1 2

Precise Positioning Using Model-Based Maps

Paul MacKenzie and **Gregory Dudek**
Centre for Intelligent Machines
McGill University, 3480 University St.,
Montréal, Québec, Canada H3A 2A7
email: {dudek,mackenzi}@cim.mcgill.edu

Abstract

This paper addresses the coupled tasks of constructing a spatial representation of the environment with a mobile robot using noisy sensors (sonar) and using such a map to determine the robot's position. The map is not meant to represent the actual spatial structure of the environment so much as it is meant to represent the major structural components of what the robot "sees". This can, in turn, be used to construct a model of the physical objects in the environment. One problem with such an approach is that maintaining an absolute coordinate system for the map is difficult without periodically calibrating the robot's position.

We demonstrate that in a suitable environment it is possible to use sonar data to correct position and orientation estimates on an ongoing basis. This is accomplished by incrementally constructing and updating a model-based description of the acquired data. Given coarse position estimates of the robot's location and orientation, these can be refined to high accuracy using the stored map and a set of sonar readings from a single position. This approach is then generalized to allow global position estimation, where position and orientation estimates may not be available.

We consider the accuracy of the method based on a single sonar reading and illustrate its region of convergence using empirical data.

1 Introduction

Despite their potential utility, complete *a priori* maps of a mobile robot's environment are rarely available. Even when maps (such as architect's floor plans) are available in a form that can be used, they often fail to accurately portray the environment in a manner consistent with typical robotic sensing devices. For example, commonly-used sonar devices fail to detect many fixtures and may "detect" many structures that are not physically present (such as illusory walls in corners). For these reasons, it is important for an autonomous (mobile) robot to construct and maintain a map of its novel environment in terms of its own perceptual mechanisms.

Most simple devices for measuring position and distance are relative measurement tools (e.g. odometers). Imperfect estimates of orientation, distance and velocity must be integrated over time and hence errors in absolute pose (position and orientation [5]) accumulate disastrously with successive motions and make the general problem of maintaining an accurate absolute coordinate system very difficult. For these reasons, map construction and long-term localization are dependent on the use of sensory data for recalibrating a robot's sense of its own location within the environment.

Given approximate estimates of a robot's position based on odometry and dead-reckoning, we show in this paper how a geometric map can be constructed and used for ongoing re-calibration. This approach is then extended to allow global localization, where the robot is not provided with any initial pose estimate. Our construction is based on the use of stable detectable structures of sizable spatial extent in the environment and avoids intervention such as the placement of beacons. Although the method is described using sonar sensors, its applicability is not restricted to this sensing modality. The fundamental issue is that if the robot is able to sense its location accurately within a familiar geometrically modeled region, it can compensate for the cumulative errors that result from uncertainties in its movements as it travels within and between regions.

Some other work in this area has involved the use of Kalman filters to track environmental features [7, 12]. Other approaches have relied on the calculation of a transformation matrix to match observed signals and to localize a robot [2, 6]. Leonard, Durrant-Whyte and Cox incorporate a sonar model into their system to anticipate sensor readings [7]. While this is not necessary to perform the localization in this paper, it does allow for dynamic map keeping by keeping track of particular targets that may disappear or new ones that may appear [8]. This is useful for environments that change periodically, and could easily be adapted into the system described in this paper (since it would be independent of localization). The work described here differs from previous work not only in the technical details of the algorithm, but also in its combined ability to combine data from multiple readings in a single computation, to exploit measurements that may not arise only from simple reflections, to dynamically construct a map (shared by some existing approaches), its strong convergence properties, the ability to perform global localization and in its investigation of the

*This paper appears in the Proceeding of the IEEE International Conference on Robotics and Automation, San Diego, CA, May 1994, pp. 1615-1621.

†The authors gratefully acknowledge the financial support of the Natural Sciences and Engineering Research Council.

region of convergence of the algorithm.

In this paper we begin by outlining the manner in which a map can be constructed using dynamically acquired sonar range data. We go on to show how the re-calibration of pose estimates can be achieved based on a single set of measurements. Finally, we extend this to global localization and examine some properties of the algorithm.

2 Environmental Models

We will consider here the case of two-dimensional environmental modeling only. Line segments are used to model collections of observations of the environment. Each segment can be thought of as representing a section of a wall or other obstacle although, in fact, some linear collections of observations may not correspond directly to existing structures. Line segment models for sonar use are appropriate given the characteristics of simple threshold-based sonar sensing [1, 3, 4], where even a small object will produce a collection of measurements at similar distances that are nearly linear in structure. (In fact, the measurements from a single position are often arranged in the form of circular arcs of low curvature [7]. For noisy data these can be well approximated by line segments with much less computational overhead than circular models. Furthermore, when data from multiple positions is integrated the linear nature of indoor structures (walls) tends to dominate.) Raw sonar data obtained from a robot with a rotatable ring of 12 Polaroid sonar transducers is used in the experiments described below.

