Visually-Guided Navigation by Comparing Two-Dimensional Edge Images*

Daniel P. Huttenlocher
Michael E. Leventon
William J. Rucklidge

TR 94-1407
January 1994

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

* This work was supported in part by National Science Foundation FYI grant IRI-9057928 and matching funds from Xerox Corp., and in part by Air Force contract AFOSR-91-0328.

Visually-Guided Navigation by Comparing Two-Dimensional Edge Images *

Daniel P. Huttenlocher, Michael E. Leventon and William J. Rucklidge
Computer Science Department
Cornell University
Ithaca NY 14853 USA

Abstract

We present a method for navigating a robot from an initial position to a specified landmark in its visual field, using a sequence of monocular images. The location of the landmark with respect to the robot is determined using the change in size and location of the landmark in the image, as a function of the motion of the robot. The landmark location is estimated after the first three images are taken, and this estimate is refined as the robot moves. The method can correct for errors in the robot motion, as well as navigate around obstacles. The obstacle avoidance is done using bump sensors, sonar and dead reckoning, rather than visual servoing. The method does not require prior calibration of the camera. We show some examples of the operation of the system.

*This work was supported in part by National Science Foundation PYI grant IRI-9057928 and matching funds from Xerox Corp., and in part by Air Force contract AFOSR-91-0328.

1 Introduction

In this paper we describe a method for using two-dimensional shape information to determine the location of a mobile robot with respect to some visual landmark in the world. The task is for the robot to navigate to a specified target or landmark in its visual field, possibly in the presence of obstacles. The landmark is initially specified either by marking some portion of an image (containing the landmark) or by providing a prior model. The location of the landmark with respect to the robot is recovered from the change in apparent size and position of the landmark in the image, as a function of the motion of the robot. The size and position of the landmark are determined by comparing two-dimensional shapes from successive images taken as the robot moves.

The key aspects of the method are,

1. The position of the landmark is expressed in terms of the robot-centered quantities range (distance) and bearing (orientation), rather than world coordinates.

2. The range and bearing are calculated using the change in size and location of the object in the image, as a function of the translation and rotation of the robot. The methods do not require camera calibration.

3. The identification of the landmark in the image data is performed by comparing two-dimensional shapes that can translate and scale, without the use of three-dimensional shape information. The range and bearing to the landmark are used to predict its size and location in the image, in order to speed up the processing.

Our approach differs from much of the previous work in visually guided robot navigation since it uses recognition of a specific object in the world instead of extracting some more global image properties; see, for example, the use of stereo in [2] and image deformations in [6]. While this only gives us essentially a single depth reading, it allows us to concentrate on the landmark, and so obtain a high degree of overall accuracy in motion, as we continually confirm that we are on track towards the specified target. In contrast with work such as [7], we do not construct any explicit three-dimensional models. Our work also complements the system described in [8] for finding distinctive landmarks along a route, by providing an effective means of navigating from one landmark to the next.

The key observation underlying the approach is that when a camera moves directly towards an object, the range to that object is given by $m/(s-1)$ where m is the distance that the camera moved and s is the change in the apparent size of the object in the image. Thus a straightforward method for navigating to a landmark is to determine the bearing to that landmark, rotate so that the robot (and camera) is heading in that direction, move forward some distance, use the change in apparent size of the landmark to compute the range to the landmark, move to the landmark. As the robot moves towards the landmark the range and bearing estimates can be updated and the path corrected based on these measurements. By correcting the path as the robot moves, the method can compensate for the camera being misaligned (not pointing in the same direction as the "forward" motion of the robot) and for the robot not moving in exactly the commanded direction.

The configuration of our system is a camera mounted on a wheeled-robot that moves across a relatively flat surface. The camera is mounted at a fixed position on the robot, with its focal point at approximately the center of rotation of the robot, its optic axis approximately in the direction of forward motion of the robot, and its image plane approximately perpendicular to the ground plane. There are thus two degrees of freedom for the camera: a translation in a plane parallel to the ground plane (and perpendicular to the image plane of the camera), and a rotation about the focal point. The translational degree of freedom has the effect of changing the imaged size of an object, and the rotational degree of freedom has the effect of changing the location of the image x-coordinate of an object.

The overall operation of the navigation method consists of the following steps: (1) grab an image of the current visual field of the robot, (2) use a two-dimensional model to localize the landmark in the current image, (3) compute the range and bearing using the localized landmark and the robot motion since the previous image was taken, (4) construct a new two-dimensional model to use in localizing the landmark in the next image, and (5) command the robot to make the next motion (a forward motion or a rotation), whereupon return to step 1. Initially we will describe the method assuming that the robot stops moving during steps 1-4, but in practice these operations actually happen concurrently with the robot motion.

We will first discuss the computation of the range from the change in size of the landmark in the image, and then consider the computation of the bearing from the location (x-position) of the landmark in the image. In Section 4 we then describe the shape comparison method that is used to determine the scale and location of the landmark in the image, and discuss the use of the range and bearing estimates to speed up the shape comparison by predicting the location and size of the landmark in the image. Sections 5 and 6 discuss the navigation method, including the obstacle avoidance, and present some examples. The navigation method has been implemented on a mobile robot platform (developed by [5]); the shape comparison computation is done off-board on a Sun SPARCstation. We have experimented extensively with the navigation system, and it generally brings the robot into contact with a target that is approximately half a meter in each dimension.

2 Determining the Range to the Landmark

In this section we describe the method used to compute the range, r, to a landmark, and discuss the assumptions underlying the method. The only quantities that are measured are the change (increase) in the apparent size of the landmark in the image, s, and the distance that the robot moved, m. Consider a pinhole camera with focal length f, and a landmark of length L whose center lies D away from the optical axis, as illustrated in Figure 1. The camera moves a distance m in the direction of the optic axis (thus towards the landmark, as the landmark is visible). Before the motion, the image of the landmark is of length l, and after the motion is of length l'. The offset of the center of the landmark from the center of the image is d in the first image and d' in the second. From the projection equations,

