Localization and Homing using Combinations of Model Views

Ronen Basri*
Dept. of Applied Math
The Weizmann Institute of Science
Rehovot 76100, Israel
Internet: ronen@wisdom.weizmann.ac.il

Ehud Rivlin†
Computer Science Dept.
Technion
Haifa 32000, Israel
Internet: ehudr@cs.technion.ac.il

Abstract

Navigation involves recognizing the environment, identifying the current position within the environment, and reaching particular positions. We present a method for *Localization* (the act of recognizing the environment), *positioning* (the act of computing the exact coordinates of a robot in the environment), and *homing* (the act of returning to a previously visited position) from visual input. The method is based on representing the scene as a set of 2D views and predicting the appearances of novel views by linear combinations of the model views. The method accurately approximates the appearance of scenes under weak-perspective projection. Analysis of this projection as well as experimental results demonstrate that in many cases this approximation is sufficient to accurately describe the scene. When weak-perspective approximation is invalid, either a larger number of models can be acquired or an iterative solution to account for the perspective distortions can be employed.

The method has several advantages over other approaches. It uses relatively rich representations; the representations are 2D rather than 3D; and localization can be done from only a single 2D view without calibration. The same principal method is applied for both the localization and positioning problems, and a simple "qualitative" algorithm for homing is derived from this method.

*This report describes research done in part at the Massachusetts Institute of Technology within the Artificial Intelligence Laboratory and the McDonnell-Pew Center for Cognitive Neuroscience. Support for the laboratory's artificial intelligence research is provided in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research Contract N00014-91-J-4038.

†This report describes research done in part at the University of Maryland within the Computer Vision Laboratory in the Center for Automation Research. The second author was supported in part by the Defense Advanced Research Projects Agency (ARPA Order No. 8459) and the U.S. Army Engineer Topographic Laboratories under Contract DACA76-92-C-0009.

1 Introduction

Basic tasks in autonomous robot navigation are localization, positioning, and homing. *Localization* is the act of recognizing the environment, that is, assigning consistent labels to different locations, and *positioning* is the act of computing the coordinates of the robot in the environment. Positioning is a task complementary to localization, in the sense that position (e.g., "1.5 meters northwest of table T") is often specified in a place-specific coordinate system ("in room 911"). *Homing* is the task of returning to a previously visited position.

A method for localization, positioning, and homing in visually-guided navigation systems is presented. The method, based on [20], represents scenes by sets of their 2D images. Localization is achieved by comparing the observed image to linear combinations of model views. The position of the robot is computed by analyzing the coefficients of the linear combination that aligns the model to the image. Also, a simple, qualitative solution to the homing problem using the same scheme is presented.

Visually-guided navigation systems can be classified according to the type of scene representations utilized. We distinguish between two types of representations, signatures and 3D models. Systems that represent the scene using a set of signatures usually generate from images of the scene a representation that is invariant over a relatively large range of transformations. These invariant representations often are obtained by projecting the image data onto a lower dimensional subspace or by computing a set of measurements from the data. Localization is achieved by generating signatures from the observed images and comparing the obtained signatures with the stored signatures in a straightforward way.

Sarachik [17] computes and stores the dimensions of the navigated offices. Engelson *et al.* [6] use blurred images of the scene as signatures. Nelson [14] generates signatures from averaged orientations of edges in different regions of the image. Braunegg [4] recovers a depth map of the scene from which he generates an occupancy map obtained by projecting the 3D edges onto "the floor". Hong *et al.* [9] generate signatures from panoramic views of the scene by projecting them onto a 1D circle.

Other systems store complete 3D descriptions of the scene. To recognize the scene the systems must first recover the transformation that relates between the model and the incoming images. Ayache and Faugeras [1] use a trinocular stereo system to recover the 3D structure of the scene before it is compared with the model. Onoguchi *et al.* [15] use a stereo system to recover a depth map of the observed scene. In order to align the stereo image with the model a set of landmarks is first located by the system and their positions are used to derive the transformation that relates the model to the image. Fennema *et al.* [7]) compares the 3D models of the scene to sequences of 2D images. Gray-scaled templates of selected landmarks are generated from the model, and the location of these landmarks is computed by means of correlation and tracking.

The method presented in this paper does not generate signatures of the scene. However, rather than using explicit 3D descriptions of the scene, the scene is represented by sets of its 2D images. Predicting the appearances of novel views is obtained by combining the model views.

Homing was recently addressed in several studies. Nelson [14] and Zipser [21] proposed to handle this problem by generating signatures of the scene from single images and storing them along with vectors directing the robot toward the target location. At runtime whenever the robot encounters a signature similar to one or more of the stored signatures it follows the precomputed direction vectors associated with these signatures. Hong *et al.* [9] perform homing by comparing signatures obtained from a panoramic view of the scene with a similar signature obtained at the target location. The robot is then instructed to move so as to bring the observed signature and the target signature into alignment.

The method for homing presented in this paper differs from previous algorithms by that it does not use signatures to represent the scene. Homing is achieved by moving the robot so as to align the observed images of the scene with an image taken from the target position. Like [9], our algorithm computes the direction of motion "on the fly". The algorithm is qualitative in nature, and it is designed so as to gradually bring the current and the target images into alignment.

The rest of the paper is organized as follows. The method for localization is presented in Section 2, where we propose a method that works accurately under weak-perspective approximation and an iterative scheme to account for perspective distortions. Positioning is addressed in Section 3, and the algorithm for homing is described in Section 4. Constraints imposed on the motion of the robot as a result of special properties of indoor environments can be used to reduce the complexity of the method presented here. This topic is covered on Section 5. Experimental results follow.

2 Localization

The problem of localization is defined as follows: given P, a 2D image of a place, and $\mathcal{M}$, a set of stored models, find a model $M^i \in \mathcal{M}$ such that P matches M^i. One problem a system for localization should address is the variability of images due to viewpoint changes. The inexactness of practical systems makes it difficult for a robot to return to a specified position on subsequent visits. The visual data available to the robot between visits varies in accordance with the viewing position of the robot. A localization system should be able to recognize scenes from different positions and orientations.

Another problem is that of changes in the scene. At subsequent visits the same place may look different due to changes in the arrangement of the objects, the introduction of new objects, and the removal of others. In general, some objects tend to be more static than others. While chairs and books are often moved, tables, closets, and pictures tend to change their position less frequently, and walls are almost guaranteed to be static. Static cues naturally are more reliable than mobile ones. Confining the system to static cues, however, may in some cases result in failure to recognize the scene due to insufficient cues. The system should therefore attempt to rely on static cues, but should not ignore the dynamic cues.

We are interested in a system that can recognize the environment from different viewing positions and that can update its representations dynamically to accommodate changes in the scene. A common approach to handling the problem of recognition from different viewpoints is by comparing the stored models to the observed environment after the viewpoint is recovered and compensated for. This approach, called *alignment*, is used in a number of studies of object recognition [3, 8, 10, 13, 18, 19]. We apply the alignment approach to the problem of localization. Below we describe a localization system based on the "Linear Combinations" scheme [20]. The presentation is divided into two parts. In the first part (Section 2.1) we describe the basic system that works under weak-perspective approximation. The second part (Section 2.2) proposes a method for handling large perspective distortions.

2.1 Localization under a Weak-Perspective Assumption

The scheme for localization is the following. Given an image, we construct two view vectors from the feature points in the image, one contains the x-coordinates of the points, and the other contains the y-coordinates of the points. An object (in our case, the environment) is modeled by a set of such views, where the points in these views are ordered in correspondence. The appearance of a novel view of the object is predicted by applying linear combinations to the stored views. The coefficients of this linear combination are recovered using a small number of model points and their corresponding image points. To verify the match, the predicted appearance is compared with the actual image, and the object is recognized if the two match. A large number of points (or line segments) are used for verification. The advantage of this method is twofold. First, viewer-centered representations are used rather than object-centered ones; namely, models are composed of 2D views of the observed scene. Second, novel appearances are predicted in a simple and accurate way (under weak-perspective projection).

Formally, given P, a 2D image of a scene, and $\mathcal{M}$, a set of stored models, the objective is to find a model $M^i \in M$ such that $P = \sum_{j=1}^{k} \alpha_j M_j^i$ for some constants $\alpha_j \in \mathcal{R}$. It has been shown that this scheme accurately predicts the appearance of rigid objects under weak-perspective projection (orthographic projection and scale) [20]. The limitations of this projection model are discussed later in this paper.