The construction of an environmental map entails the following main steps:

1. Conversion of sonar time-of-flight readings into distance measurements using simple thresholding.

2. Spatially clustering of groups of neighbouring unexplained measurements. This serves to associate measurements that may arise from the same object (or interaction); disparate measurements from the same object may be grouped subsequently.

3. Generation of line segments to represent data using a fitting procedure for each cluster combined with a split-and-merge segmentation process.

4. Combination of new line segment models with existing models.

5. Detection of higher-level map features (for example corners).

2.1 Clustering Algorithm

The first stage in describing a collection of data points is to perform clustering using an adaptation of the sphere-of-influence graph [11, 4]. This technique divides measurements into unconnected subsets which are considered for independent modeling. This technique avoids *a priori* thresholds and has a complexity of $\mathcal{O}(n \log n)$, when n is the number of data points.

2.2 Fitting Line Segment Models to Data

Assigning line segment models to the data clusters is done with a *fit-split-merge* strategy. It functions by fitting lines to entire clusters of measurements (produced by the prior computation) and then recursively subdividing the clusters to maximize a goodness-of-fit measure (this is interleaved with a merging phase, below). An *a priori* stopping criterion is needed to prevent splitting of the clusters into excessively small groups (in the worst case, two points each). After all line segments have been fit, any that are close together and co-linear are merged into a larger single line.

The coordinate-system independent line fitting algorithm is based on a least mean squares fit using the singular values (eigenvectors) of the covariance matrix of the data [10]. This is analogous to fitting an *ellipse* to the data and finding the major and minor axes which are, respectively, the directions of maximum and minimum variance. The best fit line is described by the principal eigenvector (major axis), while splitting is performed by subdivision (if necessary) and is performed in the orthogonal direction. The end points of the line segment are assigned so that the smallest line segment is generated such that all the data points for that line may perpendicularly project onto it (a more comprehensive model includes an estimate of the uncertainty in the sonar measurements defining the end points of the line but is beyond the scope of this paper).

When a cluster cannot support a single line segment of high enough line quality, it is split into two parts and a line segment fit is attempted on each half. There are two criteria for this decision. First, if the fit ellipse is very elongated (where elongation is simply the ratio of eigenvalues of the covariance matrix of the data [10]), then the cluster is line-shaped and we do not wish to split it. Secondly, an extremely long line segment may still have a significant orthogonal variance. Relative to the line it may be small, but in absolute terms it may be comparable in size to the robot itself (i.e. more than just a few centimetres). Therefore, long lines warrant splitting just in case this detail exists. If it does not and the data is indeed shaped like a long line segment, then the merging of line segments after the fitting process will give this result (it also allows old models and models of new data to be combined when appropriate). Taking both of these considerations into account allows a decision to be made regarding the splitting of clusters [4]. This process of selecting the minimum number of sufficiently good lines is related to minimal length encoding.

Figure 1 illustrates the results of the modeling process in a simple environment. The area was scanned by moving the robot around a square obstacle in the centre of a partially enclosed area (roughly 4 square meters) along the indicated trajectory. The linear models corresponding to major structures in the environment (such as walls) are evident. In addition, line segments can be observed modeling non-physical collections of data points that correspond to artifacts of the sonar measurement process. Not only is it is unnecessary to discard these apparently spurious data, but they prove to be valuable in modeling (this illustrates the importance of deriving a map from the actual sensors to be used rather than from *a priori* information).

