Real-Time Vision-Based Robot Localization

MS-CIS-90-79
GRASP LAB 240

Sami Atiya
(Universität Karlsruhe)

Greg Hager
(University of Pennsylvania)

Department of Computer and Information Science
School of Engineering and Applied Science
University of Pennsylvania
Philadelphia, PA 19104-6389

October 1990

Real-Time Vision-Based Robot Localization

Sami Atiya
Universität Karlsruhe (TH)
Institut für Algorithmen und Kognitive Systeme
and
Fraunhofer-Institut für Informations- und Datenverarbeitung (IITB)
Fraunhoferstr.1
7500 Karlsruhe 1, FRG

Greg Hager
University of Pennsylvania
GRASP Lab - Room 301C
3401 Walnut St.
Phila., PA 19104/6228

September 14, 1990

Abstract

In this article we describe an algorithm for robot localization using visual landmarks. This algorithm determines both the correspondence between observed landmarks (in this case vertical edges in the environment) and a pre-loaded map, and the location of the robot from those correspondences. The primary advantages of this algorithm are its use of a single geometric tolerance to describe observation error, its ability to recognize ambiguous sets of correspondences, its ability to compute bounds on the error in localization, and fast performance. The current version of the algorithm has been implemented and tested on a mobile robot system. In several hundred trials the algorithm has never failed, and computes location accurate to within a centimeter in less than half a second.

Submitted to the 1991 International Conference on Robotics and Automation. Please address all correspondence to the first author at the address listed above or to greg@grip.cis.upenn.edu.

1 Introduction

A core problem in robotics is the determination of the location (sometimes referred to as the *pose*) of a mobile robot in its environment using passively acquired sensor data. This process, often referred to as *localization*, is a basic operation which must be successfully carried out in complex environments using imprecise, contaminated data. For these reasons a solution to the localization problem must be

- Tolerant of errors in measurements.
- Tolerant of falsely detected features or landmarks.
- Simple enough to perform quickly and efficiently.

Furthermore, the solution should have a low fundamental complexity so that good performance can be maintained over a wide range of situations.

We can break the localization problem into two distinct but closely related subproblems:

1. Establishment of the correspondence between sensor observations and known landmarks in the surrounding environment.
2. Determination of robot location relative to an external, fixed coordinate frame using recognized landmarks.

In addition to providing information required for establishing the pose of the robot, the solution to the first problem may provide useful information for planning or navigation. For instance, the landmarks may correspond to a door or other opening which must be navigated, or they may indicate a docking site or other task-relevant structure.

The localization problem must be solved in two modes: static and dynamic. In the static case, the system is presented with sensory data, must determine a labeling for this data, and then, from this information, compute its pose in the world coordinate frame. In dynamic mode, it may be assumed that the system has solved both problems in a previous step, and that the new situation is a slight perturbation of the previous situation. This "temporal coherence" provides a constraint which, when properly exploited, can make a solution to the dynamic problem simpler and more reliable than a solution to the static problem. There are several experimental and commercially available systems which are able to accomplish static and/or dynamic localization. For example, [Krishnamurthy et.al., 1988] uses structured light, sonar, and active and passive vision to recognize landmarks on walls, ceilings, and other surfaces in the environment. [Warnecke, 1987; Crowley, 1989; Leonard & Durrant-Whyte, 1989] determine the position of a robot relative to a stored map from sonar data. However, as these examples indicate, most commercial and many experimental systems depend on some type of artificial "beacons" in the environment and/or employ active or intrusive sensing.

It is our goal to employ non-intrusive sensing, in this case vision, to solve the localization problem in typical, unaltered indoor environments. Examples of solutions to the static problem using visual data include [Sugihara, 1988; Krotkov, 1989a] while solutions in the dynamic case include [Ayache & Faugeras, 1987; Chatila & Moutarlier, 1989]. We note, however, that most solutions to the dynamic problem inherently assume a "good" prior solution and expect only small perturbations from this prior solution. If a good prior solution does not exist, the methods cited above can fail, and moreover it is difficult to automatically detect when such failures occur. Thus, static localization is normally required to provide an initial solution and is therefore fundamental to the solution of the dynamic case.

In this article we describe:

- The development of simple methods for determining data/landmark correspondences and robot location which
  - Can be implemented in real time (the current version requires less than 0.5 seconds for a solution to recognize landmarks and localize the robot).
  - Treats errors using tolerances, thereby avoiding the difficult issues surrounding the time-series modeling required to employ statistical techniques.
  - Computes both robot location *and* a geometric, worse-case accuracy.
  - Determines when ambiguities arise which make it impossible to solve the problem.
- The presentation of simulated and real tests of these methods which indicate that it is very robust to observation error and geometric ambiguity.
- The extension of our results for the static localization problem to the dynamic localization problem.

In addition, we describe how these algorithms have been integrated into a working mobile robot navigation system. Finally, we note that many of the ideas and methods developed in this article are, in fact, independent of vision and could be used with other sensing equipment.

In the next section we formulate the localization problem precisely. Following that, we present our solution and analyze its complexity as well as its limitations. In the fourth section we describe a mobile robot system and present the results of several simulations and experimental trials. In the final section we discuss our results and describe a set of problems that we plan to address in future research.

2 Problem Formulation

Following Krotkov and Sugihara, let $p_1, p_2, \ldots p_n$ denote the positions of fixed landmarks or beacons expressed in a fixed two-dimensional world coordinate system W. Let $\Gamma = (x, y, \theta)$ parameterize the location of a local robot coordinate system R with respect to W. We assume that the robot is equipped with a camera system capable of taking stereo images. Let $o = (o^l, o^r)$ denote the horizontal position of a vertical edge in two camera images which we label "left" and "right", and let $o_1, o_2, \ldots, o_m$ denote a series of these observations. We will model the imaging of points by the camera in a coordinate system C using a linear spatial transformation ${}^CT_W = {}^CT_R{}^RT_W$ followed by a nonlinear imaging transformation $I(\cdot)$ which computes both the left and right image locations. Unless otherwise noted, all positional quantities (including robot location) will be expressed in millimeters and all angles will be expressed in degrees.

A *correspondence* between an observation o and a world point p is a tuple $\lambda = \langle o, p \rangle$, and a labeling Λ is a set of such correspondences. A labeling is *consistent* if each observation is in correspondence with no more than one feature in the world coordinate frame.

We now precisely formulate the localization problem:

Problem 0: Given n points $p_1, p_2, \ldots, p_n$ in a world coordinate system and m observations $o_1, o_2, \ldots, o_m$ taken at two camera positions with known relative relationship, determine if there is a unique, consistent labeling Λ and fixed pose Γ such that $o = I({}^{C}T_{W}p)$ for all $\langle o, p \rangle \in \Lambda$.

This problem is formulated for the ideal case where the observation of landmarks is error free. In practice we must be prepared to accommodate errors in edge localization as well as culling observations which have no corresponding landmark in the map. To accommodate the former we will introduce an observation *tolerance*, ϵ, indicating that the matching criteria must be unique and satisfiable up to this tolerance. This modification leads to the following reformulation of the ideal problem as two separate subproblems:

Problem 1: Given n points $p_1, p_2, \ldots, p_n$ in a world coordinate system and m observations $o_1, o_2, \ldots, o_m$ taken at two camera positions with known relative relationship, determine if there is a unique, consistent labeling Λ so that for *some* fixed location Γ,

$$(o_i - I({}^{C}T_{W}p_i)) \in [-\epsilon, \epsilon] \times [-\epsilon, \epsilon]$$

for all $\langle o_i, p_i \rangle \in \Lambda$.