More concretely, let $p_i = (x_i, y_i, z_i)$, $1 \leq i \leq n$, be a set of n object points. Under weak-perspective projection, the position $p_i' = (x_i', y_i')$ of these points in the image are given by

$$\begin{aligned} x_i' &= sr_{11}x_i + sr_{12}y_i + sr_{13}z_i + t_x \\ y_i' &= sr_{21}x_i + sr_{22}y_i + sr_{23}z_i + t_y \end{aligned} \tag{1}$$

where r_{ij} are the components of a 3×3 rotation matrix, and s is a scale factor. Rewriting this in vector equation form we obtain

$$\begin{aligned} \mathbf{x}' &= sr_{11}\mathbf{x} + sr_{12}\mathbf{y} + sr_{13}\mathbf{z} + t_x\mathbf{1} \\ \mathbf{y}' &= sr_{21}\mathbf{x} + sr_{22}\mathbf{y} + sr_{23}\mathbf{z} + t_y\mathbf{1} \end{aligned} \tag{2}$$

where $\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{x}', \mathbf{y}' \in R^n$ are the vectors of x_i, y_i, z_i, x'_i and y'_i coordinates respectively, and $\mathbf{1} = (1, 1, \ldots, 1)$. Consequently,

$$\mathbf{x}', \mathbf{y}' \in span\{\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{1}\} \tag{3}$$

or, in other words, $\mathbf{x}'$ and $\mathbf{y}'$ belong to a four-dimensional linear subspace of $\mathcal{R}^n$. (Notice that $\mathbf{z}'$, the vector of depth coordinates of the projected points, also belongs to this subspace. This fact is used in Section 2.2 below.) A four-dimensional space is spanned by any four linearly independent vectors of the space. Two views of the scene supply four such vectors [16, 20]. (See also [11].) Denote by $\mathbf{x}_1$, $\mathbf{y}_1$ and $\mathbf{x}_2$, $\mathbf{y}_2$ the location vectors of the n points in the two images; then there exist coefficients a_1, a_2, a_3, a_4 and b_1, b_2, b_3, b_4 such that

$$\begin{aligned} \mathbf{x}' &= a_1\mathbf{x}_1 + a_2\mathbf{y}_1 + a_3\mathbf{x}_2 + a_4\mathbf{1} \\ \mathbf{y}' &= b_1\mathbf{x}_1 + b_2\mathbf{y}_1 + b_3\mathbf{x}_2 + b_4\mathbf{1} \end{aligned} \tag{4}$$

(Note that the vector $\mathbf{y}_2$ already depends on the other four vectors.) Since R is a rotation matrix, the coefficients satisfy the following two quadratic constraints:

$$\begin{gathered} a_1^2 + a_2^2 + a_3^2 - b_1^2 - b_2^2 - b_3^2 = 2(b_1b_3 - a_1a_3)r_{11} + 2(b_2b_3 - a_2a_3)r_{12} \\ a_1b_1 + a_2b_2 + a_3b_3 + (a_1b_3 + a_3b_1)r_{11} + (a_2b_3 + a_3b_2)r_{12} = 0 \end{gathered} \tag{5}$$

To derive these constraints the transformation between the two model views should be recovered. This can be done under weak-perspective using a third image. Alternatively, the constraints can be ignored, in which case the system would confuse rigid transformations with affine ones. This usually does not prevent successful localization since generally scenes are fairly different from one another.

Note that we incorporate in the model only points that appear in both model images. Points that are not visible in one of the images due to occlusion are excluded from the model. We can extend the models with additional points by taking more then two images of the scene. (See [20].)

To summarize, we model the environment by a set of images with correspondence between the images. For example, a spot can be modeled by two of its corresponding views. The corresponding quadratic constraints may also be stored. Localization is achieved by recovering the linear combination that aligns the model to the observed image. The coefficients are determined using four model points and their corresponding image points by solving a linear set of equations. Three points are sufficient to determine the coefficients if the quadratic constraints are also considered. Additional points may be used to reduce the effect of noise. After the coefficients are recovered we use them to predict the appearance of the model. All the points of the model can be used at this stage. The predicted appearance is then compared to the actual image to verify the match. The worst case time-complexity of the localization process when the quadratic constraints are ignored is $k(m^4n^4)m'$, where k is the number of models considered, m is the number of model points, n is the number of image points, and m' is the number of points considered for verification. This complexity is typical to alignment schemes. Applying the constraints proposed in Section 5 considerably reduces this complexity.

The recovery of the alignment coefficients is defined as follows. Denote by

$$M = [\mathbf{x}_1, \mathbf{y}_1, \mathbf{x}_2, \mathbf{1}] \tag{6}$$

the matrix of model points, and let $\mathbf{a}$ and $\mathbf{b}$ denote the vectors of coefficients, then

$$\begin{aligned} \mathbf{a} &= M^{+}\mathbf{x}' \\ \mathbf{b} &= M^{+}\mathbf{y}' \end{aligned} \tag{7}$$

where $M^{+} = (M^T M)^{-1} M^T$ is the pseudo-inverse of M. ($M^{+} = M^{-1}$ when only four points are used.) Note that for the recovery stage M, $\mathbf{x}'$, and $\mathbf{y}'$ should contain only the coordinates of those points used for the recovery process, e.g., of the hypothesized match. The sensitivity to errors of this recovery process is determined by the condition number of M. The robustness of the recovery process can be increased by choosing quadruples of model points arising from non-planar configurations and by extending the set of matches with additional points to generate an overdetermined system

In our scheme we distinguish between static, semi-static, and dynamic cues. To handle the different types of features we assign weights to the model points reflecting their reliability. We can use several different criteria to determine the weights of points, such as, the number of occurrences in subsequent visits or the height of points in the scene (higher points tend more to be static). The weights are incorporated in both stages of recovering the coefficients and verification. In the recovery stage, let $\mathbf{w}$ be a vector of weights assigned to the model points, and let $W = diag\{\mathbf{w}\}$ then

$$\begin{aligned} \mathbf{a} &= (WM)^{+}W\mathbf{x}' \\ \mathbf{b} &= (WM)^{+}W\mathbf{y}' \end{aligned} \tag{8}$$

In the verification stage, distances between predicted positions of model features and their matched positions in the image are weighed according to $\mathbf{w}$.

Our scheme for localization uses viewer-centered models, that is, representations that are composed of images. It has a number of advantages over methods that build full three-dimensional models to represent the scene. First, by using viewer-centered models that cover relatively small transformations we avoid the need to handle occlusions in the scene. If from some viewpoints the scene appears different because of occlusions we utilize a new model for these viewpoints. Second, viewer-centered models are easier to build and to maintain than object-centered ones. The models contain only images and correspondences. By limiting the transformation between the model images one can find the correspondence using motion methods (e.g., epipolar constraints [2, 12]). If large portions of the environment are changed between visits a new model can be constructed by simply replacing old images with new ones.

The number of models required to cover the scene from all possible viewing positions depends on the complexity of the scene. A complex scene (containing many aspects) may require a relatively large number of views. In practice, however, navigation may require only a relatively small number of models. Specifically, to recognize its rough location in the environment the robot may need to represent the environment as it appears from the access routes only. For example, to recognize a room the robot can represent the appearance of the room from the threshold. One model may therefore be sufficient in this case. (See Section 5.)

One problem with using the scheme for localization is due to the weak-perspective approximation. (An analysis of the weak-perspective assumption under this scheme is given in Appendix A.) In contrast with the problem of object recognition, where we can often assume that objects are small relative to their distance from the camera, in localization the environment surrounds the robot and perspective distortions cannot be neglected. The limitations of the weak-perspective modeling are discussed both mathematically and empirically through the rest of this paper. It is shown that in many practical cases weak-perspective is sufficient to enable accurate localization. The main reason is that the problem of localization does not require accurate measurements in the entire image; it only requires identifying a sufficient number of spots to guarantee accurate naming. If these spots are relatively close to the center of the image, or if the depth differences they create are relatively small (as in the case of looking at a wall when the line of sight is nearly perpendicular to the wall), the perspective distortions are relatively small, and the system can identify the scene with high accuracy. Also, views related by a translation parallel to the image plane form a linear space even when perspective distortions are large. This case and other simplifications are discussed in Section 5.

By using weak-perspective we avoid stability problems that frequently occur in perspective computations. We can therefore compute the alignment coefficients by looking at a relatively narrow field of view. The entire scheme can be viewed as an accumulative process. Rather than acquiring images of the entire scene and comparing them all to a full scene model (as in [4]) we recognize the scene image by image, spot by spot, until we accumulate sufficient convincing evidence that indicates the identity of the place.