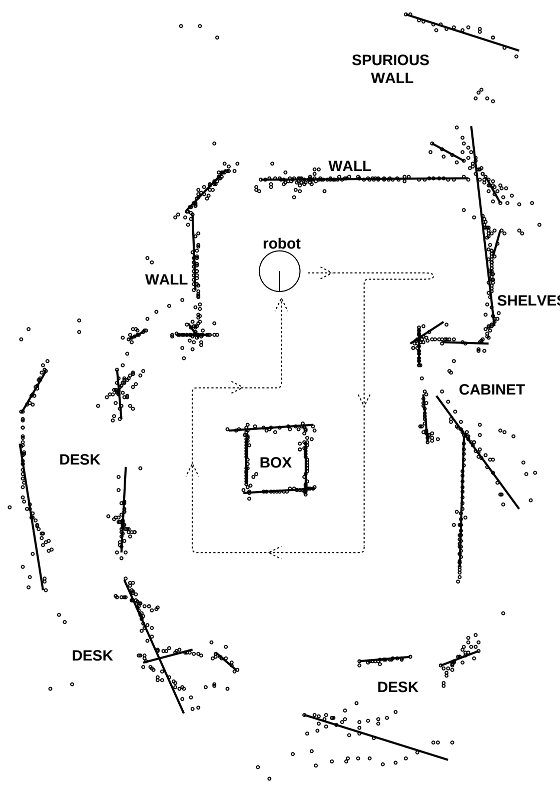


Figure 1: Modeling of Range Data with Line Segments. Dots are sonar measurements, thick lines are inferred models, and the dotted line shows the path of the robot when scanning the environment. The map was constructed incrementally from various individual scans rather than formed from this data point set as a whole.

3 Localization

Precise localization, the process whereby a robot can determine pose with respect to a mapped environment, is performed in several stages. We present an approach to position refinement (or localization) based on an a priori coarse position estimate, and go on the develop a global method from this. All stages assume the existence of a map constructed by the method described in section 2. The first stage, position correction, assumes an estimate of the robot's position is available and that there is no error in the robot's orientation. The pose estimation problem is formulated as an optimization in terms of the extent to which the map explains the observed measurements. Position and orientation corrected simultaneously, given an initial coarse estimate, is referred to as *local pose correction*. Using the local pose corrector and an associated measure of quality-of-estimate (consistency between map and measurements) allows global localization (where the robot has no initial estimate of its pose) to be formulated as a secondary optimization problem.

3.1 Position Correction

As noted above, position correction assumes a coarse initial position estimate is available, and estimates the correct position assuming the orientation is known. There are two phases involved: 1) Classification of Data Points and 2) Weighted Voting of Correction Vectors. The classification stage entails examining sensed data points and *classifying* each to a *target* line segment (i.e. the line segment most likely representing the same object that the sensed data represents). The assumption is made that the closest line segment to a given sensed data point is the most likely target for that point [2]. The voting stage consists of computing a vector difference between each point and its target model and deriving a weighted sum of these *correction vectors* to give an estimate of the robot's true position. This process is then repeated until the position can no longer be further refined.

3.1.1 Classification of Data Points

The purpose of classification is to match range data points with models representing the most likely object in the environment to which the sensor responded. The target model of each point is that model to which the point is closest (close in a Euclidean distance sense). Assuming a small positional error, the data points will not usually be too far away from the objects they represent. This is somewhat analogous to the linearization of a system of non-linear equations about an operating point. In this case, the operating point is the roughly accurate position estimate.

From each data point and target line segment, we obtain a *correction vector*: the vector difference between the data point and its perpendicular projection onto the infinite line passing through the line segment. This vector represents the offset required to exactly match the point to the line through the model, i.e. correct the error in that point. A combination of the correction vectors from all the measurements used in the position estimate provides the estimate of the refined position.

The one-dimensional position constraint (and hence one-dimensional uncertainty) provided by each measurement derives from the geometric constraint (or lack of constraint) that derives from matching to a one dimensional (line) model (if fact, there is a weak constraint in the orthogonal direction but we will omit it here in the interest of clarity). This problem which manifests itself in this context as the *long hallway effect*, is analogous to the *aperture problem* in motion estimation [9]. Observation of position (or motion) of a section of a straight line provides information only in the direction of the normal to the line (akin to normal flow in motion estimation). In practice, a robot in the middle of a long hallway (such that it cannot see the ends) can only correct its position in the direction perpendicular to the main axis of the hallway. Movement *along* the axis of the hallway gives no displacement information since all parts of the walls look identical, and therefore cannot be distinguished in order to calculate a displacement. In practice, Kalman filtering can be used in such situations to combine dead-reckoning information with sensory input [7].

