Fast Localization of Mobile Robots using Visibility Sectors *

Sooyong Lee sooyong@cs.tamu.edu
Nancy M. Amato amato@cs.tamu.edu
James Fellers jpf9138@cs.tamu.edu

Technical Report 00-002
Department of Computer Science
Texas A&M University
January 17, 2000

Abstract

This paper presents a rapid localization method for mobile robots. Localization, i.e., absolute position measurement, is an important issue since odometer errors render it impossible for any robot to precisely follow a specified trajectory, resulting in a growing difference between the actual configuration and the calculated configuration as the robot travels. Periodic localization is required to correct these errors.

In this paper, we propose a new localization method using range sensor data which is based on simple geometric properties of the environment. In many common situations, information regarding the environment is provided *a priori* for path planning. During processing, the method proposed here utilizes this information to partition the workspace into sectors using simple visibility computations, and a small identifying label is computed for each sector. The localizer analyzes range sensor readings (distances) and extracts characteristic points, which are compared with the pre-computed sector labels to localize the robot, first to a sector, and then to a particular configuration within that sector. Advantages of this two step process are that it is computationally very simple, and that it allows precise localization without any landmarks from any configuration in the environment. This localization method also provides opportunities for the global navigation procedure to analyze and select trajectories in terms of their tolerance to localization errors.

*This research supported in part by NSF CAREER Award CCR-9624315, NSF Grants IIS-9619850, EIA-9805823, and EIA-9810937, and by the Texas Higher Education Coordinating Board under grant ARP-036327-017.

1 Introduction

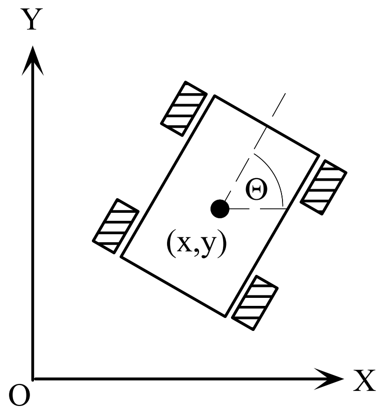


Figure 1: Coordinates

Automatic navigation by mobile robots involves following a planned trajectory taking the robot from a known start configuration to a goal configuration. The desired trajectory can be viewed as a collection of robot configurations, which describe the robot's location (x, y) and orientation (Θ) with respect to the world coordinate system (see Figure 1).

Unfortunately, it is not possible for robots to precisely follow planned trajectories. This is due to unavoidable measurement errors which prevent the robot from precisely determining its current configuration. In particular, mobile robots have encoders to accurately measure/control the rotation of their wheels. Based on its kinematics, the robot's configuration is calculated from the encoder reading. However, the error between the actual configuration of the robot and the estimated configuration gets larger as the robot travels, mainly due to encoder errors caused by the slip between the wheel and the surface [22]. Since the robot cannot follow the given trajectory exactly, periodic localization which resets the error to zero is necessary.

Much work has been done on mobile robot localization. A number of landmark-based or map-based approaches have been proposed which use ultrasonic range or sonar sensors. In [9], a position estimation method based on principal component analysis of laser range data is proposed. The structure of an environment is represented as a family of surfaces in an eigenspace, which is constructed from the principal components of a large number of range data sets. The Generalized Voronoi Graph (GVG) can be used for localization [18] by comparing the coordinates of the current meet point (vertex of the GVG) with the coordinates of previously discovered meet points. If there is a match, then the robot can locate itself on the partially explored GVG. The disadvantage of this approach is that localization is possible only at meet points. A fast localization algorithm for dynamic environments is proposed in [23]; it is based on the definition of a very small landmark, which is calculated from the circular depth function using a sonar sensor. Sonar sensor data is used in [10] to build a multi-level description of the robot's surroundings. The resulting two-dimensional maps are used for path planning and navigation. The localization method in [20] uses sensor data and the current odometry reading to build a series of local perception grids, and then registers the local and the global grids. They assume the mobile robot has a map of its environment represented as an evidence grid (i.e., each grid cell is marked as either free or occupied).

The basic principles of landmark-based and map-based positioning also apply to vision-based positioning which relies on optical sensors instead of ultrasonic range sensors and inertial sensors.

Common optical sensors include photometric cameras using CCD arrays. Visual sensing provides a tremendous amount of information about a robot's environment and it is potentially the most powerful source of information among all the sensors used on robots. Due to the wealth of information, however, extraction of visual features for positioning is not an easy task. The problem of localization by vision has received considerable attention and many techniques have been suggested. Most localization techniques using vision provide absolute/relative configuration of sensors. Techniques vary substantially, depending on the sensors, their geometric models and the representation of the environment. Sugihara [21] considered two cases of point location problems, and [14] followed this approach and formulated the positioning problem as a search in a tree of interpretations. In [5], an algorithm is proposed for localization based on ray angle measurement using a single camera, and then in [7], an odometric sensor was added to landmark-based ray measurements and an extended Kalman filter was used to combine vision and odometric information. A method was developed in [3] that uses a stereo pair of cameras to determine the correspondence between the observed landmarks and the preloaded map and to estimate the two-dimensional location of the sensor from the correspondence. A system for navigation in a partially modeled environment is outlined in [11], and in [4], trinocular stereo and three-dimensional line features are used for building, registering, and fusing noisy visual maps.

Most of the vision-based approaches use landmarks, whose locations are previously defined in world coordinates. Some of the map-based localization techniques only provide for localization in restricted regions of the environment (e.g., only at previously known or discovered landmarks such as meet points). To our knowledge, no localization method has been proposed which exploits pre-computable geometric properties of (partially) known environments to enable localization at virtually any point in the environment using only range sensor data.

1.1 Our Approach

In this paper, we propose a new localization method using range sensor data (i.e., distance measurements) which is based on simple geometric properties of the environment. In many common situations, information regarding the environment is provided *a priori* for path planning. Our goal is to utilize this information for localization. Our localization method is computationally very simple and enables the robot to estimate its configuration at any place in the workspace. During preprocessing, the workspace is partitioned into sectors using simple visibility computations, and a small identifying label is computed for each sector. The localizer analyzes the range sensor readings and extracts characteristic points, which are compared with the pre-computed sector labels to localize the robot, first to a sector, and then to a particular configuration within that sector. This two step process is computationally very simple, and allows precise localization without any landmarks (beacons). Another advantage of this localization method is that it provides opportunities for the global navigation procedure to analyze and select trajectories in terms of their tolerance to localization errors.