$$\frac{f}{d + l/2} = \frac{r + m}{D + L/2}$$

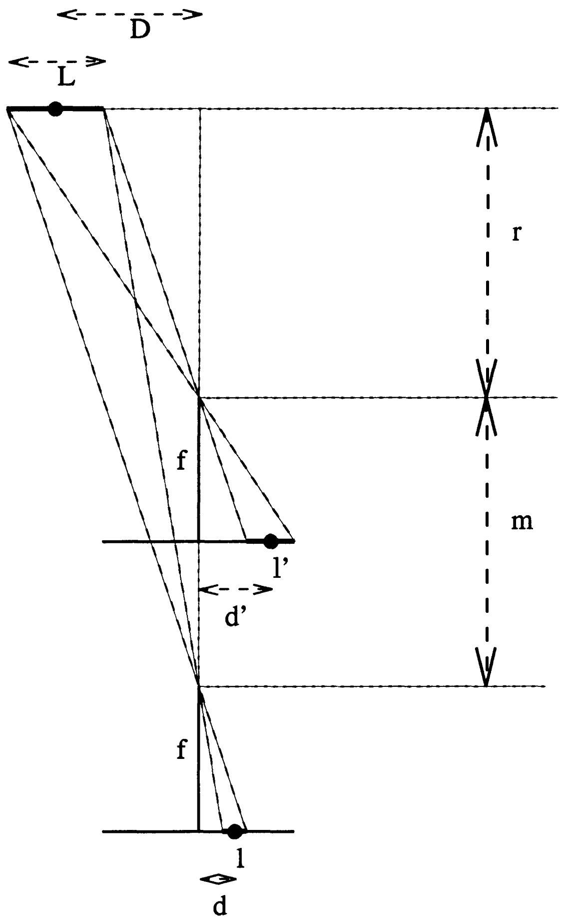


Figure 1: Determining the range to the landmark. A top-down view of the camera as it it moves (translates) distance m in the direction of the camera optic axis. The landmark is of length L with a reference point at its center that is distance D from the optic axis.

and

$$\frac{f}{d' + l'/2} = \frac{r}{D + L/2}$$

and thus

$$(r + m)(d + l/2) = r(d' + l'/2).$$

Similarly $f/d = (r + m)/D$ and $f/d' = r/D$, and thus $(r + m)d = rd'$. From this we obtain $(r + m)l/2 = rl'/2$ and rewriting in terms of r yields

$$r = \frac{m}{\frac{l'}{l} - 1} = \frac{m}{s - 1} \tag{1}$$

where $l'/l = s$ is the change in size of the landmark in the image.

The quantity $r = m/(s-1)$ depends only on the size change s and the distance moved m, but not the camera parameters. Implicitly, however, the accuracy of the range measurement depends on the motion being nearly directly towards the landmark. This is because r only measures the actual range to the landmark when the motion is directly towards the object (i.e., the quantity D is zero). There are two sources of inaccuracy when $D \neq 0$. The first is that r only measures the component of the range in the direction of the motion. That is, the true range is $\sqrt{r^2 + D^2}$ whereas the computed distance is r.

The second source of inaccuracy when $D \neq 0$ comes from that fact that the derivation of $r = m(s-1)$ assumes that the landmark is in a plane perpendicular to the direction of motion of the camera (see Figure 1). When $D = 0$ the landmark projects directly onto the optical center, and this assumption is not necessary. Thus when D is nonzero and the landmark is in a plane that is not perpendicular to the direction of motion (or for instance is not planar) there will be error in computing the range. These two sources of inaccuracy thus lead to a navigation strategy of heading directly towards the landmark, within some tolerance, so that the resulting errors are small. Experimentally, we have determined that the robot generally navigates successfully to the target when the landmark is allowed to be no more than ± 20 pixels of the camera center in an image 360 pixels wide (until the robot is close enough that the object fills much of the field of view, at which point the landmark is kept within a pixel of the center because small rotation errors at such a close range can cause the object to be lost).

Another possible source of inaccuracy in the computation of the range arises from the fact that the image of the landmark may change shape in ways other than translation and scaling. In particular, perspective effects and the fact that different parts of the object will be visible in successive images will result in changes in the image shape. This may influence the computation of the scale change that is used in computing the range. We use a shape comparison method that operates on dense sets of features (intensity edges), and that does not compute a correspondence between features of two shapes being compared (see Section 4). Thus, the method is less sensitive to errors in the locations of individual features than are methods based on correspondences between a small number of features (where errors in a few features may cause a significant error in the shape comparison). Using this shape comparison method, we have found the effects of perspective and correspondence to be negligible until the camera is quite close to the object (within a meter or so using a camera with a 16 mm lens and an image width of about $\pm 15°$), as will be seen in the experimental results section below.

3 Centering the Landmark in the Image

The above method for computing the range to a landmark requires centering the landmark in the image, so that the robot can head directly towards the landmark in order to estimate the range. Centering the landmark in the image is straightforward given an estimate of the focal length of the camera, and assuming that the camera is mounted as described in the introduction, such that the x-axis of the image is approximately parallel to the ground plane and the focal point of the camera is near the center of rotation of the robot. Small inaccuracies in these assumptions are not an issue (e.g., we do not do any careful alignment or calibration of the camera with respect to its mount on the robot).

The assumption that the robot moves in a straight line can be relaxed, as discussed below. Given these assumptions, a rotation of the robot simply changes the x-coordinate of an object in the image. Thus if the landmark is centered at some image location $c = (c_x, c_y)$,

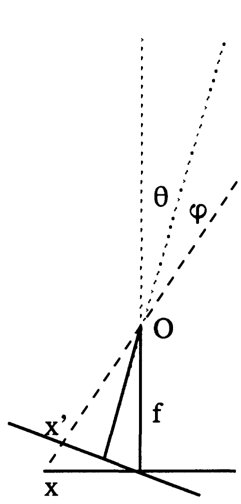


Figure 2: Computing focal length.

a rotation by

$$\theta = \tan^{-1}\left(\frac{c_x}{f}\right) \tag{2}$$

will place the landmark at the image center ($x = 0$), where f is the focal length of the camera. We do not assume that f is known, but rather estimate it from the motion of the landmark in the image as a function of the rotation of the camera. The initial estimate of f is obtained by rotating the camera a fixed, known, amount, and is then refined during subsequent motion towards the landmark, until the robot is close enough that perspective effects and other sources of error become significant (when partial matching starts; see Subsection 5.3).

Figure 2 illustrates the manner in which we compute an approximation to f by rotating the camera some given amount about the focal point. Consider a point X in the world that projects to a point x in the image. If the camera is rotated about the focal point by some given amount θ, then X will project to some new location x' in the resulting image. Let φ be the angle between the optic axis of the rotated camera and the line $\overline{XO}$ (where O is the focal point of the camera). Let $\psi = \theta + \varphi$ be the angle between the optic axis of the original camera and the line $\overline{XO}$.

