recover the position on the viewing sphere from which unknown images come. Although results in the pan-tilt cases are good, space does not permit their discussion here.

In the case of the mobile robot position estimates, the camera was pointing forward (along the y-axis) in all cases and hence the problem is to recover the viewing location using images that may be displaced either laterally or in the fronto-parallel direction with respect to the closest examples in the training set. The solution to the orientation problem is described later.

[2] Our implementation uses a 3-layer network with full connectivity between layers. Our implementations use roughly 16 units in the hidden layer.

The data for the specific localization experiments described here was acquired by moving a RWI B-12 mobile robot about a workspace of roughly one meter square. Images were sampled, 2 per position, in a grid with a spacing of 5 cm with 20 *per cent* of the positions reserved for testing cases, not visited during training. Because the camera was pointing along the y-axis in all cases, estimates of position in the y direction involve an implicit estimate of dilation. Positions of the robot were measured by hand and are accurate to 0.2 cm. The arrangement is illustrated in Figure 1.



(b) View A

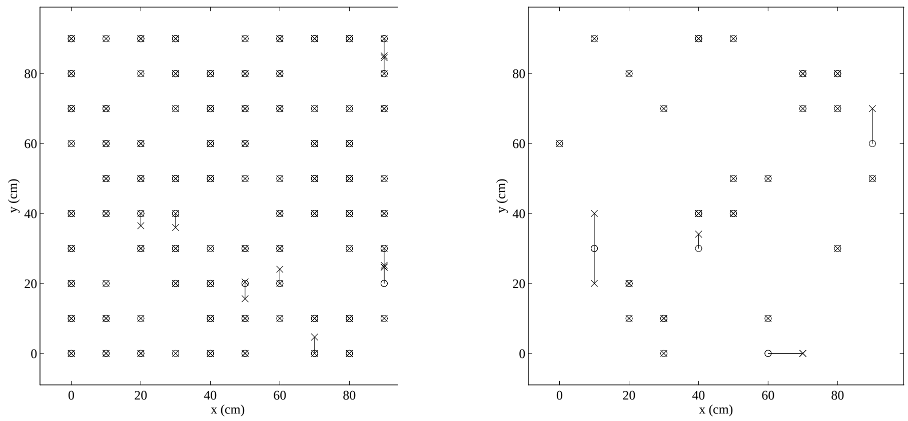
(c) View B

Figure 1: Overhead view of experimental layout for mobile robot localization. Field of view for images from various positions labelled (A) and (B) is shown.

3 Performance

Typical experimental results for Cartesian localization (translation only) are shown in Figure 2. The results are very good with errors in training data averaging 0.3 cm (under 0.4 *per cent* of the output range) and an average error over test cases of 0.8 cm (under 1 per cent of the output range). In contrast, a localization method based on tracking "geometric beacons" using sonar data has an accuracy of roughly 4cm in the same environment and is much more restricted in the kinds of environment it can handle [MD94].

Once training of the network has been computed, pose estimates can be computed rapidly. The time re-



(a) Training

(b) Test

Figure 2: Experimental results for uncorrected localization; line lengths correspond to errors. *Lines connect position estimates marked by crosses to "actual" position estimates marked by circles. (Estimates of "actual" position have accuracies between roughly 0.5 and 0.8 cm each). Training data is shown at left, test data (not visited during training) at right.* Note: most lines have almost zero length. Application of the consistency check can remove the few poor estimates (see text).

quired is that for image acquisition (1/30 s), edge detection (roughly 1 sec for our non-optimized implementation on a low-end SGI INDY workstation), statistical feature computation (roughly 2 sec.), and a single feed-forward pass through the network (several hundred multiplies). In short, pose estimates can be computed in near real-time as the robot moves.

4 Orientation estimation and other refinements

As a pose estimator for a mobile camera, two serious issues have not yet been described: how orientation estimation is accomplished in the mobile robot (translation estimation) context, and how outlier estimates or images are handled. The above approach to pose estimation for a mobile observer assumes that the camera orientation is fixed. Although orientation might be estimated using the same approach used for determining the other components of pose (i.e. via explicit training), this has disadvantages such as the requirement for additional training data and a more complex system. In fact, we can exploit the existing network and its response to incorrect orientation estimates. Figure 3(a) illustrates the effects of acquiring the input image at the incorrect orientation – images from incorrect orientations are associated with (essentially arbitrary) incorrect position estimates. How we can exploit this is described below in the context of outlier elimination.

The quality of the results from the network is good, but there are occasional outliers in the position estimates. These may be due to views that resemble other views. Examples of those are sets of images that result when the mobile robot is looking at bare section of wall from various parts of a workspace. Although some of these may be detectable *a priori*, the possibility that incorrect estimates will sometimes be produced still remains. This potential difficulty is resolved by exploiting the fact that dead reckoning is accurate over small intervals in space or time; the approach is loosely related to both Kalman and median filtering [LDW92, RK76].