The paper is outlined as follows. Section 2 describes visibility sectors, and localization based on visibility sectors is described in Section 3. The global navigator is described in Section 4. Prelimary experimental results are presented in Section 5, and some conclusions are given in Section 6.

2 Visibility Sectors

To facilitate subsequent localization, it is convenient during preprocessing, to partition the environment into sectors that are distinguished by the features of the environment that are visible from the sector.

We assume that a model of the environment is provided *a priori*, that the portion of the workspace relevant for navigation can be considered to be planar, and that obstacles can be modeled by simple polygons. In particular, we assume that a description of the environment is available as an embedded planar graph $G = (V, E)$, $|V| = n$ and $|E| = m$, which decomposes the plane into regions that are *free* (i.e., contain no obstacles) or *blocked* (i.e., filled with obstacles). The union of the free (blocked) regions of the environment is called the *workspace* (obstacle space) and is denoted by $\mathcal{W}$ ($\mathcal{B}$).

Two points p_1 and p_2 are *visible* if $\overline{p_1 p_2}$ does not properly intersect $\mathcal{B}$. The *point visibility polygon* $V(p)$ is the set of all points visible from p [19] (see Figure 2(a)).

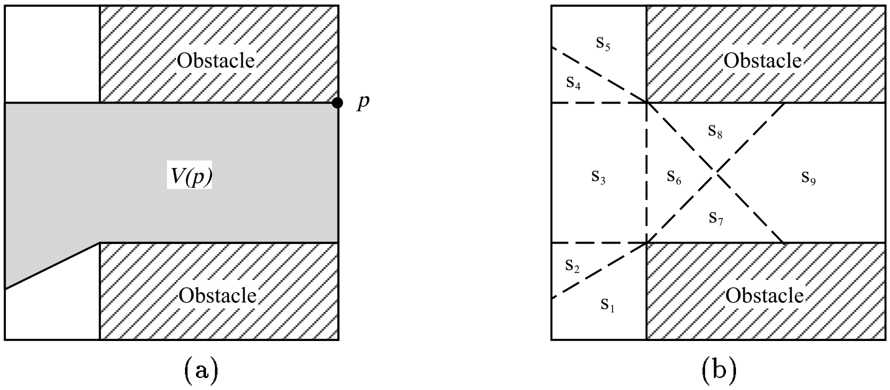


Figure 2: (a) A point visibility polygon, and (b) the visibility sectors.

A *visibility sector* $s_i \subseteq \mathcal{W}$ is a maximal region such that every point $p \in s_i$ is visible from the same set of environment vertices $V_{s_i} \subseteq V$. Thus, the visibility sectors of $\mathcal{W}$ are the faces in the planar subdivision that is obtained by overlaying the point visibility polygons for all $v \in V$ (see Figure 2(b)).

2.1 Computing Visibility Sectors

As mentioned above, the visibility sectors $S = \{s_1, s_2, \ldots, s_k\}$ are the faces in the planar subdivision which is obtained by overlaying the point visibility polygons for all environment vertices. Thus, one approach is to first compute all point visibility polygons, and then compute their common intersection.

All point visibility polygons $V(v)$, $v \in V$, can be computed in $O(n^2)$ time using Lee's [15] optimal $O(n)$ time algorithm for computing a single point visibility polygon. Let E_v denote the edges in v's visibility polygon, let $E' = \cup_{v \in V} E_v$, and let $n' = |E'|$. Then, the common intersection of the $V(v)$, for all $v \in V$, can be obtained in $O(n' \log n' + k')$ time, where k' is the number of intersection points, using one of a number of deterministic [2, 6] or randomized [8, 17] algorithms for line segment intersection. Thus, the total time required to determine the visibility sectors is $O(n^2 + n' \log n' + k')$. In many practical situations, $n' = O(n)$, and so the total construction time would actually be $O(n^2)$. We remark that a more efficient algorithm might be obtained by exploiting the fact that the visibility polygons are planar subdivisions. For example, the overlay of two planar subdivisions can be computed in $O(n + k)$ time [12] as compared to $O(n \log n + k)$ time in the general case.

However, since the environments we consider are quite small, and these computations are performed during preprocessing, it is sufficient for us to use a more naive, less efficient algorithm. In particular, we simply consider all pairs of vertices $v_i, v_j \in V$ and determine what contribution, if any, the line containing them makes to $V(v_i)$ or $V(v_j)$. After determining all such segments, we add them to our description of the environment $G = (V, E)$ to obtain a new graph $G' = (V, E')$ (intersection points are not introduced yet). Finally, we use the LEDA [16] routine to construct a planar map, whose faces correspond to our visibility sectors, from the graph G'.

2.2 Visibility Sector Labels

Our localization algorithm is a two step process that first localizes the robot to a particular visibility sector, and then within that sector. To aid this process, we compute, during preprocessing, a *label* for each sector that includes a component for each vertex and edge of the environment visible from that sector.

The component for each feature is classified as one of four types of *characteristic points*: a local maxima (`M`), a discontinuity (`D`), a local minima (`m`), or a connection (`c`) point. The *local maxima* represent the convex vertices of the environment visible from the sector, while the *discontinuity points* are the concave environment vertices that inhibit the view of some portion of the environment from the sector. The *Local minima* of a sector are the visible concave vertices that are not discontinuity points or are points on edges which are perpendicularly visible from a sector (i.e., if a perpendicular segment from the edge can reach the sector without intersecting any obstacle). Finally, each discontinuity point `D` has a corresponding *connection point* `c`, which represents the partially visible edge whose visibility is blocked by `D` (i.e., `c` represents the 'landing' point of a ray from the sector through `D`). Note that if a point could be both a discontinuity or a connection point and a local minimum or maximum, distinction as a discontinuity or connection point takes precedence.