When perspective distortions are relatively large and weak-perspective is insufficient to model the environment, two approaches can be used. One possibility is to construct a larger number of models so as to keep the possible changes between the familiar and the novel views small. Alternatively, an iterative computation can be applied to compensate for these distortions. Such an iterative method is described in Section 2.2.

2.2 Handling Perspective Distortions

The scheme presented above accurately handles changes in viewpoint assuming the images are obtained under weak-perspective projection. Error analysis and experimental results demonstrate that in many practical cases this assumption is valid. In cases where perspective distortions are too large to be handled by a weak-perspective approximation, matching between the model and the image can be facilitated in two ways. One possibility is to avoid cases of large perspective distortion by augmenting the library of stored models with additional models. In a relatively dense library there usually exists a model that is related to the image by a sufficiently small transformation avoiding such distortions. The second alternative is to improve the match between the model and the image using an iterative process. In this section we consider the second option.

The suggested iterative process is based on a Taylor expansion of the perspective coordinates. As is described below, this expansion results in a polynomial consisting of terms each of which can be approximated by linear combinations of views. The first term of this series represents the orthographic approximation. The process resembles a method of matching 3D points with 2D points described recently by DeMenthon and Davis [5]. In this case, however, the method is applied to 2D models rather than 3D ones. In our application the 3D coordinates of the model points are not provided; instead they are approximated from the model views.

An image point $(x, y) = (fX/Z, fY/Z)$ is the projection of some object point, (X, Y, Z) in the image, where f denotes the focal length. Consider the following Taylor expansion of $1/Z$ around some depth value Z_0:

$$\begin{aligned} \frac{1}{Z} &= \sum_{k=0}^{\infty} \frac{1}{k!} f^{(k)}(Z_0)(Z - Z_0)^k \\ &= \sum_{k=0}^{\infty} \frac{1}{k!} \frac{(-1)^k k!}{Z_0^{k+1}} (Z - Z_0)^k \\ &= \frac{1}{Z_0} \sum_{k=0}^{\infty} \left(1 - \frac{Z}{Z_0}\right)^k \end{aligned} \tag{9}$$

The Taylor series describing the position of a point x is therefore given by

$$x = \frac{fX}{Z} = \frac{fX}{Z_0} \sum_{k=0}^{\infty} \left(1 - \frac{Z}{Z_0}\right)^k \tag{10}$$

Notice that the zero term contains the orthographic approximation for x. Denote by $\Delta^{(k)}$ the kth term of the series:

$$\Delta^{(k)} = \frac{fX}{Z_0} \left(1 - \frac{Z}{Z_0}\right)^k \tag{11}$$

A recursive definition of the above series is given below.

Initialization:

$$x^{(0)} = \Delta^{(0)} = \frac{fX}{Z_0}$$

Iterative step:

$$\begin{aligned} \Delta^{(k)} &= \left(1 - \frac{Z}{Z_0}\right) \Delta^{(k-1)} \\ x^{(k)} &= x^{(k-1)} + \Delta^{(k)} \end{aligned}$$

where $x^{(k)}$ represents the kth order approximation for x, and $\Delta^{(k)}$ represents the highest order term in $x^{(k)}$.

According to the orthographic approximation both X and Z can be expressed as linear combinations of the model views (Eq. (4)). We therefore apply the above procedure, approximating X and Z at every step using the linear combination that best aligns the model points with the image points. The general idea is therefore the following. First, we estimate $x^{(0)}$ and $\Delta^{(0)}$ by solving the orthographic case. Then at each step of the iteration we improve the estimate by seeking the linear combination that best estimates the factor

$$1 - \frac{Z}{Z_0} \approx \frac{x - x^{(k-1)}}{\Delta^{(k-1)}} \tag{12}$$

Denote by $\mathbf{x} \in \mathcal{R}^n$ the vector of image point coordinates, and denote by

$$P = [\mathbf{x}_1, \mathbf{y}_1, \mathbf{x}_2, \mathbf{1}] \tag{13}$$

an $n \times 4$ matrix containing the position of the points in the two model images. Denote by $P^+ = (P^T P)^{-1} P^T$ the pseudo-inverse of P (we assume P is overdetermined). Also denote by $\mathbf{a}^{(k)}$ the coefficients computed for the kth step. $P\mathbf{a}^{(k)}$ represents the linear combination computed at that step to approximate the X or the Z values. Since Z_0 and f are constant they can be merged into the linear combination. Denote by $\mathbf{x}^{(k)}$ and $\mathbf{\Delta}^{(k)}$ the vectors of computed values of x and Δ at the kth step. An iterative procedure to align a model to the image is described below.

Initialization:
Solve the orthographic approximation, namely

$$\begin{aligned} \mathbf{a}^{(0)} &= P^+ \mathbf{x} \\ \mathbf{x}^{(0)} = \mathbf{\Delta}^{(0)} &= P\mathbf{a}^{(0)} \end{aligned}$$

Iterative step:

$$\begin{aligned} \mathbf{q}^{(k)} &= (\mathbf{x} - \mathbf{x}^{(k-1)}) \div \Delta^{(k-1)} \\ \mathbf{a}^{(k)} &= P^+ \mathbf{q}^{(k)} \\ \Delta^{(k)} &= (P\mathbf{a}^{(k)}) \otimes \mathbf{\Delta}^{(k-1)} \\ \mathbf{x}^{(k)} &= \mathbf{x}^{(k-1)} + \mathbf{\Delta}^{(k)} \end{aligned}$$

where the vector operations $\otimes$ and $\div$ are defined as

$$\begin{aligned} \mathbf{u} \otimes \mathbf{v} &= (u_1 v_1, \ldots, u_n v_n) \\ \mathbf{u} \div \mathbf{v} &= (\frac{u_1}{v_1}, \ldots, \frac{u_n}{v_n}) \end{aligned}$$

The method presented above is meant to improve the match between the model and the image by reducing perspective effects. One problem with applying this method is that we may mistake false matches for errors due to perspective distortion. In general, one cannot distinguish a-priori between the two kinds of errors (unless the iterative procedure is run to convergence). Certain heuristics and probabilistic methods may be used to reduce the chances of assigning false matches. Such methods include limiting the distortions allowed as a function of position in the image and expected position in the model (using the analysis in Appendix A). Other cues (such as stereo, color, texture, or previous knowledge) and instruments (e.g., sonar) may also be used to detect where large variations due to perspective distortions should be anticipated.

3 Positioning

Positioning is the problem of recovering the exact position of the robot. This position can be specified in a fixed coordinate system associated with the environment (i.e., room coordinates), or it can be associated with some model, in which case location is expressed with respect to the position from which the model views were acquired. In this section we derive the position of a robot from the alignment coefficients.

We assume a model composed of two images, P_1 and P_2; their relative position is given. Given a novel image P', we first align the model with the image (i.e., localization). By considering the coefficients of the linear combination the robot's position relative to the model images is recovered. To recover the absolute position of the robot in the room the absolute positions of the model views should also be provided. Note that the computation is done in "image coordinates" (that is, assuming a unit focal length). Positions should be normalized if world coordinates are used.

Assume P_2 is obtained from P_1 by a rotation R, translation $\mathbf{t} = (t_x, t_y, t_z)$, and scaling s. (Denote the average distance of the camera in P_1 to the scene by Z_0, s is given by $Z_0/(Z_0 + t_z)$.) The coordinates of a point in P', (x', y'), can be written as linear combinations of the corresponding model points in the following way:

$$\begin{aligned} x' &= a_1x_1 + a_2y_1 + a_3x_2 + a_4 \\ y' &= b_1x_1 + b_2y_1 + b_3x_2 + b_4 \end{aligned} \tag{14}$$

Substituting for x_2 we obtain

$$\begin{aligned} x' &= a_1x_1 + a_2y_1 + a_3(sr_{11}x_1 + sr_{12}y_1 + sr_{13}z_1 + t_x) + a_4 \\ y' &= b_1x_1 + b_2y_1 + b_3(sr_{11}x_1 + sr_{12}y_1 + sr_{13}z_1 + t_x) + b_4 \end{aligned} \tag{15}$$

and rearranging these equations we obtain

$$\begin{aligned} x' &= (a_1 + a_3sr_{11})x_1 + (a_2 + a_3sr_{12})y_1 + (a_3sr_{13})z_1 + (a_3t_x + a_4) \\ y' &= (b_1 + b_3sr_{11})x_1 + (b_2 + b_3sr_{12})y_1 + (b_3sr_{13})z_1 + (b_3t_x + a_4) \end{aligned} \tag{16}$$