We know that $\tan(\psi) = x/f$ and $\tan(\varphi) = x'/f$, and hence

$$\tan(\psi) - \tan(\varphi) = \frac{x - x'}{f}.$$

This can be used to approximate f by noting that since $\theta = \psi - \varphi$, when $\tan(\psi) - \tan(\varphi) \approx \tan(\psi - \varphi)$ then

$$f \approx \frac{x - x'}{\tan(\theta)}. \tag{3}$$

This approximation is close when $\cos(\psi - \varphi) \gg \sin(\psi)\sin(\varphi)$ which clearly holds when ψ and φ are both small, but which also holds relatively well over the range of rotations that occur with our camera (which has a field of view of about $\pm 15°$). We choose the rotation of the camera to be $\theta = \pm 8°$ (where the sign of the rotation is in the direction that will move the object towards the image center). Thus if the initial angle is, for example, $\psi = 12°$, then $\varphi = 4°$ and $\cos(\psi - \varphi) \approx .99 \gg .01 \approx \sin(\psi)\sin(\varphi)$. In an image with a visual angle of $\pm 15°$ the initial angle between the optic axis and the image of the landmark is rarely more than 12° because the landmark has some width, and thus its center (which is the reference point on the landmark) is generally not all the way on the edge of the image.

If the robot does not move in a straight line (which we discovered was the case for our robot — it actually moves on a large circle), then when the robot is traveling large distances to the landmark, the landmark will drift away from the center of the image as the robot moves. Thus this effect is most pronounced when the robot begins at considerable distance from the landmark. We have observed it most when starting 8-10 meters away from the landmark. This source of error can be corrected for using the amount that the center of the landmark translates in the image, assuming that the landmark is stationary. We initially center the landmark ($x = 0$), then move forward a distance of m and measure the new position x' of the landmark. The quantity

$$R = \frac{\tan^{-1}(x'/f)}{m}$$

gives the degrees/meter of compensating rotation needed to keep the robot moving in a straight line.

We estimate the drift, R, only when the robot is initially 8 or more meters from the landmark, and then use it to correct the robot's motion locally throughout the path (i.e., local differential rotations are performed during a translation to keep the robot headed in the correct direction). When the robot is initially closer than 8 meters, we have experimentally found that the drift computation is not very accurate. Note that when the drift is not computed, the robot will generally still find the target; however some additional error and expense is introduced because the robot drifts away from the target direction, re-centers, and so forth. As the error in estimating range is proportional to the degree to which the heading is off, it is better to estimate and correct for the drift directly when possible.

4 Locating the Landmark in the Image

As the robot moves, the landmark changes size and position in the camera image. After every motion, the robot recomputes the range and bearing to the landmark so that it may adjust its course, in order to ensure that it continues to move directly towards the landmark.

The robot's time frame is divided into time steps; each time step corresponds to the robot completing a motion (either a forward translation or a rotation). At the beginning of time step t, the robot has a model of the landmark object M_t, an estimate r_t of the range to the landmark, and the position x_t of the landmark in the camera image (i.e. the bearing to the landmark). It performs some motion, then acquires a new frame from the camera and detects the intensity edges (using a detector similar to [1]). This produces a binary edge image I_{t+1}, in which the robot must locate the landmark, using its previous model M_t.

Since the robot has moved, the landmark will have moved (translated) in the image. It will also have enlarged, as the robot is getting closer to it. Also, if the robot rotates about a vertical axis, or does not move directly towards the landmark, then the landmark may appear to scale slightly in x. We therefore search for a transformation $T = (p_x, p_y, s_x, s_y)$ of the model, where each point $w = (w_x, w_y)$ of M_t is mapped to $T(w) = (s_x w_x + p_x, s_y w_y + p_y)$. The difference between s_x and s_y will, however, be small, as the primary influence on these scale values is the overall enlarging of the landmark as the robot approaches.

The stages in the processing of I_{t+1} are as follows. First, the robot uses the information that it knew from the previous time step (the size and position of the landmark) together with its knowledge of the motion it performed to estimate a new size and position, and searches around that location for a section of I_{t+1} which closely resembles M_t. It then finds the scaling and translation of M_t which brings it into the best possible alignment with I_{t+1}. Next, the p_x component of this transformation is converted into the new bearing to the landmark, as described in Equation 2 and the two scale values s_x and s_y are averaged together to give s, the overall scale, which is converted into the new estimate of the range to the landmark, as described in Equation 1; it has now computed r_{t+1} and x_{t+1}. The robot then builds a new model of the landmark, M_{t+1}, determines what motion should be carried out next, and proceeds.

The following subsections explain in detail how each of these steps is carried out.

4.1 The Hausdorff Distance

In order to precisely locate M_t in I_{t+1}, we must have a measure of similarity. Here, we use the minimum Hausdorff distance under translation and scaling of the model M_t.

M_t and I_{t+1} will be finite point sets, where each point represents a pixel where an edge was detected. The Hausdorff distance between the image I_{t+1} and a transformation of the model $T(M_t)$ is defined as

$$H(I_{t+1}, T(M_t)) = \max(h(I_{t+1}, T(M_t)),\ \ h(T(M_t), I_{t+1})) \tag{4}$$

where

$$h(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\|, \tag{5}$$

and $\|\cdot\|$ is some underlying norm. Here, we use the L_2 or Euclidean norm.

The function $h(A, B)$ is called the *directed* Hausdorff distance from A to B. In effect, $h(A, B)$ ranks each point of A based on its distance to the nearest point of B, and then the largest ranked such point (the most mismatched point of A) specifies the value of the distance. Intuitively, if $h(A, B) = \epsilon$, then each point of A must be within distance ϵ of some point of B, and there also is some point of A that is exactly distance ϵ from the nearest point of B (the most mismatched point). Note that in general $h(A, B)$ and $h(B, A)$ can attain very different values (the directed distances are not symmetric).

The computation of $H(I_{t+1}, T(M_t))$ does not involve determining an explicit pairing (or correspondence) of points of I_{t+1} with points of $T(M_t)$ (for example many points of I_{t+1} may be close to the *same* point of $T(M_t)$). This contrasts with most model-based recognition methods which determine a correspondence between points of the model and the image.