Sec.	Label
1	mDcmMmMmMmM
2	mDcMmMmMmMmM
3	mMmMmMmMmMmM
4	mMmMmMmMmMcD
5	mMmMmMmMmcD
6	mMmDcMmMcDmM
7	mMmDcMmcDmM
8	mMmDcmMcDmM
9	mMmDcmcDmM

Table 1: Sector labels for the environment of Figure 2(b).

A sector's label is a string of mark characters (`M`, `m`, `D` and `c`), one for each characteristic point, which appear in the order the characteristic points are seen in a counter-clockwise scan with an initial orientation of 0 (i.e., eastern point first). For example, Table 1 shows the sector labels for the environment of Figure 2(b).

3 Localization

Our localization method is actually a two-step process. First the robot is localized to a particular visibility sector, and then its precise position and orientation in that sector are computed. Both these steps make use of the visibility sector information computed during preprocessing and are accomplished using only distance readings obtained by simple range sensors.

To simplify the exposition, in this section we make the following assumptions: (i) each visibility sector in the environment has a unique label, and (ii) the robot's range sensors provide at least three readings for each edge visible from the robot's current position. While the first assumption is in general not valid, the second may be practically true for laser scanning sensors in certain environments. These assumptions are removed in Section 4.

3.1 Range sensor data

The 'visibility' information required by our localization algorithm can be obtained using simple, inexpensive range sensors. Ultrasonic range sensors or laser scanning sensors are commonly used for mobile robots to measure the distance to nearby (visible) obstacles or to avoid collision. Laser scanning sensors provide distance measurements of 2π radians with a very high sampling rate. Some mobile robots are equipped with a rotating table so that the ultrasonic sensor mounted on top gives 2π radians sensor readings. Figure 3 shows an example sensor reading. The center is the origin of the sensor coordinates. If measured N times for 2π radians, the resolution is $\frac{2\pi}{N}$ radians and the $i-th$ sensor reading gives distance information (r_i) at its angle (θ_i) with respect to the sensor coordinates. We denote the set of N sensor readings by SD:

$$SD = \{ (r_i, \theta_i) | \, r_i \in R, \theta_i \in \theta, 0 \le i \le N-1 \} . \tag{1}$$

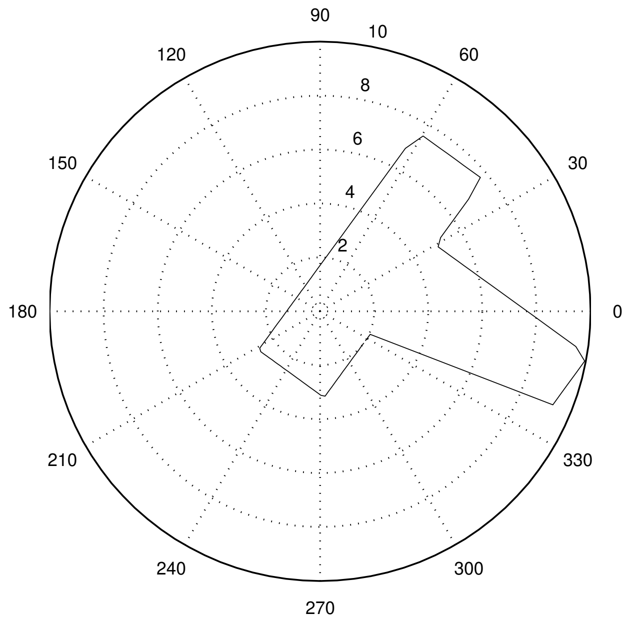


Figure 3: Range Sensor Reading

3.2 Localization to a visibility sector

To localize the robot to a particular visibility sector of the environment, we extract a label ℓ_{R} from the distance data R scanned by the robot and match this label with a label ℓ_{vs} for one of the environment's visibility sectors (recall in this section we assume each sector has a unique label). This is accomplished by extracting characteristic points from the measured distance data, and then composing them into a label.

Figure 4 shows the measured distance data scanned from 0 to 2π radians. If we scan these readings in sorted angular order, then the local minima and maxima readings directly correspond to local minima and maxima characteristic points `m` and `M`, respectively. The discontinuity and connection characteristic points `D` and `c`, respectively, can be found by a similar scan. In this case, if two consecutive distance readings differ by some predetermined threshold value ε_T, then the smaller reading corresponds to `D` and the larger to `c`. This process is outlined in the pseudo-code below, where $R = \{r_0, r_1, \ldots, r_{N-1}\}$ is the set of N distance readings sorted in angular order; all index arithmetic is done modulo N, and $\circ$ denotes string concatenation.

<u>CONSTRUCT SCAN LABEL</u>(R)

$\ell_{\mathrm{R}} := \emptyset$
for $i := 0$ to N
    if $(r_i - r_{i+1} > \varepsilon_T)$ **then** $\ell_{\mathrm{R}} := \ell_{\mathrm{R}} \circ$ `cD`
    elseif $(r_{i+1} - r_i > \varepsilon_T)$ **then** $\ell_{\mathrm{R}} := \ell_{\mathrm{R}} \circ$ `Dc`
    elseif $(r_{i-1} > r_i$ **and** $r_i < r_{i+1})$ **then** $\ell_{\mathrm{R}} := \ell_{\mathrm{R}} \circ$ `m`
    elseif $(r_{i-1} < r_i$ **and** $r_i > r_{i+1})$ **then** $\ell_{\mathrm{R}} := \ell_{\mathrm{R}} \circ$ `M`
    endif
endfor