Using these equations we can derive all the parameters of the transformation between the model and the image. Assume the image is obtained by a rotation U, translation $\mathbf{t}_n$, and scaling s_n. Using the orthonormality constraint we can first derive the scale factor

$$\begin{aligned} s_n^2 &= (a_1 + a_3sr_{11})^2 + (a_2 + a_3sr_{12})^2 + (a_3sr_{13})^2 \\ &= a_1^2 + a_2^2 + a_3^2s^2 + 2a_3s(a_1r_{11} + a_2r_{12}) \end{aligned} \tag{17}$$

Note that we can also extract the scale factor by applying the same constraint to the b's:

$$s_n^2 = b_1^2 + b_2^2 + b_3^2 s^2 + 2b_3 s(b_1 r_{11} + b_2 r_{12}) \tag{18}$$

We can use the two equations to verify that the weak-perspective approximation is valid. The orthogonality constraint (Eq. 5) can also be used for the this purpose. From Equations (16) and (17), by deriving the components of the translation vector, $\mathbf{t}_n$, we can obtain the position of the robot in the image relative to its position in the model views:

$$\begin{aligned} \Delta x &= a_3 t_x + a_4 \\ \Delta y &= b_3 t_y + b_4 \\ \Delta z &= t_z\left(\frac{1-\frac{1}{s_n}}{1-\frac{1}{s}}\right) \end{aligned} \tag{19}$$

Note that Δz is derived from the change in scale of the object. The rotation matrix U between P_1 and P' is given by

$$\begin{aligned} u_{11} &= \frac{a_1 + a_3 s r_{11}}{s_n} & u_{12} &= \frac{a_2 + a_3 s r_{12}}{s_n} & u_{13} &= \frac{a_3 s r_{13}}{s_n} \\ u_{21} &= \frac{b_1 + b_3 s r_{21}}{s_n} & u_{22} &= \frac{b_2 + b_3 s r_{22}}{s_n} & u_{23} &= \frac{b_3 s r_{23}}{s_n} \end{aligned} \tag{20}$$

As has already been mentioned, the position of the robot is computed here relative to the position of the camera when the first model image, P_1, was acquired. Δx and Δz represent the motion of the robot from P_1 to P', and the rest of the parameters represent its 3D rotation and elevation. To obtain this relative position the transformation parameters between the model views, P_1 and P_2, are required. Consequently, positioning, unlike localization, requires calibration of the model images.

One should note that the results of the positioning process depend on the precision of the alignment coefficients, which may be erroneous due to either a bad choice of correspondences or to an invalid orthographic approximation. In cases of errors in the coefficients the recovery of Δx and Δy would depend linearly on the errors, while Δz is inversely dependent on the errors. This sensitivity of Δz is typical in processes of recovering depth such as stereo and motion. We should note, however, that positioning in general is performed after localization is achieved, and so the estimate of the coefficients can be improved by using a large number of points. Section 4 below presents an alternative process to lead the robot to desired positions which, due to the use of feedback, is less sensitive to errors and does not require calibration of the model images.

4 Homing

The *homing* problem is defined as follows. Given an image, called the *target image*, position yourself in the location from which this image was observed. One way to solve this problem is to extract the exact position from which the target image was obtained and direct the robot to that position. In this section we are interested in a more qualitative approach. Under this approach position is not computed. Instead, the robot observes the environment and extracts only the direction to the target location. Unlike the exact approach, the method presented here does not require the recovery of the transformation between the model views.

We assume we are given with a model of the environment together with a target image. The robot is allowed to take new images as it is moving towards the target. We begin by assuming a horizontally moving platform. (In other words, we assume three degrees of freedom rather than six; the robot is allowed to rotate around the vertical axis and translate horizontally. The validity of this constraint is discussed in Section 5.) Later in this section we shall consider homing in the full 3D case. Below we give a simple computation that determines a path which terminates in the target location. At each time step the robot acquires a new image and aligns it with the model. By comparing the alignment coefficients with the coefficients for the target image the robot determines its next step. The algorithm is divided into two stages. In the first stage the robot fixates on one identifiable point and moves along a circular path around the fixation point until the line of sight to this point coincides with the line of sight to the corresponding point in the target image. In the second stage the robot advances forward or retreats backward until it reaches the target location.

Given a model composed of two images, P_1 and P_2, P_2 is obtained from P_1 by a rotation about the Y-axis by an angle α, horizontal translation t_x, and scale factor s. Given a target image P_t, P_t is obtained from P_1 by a similar rotation by an angle θ, translation t_t, and scale s_t. Using Eq. (4) the position of a target point (x_t, y_t) can be expressed as (see Figure 1)

$$\begin{aligned} x_t &= a_1 x_1 + a_3 x_2 + a_4 \\ y_t &= b_2 y_1 \end{aligned} \tag{21}$$

(The rest of the coefficients are zero since the platform moves horizontally.) In fact, the coefficients are given by

$$\begin{aligned} a_1 &= \tfrac{s_t \sin(\alpha-\theta)}{\sin\alpha} \\ a_3 &= \tfrac{s_t \sin\theta}{s \sin\alpha} \\ a_4 &= t_t - \tfrac{t_x s_t \sin\theta}{s \sin\alpha} \\ b_2 &= s_t \end{aligned} \tag{22}$$

(The derivation is given in Appendix B.)

At every time step the robot acquires an image and aligns it with the above model. Assume that an image P_p is obtained as a result of a rotation by an angle ϕ, translation t_p, and scale s_p. The position of a point (x_p, y_p) is expressed by

$$\begin{aligned} x_p &= c_1 x_1 + c_3 x_2 + c_4 \\ y_p &= d_2 y_1 \end{aligned} \tag{23}$$

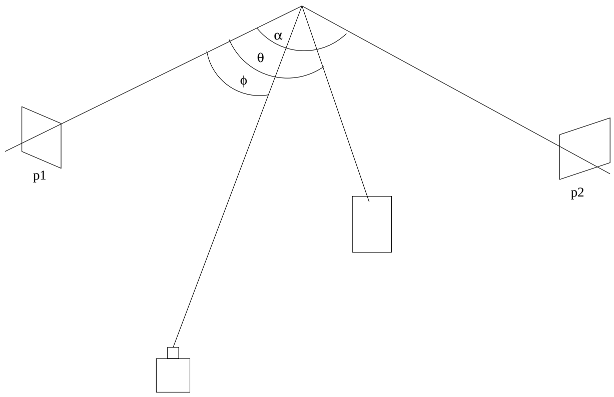

Figure 1: Illustration of the homing task. P_1 and P_2 are the two model images separated by an angle α. The target image is separated from P_1 by an angle θ, and the robot is positioned at an angle ϕ of P_1.

where the coefficients are given by

$$\begin{aligned} c_1 &= \frac{s_p \sin(\alpha-\phi)}{\sin\alpha} \\ c_3 &= \frac{s_p \sin\phi}{s \sin\alpha} \\ c_4 &= t_p - \frac{t_x s_p \sin\phi}{s \sin\alpha} \\ d_2 &= s_p \end{aligned} \tag{24}$$

The step performed by the robot is determined by

$$\delta = \frac{c_1}{c_3} - \frac{a_1}{a_3} \tag{25}$$

That is,

$$\delta = \frac{s\sin(\alpha-\phi)}{\sin\phi} - \frac{s\sin(\alpha-\theta)}{\sin\theta} = s\sin\alpha(\cot\phi - \cot\theta) \tag{26}$$

The robot should now move so as to reduce the absolute value of δ. The direction of motion depends on the sign of α. The robot can deduce the direction by moving slightly to the side and checking if this motion results in an increase or a decrease of δ. The motion is defined as follows. The robot moves to the right (or to the left, depending on which direction reduces $|\delta|$) by a step Δx.

A new image P_n is now acquired, and the fixated point is located in this image. Denote its new position by x_n. Since the motion is parallel to the image plane the depth values of the point in the two views, P_p and P_n, are identical. We now want to rotate the camera so as to return the fixated point to its original position. The angle of rotation, β, can be deduced from the equation

$$x_p = x_n \cos\beta + \sin\beta \tag{27}$$

This equation has two solutions. We chose the one that counters the translation (namely, if translation is to the right, the camera should rotate to the left), and that keeps the angle of rotation small. In the next time step the new picture P_n replaces P_p and the procedure is repeated until δ vanishes. The resulting path is circular around the point of focus.