Note that $h(T(M_t), I_{t+1}) \leq \epsilon$ iff $T(M_t) \subseteq I^{\epsilon}_{t+1}$, where I^{ϵ}_{t+1} is the set I_{t+1} where every point has been replaced by a disk of radius ϵ; in other words, $h(T(M_t), I_{t+1}) \leq \epsilon$ iff every point of $T(M_t)$ is within ϵ of some point of I_{t+1}. We therefore extend the Hausdorff distance by defining, for some ϵ,

$$f_\epsilon(T(M_t), I_{t+1}) = \frac{\#(\{w \in T(M_t) | w \in I^\epsilon_{t+1}\})}{\#(T(M_t))}$$

where $\#(S)$ is the number of points in the set S. This is the *fraction* of the points of $T(M_t)$ which lie inside I^ϵ_{t+1}, or (equivalently) the fraction of the points of $T(M_t)$ that lie within ϵ of some point of I_{t+1}

M_t is enclosed by some box, defining its maximum extent. This also defines a box which bounds $T(M_t)$. We are only really interested in the points of I_{t+1} which lie inside this box, as points lying outside it are likely to be parts of some other object which is also present in the image. We therefore define $f'_\epsilon(I_{t+1}, T(M_t))$ to be the fraction of the points of I_{t+1} lying inside $T(M_t)$'s box which fall within ϵ of some point of $T(M_t)$.

4.2 Rasterizing Transformation Space

We impose a raster grid on the four-dimensional space of translations and scales, and search on this grid for the new location and scale of the landmark. We do not search the entire space of possible translations and scales, but instead bound the range of allowable transformations; we describe this in the following subsection.

If T and T' are two transformations which are adjacent in the rasterized transformation space (i.e. neighbors on the grid of transformations, considered here to be 8-connected: each location is considered to be connected to its immediate neighbours only), we want the points of $T(M_t)$ to be adjacent (on the image grid) to the points of $T'(M_t)$. This leads us to the following rules:

1. The translational component of the translation should be rasterized to an accuracy of one pixel: translations should have integer components.

2. If for each point $w = (w_x, w_y) \in M_t$, we have $0 \leq w_x \leq x_{\max}$ and $0 \leq w_y \leq y_{\max}$, then we should rasterize the x-scale component to an accuracy of $1/x_{\max}$ and the y-scale component to an accuracy of $1/y_{\max}$. Thus, if $T = (p_x, p_y, s_x, s_y)$ and $T' = (p_x, p_y, s_x, s'_y)$ are adjacent in the grid of transformations, $s'_y = s_y \pm 1/y_{\max}$ and so for any $w \in M_t$, the y-coordinate of $T'(w)$ will be at most $w_y/y_{\max} \leq 1$ away from the y-coordinate of $T(w)$. Our rule is therefore that the x-scale component should be an integer multiple of $1/x_{\max}$, and the y-scale component should likewise be an integer multiple of $1/y_{\max}$. Note that this rasterization depends on the model M_t.

Transformations can therefore be represented by four integers; the quadruple (i_x, i_y, j_x, j_y) would represent the transformation $(i_x, i_y, j_x/x_{\max}, j_y/y_{\max})$.

4.3 Predicting the Scale and Location of the Match

We can bound the range of the search on the grid of transformations by predicting the position and scale of the landmark, based on the previous values for its size, range and bearing. Initially, suppose that our estimate of the range is r_t, the size (of the image of the landmark) is l_t, and the x-position of the center of the landmark is x_t. We wish to compute $\tilde{r}_{t+1}$, $\tilde{l}_{t+1}$ and $\tilde{x}_{t+1}$ which are predictions of r_{t+1}, l_{t+1} and x_{t+1}, the range, size and x-position of the landmark after a forward motion of m. The motion will be close to directly towards the landmark.

$\tilde{r}_{t+1}$ is simply $r_t - m$, as we are moving towards the landmark. The predicted value of the size of the landmark and thus the next scale factor can then be calculated as a function of $\tilde{r}_{t+1}$ and m: $\tilde{r}_{t+1} = m/(\tilde{l}_{t+1}/l_t - 1)$, and so $\tilde{l}_{t+1} = l_t(m/\tilde{r}_{t+1} + 1)$; the scale is predicted to be $\tilde{l}_{t+1}/l_t$.

The predicted position of the landmark can also be determined from its previous position. Ideally, the landmark would have been centered in the image, and the robot would have moved directly towards it; in the next image, it would therefore continue to be centered. In practice, though, it will tend to move outward from the image center as the robot moves, as it will not have been exactly centered, and the robot may not have moved directly forwards. Our prediction of its new position is simply $\tilde{x}_{t+1} = r_t x_t/\tilde{r}_{t+1}$, as the drift is proportional to the distance from the camera center. We then use the predicted location of the landmark in order to restrict the search space: we search for a match only in a small portion of the transformation space, surrounding the anticipated position and scale. Since the x and y scales of the landmark will typically be close, we also only consider transformations where they are within 1% of each other. This allows us to correct for small unanticipated effects due to errors in the scale and position estimates, or errors in motion (including possible motion of the landmark).

Given two fractions f_1 and f_2, and some threshold ϵ, we consider all transformations (lying on the grid) inside the current transformation bounds for which $f_\epsilon(T(M_t), I_{t+1}) \geq f_1$ and $f'_\epsilon(I_{t+1}, T(M_t)) \geq f_2$. For each such transformation T, we also compute

1. ϵ_0, the minimum value of ϵ for which $f_\epsilon(T(M_t), I_{t+1}) \geq f_1$
2. $f_{\epsilon_0}(T(M_t), I_{t+1})$
3. ϵ'_0, the minimum value of ϵ for which $f'_\epsilon(I_{t+1}, T(M_t)) \geq f_2$
4. $n_{\epsilon'_0}(I_{t+1}, T(M_t))$, the actual number of points of I_{t+1} which lie inside $M_t^{\epsilon'_0}$.

Such a search can be performed very efficiently using methods described in [3]. These transformations are then ranked based on these four items, in lexicographic order (i.e. ϵ_0 is most important, followed by $f_{\epsilon_0}(T(M_t), I_{t+1})$, etc). All but the top 10% are discarded. The remaining transformations are grouped into connected components in transformation space (using 80-connectedness adjacency on the four-dimensional grid: each transformation is considered to be connected to all those transformations whose parameters differ by no more than one in each dimension) and the centroid of each component is found. Within each component, each of these values is averaged over all the matches in that component, giving four average values. The components are then ranked (lexicographically) by these four values; the best component is then chosen as the one most likely to contain the image of the landmark, and its centroid determines the new values for the size and position of the landmark, and thus its range and bearing.

If the landmark is not found in this search (i.e. no transformations satisfying these criteria can be found inside the search range), then the range of the search is enlarged, f_1 and f_2 are lowered, and the search is repeated until the landmark is located.