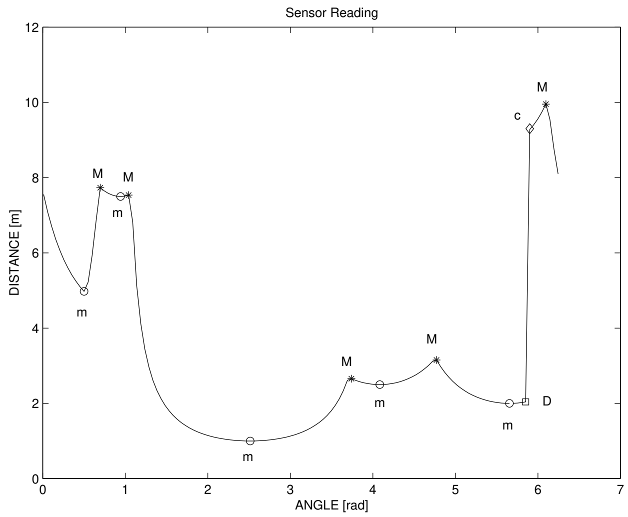


Figure 4: Sensor Data Analysis

The characteristic points identified by CONSTRUCT SCAN LABEL are marked in Figure 4 and Figure 5 shows the sensor reading mapped into the workspace. In this case, a total of 128 samples are measured. The label constructed for this sensor reading is

$$\ell_{\mathrm{R}} = \mathtt{mMmMmMmMmDCm}$$

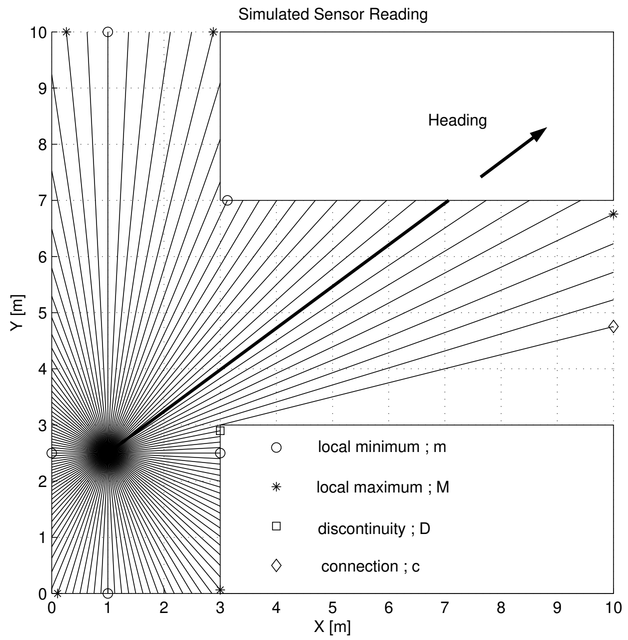


Figure 5: Characteristic Points

where the initial orientation $\theta = \frac{\pi}{5}$ (note the heading direction in Figure 5).

Once the label ℓ_{R} is composed from the scan, it is then matched with a label ℓ_{vs} for one of the environment's visibility sectors. Our assumption that the range sensors provide at least three distance readings for all visible edges of the surrounding environment implies that we will have a distance reading for each characteristic point visible from the robot. However, since the orientation of the robot is not known, our matching of ℓ_{R} with the environment's sectors labels must consider cyclic shifts of ℓ_{R} when performing the matching. For example, in the example considered above, the label ℓ_{R} matches the label for visibility sector s_2, but only after cyclically shifting ℓ_{R}.

3.3 Localization in a visibility sector

Once the visibility sector containing the robot is identified, we now must determine the position and orientation of the robot in that sector, i.e., we must determine its configuration (x_o, y_o, Θ_o) (see Figure 1). Note that we know the world coordinates of the vertices and edges visible in this sector. In addition, we have sensor readings for each visible edge, and moreover, after the label matching, we know the edge associated with each distance measurement.

If our sensor readings corresponding to the local maxima `M` characteristic points exactly coincided with the corresponding vertices of the environment, then localization would be trivial. For example, let (X_M, Y_M) and (x_M, y_M) denote the coordinates of the scanned local maximum point in the world and the robot's local coordinate systems (X, Y) and (X_r, Y_r), respectively. Then, the robot configuration (x_o, y_o, Θ_o) denotes the linear transformation from (X, Y) to (X_r, Y_r) with translation (x_o, y_o) and rotation (Θ_o), which can represented as shown in Equation 2:

$$\begin{bmatrix} -\cos\Theta_o & \sin\Theta_o & X_M \\ -\sin\Theta_o & -\cos\Theta_o & Y_M \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_M \\ y_M \\ 1 \end{bmatrix} = \begin{bmatrix} x_o \\ y_o \\ 1 \end{bmatrix} \tag{2}$$

In the above matrix representation, there are three unknown variables (x_o, y_o, and Θ_o), and two equations. Therefore, in order to solve for (x_o, y_o, Θ_o) we require the local coordinates of at least two environment vertices (`M` characteristic points). Without loss of generality, let M_1 and M_2 be two such points. Solving these four equations for the three unknowns, we obtain:

$$\begin{aligned} \theta &= \sin^{-1}\left(\frac{X_{M2} - X_{M1}}{\sqrt{(x_{M2} - x_{M1})^2 + (y_{M2} - y_{M1})^2}}\right) \\ &+ \tan^{-1}\left(\frac{x_{M2} - x_{M1}}{y_{M2} - y_{M1}}\right) \end{aligned} \tag{3}$$

$$x_o = -\cos\Theta x_{M1} + \sin\Theta y_{M1} + X_M \tag{4}$$

$$y_o = -\sin\Theta x_{M1} - \cos\Theta y_{M1} + X_M \tag{5}$$

Unfortunately, due to the finite number of scans, our readings do not exactly coincide with environment vertices. However, we can use the readings to compute the local coordinates of (two) environment vertices, and then use these to compute (x_o, y_o, Θ_o) as described above. Consider the situation shown in Figure 6, where we have scans on the edges adjacent to the local maximum vertex $M = (x_M, y_M)$. In the figure, (X_r, Y_r) is the robot's local coordinate system and (X, Y) is

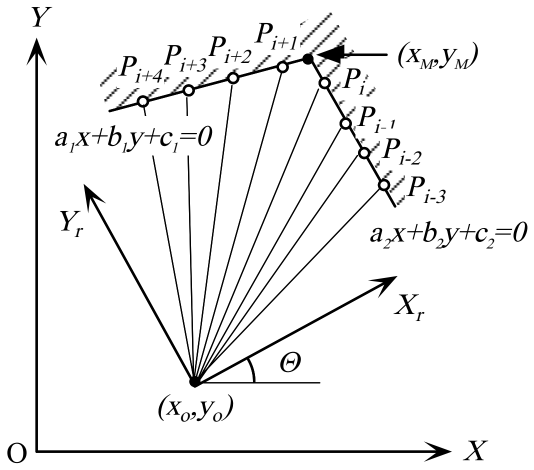


Figure 6: Computing the local coordinates of M from scans on its incident edges.