Problem 2: Given a labeling Λ as described above, determine the complete set of robot positions, $\mathcal{P}$, consistent with Λ.

We note this problem formulation differs from Krotkov's and Sugihara's in that we assume information from two camera positions with known relative relationship. This allows us to explicitly compute depth. Furthermore, we explicitly include observation error and observation of non-landmark points in our problem formulation.

3 Our Solution

Our approach to the problem is to transform both observed data and stored map points into a representation that is invariant to translation and rotation and thereby permits direct comparison of observed and stored entities. The original motivation for this approach came from [Richter, 1986] where the labeling of star fields was done from sighting data. The idea of invariant transformation is quite general and appears in many vision applications. It is the basis of the well-known Hough transform techniques for parameter determination, though our algorithm should not be confused with Hough-based methods as we do not quantize the parameter space or make explicit use of accumulation techniques.

In overview, we first note that any three non-colinear points in the plane determine a triangle with three angles α, β, γ and three sides L, R, B of length l, r, b, respectively (see Figure 1). These six values are translation and rotation invariant, and therefore independent of the coordinate frame in which the points are expressed. Hence, comparison of these quantities for three points expressed in the world frame with the corresponding values for three points expressed in the camera frame can be employed to determine if the two clusters of points lie in the same geometric configuration relative to one another. Furthermore, we can incorporate tolerances on lengths and angles based on a given observation error tolerance and thereby make the comparison tolerant to observation errors.

Thus, an algorithm for determining the solution to Problem 1 is to store a list of angles and distances between the points in a map of the environment. At runtime, for every combination

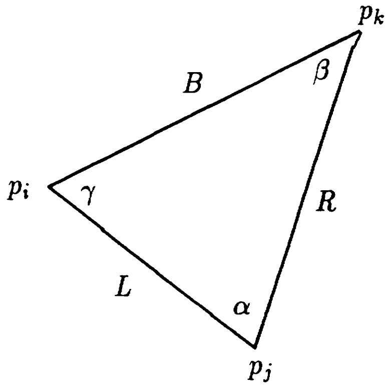

Figure 1: Our triangle labeling conventions.

of three stereo data pairs, we can compute the same quantities and compare the network of observed points, now encoded as *intervals* on angles and distances, with the pre-stored map. From this comparison, we can determine all possible correspondences between observed and stored points up to the specified observation error. Given a set of matched points, we can determine robot location and use the tolerance on observation to compute an error bound on that localization.

We now describe the complete matching process in detail.

3.1 Computing Correspondences

Imaging Model and Calibration Our camera system, due to the nature of its use, requires a very wide field of view and consequently the lens suffers from significant distortion effects. Thus, rather than using the simple pin-hole imaging model, we include a second-order distortion component [Lenz & Tsai, 1988]. The imaging process can be subdivided into four steps:

1. the geometric transformation $e = {}^{C}T_{W}p$ depending on six parameters describing the spatial transform ${}^{C}T_{W}$,

2. the perspective transformation of the point $e = (x_c, y_c, z_c)$ into undistorted image coordinates

$$\begin{bmatrix} u_u \\ v_u \end{bmatrix} = \frac{f}{z_c} \begin{bmatrix} x_c - b \\ y_c \end{bmatrix} \tag{1}$$

   depending on the focal length f and baseline b,

3. the mapping of (u_u, v_u) into the radially distorted point (u_d, v_d) depending on the distortion coefficient κ:

$$\begin{bmatrix} u_d \\ v_d \end{bmatrix} = \frac{2}{1 + \sqrt{1 - 4\kappa R^2}} \begin{bmatrix} u_u \\ v_u \end{bmatrix}; \quad R^2 = u_u^2 + v_u^2. \tag{2}$$

4. Conversion from the image coordinates (u_d, v_d) to pixel coordinates (u, v) depending on image center (c_u, c_v) and scale factors (s_u, s_v):

$$u = u_d/s_u + c_u; \qquad v = v_d/s_v + c_v; \tag{3}$$

The camera is mounted on a sliding platform. For simplicity, we assume that the camera mounting is attached so that the camera x-axis is parallel to the slider direction of motion. In this case, the transform ${}^{C}T_{R}$ describes the camera at the slider origin, and slider motions only require adjustment of the x translation parameter by an offset b given by the slider controller.

We take the focal length of the camera to be that given by the manufacturer and the baseline is assumed to be supplied by the slider controller, so the camera calibration process involves the determination of $6+1+4 = 11$ parameters.[1] This calibration is carried out by observing several (approximately 20) points at known positions on two different planes with the robot positioned at the origin of the world coordinate system. Performing a nonlinear least squares regression on the observed data yields the 11 required parameters, and since the robot is aligned with with the world coordinate system, the 6 transformation parameters can be taken to describe ${}^{C}T_{R}$.

In the sequel, the previously mentioned function $I(p)$ is assumed to perform steps 2 through 4 on p using baseline parameters 0 and b and return the u components of the two resulting image coordinate vectors.

Stereo-Based Position Determination We now reduce the imaging geometry to 2 dimensions by assuming that vertical edges are imaged at a fixed height $y_c = 0$ corresponding to the scan line $v = c_v$.

Let $o^r = (u,v)^r$ be the imaging of a vertical edge at the slider origin and $o^l = (u,v)^l$ be the imaging of the same vertical edge at a distance b from the origin. We first invert (3) to get distorted image coordinates (u_d, v_d). We then invert (2)

$$u_u \quad = -u_d/(1+\kappa R_d^2); \quad v_u \quad = \quad v_d/(1+\kappa R_d^2); \quad R_d^2 \quad = \quad u_d^2 + v_d^2 \tag{4}$$

to compute the distortion-corrected image coordinates $(u_u, v_u)^r$ and $(u_u, u_u)^l$. Using (1) we can now compute the (planar) location $e = (z_c, x_c)$ of an observed point in the camera coordinate system as:

$$z_c = \frac{bf}{u_u^r - u_u^l} \qquad\qquad x_c = \frac{z_c u_u^r}{f} \tag{5}$$

Now, recall that we assume to know *a priori* tolerances on observation errors. Examining the above equation we see that, under the assumption that image distortion is locally constant, perturbing u_d^r and u_d^l yields a diamond-like area defined by the four points computed from all combinations of $u_d^r \pm \epsilon$ and $u_d^l \pm \epsilon$ [Solina, 1985; Matthies & Shafer, 1987]. The region enclosed by this polytope, which we will refer to as a *stereo region*, contains all possible locations for the observed point up to sensing error. We note that, in principle, we could perform the same analysis with other parameters such as the distortion coefficient or the baseline, thereby accounting for other sensor inaccuracies. In practice, we have found a single tolerance on observation error to suffice.