Once the landmark has been located in I_{t+1}, we build M_{t+1}, which will be used to find the landmark in the next time step. This is done by simply cutting out a rectangular box around the location in I_{t+1} where M_t was found.

Initially, ϵ, f_1 and f_2 are set to fixed values (currently 2 pixels, 90% and 85% respectively). After each time step, the average values of ϵ_0 etc. generated by the best component are used to determine the values of ϵ etc. to be used in the next time step.

5 The Overall Navigation Method

Initially the robot grabs an image of the visual field which is presumed to contain the landmark. An initial model is given, which consists of a set of two-dimensional intensity edges (as discussed above this model can be extracted from an image of the scene, or can be obtained in some other manner such as from a modeling system). The navigation process is divided into an initialization stage and an approach stage. In the initialization stage a crude approximation to the range is obtained and the focal length of the camera is estimated. In the approach stage, the robot moves in a direction that is the current best estimate of the heading to the landmark, and updates the range and focal length values, as well as computing some other quantities discussed below. The approach stage is divided into steps, where at each step the robot grabs an image of the visual field and matches a model from the previous step to this image, in order to update the range and bearing estimates.

The initialization stage consists of two pre-programmed movements that are used to calculate the range to the landmark and the focal length of the camera. Before the robot executes either of these movements, an image is grabbed and the initial model is matched to this image, in order to determine the initial model size. The robot then moves forward a known distance, currently one meter, and grabs another image. The initial specified model is matched to this image and which yields a location and size of the model. This size and the size in the initial image determine the scale change s of the model over a one meter motion. The range to the landmark is calculated from s and the motion using Equation 1. A new model is then generated by selecting the rectangular portion of the image where the landmark was found.

The robot then executes the second of the pre-programmed initialization movements, in order to estimate the focal length of the camera. The robot rotates a known amount, currently 8 degrees, in the direction that would result in the landmark being closer to the center of the image frame and grabs an image. After matching the model to the image, the robot uses the amount of rotation and the landmark's change in x-position in the image to calculate an estimate of the camera's focal length, as described by Equation 3.

In general the object will not be centered in the image frame after the two initial movements. Therefore, before beginning the approach, the robot calculates the angle needed to rotate in order to center the landmark, using the focal length of the camera and the number of pixels that the object is off center by Equation 2. After rotating this amount, the robot grabs an image and searches for the landmark to be sure that the centering worked and to recalculate the focal length of the camera.

At this point, the landmark is centered in the frame, and the robot enters the approach phase in which it proceeds towards the object. Ideally, moving forward when the landmark is centered in the image frame should always keep the robot on a direct course, until the robot eventually makes contact with the landmark. However, factors such as the object not being completely centered and the robot not moving exactly as commanded may cause the landmark to drift off center in the image. During the approach, the landmark may drift in the image frame, signifying that the robot is slightly off target. When the drift is more than some threshold, currently ± 20 pixels in a ± 180 pixel image, away from the camera center, it is necessary to stop the forward approach and recenter the landmark.

During each step of the approach process, the scale change between the current image and the previous image is computed. Rather than simply computing the range from this scale change, the scale changes for the last several images are combined in order to produce a more accurate estimate of the range. That is, the overall scale change across the sequence of images and the total motion during that sequence are used to compute the range from Equation 1. All of the images taken since the last rotation of the robot are used in this process.

5.1 Concurrent Moving and Matching

In practice, the amount of time required for matching is comparable to the amount of time the robot takes to move one meter. Furthermore, the moving and matching are independent, and can occur in parallel, as there is no need for the robot to wait for the match to finish. Given that the object has been centered before the approach begins, moving forward is the correct motion to perform almost all the time. However, if the robot is moving forward while matching and is slightly off course, then the robot will unknowingly continue off course until the match has completed. Therefore, there is a risk accompanying this strategy which must be accounted for. Placing a limit on the distance traveled since the most recent completed match limits the amount that the robot can drift off course. Once the robot has moved a certain amount since the last successful match, it will halt and wait for the match to complete. The maximum "safe" traveling distance between matches is a function of the range to the landmark.

If the most recent match has completed while the robot is still moving forward, and the match reveals that the robot continues on target, then the robot grabs a new frame while moving, resets the odometer of distance traveled since the last frame, and continues its approach. Meanwhile, the next model is generated and the image frame is searched from the the landmark.

With the concurrent moving and matching, if the most recent match places the landmark more than some threshold away from the center of the frame, recentering can no longer be done with respect to this frame, since the robot has continued moving off target since the frame was grabbed. Therefore, in order to recenter the landmark, the robot halts, grabs another frame, and identifies the landmark's new position and scale. From this updated information about the object's position in the frame, the robot can rotate the appropriate amount to recenter the object (as described above) and then resume the approach. Given that the distance traveled between each two successive matches is not too large, the landmark should still be visible in the frame.

5.2 Obstacle Avoidance

The obstacle avoidance algorithm was added to handle the case when there are obstacles in the robot's path to the landmark. Since our navigation is completely visually guided, we assume that the obstacles do not interfere with the camera's line of sight to the landmark, but just impede the robot's path. We also assume that the surfaces of the obstacles that the robot bumps into are flat.

As the robot is moving forward, it will stop and register an impediment if one of the contact bumpers has been pressed. The robot uses its estimate of range to determine whether the impediment is an obstacle or the landmark. If the calculated range is greater than some value, currently one meter, then the robot assumes the impediment is an obstacle and begins its obstacle avoidance routine. If the range from the target is less then one meter, the robot assumes that it has arrived, and returns a successful hit.

The goal of the obstacle avoidance is to successfully move around the obstacle, while keeping the landmark in the image frame. The first step in the obstacle avoidance routine is for the robot to align itself with the obstacle. The robot we used has eight contact bumpers attached around the front portion of the cylindrical body. The robot is considered aligned when the robot's heading is normal to the face of the obstacle. This active alignment can be achieved by rotating while remaining in contact with the object until only the middle two contact bumpers are pressed [4].

At this point, the robot backs up a small amount and rotates 90 degrees in the direction that would still be making progress towards the landmark. The robot begins repeatedly pinging the sonar on the side where the obstacle lies, as it translates forward. By the flat surface assumption, the sonar on the side of the robot facing the obstacle will register a large change in distance once the robot has cleared the obstacle.

If another obstacle is encountered as the robot is translating to clear the first obstacle, then the robot rotates 180 degrees, and attempts to move around the obstacle in the opposite direction in a similar way. If the robot encounters yet another obstacle as it attempts to clear the box in this direction, then the robot is boxed in (in that it must actually move away from the target to get to it), and gives up.