the world coordinate system. We can derive the equation (Eq. 6) for the the line containing the left edge using the series of sensed (local) points $P_{i+1}, P_{i+2}, P_{i+3}, P_{i+4}$. Only two points are required to get the equation of the line, but multiple points are used for a least square error method to reduce the error. Similarly, the equation for the line through the right edge (Eq. 7) can be derived from $P_i, P_{i-1}, P_{i-2}, P_{i-3}$.

$$a_1 x + b_1 y + c_1 = 0 \tag{6}$$

$$a_2 x + b_2 y + c_2 = 0 \tag{7}$$

The local coordinates of $M = (x_M, y_M)$ can then be calculated as:

$$x_M = \frac{-c_1 b_2 + c_2 b_1}{a_1 b_2 - a_2 b_1} \tag{8}$$

$$y_M = \frac{-a_1 c_2 + a_2 c_1}{a_2 b_1 - a_1 b_2} \tag{9}$$

Note that all the variables, $x_M, y_M, a_1, b_1, c_1, a_2, b_2, c_2$, are calculated from the sensed data.

In principle, one could use any two convex environment vertices (local maxima `M` points) for the above calculations. In practice, it is convenient to select to select adjacent vertices so that the equations for only 3 lines need to be computed, i.e., we use the edge between the vertices in both cases.

4 Global Navigation

The global navigator includes the *localizer* and the *path planner/updater* modules, which both use the environment's visibility sector information. With the given `start` and `goal` configurations, and the known environment information, the global navigator plans the path, and outputs a trajectory for the mobile robot. As the robot traverses the path, it (periodically) sends sensor readings to the localizer module. From the sensor data, the localizer estimates the current configuration, and sends it to the path updater, which replans the path, if necessary, and outputs a new trajectory for the mobile robot.

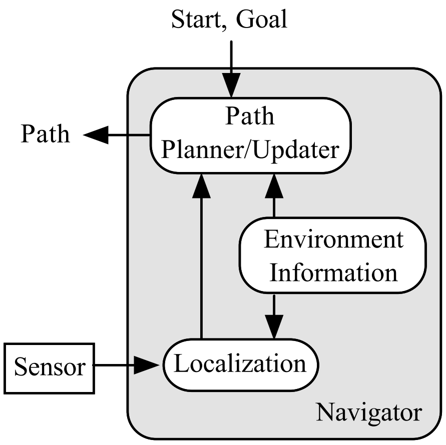


Figure 7: Global Navigator

4.1 Ambiguous Sector Labels and Error Estimation

If each sector in the environment has a unique label, then the localization procedure described in Section 3 is sufficient to localize the robot to a sector. We note that in many situations, the labels of the visibility sectors will in fact be unique. For example, in the more complex office environment shown in Figure 8, which contains 82 visibility sectors, each sector's label is unique.

However, there exist environments in which multiple sectors have the same label. To deal with this case safely, the global navigator should plan the path carefully so that no ambiguities will arise, or it should ask the robot to localize sufficiently often so that accurate localization is always possible. This section describes how the global navigator deals with these issues.

Many researchers have developed algorithms that estimate the position uncertainty of a dead-reckoning robot [13, 22]. With this approach, each computed robot position is surrounded by a characteristic *error ellipse* which indicates a region of uncertainty for the robot's actual position [1, 22]. Typically, these ellipses grow with travel distance, until an absolute position measurement (localization) reduces the growing uncertainty and thereby *resets* the size of the error ellipse.

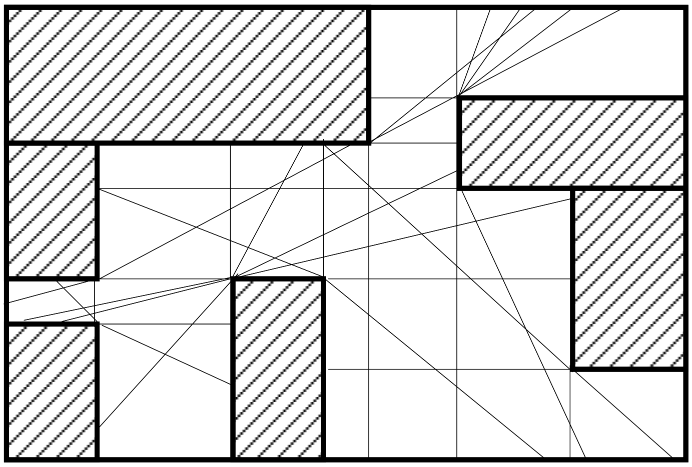


Figure 8: Sectors for an Office Environment

In Figure 9, the mobile robot started or localized itself at time t_1, therefore, there exists no position uncertainty. Sector i and sector j have the same label. If the robot is in either of these

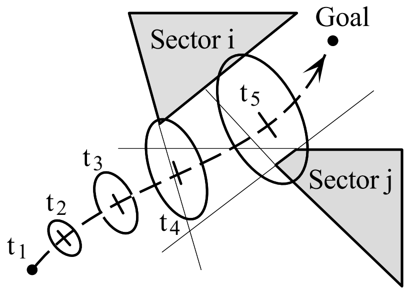


Figure 9: Uncertainty Ellipses and Ambiguous Sectors

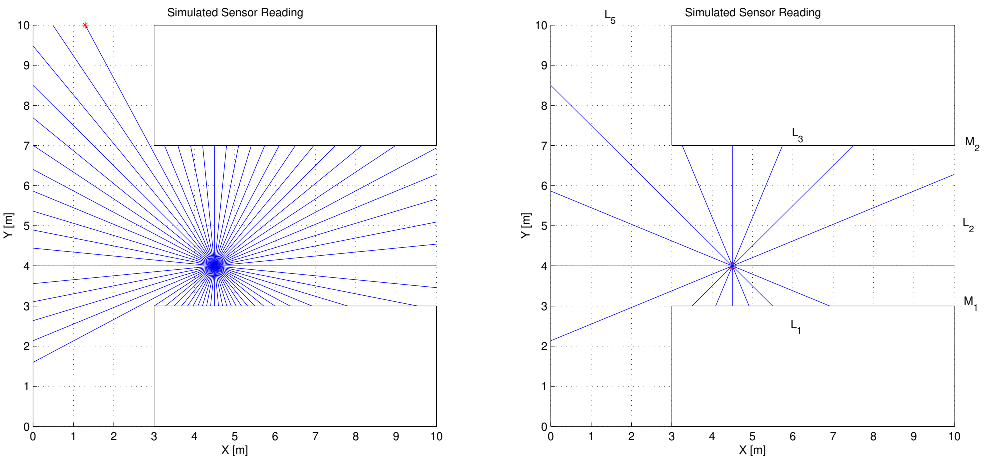
sectors, which cannot be identified simply from the label information. As the robot moves, the position uncertainty ellipse grows and it touches sector i at t_4. At t_5, the ellipse covers both sector i and j.