4.1 Assuring consistency

For small motions, robotic vehicles or manipulators can usually accurately recover their motion parameters (our RWI B-12 mobile robot is accurate to a fraction of a centimeter for straight-line motions of less than a meter on hard flooring). For two images acquired by a camera on a mobile robot, if we acquire a first image $\mathbf{i}_1$, move straight ahead and acquire a second image $\mathbf{i}_2$, we can compute feature measurement sets $\mathbf{G}_1$ and $\mathbf{G}_2$ and can determine $\mathbf{N}(\mathbf{G}_1) = (x_1, y_1)$ and $\mathbf{N}(\mathbf{G}_2) = (x_2, y_2)$. For a forward motion of distance s at orientation κ, we must have

$$x_2 = x_1 + s\ cos(\kappa) \quad y_2 = y_1 + s\ sin(\kappa) \tag{9}$$

Thus we can define the **inconsistency** of a pair of measurements in this case

$$\gamma_{trans}(\kappa) = ||\mathbf{N}(\mathbf{G}_1) + s(cos(\kappa), sin(\kappa)) - \mathbf{N}(\mathbf{G}_2)|| \tag{10}$$

recalling that the vectors $\mathbf{G}_1$ and $\mathbf{G}_2$ are also functions of κ.

More generally, we can define the *inconsistency* of a *series* of measurements (from a set of images) $i = 1...n-1$ as

$$\gamma = \sum_{i=1}^{n-1} ||\mathbf{N}(\mathbf{G}_i) + (\delta(\mathbf{q}))_{i,i+1} - \mathbf{N}(\mathbf{G}_{i+1})|| \tag{11}$$

where $(\delta(\mathbf{q}))_{i,i+1}$ is the camera pose offset between steps i and $i+1$. When the value of inconsistency is high, it is likely that one or more of the position estimates is in error and should be rejected. This approach can readily be generalized to arbitrary robot verification trajectories including combined rotational and translational motion[3]. Measurements that are not consistent can thus be discarded. Note also, that in the case of a degenerate path $\delta(\mathbf{q}) = 0$ this reduces to simply taking multiple measurements from a single location and selecting the mode, akin to median filtering.

4.2 Estimating orientation

Not only does the inconsistency measure allow outliers or incorrect pose estimates to be rejected, it also provides a mechanism for orientation estimation for a moving camera without the need to train the system using data from multiple orientations. The problem of computing orientation is posed as one of finding the transformation between an arbitrary local reference angle of the robot κ and the true (absolute) reference orientation κ^* at which the system was trained. In the case of translation estimation, the estimate of the function $\mathbf{f}^{-1}$ implemented by the neural net $\mathbf{N}$ assumes that the orientation of the robot is correct: this can, however, be seen as an implicit *parameter* to the computation; this insight is the key to orientation estimation.

By acquiring images $\mathbf{i}_1(\theta)$ at a set of orientations at a fixed (unknown) position $\mathbf{q}_1$, and using each as input to the localization computation, a series of position estimates $\mathbf{q}(\theta)$ may be obtained. For orientations that do not correspond to the training scenario, the estimates produced by the interpolator will be incorrect, as shown in Figure 3(a). At least one of these, however, $\mathbf{q}(\theta^*)$ for $\theta^* = \kappa^* - \kappa$ will be correct (i.e. $\mathbf{q}(\theta^*) = \mathbf{q}_1$). If the camera moves and makes a dead-reckoning estimate of the motion, it can acquire another set of images $\mathbf{i}_2(\theta)$ and the inconsistency computation can be applied for measurements on pairs of images acquired at corresponding values of θ, giving a parameterized inconsistency function $\gamma(\theta)$. Recall that we can safely assume dead-reckoning errors are small over short distances. Those orientations that produce low inconsistency are candidates for the references orientation. In practice, our experiments indicate that only 2 to 4 positions are needed to uniquely determine the reference orientation (shown in Figure 3(b)).

[3] A variety of more sophisticated fitting methods could also be used for computing *inconsistency*, but this does not appear necessary.