During this process, the robot maintains the angle and distance traveled since first contact with the obstacle. From these parameters and the previous range to the landmark, the robot calculates the new range and bearing to the target, and rotates to that bearing. The robot then grabs an image, locates the landmark, recenters, and resumes the approach.

5.3 Partial Matching

As the robot approaches the object, the landmark gets larger in the image frame, and at some range portions of the landmark may begin to fall outside the image. When this happens, the model from a given frame will not match the image at the next frame very well, because parts of the object will be missing. At this point, the matching process is changed to allow for partial matches of the model (by lowering the fraction of model points that must be near points of the image, recall this fraction f was discussed in Section 4).

The navigation system goes into the partial matching phase once the robot has gotten close enough to the object that part of it lies to the boundary of the image (within 15 pixels of any border). In this phase, the matching method only requires an initial fraction of $f = .8$ when looking for the model in the image. In addition, the robot moves smaller intervals between successive images (one half meter rather than one meter). The motivation for this is that once the robot is close enough that only part of the object is visible, the image of the object may change shape greatly with a significant sized motion.

6 Some Experimental Results

Here we describe the results of some experiments we have performed using this technique. In each case, the robot was started some distance away from the landmark, with the landmark in view. The user then outlined a rectangular area which enclosed the landmark; this was used as the first model.

The first example shows the robot navigating to its target, a box of paper. Figures 3 and 4 show the images used by the robot. Each line contains three images: the image acquired by the robot at that time step (I_t), the model from the previous time step overlaid on the image at the location where it was found (M_{t-1} overlaid on I_t), and the model extracted from the image (M_t). There is also a description of the actions the robot performed as a result of these images (i.e. the actions it performed between taking this image and taking the next). The distances and angles given here are taken from the robot's internal odometry. Note that because the motion and computation are overlapped, the robot is in motion most of the time; commanded actions therefore may not take place immediately; for example, if the robot is commanded to stop then rotate, it will actually continue forward for a short time before doing so.

The first line does not have a previous model, so no overlay is shown; the last line was taken when the robot was in contact with the target (as reported by the bumpers) and so no model was extracted.

The second example, shown in Figures 5 and 6 shows a sequence where the robot had to navigate around two obstacles before reaching the goal.

The time between images in these examples was typically around 5 seconds, in the cases that the robot was able to move straight forward. The matching also took about 5 seconds in most cases, and so the results of the matching were generally available when the robot finished its planned movement; this meant that it often was able to begin the next movement without actually coming to a stop.

If the robot encountered an obstacle and had to navigate around it, this of course took more time. The robot also needed to pause after a rotational correction, since it had to ensure that the landmark was properly centered before proceeding.

7 Summary

We have presented a method for using visual information to navigate a mobile robot to a landmark in its visual field. The method operates by comparing two-dimensional edge images in order to recover the change in location and size of the landmark in the image as the robot moves. The change in image location is used to keep the landmark centered in the robot's visual field (and thereby keep the robot moving straight towards the landmark). The change in image scale is used to estimate the range to the landmark (given that the robot moves straight towards the landmark).

The method does not make use of absolute world coordinates, or of any three-dimensional information. The robot always maintains an estimate of the range and bearing to the landmark from the last place that a picture was taken, and updates that estimate after each successive image is obtained. The method has been used in our laboratory to control a mobile robot, and some examples of its operation were presented. The matching technique is fast enough (on a SPARCstation) that it can run concurrently with them motion of the robot. Overall, the method is quite simple, not requiring complex representations of objects, or accurate calibration of the camera system.

References

[1] J.F. Canny. A computational approach to edge detection. *IEEE Trans. Pat. Anal. and Mach. Intel.*, 8(6):34–43, 1986.

[2] Enrico Grosso, Missimo Tistarelli, and Giulio Sandini. Active/dynamic stero for navigation. In *Proc. 2nd European Conf. on Computer Vision*, pages 516–525, Santa Margherita Ligure, Italy, May 1992.

[3] D.P. Huttenlocher and W.J. Rucklidge. A multi-resolution technique for comparing images using the Hausdorff distance. In *Proc. Computer Vision and Pattern Recognition*, pages 705–706, New York, NY, 1993.

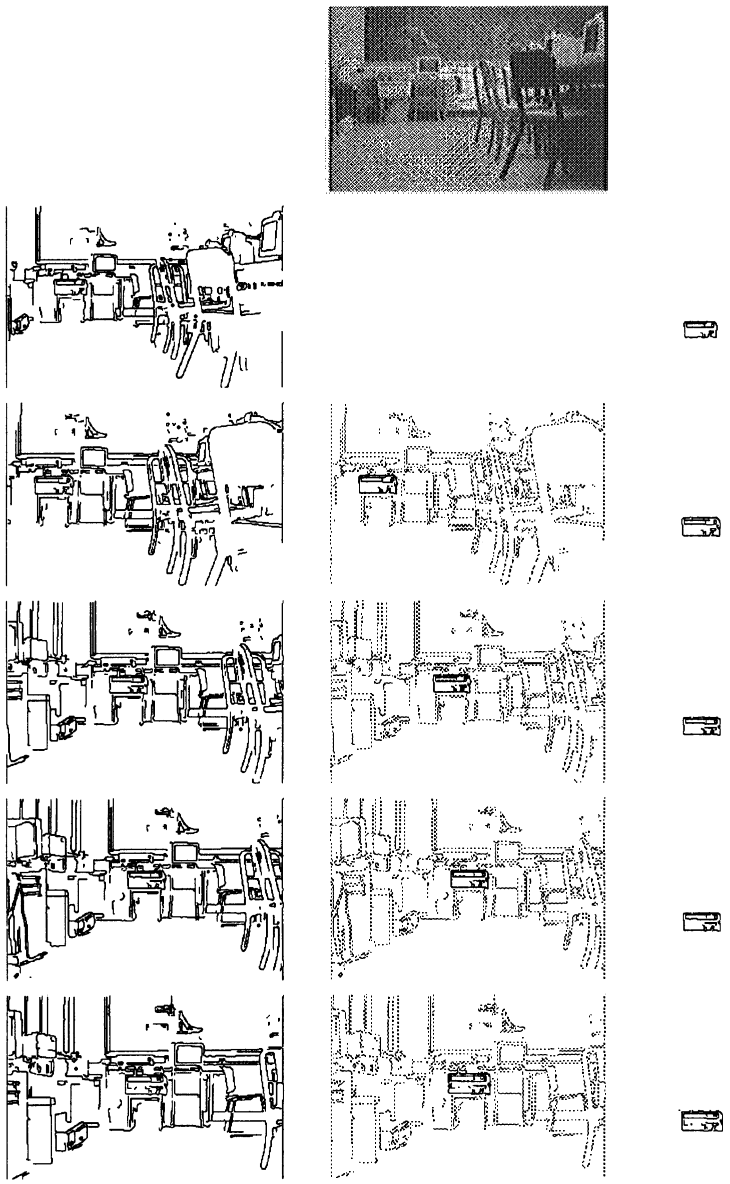
[4] J. Jennings and D. Rus. Active model acquisition for near-sensorless manipulation with mobile robots. In *Proc. IASTED International Conference on Robotics and Manufacturing*, Oxford, England, September 1993.