In order to avoid this situation, the global navigator should attempt to plan the path so that the position uncertainty ellipse will never intersect two ambiguous sectors simultaneously. If that is not possible, then the navigator must determine when the robot should be localized to prevent such a situation from arising. Finally, even if the pre-planned path avoids all ambiguous situations, the robot must still be localized and the trajectory updated to correct for the accumulated positioning errors.

4.2 Limited number of scans

In this section, we consider the issue of determining the minimum number of scans necessary for accurate localization. This is an important question for two reasons. First, minimizing the sensing time will minimize the localization time as well. Second, in practice, only a limited number of scans can performed by sonar range sensors due to limits on the sensor bandwidth and on the resolution of the robot's rotating table.

The scan label constructed from sensor data will not be correct if every edge visible to the robot is not scanned in such way that its corresponding characteristic points can be identified. In particular, the label constructed by the scan will be correct if every edge containing a connection point `c` is scanned at least once, every edge which is perpendicularly visible is scanned at least three times (to get both `M`'s and the intervening `m`), and every other edge is scanned at least twice. For example, Figure 10 shows two scans, with 64 and 16 readings, respectively, taken from the same position in sector s_7. The scan with 64 readings satisfies the requirements given above, while the one with 16 readings does not. The labels constructed from the scans are $\ell_{64} =$ `mMmDcMmcDmM` and



(a) (b)

Figure 10: Sensor scan of sector s_7 with (a) 64 scans, and (b) 16 scans.

$\ell_{16} =$ `MmDcmcDm`, respectively. While the label ℓ_{64} correctly matches the pre-computed label for sector s_7, the label ℓ_{16} does not, and in fact matches the label for sector s_9 instead.

Based on the above observations, a conservative requirement to ensure the correct label is constructed is to ensure that each edge is scanned at least three times. For example, consider the situation shown in Figure 11(a). Given the position of the robot, we can determine the *scan interval* for each edge (i.e., the visible angle of that edge from the given robot position). Letting α denote the minimum such angle, the minimum number of scans is

$$N_{min} = \frac{6\pi}{\alpha} \tag{10}$$

However, the angle, α varies depending on the location in sector s_7. As the robot moves towards the edge $\overline{p_1 p_3}$, α goes to zero and N_{min} becomes ∞, which is not possible.

Recall, however, that our goal is correct localization, not correct label construction. For localization, as outlined in Section 3.3, we need to compute the local coordinates of two environment vertices. This requires that we scan at least two points (not three) on the edges adjacent to those vertices (used to get the equations of the lines whose intersection determines the local coordinates of the vertex). For example, the local coordinates of the local maximum points M_1 and M_2 can be obtained from the 16 reading scan (see Figure 10(b)). In this case, although the label constructed for the 16 reading scan taken in sector s_7 incorrectly matches sector s_9 (since the edge L_5 was not scanned), it still correctly identifies the environment vertices corresponding to M_1 and M_2, and this is all that is needed if the localization is performed using M_1 and M_2. In fact, correct localization will result if the computed label matches the label from a different sector which uses the *same* local