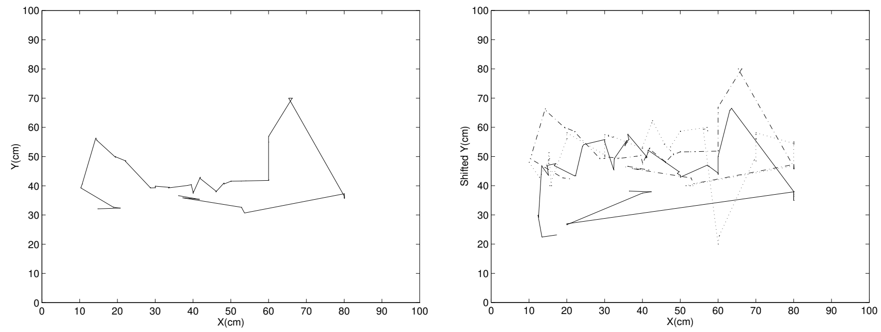
5 Conclusions

We have described a technique for computing camera position from image data. This approach to estimating camera position has the advantage of not requiring either an *a priori* model of the environment nor an environment that is consistent with some simple model of scene or object geometry. The basis for the position estimate is an inferred statistical relationship between a collection of global parameters extracted from an edge map and the camera position over a set of training examples. A critical aspect of the method is the use of local consistency to eliminate spurious estimates.

In this paper, we have ignored the issues of how a large environment should be explored or subdivided. We have concentrated here, rather, on the problem of using visual input data to recover the pose of the robot within a (local) environment that has been previously visited. This is related to work of model-based position estimation from image data [BR93, KMK93] as well as techniques for tracking and recognition based on principal components analysis [TP89, NMN94]. Our approach, however, does not depend on the selection of specific robust geometric features or an initial estimate of the robot's position.

Where the method will be most suitable is for environments that are too complex or poorly structured for model-based methods, where a prior pose estimate is unavailable, where computational cost is a major factor or, perhaps, where local structures (as used in model-based methods) are not sufficiently reliable.

Issues that remain to be considered are the scaling behaviour of the approach to more complex environments, and approaches to the automated on-line selection of optimal input features. One difficulty is that



(a) Single position

(b) Successive positions

Figure 3: Position estimates for images taken at various orientations. The correct position is roughly the center of the figure, position (40,40). **(a)** At a single position. For incorrect angles, the estimated position is incorrect and varies with angle. **(b)** Different curves as a function of angle, drawn on the plane, show estimates from successive positions, shifted so that they should intersect at the correct location. Automatic detection of this point of maximum consistency, where all estimates agree, allows computation of absolute orientation accurate to within 3 degrees. Note that an acceptable intersection corresponds to an intersection at the same location (as shown on the graph) but also for the same angle of the robot's angle (i.e. position along the curve, which is not shown).

there may be regions of the environment where the technique provides no information, but this is a difficulty with model-based approaches as well. Key advantages over landmark-based methods are the reduced reliance on any single landmark and the system's ability to automatically learn the relationship between its percepts and the environment.

References

[BEHW86] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Classifying learnable geometric concepts with the vapnik-chervonenkis dimension. *Proc of the Eighteenth Annual ACM Symposium on Theory of Computing*, pages 273–282, Berkeley 1986, 1986. ACM, Salem.

[BR93] R. Basri and E. Rivlin. Homing using combinations of model views. *Proc of the International Joint Conf of Artificial Intelligence*, pages 1656–1591, Chambery, France, 1993.

[Bro85] R.A. Brooks. *Visual Map Making for a Mobile Robot.* 1985.

[Can86] J.F. Canny. A computational approach to edge detection. *IEEE Trans Pattern Analysis and Machine Intel.*, PAMI-8:679–698, 1986.

[DH73] Duda and Hart. *Pattern Classification and Scene Analysis.* New York. NY, 1973.

[JPT93] T. Jochem, D.A. Pomerleau, and C.E. Thorpe. Maniac: A next generation neurally based autonomous road follower. *Proc of the Conf on Intelligent Autonomous Systems (IAS-3)*, pages 592–599, Pittsburgh, PA, February 1993.

[KMK93] A. Kosaka, M. Meng, and A. C. Kak. Vision-guided mobile robot navigation using retroactive updating of position uncertainty. *Proc of the International Conf of Robotics and Automation, Volume 2*, pages 1–7, Atlanta, GA, May 1993.

[LDW92] J. J. Leonard and H. F. Durrant-Whyte. *Directed sonar sensing for mobile robot navigation.* 1992.

[MD94] P. MacKenzie and G. Dudek. Precise positioning using model-based maps. *Proc of the International Conf on Robotics and Automation*, San Diego, CA, 1994.

[NMN94] S.K. Nayar, H. Murase, and S.A. Nene. Learning, positioning, and tracking visual appearance. *Proc. IEEE Conf of Robotics and Automation*, pages 3237–3244, San Diego, CA, May 1994.

[RHW86] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by back-propagating errors. *Nature*, *323*:533–536, 1986.

[RK76] A. Rosenfeld and A.C. Kak. *Digital Picture Processing.* New York, 1976.

[TP89] M. Turk and A. Pentland. Face processing: Models for recognition. *Mobile Robotics IV*, Nov. 1989.

[VC71] V.N. Vapnik and A.Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and Its Applications*, 16:264–280, 1971.