Once the robot arrives at a position for which $\delta = 0$ (namely, its line of sight coincides with that of the target image, and $\phi = \theta$) it should now advance forward or retreat backward to adjust its position along the line of sight. Several measures can be used to determine the direction of motion; one example is the term c_3/a_3 which satisfies

$$\frac{c_3}{a_3} = \frac{s_p}{s_t} \tag{28}$$

when the two lines of sight coincide. The objective at this stage is to bring this measure to 1.

A similar process can be formulated in the full 3D case. Given a model composed of two images, P_1 and P_2, P_2 is obtained from P_1 by a rotation matrix R, translation vector $\mathbf{t}$, and scaling s. Given a target image P_t, P_t is obtained from P_1 by a rotation U, translation $\mathbf{t}_t$, and scaling s_t. As before, at every time step the robot acquires an image and aligns it with the above model. Assume that an image P_p is obtained as a result of a rotation U', translation $\mathbf{t}_p$, and scaling s_p. Again, the robot takes a circular path attempting to minimize simultaneously the absolute value of the four terms

$$\begin{aligned} \delta_1 &= \tfrac{c_1}{c_3} - \tfrac{a_1}{a_3} & \qquad \delta_3 &= \tfrac{d_1}{d_3} - \tfrac{b_1}{b_3} \\ \delta_2 &= \tfrac{c_2}{c_3} - \tfrac{a_2}{a_3} & \qquad \delta_4 &= \tfrac{d_2}{d_3} - \tfrac{b_2}{b_3} \end{aligned} \tag{29}$$

As is shown in Appendix B,

$$\begin{aligned} \delta_1 &= (\tfrac{u'_{11}}{u'_{13}} - \tfrac{u_{11}}{u_{13}})sr_{13} & \qquad \delta_3 &= (\tfrac{u'_{21}}{u'_{23}} - \tfrac{u_{21}}{u_{23}})sr_{13} \\ \delta_2 &= (\tfrac{u'_{12}}{u'_{13}} - \tfrac{u_{12}}{u_{13}})sr_{13} & \qquad \delta_4 &= (\tfrac{u'_{22}}{u'_{23}} - \tfrac{u_{22}}{u_{23}})sr_{13} \end{aligned} \tag{30}$$

where the term sr_{13} depends on the model and so it is constant throughout the computation. The sign of δ_k $(k = 1, ..., 4)$ therefore depend only on the rotation components of the current and the target image. Note that since only the rotation components determine the sign of δ_k there exists a circular path that decreases the absolute values of all four terms simultaneously.

Once the robot arrives at a position where $\delta_k = 0$ $(k = 1, ..., 4)$ the rotation matrix corresponding to the current image, P_p, and that corresponding to the target image, P_t, become equal, namely, $U' = U$. This is shown in the following claim.

Claim: $\delta_k = 0$ $(k = 1, ..., 4)$ implies that $U' = U$.

Proof: $\delta_1 = 0$ implies that

$$\frac{u'_{11}}{u'_{13}} = \frac{u_{11}}{u_{13}}$$

and $\delta_2 = 0$ implies that

$$\frac{u'_{12}}{u'_{13}} = \frac{u_{12}}{u_{13}}$$

As a result, the two following vectors are identical

$$(\frac{u'_{11}}{u'_{13}}, \frac{u'_{12}}{u'_{13}}, 1) = (\frac{u_{11}}{u13}, \frac{u_{12}}{u_{13}}, 1)$$

Notice that the top rows of U' and U are the normalized versions of these two vectors, and so clearly they also must be equal:

$$(u'_{11}, u'_{12}, u'_{13}) = (u_{11}, u_{12}, u_{13})$$

Similarly, $\delta_3 = \delta_4 = 0$ implies that the middle rows of U and U' are equal, namely

$$(u'_{21}, u'_{22}, u'_{23}) = (u_{21}, u_{22}, u_{23})$$

and since the third row of a rotation matrix is given by the cross product of the first two rows we obtain that

$$U' = U$$

□

Consequently, after the robot reaches a position where all δ_k vanish the line of sight of the robot coincides with the line of sight at the target image. In order to reach the target position the robot should now advance forward or retreat backward to adjust its position along the line of sight. Again, the measure c_3/a_3 can be used for this purpose since

$$\frac{c_3}{a_3} = \frac{s_p}{s_t} \tag{31}$$

when the two lines of sight coincide. The objective at this stage is to bring this measure to 1.

5 Imposing Constraints

Localization and positioning require a large memory and a great deal of on-line computation. A large number of models must be stored to enable the robot to navigate and manipulate in relatively large and complicated environments. The computational cost of model-image comparison is high, and if context (such as path history) is not available the number of required comparisons may get very large. To reduce this computational cost a number of constraints may be employed. These constraints take advantage of the structure of the robot, the properties of indoor environments, and the natural properties of the navigation task. This section examines some of these constraints.

One thing a system may attempt to do is to build the set of models so as to reduce the effect of perspective distortions in order to avoid performing iterative computations. Views of the environment obtained when the system looks relatively deep into the scene usually satisfy this condition. When perspective distortions are large the system may consider modeling subsets of views related by a translation parallel to the image plane (perpendicular to the line of sight). In this case the depth values of the points are roughly equal across all images considered, and it can be shown that novel views can be expressed by linear combinations of two model views even in the presence of large perspective distortions. This becomes apparent from the following derivation. Let $(X_i, Y_i, Z_i), 1 \leq i \leq n$ be a point projected in the image to $(x_i, y_i) = (fX_i/Z_i, fY_i/Z_i)$, and let (x'_i, y'_i) be the projected point after applying a rigid transformation. Assuming that $Z'_i = Z_i$ we obtain (assuming $f = 1$)

$$\begin{aligned} Z_i x'_i &= r_{11}X_i + r_{12}Y_i + r_{13}Z_i + t_x \\ Z_i y'_i &= r_{21}X_i + r_{22}Y_i + r_{23}Z_i + t_y \end{aligned} \tag{32}$$

Dividing by Z_i we obtain

$$\begin{aligned} x'_i &= r_{11}x_i + r_{12}y_i + r_{13} + t_x\frac{1}{Z_i} \\ y'_i &= r_{21}x_i + r_{22}y_i + r_{23} + t_y\frac{1}{Z_i} \end{aligned} \tag{33}$$

Rewriting this in vector equation form gives

$$\begin{aligned} \mathbf{x}' &= r_{11}\mathbf{x} + r_{12}\mathbf{y} + r_{13}\mathbf{1} + t_x\mathbf{z}^{-1} \\ \mathbf{y}' &= r_{21}\mathbf{x} + r_{22}\mathbf{y} + r_{23}\mathbf{1} + t_y\mathbf{z}^{-1} \end{aligned} \tag{34}$$

where $\mathbf{x}$, $\mathbf{y}$, $\mathbf{x}'$, and $\mathbf{y}'$ are the vectors of x_i, y_i, x'_i, and y'_i values respectively, $\mathbf{1}$ is a vector of all 1s, and $\mathbf{z}^{-1}$ is a vector of $1/Z_i$ values. Consequently, as in the weak-perspective case, novel views obtained by a translation parallel to the image plane can be expressed by linear combinations of four vectors.

An indoor environment usually provides the robot with a flat, horizontal support. Consequently, the motion of the camera is often constrained to rotation about the vertical (Y) axis and to translation in the XZ-plane. Such motion has only three degrees of freedom instead of the six degrees of freedom in the general case. Under this constraint fewer correspondences are required to align the model with the image. For example, in Eq. (4) (above) the coefficients $a_2 = b_1 = b_3 = b_4 = 0$. Three points rather than four are required to determine the coefficients by solving a linear system. Two, rather than three, are required if the quadratic constraints are also considered. Another advantage to considering only horizontal motion is the fact that this motion constrains the possible epipolar lines between images. This fact can be used to guide the task of correspondence seeking.

Objects in indoor environments sometimes appear in roughly planar settings. In particular, the relatively static objects tend to be located along walls. Such objects include windows, shelves, pictures, closets and tables. When the assumption of orthographic projection is valid (for example, when the robot is relatively distant from the wall, or when the line of sight is roughly perpendicular to the wall) the transformation between any two views can be described by a 2D affine transformation. The dimension of the space of views of the scene is then reduced to three (rather than four), and Eq. (4) becomes

$$\begin{aligned} \mathbf{x}' &= a_1\mathbf{x}_1 + a_2\mathbf{y}_1 + a_4\mathbf{1} \\ \mathbf{y}' &= b_1\mathbf{x}_1 + b_2\mathbf{y}_1 + b_4\mathbf{1} \end{aligned} \tag{35}$$

($a_3 = b_3 = 0$.) Only one view is therefore sufficient to model the scene.