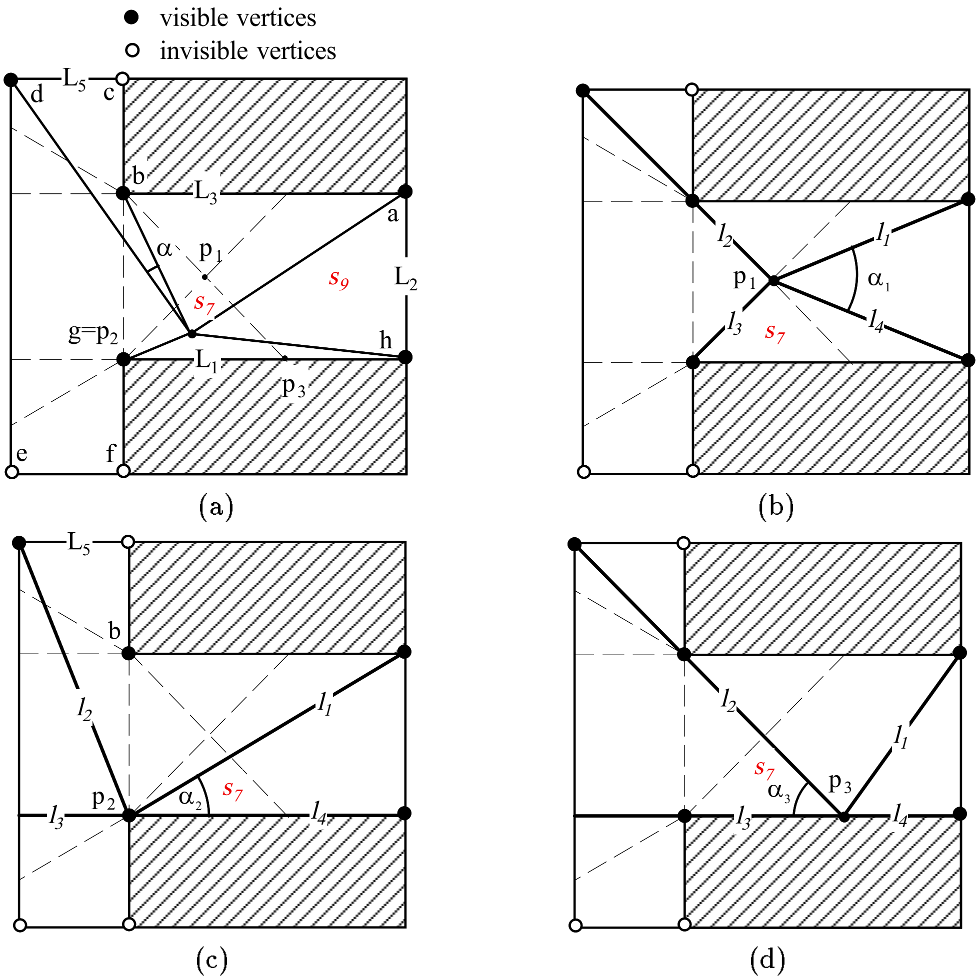


Figure 11: For sector s_7, (a) visible/invisible vertices, and minimum visible angle calculations for s_7 vertices (b) p_1, (c) p_2, and (d) p_3.

maximum points as the actual sector for localization. One situation in which this occurs is if the scan only misses edges corresponding to connection points. In these cases, our requirement now becomes that each edge, except those identified with connection points, be scanned twice. That is,

$$N'_{min} = \frac{4\pi}{\alpha} \tag{11}$$

The relaxation of the need to scan connection point edges allows us to obtain a finite requirement on the minimum number of scans for a particular sector as follows. We first note that one of the extreme points of the sector (the vertices) will achieve the minimum visible angle for a given environment edge among all points in that sector. Thus, we compute a minimum angle for each sector vertex (see Figure 11), take the minimum of these minima, and then apply Equation 11. The minimum angle of a sector vertex is the minimum angle between the rays connecting the vertex to all visible vertices from that sector except discontinuity vertices.

The procedure outlined above computes the minimum number of scans necessary for a given sector. If desired, one can determine a scan resolution valid for the entire environment as follows: compute the minimum visible angle for each sector, take the minimum of these minima, and then use this in Equation 11 to determine necessary scan resolution.

5 Preliminary Experimental Results

We are currently implementing our localization method with the mobile robot shown in Figure 12. The robot has dual differential drive with DC gear motors and encoders. Due to encoder errors caused by slip between the wheel and the surface, periodic localization is necessary. Three ultrasonic range sensors are mounted on the pan/tilt head so that each sensor gives readings of $\frac{2\pi}{3} rad$. The sensors have a minimum and maximum range of 20cm and 10.5m, respectively, with a resolution of 1%.

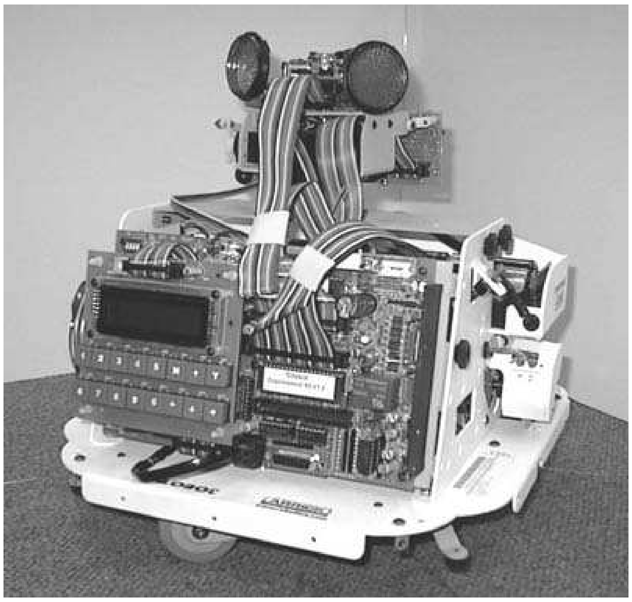


Figure 12: Mobile robot

For this initial test, we used environment identical to the running example used in the paper (e.g., Figure 11), but in reduced scale ($2.54m \times 2.54m$ as opposed to $10m \times 10m$). Figure 13 shows the robot sensing the environment from sector s_2; $N = 60$ sensor scans were obtained (20 scans by each sensor).

Figures 14 and 15 show preliminary experiment results. Note that the experimental data is very close to the simulated sensor readings shown in Figures 4 and 5. Due to crosstalk of signals, the measurement shows several incorrect readings, and postprocessing of the sensed information is currently being investigated.

6 Conclusion

In this paper, we present a new localization method for mobile robots. This method is based on range sensor data and some simple preprocessing of the environment to partition it into visibility sectors. A strength of this localization method is that it can be applied from any configuration in the environment (it does not need landmarks), and it is very simple, requiring only a comparison of some features easily extracted from the sensor reading with similar information computed for each sector in the environment. This localization method also provides opportunities for the global navigation procedure to analyze and select trajectories in terms of their tolerance to localization errors.