Transformation To Triangles For any three given points, p_i,p_j, and p_k, define $L = p_i - p_j$, $R = p_k - p_j$, and $B = p_k - p_i$. Then we can form the six vector of lengths and angles describing

[1] In practice, we in fact use two separate distortion coefficients for both x and y directions, however the values tend to agree closely enough that one value suffices.

the triangle as

$$S_{i,j,k} = \begin{bmatrix} l \\ r \\ b \\ \alpha \\ \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} \|L\| \\ \|R\| \\ \|B\| \\ \cos^{-1}(\frac{R \cdot L}{\|R\|\|L\|}) \\ \cos^{-1}(\frac{B \cdot R}{\|B\|\|R\|}) \\ \cos^{-1}(\frac{L \cdot B}{\|L\|\|B\|}) \end{bmatrix}$$

In the ideal case, this is a redundant description as a triangle is determined by any combination of three values including at least one length.

There is a family of triangles consistent with any given triplet of stereo regions. Due to the nonlinearities and couplings among the variables of the above transformations, there is no simple description of the set of angles and lengths consistent with three stereo regions. Instead, we convert stereo regions into independent *intervals* on each of the six angles and lengths of the associated triangle. The formation of independent intervals results in a loss of information (we neglect couplings among the equations), however by now using all six equations we reduce this loss through redundancy.

Given two regions a and b, the maximum possible distance between points in a and points in b occurs on the vertices defining the regions. The minimum possible distance also occurs at vertices *except* when a perpendicular to a segment of one region can be made to pass through a vertex of the other region. In this case the shortest distance is given by this perpendicular. To solve for minimum distance let b_k be a vertex of region b and a_i and a_j, $j \neq i$ be two adjacent vertices of region a. The minimum distance between the point b_k and the line through a_i and a_j is given by:

$$\min_{\lambda} \|b_k - (a_i + \lambda(a_j - a_i))\| \quad \text{with solution} \quad \lambda = \frac{b_k \cdot a_i}{\|a_j - a_i\|^2}.$$

If λ falls between 0 and 1, we take the minimum distance corresponding to that value of λ, otherwise we take $\|b_k - a_i\|$ if $\lambda < 0$ and $\|b_k - a_j\|$, otherwise. In summary, to compute the maximum distance between regions we consider all combinations of vertices of a and b (4 * 4 operations) and take the largest. To compute minimum distance we compute the above expression for all combinations of vertices taken from a and segments taken b and vice versa (4 * 4 * 2 operations), and take the minimum of those values.

Given three stereo regions a, b and c, the minimum angle between $a - b$ and $c - b$ occurs at the extreme points of all three regions,[2] while the maximum angle sometimes occurs at the extreme points and sometimes occurs by choosing two vertices of a and c and a point along a segment forming the region b. Let a_i and c_j be vertices of regions a and c respectively, and b_k and b_l be two adjacent vertices of region b. Define $s_1 = a_i - (b_k + \lambda(b_l - b_k))$ and $s_2 = c_j - (b_k + \lambda(b_l - b_k))$. Then, the latter maximization problem is

$$\max_{\lambda} \cos^{-1}\left(\frac{s_1 \cdot s_2}{\|s_1\|\|s_2\|}\right) \equiv \min_{\lambda} \frac{s_1 \cdot s_2}{\|s_1\|\|s_2\|},$$

the above equivalence holding for interior angles.

Due to the complexity of the closed-form solution for this minimization we use an approximation to compute maximal angle. For regions a, b, and c with the angle situated at b, we

[2] The only exception is when a single line passes through all three regions, a case which is easy to check for.

compute the angle between the vertices of a and c for each endpoint and ***midpoint*** of the segments comprising b (4*4*8 evaluations). We then take the maximum and minimum of these values for the upper and lower bounds on angle, respectively. In practice, this approximation is quite accurate.

By carrying out these computations for stereo regions a, b, and c, we can compute a closed interval $\mathbf{S}_{a,b,c} = [S^l_{a,b,c}, S^u_{a,b,c}]$ consisting of six components. Three points p_i, p_j, and p_k are consistent with regions a, b and c if $S_{i,j,k} \in \mathbf{S}_{a,b,c}$.

Henceforth, boldface type will be used to distinguish between point values and interval quantities as in the above expression.

Searching For Matches Each interval of lengths and angles computed from observed data will be consistent with some collection of triangles computed from map points. The crucial point of designing a good algorithm is to make the search for these matches as fast as possible.

Our algorithm for determining possible correspondences works as follows:

Initialization:

- For each unique grouping[3] of map points, compute (the point) $S_{i,j,k}$ and add it to a list M. Call the final length of this list q.
- Let $M_{j,i}$ denote the ith element of the jth vector in M. For each element M_j, $j = 1, \ldots, q$ and i, $i = 1 \ldots 6$, add the pair $\langle j, M_{j,i} \rangle$ to a list L^i.
- Sort the elements of each L^i on the second (value) component yielding six lists of pairs of indices and values sorted on value.

Runtime: For each triplet of observed stereo pairs o_a, o_b, o_c,

- Compute (the interval) $\mathbf{S}_{a,b,c} = [l, u]$ encoding the permissible range of the six triangle parameters for the given tolerance ϵ.
- For each coordinate $i = 1, \ldots, 6$
  - Find the first index r such that the value field of L^i_{r-1} is smaller than l_i.
  - Find the first index s such that the value field of L^i_{s+1} is larger than u_i.
  - For each $\langle k_j, v_j \rangle = L^i_j$, $r \leq j \leq s$, mark M_{k_j} as found.
- For each $S_{i,j,k} \in M$ that has been marked six times, add $(\langle o_a, p_i \rangle, \langle o_b, p_j \rangle, \langle o_c, p_k \rangle)$ to Λ

This algorithm computes a set of triplets of matched pairs. Some combinations of these triplets are consistent in their assignment of observed points to world points, and some are not. We partition the set of all matches into maximally consistent categories $\Lambda_1, \Lambda_2 \ldots \Lambda_c \subseteq \Lambda$. For example, we may have the following triplets of matches:

Λ_1	Λ_2	Λ_3
$o_1, o_2, o_3 \Longleftrightarrow p_1, p_2, p_3$	$o_1, o_2, o_3 \Longleftrightarrow p_1, p_2, p_4$	$o_1, o_2, o_3 \Longleftrightarrow p_1, p_2, p_5$
$o_1, o_2, o_4 \Longleftrightarrow p_1, p_2, p_4$		$o_1, o_2, o_4 \Longleftrightarrow p_1, p_2, p_4$
$o_1, o_3, o_4 \Longleftrightarrow p_1, p_3, p_4$		
$o_2, o_3, o_4 \Longleftrightarrow p_2, p_3, p_4$		

[3] Triangles that are merely a permutation of map points indices p_i, p_j, p_k corresponding to a relabeling of the triangle axes is redundant.

We can see that the matches in Λ_2 and in Λ_3 are not consistent with those in Λ_1, and that Λ_1 contains the maximal number of possible matches (4 matching triangles). In general, if $|\Lambda_i|$ is larger than $|\Lambda_j|$, for all $j \neq i$, then we would intuitively expect that Λ_i contains the correct correspondences. More specifically, if we assume that all *correct* matches will be found (which they will be if ϵ is correctly chosen), then there are two possibilities for error:

1. All detected features correspond to landmarks. In this case, multiple consistent sets of matches indicate a structural ambiguity in the map at the given error tolerance level. However, the set of *correct* correspondences can be *no smaller* than the set of correspondences for some structurally equivalent set of points.