Most office-like indoor environments are composed of rooms connected by corridors. Navigating in such an environment involves maneuvering through the corridors, entering and exiting the rooms. Not all points in such an environment are equally important. Junctions, places where the robot faces a number of options for changing its direction, are more important than other places for navigation. In an indoor environment these places include the thresholds of rooms and the beginnings and ends of corridors. A navigation system would therefore tend to store more models for these points than for others.

One important property shared by many junctions is that they are confined to relatively small areas. Consider for example the threshold of a room. It is a relatively narrow place that separates the room from the adjacent corridor. When a robot is about to enter a room, a common behavior includes stepping through the door, looking into the room, and identifying it before a decision is made to enter the room or to avoid it. The images relevant for this task include the set of views of the room from its entrance. Provided that thresholds are narrow these views are related to each other almost exclusively by rotation around the vertical axis. Under perspective projection, such a rotation is relatively easy to recover. The position of points in novel views can be recovered from one model view only. This is apparent from the following derivation. Consider a point $p = (X, Y, Z)$. Its position in a model view is given by $(x, y) = (fX/Z, fY/Z)$. Now, consider another view obtained by a rotation R around the camera. The location of p in the new view is given by (assuming $f = 1$)

$$(x', y') = \left(\frac{r_{11}X + r_{12}Y + r_{13}Z}{r_{31}X + r_{32}Y + r_{33}Z}, \frac{r_{21}X + r_{22}Y + r_{23}Z}{r_{31}X + r_{32}Y + r_{33}Z}\right) \tag{36}$$

implying that

$$(x', y') = \left(\frac{r_{11}x + r_{12}y + r_{13}}{r_{31}x + r_{32}y + r_{33}}, \frac{r_{21}x + r_{22}y + r_{23}}{r_{31}x + r_{32}y + r_{33}}\right) \tag{37}$$

Depth is therefore not a factor in determining the relation between the views. Eq. (37) becomes even simpler if only rotations about the Y-axis are considered:

$$(x', y') = \left(\frac{x \cos\alpha + \sin\alpha}{-x \sin\alpha + \cos\alpha}, \frac{y}{-x \sin\alpha + \cos\alpha}\right) \tag{38}$$

where α is the angle of rotation. In this case α can be recovered merely from a single correspondence.

6 Experiments

Figure 2: Two model views of office A.

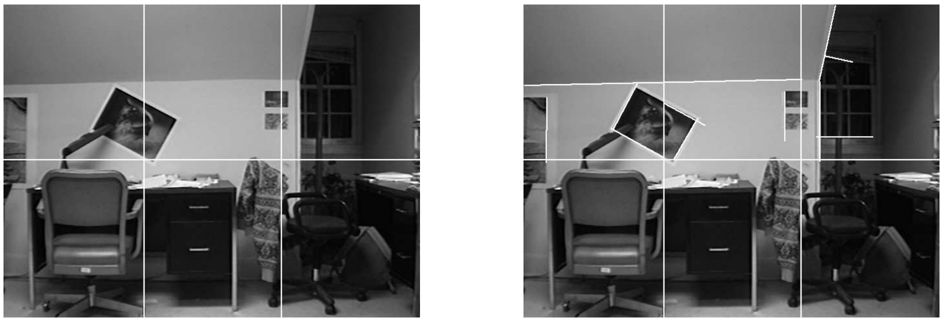

Figure 3: Lines extracted from the image. Left picture contains the search blocks. The lines were extracted from the upper three blocks only. Right picture contains the lines found by the Hough transform procedure.

Figure 4: Matching a model of office A to an image of office A (left), and matching a model of office B to the same image (right).

Figure 5: Matching a model of office A to an image of the same office obtained by a relatively large motion forward and to the right.

The method was implemented and applied to images taken in an indoor environment. Images of two offices, A and B, that have similar structures were taken using a Panasonic camera with a focal length of 700 pixels. Semi-static objects, such as heavy furniture and pictures, were used to distinguish between the offices. Figure 2 shows two model views of office A. The views were taken at a distance of about 4m from the wall. Candidates for correspondence were picked using the following method. The image was divided into six equal-size blocks. Candidates were picked from the upper three blocks only, assuming that the upper portion of the image is more likely to contain static features of the scene. In each block the dominant lines were selected and ranked using a Hough transform procedure. A line was ranked by the sum of the gradient values along its points. The results of this process are shown in Figure 3. Feature points were then obtained by intersecting the obtained lines.

Using the extracted feature points, recovering the coefficients of the linear combination that aligns the model with the image was done in a method similar to [8, 10]. Quadruples of image points were matched to quadruples of model points, and the match between the model and the image using these correspondence was evaluated. The best match obtained was selected. The results of aligning the model views to images of the two offices can be seen in Figure 4. The left image contains an overlay of a predicted image (the thick white lines), constructed by linearly combining the two views, and an actual image of office A. A good match between the two was achieved. The right image contains an overlay of a predicted image constructed from a model of office B and an image of office A. Because the offices share a similar structure the static cues (the wall corners) were perfectly aligned. The semi-static cues, however, did not match any features in the image.

Figure 5 shows the matching of the model of office A with an image of the same office obtained by a relatively large motion forward (about 2m) and to the side (about 1.5m). Although the distances are relatively short most perspective distortions are negligible, and a good match between the model and the image is obtained.

The next experiment shows the application of the iterative process presented in Section 2.2 in cases where large perspective distortion were noticeable. Figure 6 shows two model views, and Figure 7 shows the results of matching a linear combination of the model views to an image of the same office. In this case, because the image was taken from a relatively close distance, perspective distortions cannot be neglected. The effects of perspective distortions can be noticed on the right corner of the board, and on the edges of the hanger on the top right. Perspective effects were reduced by using the iterative process. The results of applying this procedure after one and three iterations are shown in Figure 8.

Another set of experimens was applied to a corridor scene. Here, because of the deep structure of the corridor, perspective distortions are noticeable. Nevertheless, the alignment results still demonstrate an accurate match in large portions of the image. Figure 9 shows two model views of the corridor. Figure 10 (left) shows an overlay of a linear combination of the model views with an image of the corridor. It can be seen that the parts that are relatively distant align perfectly. Figure 10 (right) shows the matching of the corridor model with an image obtained by a relatively large motion (about half of the corridor length). Because of perspective distortions the relatively near features no longer align (e.g., the near door edges). The relatively far edges, however, still match. Figure 11 shows the result of applying the iterative process for reducing perspective distortions on the scene. The process converged after three iterations.

The experimental results demonstrate that the method achieves accurate localization in many cases, and that when the method fails because of large perspective distortions an iterative computation can be used to improve the quality of the match.

7 Conclusions

We presented a method for localization and positioning from visual input. The method is based on representing the scene as a set of 2D views and predicting the appearance of novel views by linear combinations of the model views. The method accurately approximates the appearances of scenes under weak-perspective projection. Analysis of this projection as well as experimental results demonstrate that in many cases this approximation is sufficient to accurately describe the scene. When the weak-perspective approximation is invalid, either a larger number of models can be acquired or an iterative solution can be employed to account for the perspective distortions.

Using our method we presented a solution to the homing problem. The solution takes advantage of the 2D representation. The homing process is done in the image domain in a simple and qualitative manner. Specifically, it does not require the recovery of the transformation between the model images.

Figure 6: Two model views of office C.

Figure 7: Matching the model to an image obtained by a relatively large motion. Perspective distortions can be seen in the table, the board, and the hanger at the upper right.

Figure 8: The results of applying the iterative process to reduce perspective distortions after one (left) and three (right) iterations.

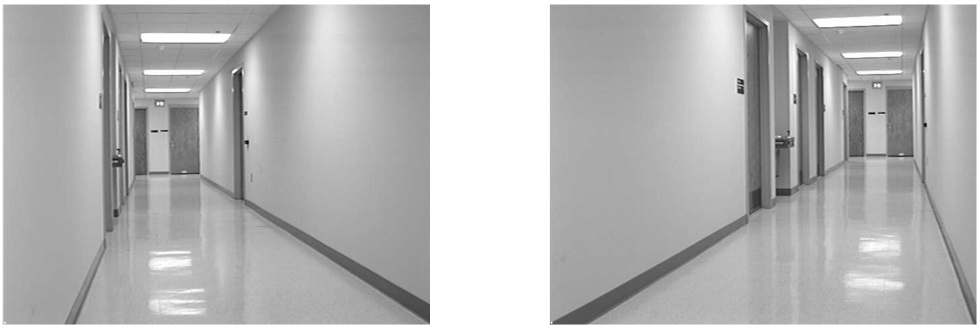

Figure 9: Two model views of a corridor.