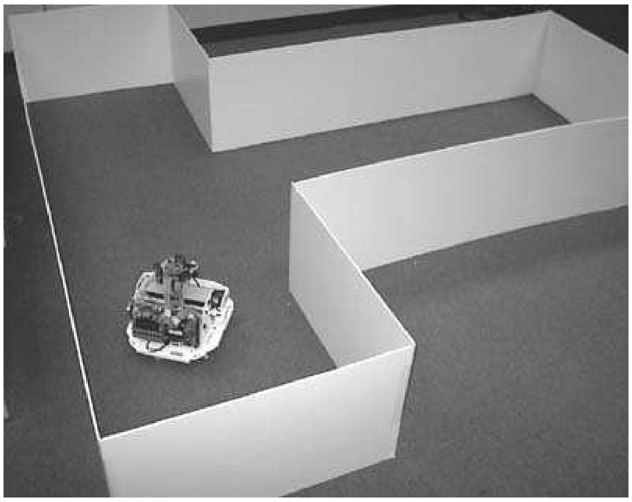


Figure 13: Experiment

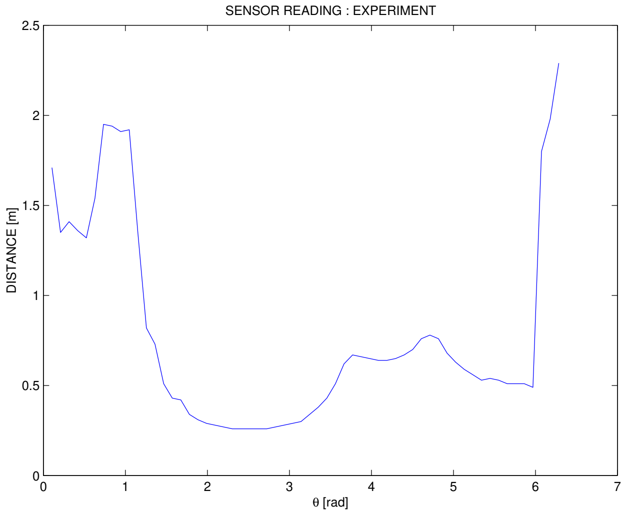


Figure 14: Sensor Data from Experiment Analysis

We believe this method is very promising, particularly in indoor environments for which good models are usually readily available. We are currently implementing this method on a mobile robot in our lab, and will report more complete experimental results in the final version of the paper. We are also working on extensions of the method so that it can be used in the presence of unknown (e.g., furniture) or moving (e.g., people) obstacles.

References

[1] M. D. Adams. Control and localization of a post distributing mobile robot. In *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, pages 150–156, 1994.

[2] N. M. Amato, M. T. Goodrich, and E. A. Ramos. Computing faces in segment and simplex arrangements. In *Proc. 27th Annu. ACM Sympos. Theory Comput.*, pages 672–682, 1995.

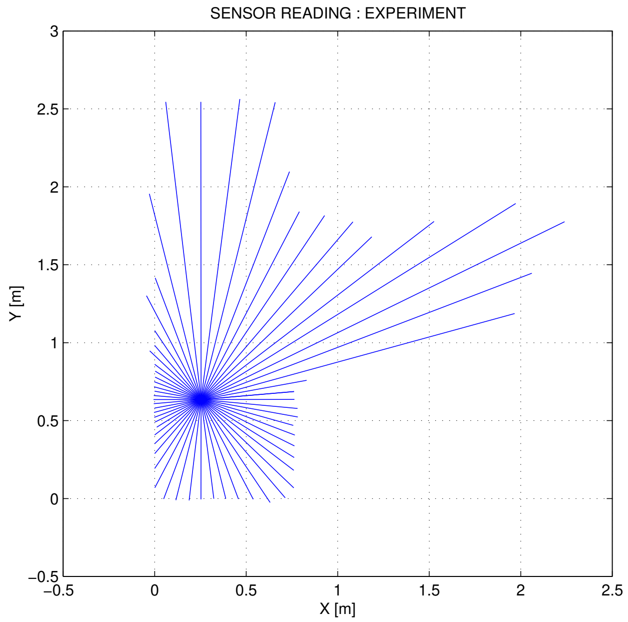


Figure 15: Sensor Reading from Experiment

[3] S. Atiya and G. Hager. Real-time vision-based robot localization. *IEEE Trans. Robot. Automat.*, 9(6):785–800, 1991.

[4] N. Ayache and O. D. Faugera. Building a consistent 3-d representation of a mobile robot environment by combining multiple stereo view. In *Proc. of Int. Joint Conf. on Aritificial Intelligence*, pages 808–810, 1987.

[5] M. Betke and L. Gurvits. Mobile robot localization using landmarks. In *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, pages 135–142, 1994.

[6] Bernard Chazelle and H. Edelsbrunner. An optimal algorithm for intersecting line segments in the plane. *J. ACM*, 39(1):1–54, 1992.

[7] F. Chenavier and J. Crowley. Position estimation for a mobile robot using vision and odometry. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 2588–2593, 1992.

[8] K. L. Clarkson and P. W. Shor. Applications of random sampling in computational geometry, II. *Discrete Comput. Geom.*, 4:387–421, 1989.

[9] J. L. Crowley, F. Wallner, and B. Schiele. Position estimation using principal components of range data. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 3121–3128, 1998.

[10] A. Elfes. Sonar-based real-world mapping and navigation. *IEEE Journal of Robot. and Automat.*, RA-3(3):249–265, 1987.

[11] C. Fennema, A. Hanson, E. Riseman, Beveridge, and R. Kuman. Model-directed mobile robot navigation. *IEEE Trans. Sys., Man, Cybern.*, 20(6):1352–1369, 1990.

[12] Ulrich Finke and Klaus Hinrichs. Overlaying simply connected planar subdivisions in linear time. In *Proc. 11th Annu. ACM Sympos. Comput. Geom.*, pages 119–126, 1995.

[13] K. Komoriya and E. Oyama. Position estimation of a mobile robot using optical fiber gyroscope. In *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, pages 143–149, 1994.

[14] E. Krotkov. Mobile robot localization using a single image. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 978–983, 1991.

[15] D. T. Lee. Visibility of a simple polygon. *Computer Vision, Graphics, and Image Processing*, 22:207–221, 1983.

[16] K. Mehlhorn and S. Näher. *LEDA: A Platform for Combinatorial and Geometric Computing.* Cambridge University Press, New York, 1998.

[17] K. Mulmuley. A fast planar partition algorithm, I. *J. Symbolic Comput.*, 10(3-4):253–280, 1990.

[18] K. Nagatani, H. Choset, and S. Thrun. Towards exact localization without explicit localization with the generalized voronoi graph. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 342–348, 1998.

[19] J. O'Rourke. *Art Gallery Theorems and Algorithms.* Oxford University Press, New York, 1987.

[20] A. C. Schultz and W. Adams. Continuous localization using evidence grids. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 2833–2839, 1998.

[21] K. Sugihara. Some location problems for robot navigation using a single camera. *Computer Vision, Graphics and Image Processing*, 42(1):112–129, 1988.

[22] Y. Tonouchi, T. Tsubouchi, and S. Arimoto. Fusion of dead-reckoning positions with a workspace model for a mobile robot by bayesian inference. In *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, pages 1347–1354, 1994.

[23] C. Urdiales, A. Bandera, R. Ron, and F. Sandoval. Real time position estimation for mobile robots by means of sonar sensor. In *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, pages 1650–1655, 1999.