2. Some detected features have no corresponding landmark. In this case, the number of correctly corresponding triples can be exceeded by the number of incorrectly corresponding triples only if there are n observed points, $n-k$ of which are "true" points and k of which are false, and there is a structure in the world such that m of the "true" points together with j of the "false" points can be placed in correspondence, and $m+j > n-k$.

Practically speaking, for maps with an even distribution of landmarks, the likelihood of the latter occurrence is very very small. In practice, we have never seen such a case occur. The former case is also seldom a problem except when the observed points are far away (> 4 meters). In this case, the stereo calculation uncertainty becomes very large, and each triplet of stereo points can match many map triplets leading to a large Λ with many multiple matches.

In order to be more tolerant of features which do not actually correspond to landmarks, we count the number of times a landmark is placed in correspondence with an observed point. If this count does not exceed $t = 50\%$ of the *expected* number of correspondences based on the number of detected features, then that landmark and all associated correspondences are removed from Λ. In the example above, the landmark p_5 occurs in only one triple, though we expect four matches from four observed stripes. Consequently this match and thereby the entire category Λ_3 can be removed. The category Λ_2 must be a structural ambiguity which, as expected, is dominated by Λ_1 which is the correct correspondence.

Though heuristic, the threshold t is a very weak criterion. Later we will discuss other methods for disambiguating matches which reduce our reliance on t.

3.2 Determining Robot Position From Matched Points

Our first approach to determining robot pose was to carry out a non-linear least-squares regression based on the imaging equation:

$$o_i = I({}^{C}T_W p_i), \quad \langle o_i, p_i \rangle \in \Lambda$$

where Λ is the consistent labeling computed from the observed data and ${}^{C}T_W$ depends on the robot pose Γ through ${}^{C}T_R$. This regression is carried out using a standard Levenberg-Marquardt gradient descent algorithm [Press *et al.*, 1986]. We choose an initial point which is close to the true point by simply examining the geometry of the observed points and choosing an orientation angle which is approximately correct. The method generally yields an answer within a second.

This approach is unsatisfying because it gives no direct, quantitative indication of the possible errors in robot positioning relative to imaging geometry and observation error. To supply this we have developed a purely geometric solution for pose determination which is consistent with our correspondence solution. As noted by Krotkov [1989b], this is a very difficult task to do precisely as the expression given above is a complex, coupled, nonlinear equation. Our approach is to approximate the true ***solution set*** $\mathcal{P}$ (as defined in Problem 2 of Section 2) by an interval on Γ which must contain that set.

Given two observed points e_i and e_j corresponding to p_i and p_j in the world coordinate system, we can compute robot position by:

1. Determining the orientations of the segment between p_i and p_j and the corresponding segment between e_i and e_j.
2. Determining the rotation that makes the segments parallel.
3. Determining the translation that causes the endpoints of the rotated segments to overlap.

This procedure will yield the robot pose. Moreover, if we examine the extreme values of the above quantities on the stereo regions, we can calculate the extreme values of computed position.

We first determine the angle interval consistent with step 1 above. To do this, the maximal and minimal angles consistent with stereo regions a_i and a_j occur when a segment of length $d = ||p_i - p_j||$ is placed such that one endpoint falls on a vertex a_i of region a_i and the other falls on a segment $(b_k - b_j)$ of region a_j or the dual case. We can determine this intersection by solving the equation:

$$||b_j + \lambda(b_k - b_j) - a_i|| = d$$

If we define $t = b_j - a_i$ and $s = b_k - b_j$, then the above equation yields a quadratic with two solutions for λ. The consistent solutions are those λ such that $\lambda \in [0, 1]$. Defining $r = p_j - p_i$, for any consistent solution, we compute the angle of the observed line, the line in the world, and their difference as

$$\theta_o = \operatorname{atan}\left(\frac{t_y + \lambda s_y}{t_x + \lambda s_x}\right), \quad \theta_d = \operatorname{atan}\left(\frac{r_y}{r_x}\right), \quad \theta = \theta_o - \theta_d.$$

For each pair of matched stereo regions, a_i and a_j, we can compute up to 32 $(2 * 4 * 4)$ values of θ, and then form the minimal interval $\boldsymbol{\theta}_{i,j}$ containing all 32 values. We then carry out this computation for all pairs of corresponding points and take the intersection of the computed intervals yielding:

$$\boldsymbol{\theta}^* = \bigcap_{i=1}^{m} \bigcap_{j=i+1}^{m} \boldsymbol{\theta}_{i,j}.$$

Given a stereo region surrounding a_i, we can easily compute a bounding interval with components $\boldsymbol{s}_i$ and $\boldsymbol{t}_i$ for that region. Then by using interval arithmetic [Moore, 1966; Alefeld & Herzberger, 1983], we calculate the intervals on robot translation as:

$$\begin{aligned} \boldsymbol{x}_{i,j} &= p_{i,x} + \sin(\boldsymbol{\theta}^*)\boldsymbol{t}_i - \cos(\boldsymbol{\theta}^*)\boldsymbol{s}_i, \\ \boldsymbol{y}_{i,j} &= p_{i,y} - \sin(\boldsymbol{\theta}^*)\boldsymbol{s}_i - \cos(\boldsymbol{\theta}^*)\boldsymbol{t}_i \end{aligned}$$

where $p_i = (p_{i,x}, p_{i,y})$ is the match for region a_i

If we have m matched pairs, we compute

$$x^* = \bigcap_{i=1}^{m} \bigcap_{j=i+1}^{m} x_{i,j}, \quad \text{and} \quad y^* = \bigcap_{i=1}^{m} \bigcap_{j=i+1}^{m} y_{i,j}.$$

We note that if the observation errors are stochastic, we can take multiple samples of the same scene, and further reduce the size of these intervals by intersection across observations. In the absence of approximation error, and assuming independence of observation, the size of these intervals will tend toward zero in the limit. In Section 4 we will return to this point.

3.3 Analysis of the Solution

The offline portion of the algorithm consumes $O(n^3 \log(n))$ time to compute and sort all triangle parameters. However, this is only done once and stored as a compiled table with space $O(n^3)$ which is read in at runtime.

At runtime, the search for the lower point of an interval takes $O(\log(n))$. The worst case of the marking phase would be if *all* map triangles are consistent with an observed triangle, yielding $O(n^3)$ marking operations. Thus, the worst case complexity is $O(m(n^3 + \log(n)))$. In practice, by keeping information about the minimal and maximal marked values, the marking and scanning can be done very efficiently and are never carried out over the entire array. We speculate that a more sophisticated set intersection algorithm could reduce this complexity. We also note that the correspondences for each triangle computed from observed data could be computed in parallel with nearly linear speedup in the number of processing elements.

The algorithm we use to partition n matched triangles is $O(n^2)$. In the worst case, this becomes combinatorial, however this limiting case is never reached. Normally $n < 60$.