[5] Jonathan Rees and Bruce Donald. Program mobile robots in Scheme. In *Proc. IEEE International Conference on Robotics and Automation*, Nice, 1992.

[6] Carlo Tomasi and Jianbo Shi. Direction of heading from image deformations. In *Proc. Computer Vision and Pattern Recognition*, pages 422–427, New York, NY, 1993.

[7] Z. Zhang and O.D. Faugeras. Building a 3d world model with a mobile robot. In *Proc. 10th Intl. Conf. Pattern. Recoc.*, 1990.

[8] J.Y. Zheng, M. Barth, and S. Tsuji. Qualitative route scene description using autonomous landmark detection. In *Proc. Third International Conference on Computer Vision*, pages 558–562, Osaka, Japan, 1990.



The original image

The first edge image, and the original model. The robot moves forward 1m to estimate the range.

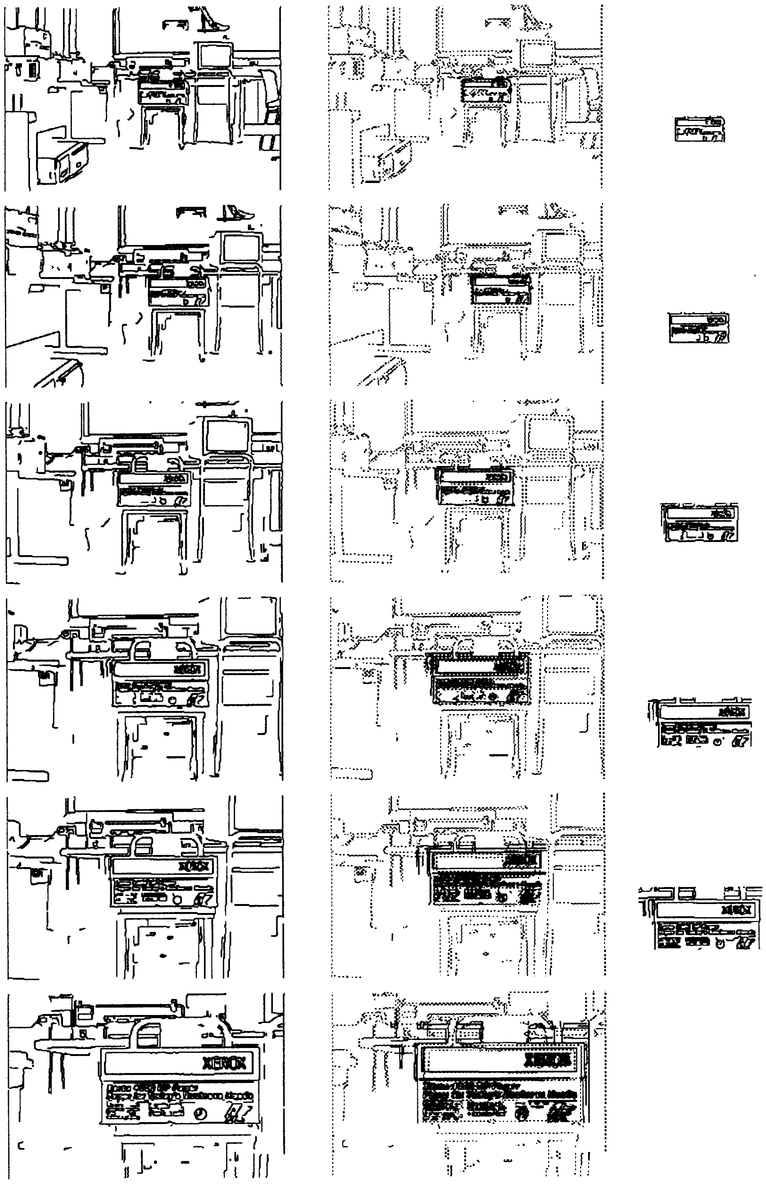
The robot rotates 8° to estimate the camera focal length

The robot rotates 2° to center the landmark.

The robot moves forward 0.9m.

The robot moves forward 1m.

Figure 3: The first example, part one.



The robot moves forward 1m.