Figure 10: Matching the corridor model with two images of the corridor. The right image was obtained by a relatively large motion forward (about half of the corridor length) and to the right. Note that the results of alignment when the picture is taken roughly under the conditions of Eq. 27 (left) are better then when these conditions are violated (right)

Figure 11: Results of applying the iterative process to reduce perspective distortions after three iterations.

The method presented in this paper has several advantages over existing methods. It uses relatively rich representations; the representations are 2D rather than 3D, and localization can be done from a single 2D view only without calibration. The same basic method is used in both the localization and positioning problems. Future work includes handling the problem of acquisition and maintenance of models, constructing indexing methods to reduce the complexity of the localization process, and building maps using visual input.

Appendix

A Projection Model – Error Analysis

In this appendix we estimate the error obtained by using the localization method. The method assumes a weak-perspective projection model. We compare this assumption with the more accurate perspective projection model. We start by deriving the error between a true perspective image and its orthographic approximation, and then we compute the error implied by assuming a weak-perspective projection for both the model and the image.

A point (X, Y, Z) is projected under the perspective model to $(x, y) = (fX/Z, fY/Z)$ in the image, where f denotes the focal length. Under our weak-perspective model the same point is approximated by $(\hat{x}, \hat{y}) = (sX, sY)$ where s is a scaling factor. The best estimate for s, the scaling factor, is given by $s = f/Z_0$, where Z_0 is the average depth of the observed environment. Denote the error by

$$E = |\hat{x} - x| \tag{39}$$

The error is expressed by

$$E = \left| fX(\frac{1}{Z_0} - \frac{1}{Z}) \right| \tag{40}$$

Changing to image coordinates

$$E = \left| xZ(\frac{1}{Z_0} - \frac{1}{Z}) \right| \tag{41}$$

or

$$E = |x| \left| \frac{Z}{Z_0} - 1 \right| \tag{42}$$

The error is small when the measured feature is close to the optical axis, or when our estimate for the depth, Z_0, is close to the real depth, Z. This supports the basic intuition that for images with low depth variance and for fixated regions (regions near the center of the image), the obtained perspective distortions are relatively small, and the system can therefore identify the scene with high accuracy. Figures 12 and 13 show the depth ratio Z/Z_0 as a function of x for $\epsilon = 10$ and 20 pixels, and Table A shows a number of examples for this function. The allowed depth variance, Z/Z_0, is computed as a function of x and the tolerated error, ϵ. For example, a 10 pixel error tolerated in a field of view of up to ± 50 pixels is equivalent to allowing depth variations of 20%. From this discussion it is apparent that when a model is aligned to

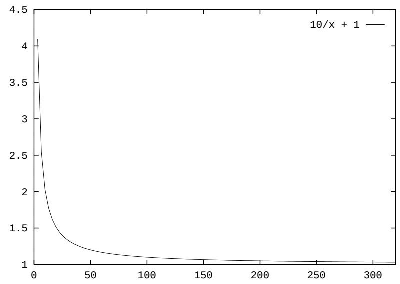

Figure 12: $\frac{Z}{Z_0}$ as a function of x for $\epsilon = 10$ pixels.

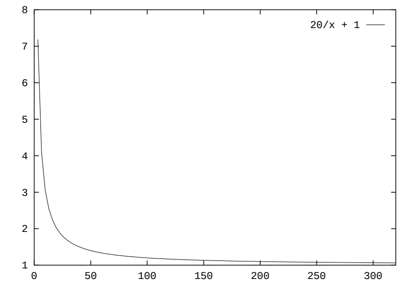

Figure 13: $\frac{Z}{Z_0}$ as a function of x for $\epsilon = 20$ pixels.

the image the results of this alignment should be judged differently at different points of the image. The farther away a point is from the center the more discrepancy should be tolerated between the prediction and the actual image. A five pixel error at position $x = 50$ is equivalent to a 10 pixel error at position $x = 100$.

So far we have considered the discrepancies between the weak-perspective and the perspective projections of points. The accuracy of the scheme depends on the validity of the weak-perspective projection both in the model views and for the incoming image. In the rest of this section we develop an error term for the scheme assuming that both the model views and the incoming image are obtained by perspective projection.

The error obtained by using the scheme is given by

$$E = |x - ax_1 - by_1 - cx_2 - d| \tag{43}$$

Since the scheme accurately predicts the appearances of points under weak-perspective projection, it satisfies

$$\hat{x} = a\hat{x}_1 - b\hat{y}_1 - c\hat{x}_2 - d \tag{44}$$

where accented letters represent orthographic approximations. Assume that in the two model

$x \setminus \epsilon$	5	10	15	20
25	1.2	1.4	1.6	1.8
50	1.1	1.2	1.3	1.4
75	1.07	1.13	1.2	1.27
100	1.05	1.1	1.15	1.2

Table 1: Allowed depth ratios, $\frac{Z}{Z_0}$, as a function of x (half the width of the field considered) and the error allowed (ϵ, in pixels).

pictures the depth ratios are roughly equal:

$$\frac{Z_0^M}{Z^M} = \frac{Z_{01}}{Z_1} \approx \frac{Z_{02}}{Z_2} \tag{45}$$

(This condition is satisfied, for example, when between the two model images the camera only translates along the image plane.) Using the fact that

$$x = \frac{fX}{Z} = \frac{fX}{Z_0}\frac{Z_0}{Z} = \hat{x}\frac{Z_0}{Z} \tag{46}$$

we obtain

$$\begin{aligned} E &= |x - ax_1 - by_1 - cx_2 - d| \\ &\approx \left|\hat{x}\tfrac{Z_0}{Z} - a\hat{x}_1\tfrac{Z_0^M}{Z^M} - b\hat{y}_1\tfrac{Z_0^M}{Z^M} - c\hat{x}_2\tfrac{Z_0^M}{Z^M} - d\right| \\ &= \left|\hat{x}\tfrac{Z_0}{Z} - (a\hat{x}_1 - b\hat{y}_1 - c\hat{x}_2)\tfrac{Z_0^M}{Z^M} - d\right| \\ &= \left|\hat{x}\tfrac{Z_0}{Z} - (\hat{x} - d)\tfrac{Z_0^M}{Z^M} - d\right| \\ &= \left|\hat{x}(\tfrac{Z_0}{Z} - \tfrac{Z_0^M}{Z^M}) - d(\tfrac{Z_0^M}{Z^M} - 1)\right| \\ &\leq |\hat{x}|\left|\tfrac{Z_0}{Z} - \tfrac{Z_0^M}{Z^M}\right| + |d|\left|\tfrac{Z_0^M}{Z^M} - 1\right| \end{aligned} \tag{47}$$

The error therefore depends on two terms. The first gets smaller as the image points get closer to the center of the frame and as the difference between the depth ratios of the model and the image gets smaller. The second gets smaller as the translation component gets smaller and as the model gets close to orthographic.

Following this analysis, weak-perspective can be used as a projection model when the depth variations in the scene are relatively low and when the system concentrates on the center part of the image. We conclude that, by fixating on distinguished parts of the environment, the linear combinations scheme can be used for localization and positioning.