The determination of robot location requires, in the worst case, $O(m^2)$ computations to determine θ^* and $O(m^2)$ computations to determine position.

Our implementation of this algorithm on a Sun IV yields the following timing figures on the various algorithm components when processing five observed landmarks with 40 stored landmarks:

Stereo Solution	< 0.02 sec
Interval Calculation	0.20 sec
Correspondence Solution	0.20 sec
Position Determination	0.07 sec
Total	0.49 sec

For 25 observed landmarks, the correspondence timing drops to under .066 seconds. The above does not include least squares position determination which consumes approximately 0.5 seconds. We expect that the timings could be improved by at least a factor of two through analysis and optimization of the algorithms.

3.4 Comparison with Related Work

The algorithm we have presented can be used to determine *all* possible consistent interpretations of the data, and can find *all* robot locations consistent with observation. Moreover, it does this *without* using any statistical information about the error in observation and without any prior information. The complexity figures we have cited are comparable with those of Krotkov ($O(mn^4)$) and Sugihara ($O(n^3 \log(n))$ and $O(n^3)$ depending on space requirements).

Recently, there have been several proposals for solving the localization problem using statistical methods [Ayache & Faugeras, 1988; Leonard & Durrant-Whyte, 1989; Crowley, 1989; Chatila & Moutarlier, 1989]. Most of these methods use *prior* knowledge about location and knowledge about system dynamics to predict what information should be observed, and to establish a threshold on maximal deviation from these predictions. Examination of the methods used, however, reveals that a good starting estimate is required to "bootstrap" the system. Our method can produce this starting estimate.

Given a system dynamic description of the form $\Gamma_{t+1} = F(\Gamma_t, V)$, where V is some bounded error, then we can lift F to an *interval function* [Alefeld & Herzberger, 1983] and project the current localization interval magnified by additional dynamic uncertainties into a new frame. All poses computed by the algorithm in the new state must be consistent with this projection. This provides both a running check on correspondence and localization solutions as well as allowing the combination of information over time. Hence, by adding this dynamic description we can also solve the localization problem in the dynamic case. Furthermore, this effectively eliminates our reliance on the the threshold t for dropping false matches.

4 Experimental Results

We have implemented and tested the above algorithms on a mobile robot system under development at the Fraunhofer Institute - IITB and the University of Karlsruhe in Karlsruhe, West Germany. In this section, we describe the system hardware and present the results of both simulation and experimental trials.

4.1 System Description

The system consists of a CCD-camera (resolution 780 by 580 pixels with an 8mm objective) mounted on a controllable slider (positioning precision to 0.02mm). The slider mounted on the Karlsruhe mobile robot (KAMRO) [Rembold, 1988]. The camera is connected to the VISTA real-time image processing system [Paul *et al.*, 1988]. This system is in turn connected to an ethernet, and sends information about images (the positions of vertical stripes) to a Sun IV computer.[4]

The camera was calibrated using the procedure described in the previous section. In justification of our second order model, we note the distortion coefficient for this lens was calculated to range between -0.094 and -0.1112. Without this coefficient, the error in stereo calculation is equivalent to an observation error of several pixels. If we were to account for this error by using a larger tolerance, the number of false matches would grow too large, thereby leading to ambiguous matches in many cases. If we neglect this error, the distortion of edges near the edge of the camera results in the rejection of matches which are correct.

Our stereo solution uses the notion of image continuity [Moravec, 1979; Baker & Bolles, 1988] to follow the path of a vertical stripe from left to right as the slider moves a pre-set distance. The VISTA system allows the frame-rate acquisition of small image slices taken with the slider in motion, and processes the resulting "band picture." This picture is processed with a low-pass (smoothing) and high-pass (differentiation) operator followed by a non-maximal

[4] Sun is a trademark of Sun Inc.

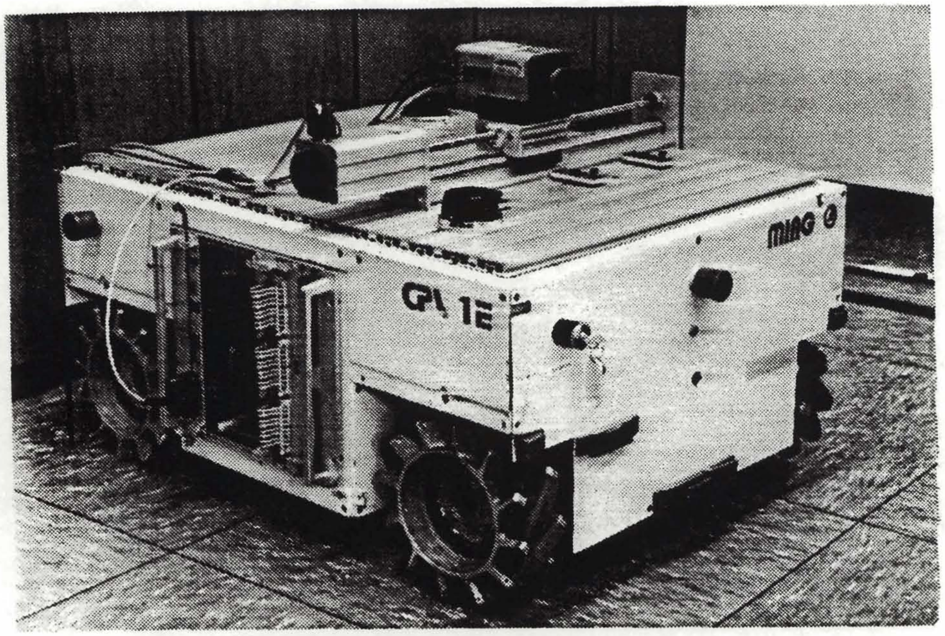

Figure 2: The University of Karlsruhe robot KAMRO with the slider stereo apparatus.

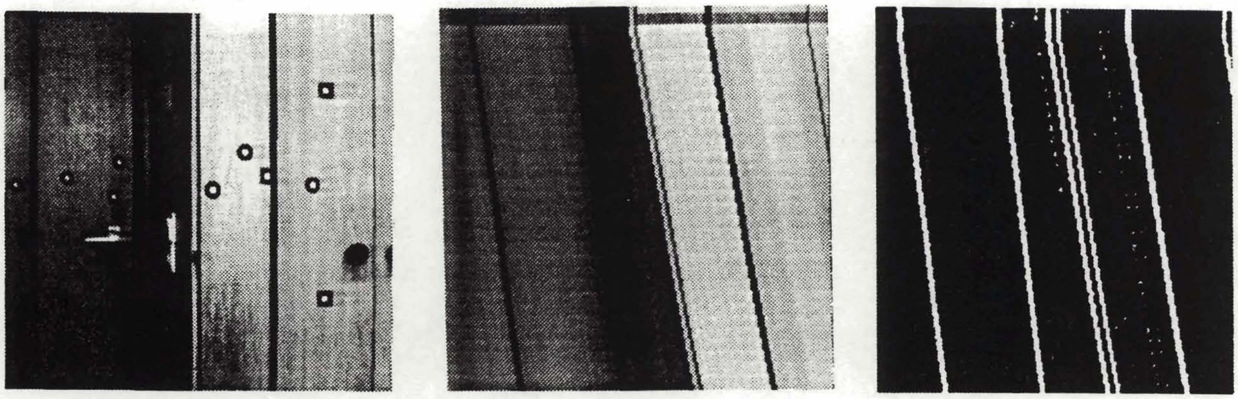

Figure 3: "Band picture" image processing steps. Left, the actual scene; middle, the "band picture"; and right, the processed image.