The robot rotates $-4°$ to center the landmark

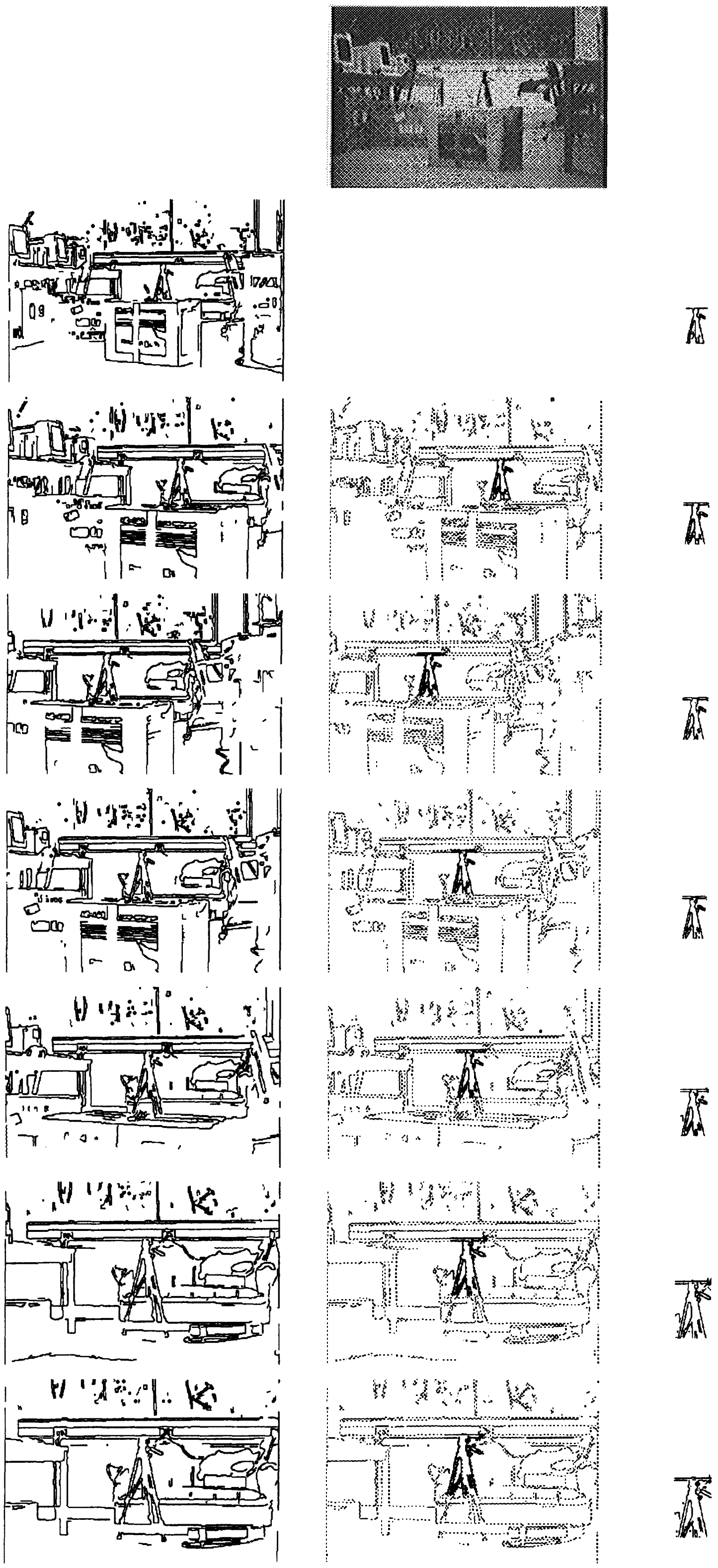
The robot moves forward 1m.

The robot moves forward 0.9m

The robot moves forward 1m.

The robot moves forward 1.0m. It reaches the landmark soon after this image is taken.

Figure 4: The first example, part two.



The original image

The first edge image, and the original model. The robot moves forward 1.1m to estimate the range

The robot rotates $-8°$ to estimate the camera focal length

The robot rotates $4°$ to center the landmark.

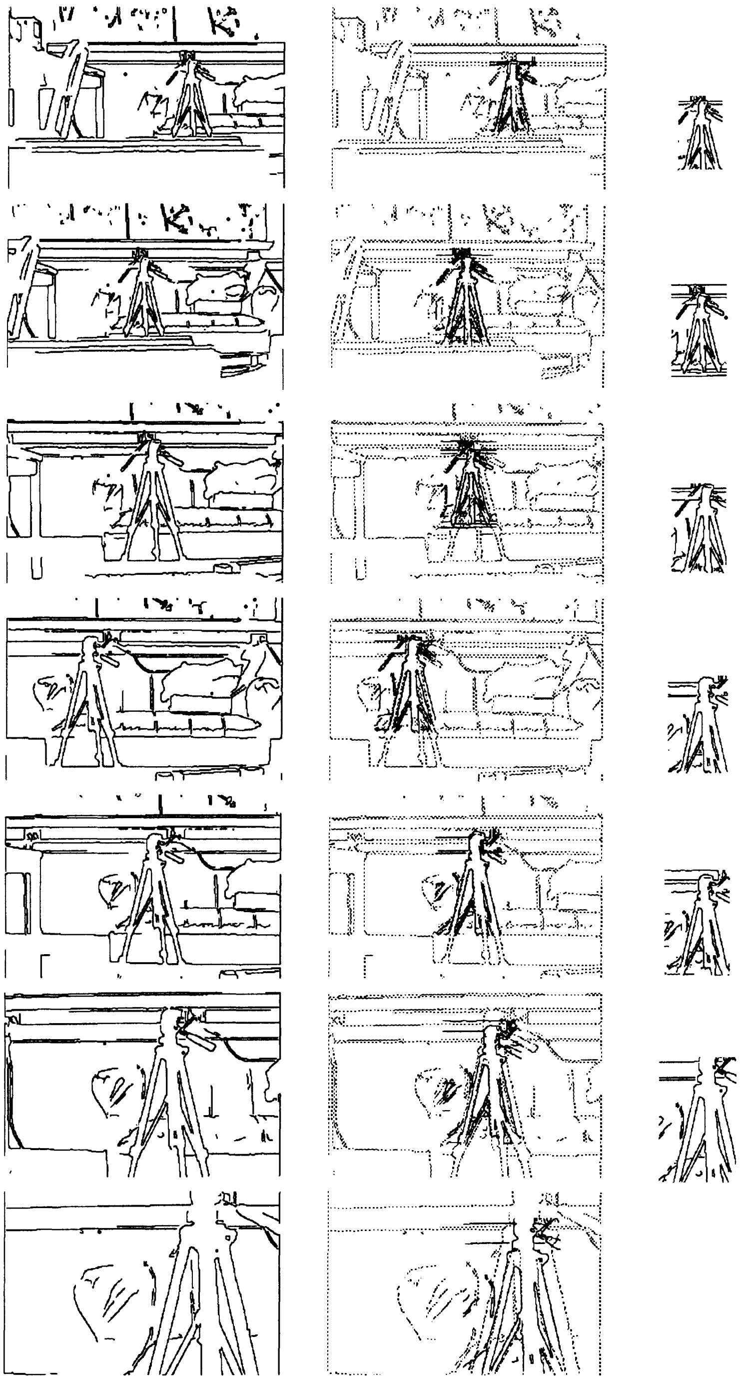
The robot moves forward 1m.

The robot moves forward 1m.

The robot moves forward 0.1m and contacts the first obstacle.

The robot navigates around the first obstacle.

Figure 5: The second example, part one. Note the obstacles visible in the first frame.



The robot rotates −5° to center the landmark.

The robot moves forward 0.9m and contacts the second obstacle.

The robot navigates around the second obstacle.

The robot rotates 7° to center the landmark.

The robot moves forward 1m

The robot moves forward 0.6m

The robot moves forward 0.6m and reaches the landmark.

Figure 6: The second example, part two.