B Coefficients Values for Homing

In this appendix we derive the explicit values of the coefficients of the linear combinations for the case of horizontal motion. Consider a point $p = (x, y, z)$ that is projected by weak-perspective to three images, P_1, P_2, and P', P_2 is obtained from P_1 by a rotation about the Y-axis by an angle α, translation t_m, and scale factor s_m, and P' is obtained from P_1 a rotation about the Y-axis by an angle θ, translation t_p and scale s_p. The position of p in the three images is given by

$$\begin{aligned}
(x_1, y_1) &= (x, y) \\
(x_2, y_2) &= (s_m x \cos\alpha + s_m z \sin\alpha + t_m, s_m y) \\
(x', y') &= (s_p x \cos\theta + s_p z \sin\theta + t_p, s_p y)
\end{aligned}$$

The point (x', y') can be expressed by a linear combination of the first two points:

$$\begin{aligned}
x' &= a_1 x_1 + a_2 x_2 + a_3 \\
y' &= b y_1
\end{aligned}$$

Rewriting these equations we get

$$\begin{aligned}
s_p x \cos\theta + s_p z \sin\theta + t_p &= a_1 x + a_2(s_m x \cos\alpha + s_m z \sin\alpha + t_m) + a_3 \\
s_p y &= b y
\end{aligned}$$

Equating the values for the coefficients in both sides of these equations we obtain

$$\begin{aligned}
s_p \cos\theta &= a_1 + a_2 s_m \cos\alpha \\
s_p \sin\theta &= a_2 s_m \sin\alpha \\
t_p &= a_2 t_m + a_3 \\
s_p &= b
\end{aligned}$$

and the coefficients are therefore given by

$$\begin{aligned}
a_1 &= \frac{s_p \sin(\alpha - \theta)}{\sin\alpha} \\
a_3 &= \frac{s_p \sin\theta}{s_m \sin\alpha} \\
a_4 &= t_p - \frac{t_m s_p \sin\theta}{s_m \sin\alpha} \\
b &= s_p
\end{aligned}$$

Similarly, we can derive terms describing the coefficients in the 3D case. Given a model composed of two images P_1 and P_2 and an image P_t, P_t is obtained from P_1 by a rotation

U, translation $\mathbf{t}_t = (t_{tx}, t_{ty}, t_{tz})$ and scaling s_t, the position of a target point (x_t, y_t) can be expressed as

$$\begin{aligned} x_t &= a_1 x_1 + a_2 y_1 + a_3 x_2 + a_4 \\ y_t &= b_1 x_1 + b_2 y_1 + b_3 x_2 + b_4 \end{aligned}$$

Using Eq. 16 (Section 3) we obtain that the coefficients are given by

$$\begin{aligned} a_1 &= s_t(u_{11} - u_{13}\tfrac{r_{11}}{r_{13}}) & b_1 &= s_t(u_{21} - u_{23}\tfrac{r_{11}}{r_{13}}) \\ a_2 &= s_t(u_{12} - u_{13}\tfrac{r_{12}}{r_{13}}) & b_2 &= s_t(u_{22} - u_{23}\tfrac{r_{12}}{r_{13}}) \\ a_3 &= \tfrac{s_t u_{13}}{s r_{13}} & b_3 &= \tfrac{s_t u_{23}}{s r_{13}} \\ a_4 &= t_{tx} - \tfrac{s_t u_{13}}{s r_{13}} t_x & b_4 &= t_{ty} - \tfrac{s_t u_{23}}{s r_{13}} t_y \end{aligned}$$

Similarly, given an image P_p obtained from P_1 by a rotation U', translation $\mathbf{t}_p = (t_{px}, t_{py}, t_{pz})$ and scaling s_p, the position of a point (x_p, y_p) is expressed by

$$\begin{aligned} x_p &= c_1 x_1 + c_2 y_1 + c_3 x_2 + c_4 \\ y_p &= d_1 x_1 + d_2 y_1 + d_3 x_2 + d_4 \end{aligned}$$

where the coefficients are given by

$$\begin{aligned} c_1 &= s_p(u'_{11} - u'_{13}\tfrac{r_{11}}{r_{13}}) & d_1 &= s_p(u'_{21} - u'_{23}\tfrac{r_{11}}{r_{13}}) \\ c_2 &= s_p(u'_{12} - u'_{13}\tfrac{r_{12}}{r_{13}}) & d_2 &= s_p(u'_{22} - u'_{23}\tfrac{r_{12}}{r_{13}}) \\ c_3 &= \tfrac{s_p u'_{13}}{s r_{13}} & d_3 &= \tfrac{s_p u'_{23}}{s r_{13}} \\ c_4 &= t_{px} - \tfrac{s_p u'_{13}}{s r_{13}} t_x & d_4 &= t_{py} - \tfrac{s_p u'_{23}}{s r_{13}} t_y \end{aligned}$$

We define the terms

$$\begin{aligned} \delta_1 &= \tfrac{c_1}{c_3} - \tfrac{a_1}{a_3} & \delta_3 &= \tfrac{d_1}{d_3} - \tfrac{b_1}{b_3} \\ \delta_2 &= \tfrac{c_2}{c_3} - \tfrac{a_2}{a_3} & \delta_4 &= \tfrac{d_2}{d_3} - \tfrac{b_2}{b_3} \end{aligned}$$

Substituting for the coefficients we obtain that

$$\begin{aligned} \delta_1 &= (\tfrac{u'_{11}}{u'_{13}} - \tfrac{u_{11}}{u_{13}}) s r_{13} & \delta_3 &= (\tfrac{u'_{21}}{u'_{23}} - \tfrac{u_{21}}{u_{23}}) s r_{13} \\ \delta_2 &= (\tfrac{u'_{12}}{u'_{13}} - \tfrac{u_{12}}{u_{13}}) s r_{13} & \delta_4 &= (\tfrac{u'_{22}}{u'_{23}} - \tfrac{u_{22}}{u_{23}}) s r_{13} \end{aligned}$$

References

[1] N. Ayache and O. D. Faugeras. Maintaining representations of the environment of a mobile robot, *IEEE Trans. on Robotics and Automation,* Vol. 5, pp. 804–819, 1989.

[2] R. Basri. On the uniqueness of correspondence from orthographic and perspective projections. *Proc. of Image Understanding Workshop*, pp. 875–884, 1992.

[3] R. Basri and S. Ullman. The alignment of objects with smooth surfaces. *Computer Vision, Graphics, and Image Processing: Image Understanding*, Vol. 57(3), pp. 331–345, 1993.

[4] D. J. Braunegg. Marvel—A system for recognizing world locations with stereo vision. *AI-TR-1229, MIT*, 1990.

[5] D. F. DeMenthon and L. S. Davis. Model-based object pose in 25 lines of code. *Proc. 2nd European Conf. on Computer Vision*, Genova, Italy, 1992.

[6] S. P. Engelson and D. V. McDermott. Image signatures for place recognition and map construction. *Proc. SPIE Symposium on Intelligent Robotic Systems*, Boston, MA, 1991.

[7] C. Fennema, A. Hanson, E. Riseman, R. J. Beveridge, and R. Kumar. Model-directed mobile robot navigation. *IEEE Trans. on Systems, Man and Cybernetics*, Vol. 20, pp. 1352–1369, 1990.

[8] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Communications of the ACM,* Vol. 24, pp. 381–395, 1981.

[9] J. Hong, X. Tan, B. Pinette, R. Weiss, and E. M. Riseman. Image-based homing. *IEEE Control Systems*, pp. 38-44, 1992.

[10] D. P. Huttenlocher and S. Ullman. Object recognition using alignment. *Int. J. on Computer Vision*, Vol 5(2), pp. 195–212, 1990.

[11] J. Koenderink and A. van Doorn. Affine structure from motion. *J. of the Optical Society of America*, Vol. 8(2) pp. 377–385, 1991.

[12] J. Lawn and R. Cipolla. Epipole estimation using affine motion parallax. *Proc. of the British Machine Vision Conference, England*, pp. 379–388, 1993.

[13] D. G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Robotics Research Technical Report 202, Courant Institute of Math. Sciences, New York University*, 1985.

[14] R. N. Nelson. Visual homing using an associative memory. *DARPA Image Understanding Workshop*, pp. 245-262, 1989.

[15] K. Onoguchi, M. Watanabe, Y. Okamoto, Y. Kuno, and H. Asada. A visual navigation system using a multi information local map. *Proc. Int. Conf. on Robotics and Automation*, Cincinnati, OH, pp. 767–774, 1990.

[16] T. Poggio. 3D object recognition: on a result by Basri and Ullman. *Technical Report 9005-03, IRST*, Povo, Italy, 1990.

[17] K. B. Sarachik. Visual navigation: constructing and utilizing simple maps of an indoor environment. *AI-TR-1113, MIT*, 1989.

[18] D. W. Thompson and J. L. Mundy. Three dimensional model matching from an unconstrained viewpoint. *Proc. Int. Conf. on Robotics and Automation*, Raleigh, NC, pp. 208–220, 1987.

[19] S. Ullman. Aligning pictorial descriptions: an approach to object recognition. *Cognition,* Vol. 32, pp. 193–254, 1989.

[20] S. Ullman and R. Basri. Recognition by linear combinations of models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 13, pp. 992–1006, 1991.

[21] D. Zipser. Biologically plausible models of place recognition and goal location. In *D. E. Rumelhart, J. L. McClelland, and the P.D.P. Group. Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 2: *Psychological and Biological Models*, M.I.T. Press, Cambridge, MA, pp. 432–471.