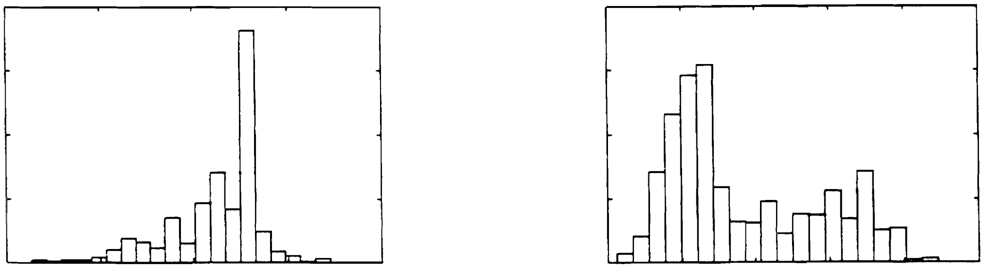

Figure 4: Two example data histograms showing the frequency distribution of the horizontal position of vertical stripes in an image.

suppression leading to a binary picture. A line is fit to contiguous patches of pixels, and from this line we calculate the coordinates of intersection with the upper and lower edges of the picture. In Figure 3 we show a "typical" scene, the resulting "band picture," and the filtered and thresholded picture.

The error in observation was determined by taking several time series of data and looking at the data spread. We assume the error is symmetrically distributed about the "true" value, and therefore read the required tolerance directly from the histogram. Figure 4 shows two representative frequency histograms of x (vertical) position in the picture. In all cases, the error was up to and including one half pixel. The effects of the slider maximal error of 0.02mm considered to act on a point observed at a distance of one meter lead to a 0.02 pixel error in picture coordinates. Finally, instead of finding maximal angles, we approximate by checking all corners and midpoints of diamonds. In order to ensure that all possible matches are found, we must inflate the observation factor slightly. We therefore adopt a tolerance of 0.55 pixels for the combined effects of quantization, approximation error, and slider positioning error. We note that, though we have not decorrelated and tested the data, the series appear to be neither Gaussian nor identically distributed as many of the previously cited methods assume.

In Figure 5 we show the environment (and its landmarks) in which all experiments take place. It consists of 2 rooms with vertical stripes formed by the edges of doors, tables, and desks. There are several large open areas with no vertical structure upon which we have introduced artificial black stripes. Both experiments and simulations will be with respect to this environment.

4.2 Simulation Tests

We have performed several simulation tests of the above-described algorithm to test its robustness against errors in setting the observation error parameter ϵ. In general, the performance of the algorithm can vary greatly depending on the geometry of the observed points. Here we detail two representative cases. The first case shows the "typical" behavior of the algorithm, and the second was chosen to demonstrate its performance in adverse circumstances.

Simulation 1: The robot position was chosen randomly from the interval $x = 560 \pm 100$, $y = 20 \pm 100$, $\theta = -26 \pm 2$ (random position 1 of Figure 5). At each position, we simulated observing the points (drawn slightly larger) in the viewing cone of the robot from this position

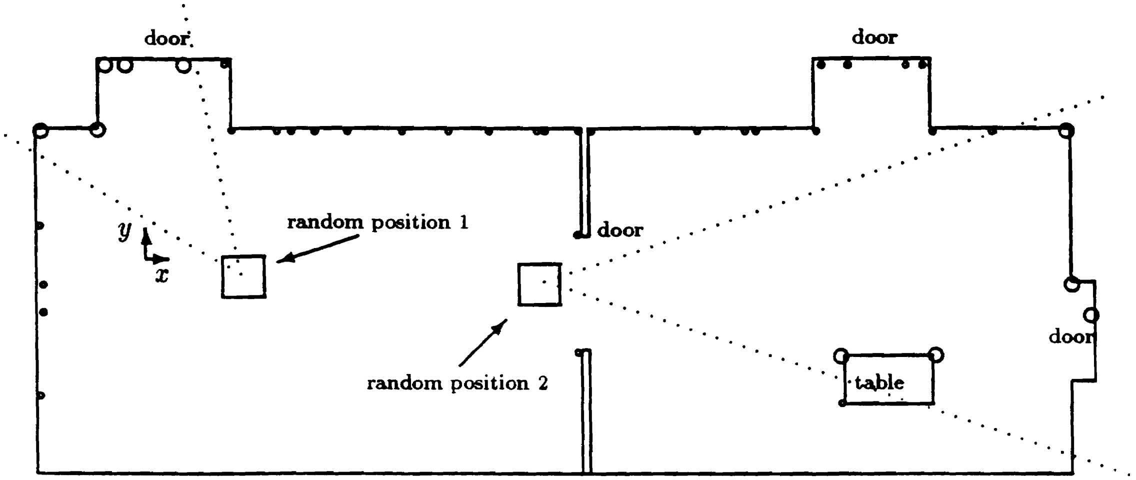

Figure 5: A map of the navigation testing area. The small circles indicated landmarks used by the robot.

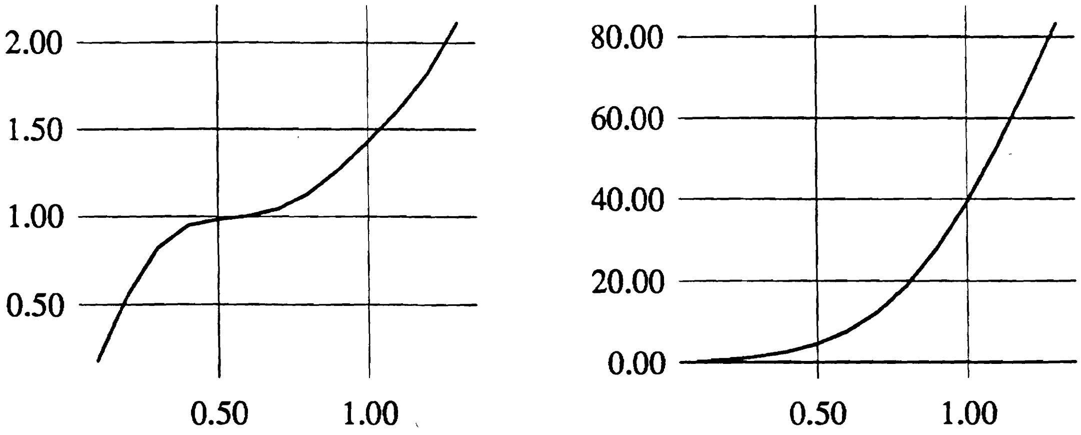

Figure 6: A graph of the number of corresponding points from two simulations.

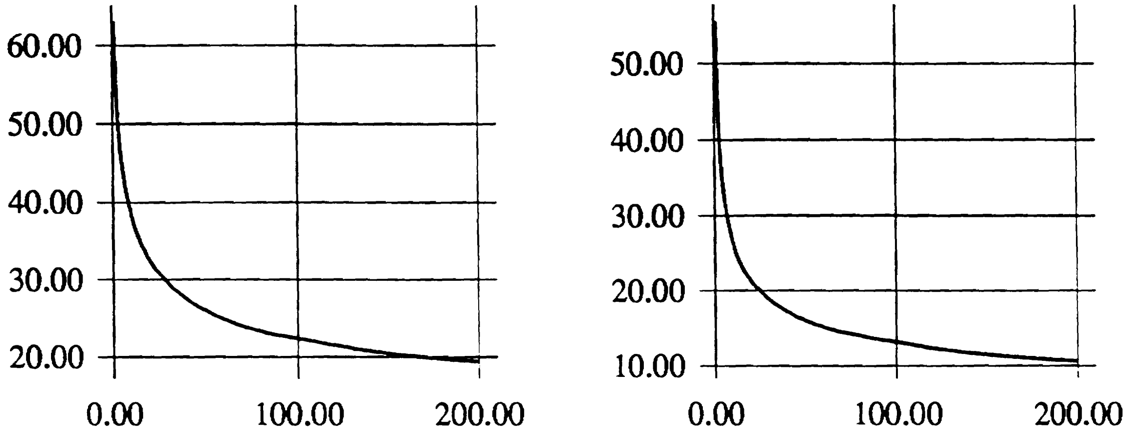

Figure 7: The time evolution of the maximal error in robot localization matched points, x component (left) and y component (right).

under uniformly varying error in the range of ±0.55 pixels. For each case we computed the number of matched points and then calculated its ratio with the ideal number of matched points. In Figure 6, the left graph shows the mean of this ratio as a function of the tolerance ϵ used in the algorithm. We observe that choosing the tolerance in a range of 0.15 pixels about the correct value leads a match ratio within 5% of optimal (1.0). Moreover, the correspondence was correctly solved in all cases up to $\epsilon = 1.3$, more that 200% of the correct value.

Simulation 2: The robot position was chosen randomly from the interval $x = 2500 \pm 100$, $y = -200 \pm 100$, $\theta = 90 \pm 2$ (random position 2 of Figure 5). At each position, we simulated observing the points (drawn larger) within the viewing cone of the robot in this position under uniformly varying error in the range of ±0.55 pixels. In Figure 6 we again graph the mean of the ratio between the number of found and the number of ideal correspondences as a function of the tolerance ϵ. In this case, due to the large distance from the observed points and structural ambiguities in the stored map, we see that even the nearly correct value of $\epsilon = 0.5$ leads to 4.7 times more matches found than are actually possible in the ideal cases. We note, however, that even with this explosive growth the correct correspondence was found for all values of ϵ up to 0.8. By slightly modifying the stored map (removing some ambiguities) the correct correspondence was found up to $\epsilon = 1.1$.

Our conclusion from these tests is that the exact choice of ϵ is not crucial to good algorithm performance, although choosing the smallest value known to be correct will improve its performance in marginal cases.

If the errors in observation across time are stochastic and independent, then continued observation of the same scene will drive the maximal error in positioning (the size of the tolerance intervals on the position parameters) toward zero. Figure 7 shows the rate of convergence over time when observing a typical scene with observation error distributed uniformly in the range $[-0.5, 0.5]$.

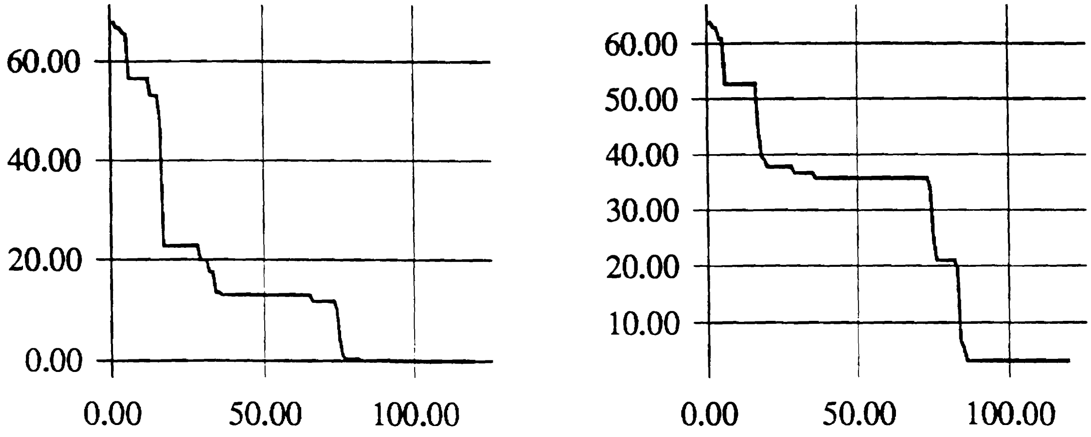

Figure 8: Position convergence with real data.

4.3 Experiments With Real Data

A typical picture taken in the test room may contain from zero to approximately seven observed vertical edges. In practice, we have never seen more than seven detected edges in a picture. If the number of observed edges is less than three, then the algorithm cannot be run (we require a minimum of three points to describe a triangle), and, unless the observed points are very close (within about a meter), we in fact require a minimum of four observed stripes to provide some redundancy. If the robot does not see a scene with four stripes, there is an error-handling procedure which rotates the robot a small amount and takes another picture. This process continues until a satisfactory picture (and in fact a unique correspondence) is found. On the average, we find the pictures contain five vertical edges, zero or one of which is a "false" stripe.

Experiment 1: With the robot in a static position, we continually sample and compute correspondence and position. We have tested this program very thoroughly (several hundred trials), and the correspondence component has almost never failed. For comparison purposes, Figure 8 shows the rate of convergence of the intervals toward a single point. For this single trial, we see that the initial localization accuracy is consistent with simulation, though the convergence is much faster than average for this trial. This experiment was carried out to 1000 observations, however the accuracy was not reduced after the 86th observation. The final accuracy of the solution was 0.5 millimeters in x position, 3.3 millimeters in y position, and 0.07 degrees of orientation. Hand measurements verified that the systematic error in location estimation was less than one centimeter.

We believe that the rate of convergence of this trial is somewhat exceptional as, by examining the histograms of the data, we see that the error is somewhat centrally distributed. Thus, we would expect the average rate of convergence to be somewhat slower than that given by the simulation.

Experiment 2: We have tested the complete system at point to point navigation in the two rooms we have shown. In particular, to move from room to room requires navigating the door opening which requires a positioning precision of 0.5 cm (we use least squares to choose the exact positioning point for these trials.) This particular path has been tested well over 100 times without failure.

In these trials we also use the notion of projecting the current position modified by robot motion and additional robot uncertainty. This allows us to introduce an additional constraint into the correspondence solution, namely that the newly computed position from correspondence must overlap the projection of the previous position. In our experience, with this modification the algorithm has never failed to find the correct correspondence.

The robot has never failed to reach the goal position, suggesting that the error in positioning is less than 0.5 cm. This was again confirmed by comparison of hand measurements of robot position to computed robot position.

5 Discussion and Future Work

We have presented interval-based algorithms for solving the problem of determining the correspondence between observed and previously stored points, and the problem of determining bounds on robot location from matched landmarks. We see the novel points of these algorithms as:

- Depending on only two parameters, the observation tolerance ϵ and the match tolerance t.
- Real-time (less than a half a second) for solution to both problems from raw camera images.
- The computation of quantitative, conservative bounds on localization error.

Furthermore, we have discussed how the solution to the static localization problem can be used to solve the dynamic localization problem.

We see these methods as competitive with the widely-published Kalman filter-based methods in terms of simplicity and execution time. Moreover, we do not rely on any statistical assumptions about the data except for the rate of interval reduction. Examination of our camera data suggests that any type of strong distributional assumptions would be difficult to support.

The computation of solution *sets* is, we believe, an important approach to robotics problems. That is, rather than computing a single point, or a single point with some type of (often heuristic) figure of merit, we compute the complete set of possible solutions modulo, of course, having sufficiently conservative observation uncertainty intervals. In practice, we have found the latter much simpler to determine than the statistical parameters required for methods such as the Kalman filter. Previous work [Hager, 1990a] describes more advanced tolerance-based computing methods. Current work [Hager, 1990b] makes significant advances on these methods.

In the next phase of our work, we plan to mount a second camera on the robot and use feature tracking to support continuous stereo. The methods presented above will be used to locate features and solve the static localization problem. Thereafter, landmarks will be tracked and a running location estimate will be computed. We expect to test both Kalman filter and interval-based methods for accuracy and suitability.

Furthermore, we plan to investigate some aspects of "active" vision. For example:

- If the correspondence cannot be solved and localization accuracy becomes unacceptable, then we can enlarge the stereo baseline. This will improve accuracy, though at the cost of having fewer landmarks common to both images. A wider baseline also complicates the feature tracking.

- When high localization accuracy is needed, a longer baseline allows a more accurate localization to be computed. Additionally, slowing the robot motion increases the effective sampling rate and thereby increases the accuracy of localization.

During the next months we plan to formalize and investigate solution to these problems using decision-theoretic methods [Berger, 1985].

Acknowledgements The navigation system for the mobile robot of the University of Karlsruhe is funded by the Deutsche Forschungsgemeinschaft as part of a cooperative research project on artificial intelligence (Sonderforschungsbereich Künstliche Intelligenz). We would like to thank Prof. Dr. H.-H. Nagel who is responsible for this project for his valuable advice and encouragement. The second author would also like to acknowledge the contribution of the Fulbright Commission which supported him as a visiting scientist for one year at the above-mentioned institutions. The following agencies also contributed to the support of the second author: DARPA Grant N0014-88-K-0630 (administered by ONR), AFOSR Grants 88-0244, AFOSR 88-0296; Army/DAAL 03-89-C-0031PRI; NSF Grants CISE/CDA 88-22719, IRI 89-06770; and Du Pont Corporation.

We wish to thank Jerome Kodjabachian and Max Mintz for their careful reading of earlier drafts of this paper, and Jerome Kodjabachian for the closed-form solution to the angle maximization problem.

References

Alefeld, G. and J. Herzberger, (1983). *Introduction to Interval Computations.* Academic Press, New York.

Ayache, N. and O. D. Faugeras, (1988). Building, registrating, and fusing noisy visual maps. *The International Journal of Robotics Research*, 7(6):45–65.

Ayache, N. and O. D. Faugeras, (1987). Maintaining representations of the environment of a mobile robot. In *Proceedings of the Fourth International Symposium on Robotics Research*, pages 109–121.

Baker, H. and R. Bolles, (1988). Generalizing epipolar-plane image analysis on the spatiotemporal surface. In *DARPA Image Understanding Workshop*, pages 1022–1030, Cambridge, MA, April 6-8.

Berger, J. O., (1985). *Statistical Decision Theory and Bayesian Analysis.* Springer-Verlag, New York, 2nd edition.

Chatila, R. and P. Moutarlier, (1989). Stochastic multisensor data fusion for mobile robot location and environment modelling. In *Proceedings of the 5th International Symposium of Robotics Research*, pages 207–216, MIT Press, Cambridge, MA.

Crowley, J. L., (1989). World modeling and position estimation for a mobile robot using ultrasonic ranging. In *Proceedings of the 1989 IEEE International Conference on Robotics and Automation*, pages 674–680, IEEE Press, Washington.

Hager, G., (1990a). *Computational Methods for Sensor Data Fusion and Sensor Planning.* Kluwer, Boston.

Hager, G. D., (1990b). The use of interval-based bisection methods for sensor data fusion. Technical report in preparation.

Krishnamurthy, B. et.al., (1988). Helpmate: a mobile robot for transport applications. In *Proceedings of the 1988 SPIE Conference*, Spie, Boston.

Krotkov, E., (1989a). *Active Computer Vision by Cooperative Focusing and Stereo.* Springer-Verlag, New York.

Krotkov, E., (1989b). Mobile robot localization using a single image. In *Proceedings of the 1989 IEEE International Conference on Robotics and Automation*, pages 978–983, IEEE Computer Society Press, Washington.

Lenz, R. K. and R. Y. Tsai, (1988). Techniques for calibration of the scale factor and image center for high accuracy 3-D machine vision metrology. *IEEE Trans. Pattern Analysis Machine Intelligence*, 10(5):713–720.

Leonard, J. L. and H. F. Durrant-Whyte, (1989). Active sensor control for mobile robotics. In *Proceedings of the First International IARP Workshop on Sensor Fusion*, Toulouse, France.

Matthies, L. and S. Shafer, (1987). Error modeling in stereo navigation. *IEEE Journal on Robotics and Automation*, RA-3(3):239–248.

Moore, R. E., (1966). *Interval Analysis.* Prentice-Hall, Englewood Cliffs, N.J.

Moravec, H., (1979). Visual mapping by a robot rover. In *Proceedings of the International Joint Conference on Artificial Intelligence 1979*, pages 598–600.

Paul, D., W. Hättich, W. Nill, S. Tatari, and G. Winkler, (1988). Vista: Visual Interpretation System for Technical Applications:Architecture and use. *IEEE Trans. Pattern Analysis Machine Intelligence*, 10:399–407.

Press, W., B. Flannery, S. Teukolsky, and W. Vetterling, (1986). *Numerical Recipes.* Cambridge University Press, New York.

Rembold, U., (1988). The Karlsruhe Autonomous Mobile Assembly Robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 598–603, Philadelphia, PA, April 24-29.

Richter, G., (1986). *Digitale Bildverarbeitung (Theoretische Grundlagen)*, chapter Identifikation von Objekten in Bildfolgen durch invariante Beschriebung der Umgebung, pages 102–107. TU Dresden.

Solina, F., (1985). *Errors in Stereo due to Quantization.* Dept. of Computer Science Report TR-85-34, University of Pennsylvania.

Sugihara, K., (1988). Some location problems for robot navigation using a single camera. *Computer Vision*, 42(1):112–129.

Warnecke, H. J., (1987). Integrierte Sensoraktions-planung als neuartige Sensor- und Steuerungsarchitektur fuer den mobilen autonomen Roboter IPAMAR. *Robotersysteme*, 3:209–217.