3.1.2 Weighted Voting of Correction Vectors

The individual correction vectors cannot be simply summed together to form an overall error vector – some kind of weighting is required. If we consider each correction vector as $(\Delta x_i, \Delta y_i)$, then the overall error vector $(\Delta X, \Delta Y)$ can be calculated as follows:

$$\Delta X = \frac{\sum_{i=1}^{n} \omega(d_i)\Delta x_i}{\sum_{i=1}^{n} \omega(d_i)} \quad (1)$$

$$\Delta Y = \frac{\sum_{i=1}^{n} \omega(d_i)\Delta y_i}{\sum_{i=1}^{n} \omega(d_i)} \quad (2)$$

where

$$\omega(d) = 1 - \frac{d^m}{d^m + c^m}, \text{ a sigmoid function} \quad (3)$$

d_i is the distance between the i^{th} range data point and its target line segment (not necessarily the distance to the infinite line through the target). A sigmoid function has values close to unity for short vectors, and approaching zero for long vectors, with a smooth transition in between governed by the value of c in equation 3, with the effect of weighting short vectors higher than long vectors. This "soft-nonlinearity" serves to reject outliers and ensures that points close to their target line segments have a greater voting strength in the overall error vector.

3.2 Convergence of the Estimate

In general, the classification and weighted summation operations must be performed iteratively, incrementally correcting the robot's position estimate. This is due to the interdependence of the solutions to the classification and estimation procedures: accurate estimation depends on accurately associating measurements and map information. As the position estimate for the robot changes, the point-target correspondences can vary substantially. Not all points may be properly classified initially, but only a few are needed to start moving the position estimate in the right direction; incorrect correspondences tend to be randomly distributed and hence are readily outweighed by correct ones. To aid in the accuracy and convergence of process, the value of c in equation 3 is decreased as the iterations proceed. This allows many measurements to participate in making a initial pose estimate while final position refinement is dependent only of measurements that are almost certain to be correctly attributed to known models. This coarse-to-fine strategy provides progressive shift in emphasis from coarse detection of an attractor to accurate estimation.

4 Local Correction Results

In order to illustrate the region of convergence from incorrect position estimates to an accurate estimate, a representative experiment using initial position estimates whose error ranged up to 300 cm (10 feet) in both the x and y directions from the robot's actual position is described using position correction.

Actual positions were measured manually to within one-half centimeter using a tiled grid on the floor of the test area. Figure 2 shows the convergence of the position estimates as a function of initial position. Each line connects an initial estimate (small circles covering the map) to a final estimate. The true robot position is in the center of the map (where many solutions converge). Figure 3 shows more clearly the region around the true location for which the estimated robot position converges correctly. For example, any initial estimate in the upper-left or within one meter of the true position converges to the correct solution The other initial estimates do not converge correctly due to incorrectly classified measurements. "Correct convergence" for this figure is defined as a final position error of less than 3 cm. Position estimation

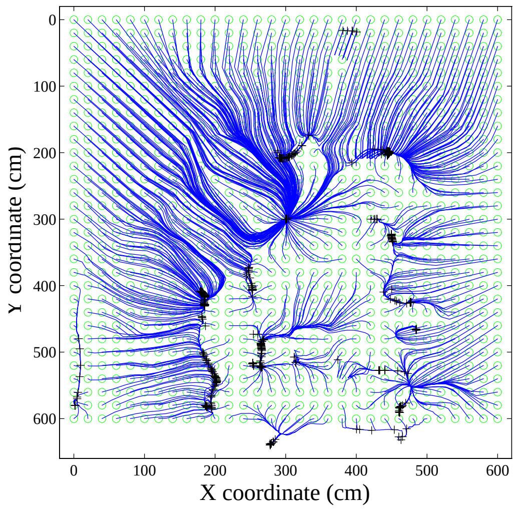


Figure 2: Paths of convergence, shown by lines originating initial position estimates (circles) leading to final position estimates. The true robot position is in the center of the rectangle.

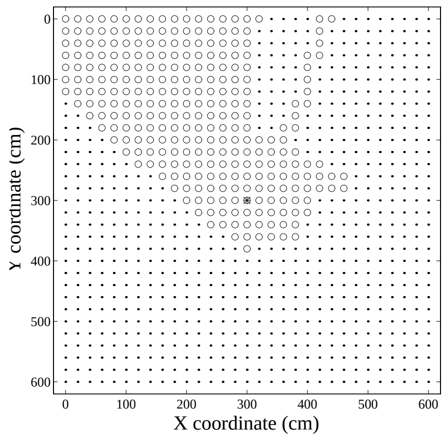


Figure 3: Initial Position Estimates that Converge to the Correct Position of the Robot (circles indicate successful convergence within 3cm of true position). The filled dot in the centre is the correct position.

errors after convergence generally tend to be on the order of 5 cm or less in moderately well structured environments such as the office space depicted earlier.

The most common source of error aside from poorly fit models (due to sparse and/or noisy data) is the "hallway effect" previously mentioned, where there may be insufficient structure to correctly estimate the position along some orientation. (As noted above, integrating odometry with sensor-based estimation can often deal with this in practice.)

5 Quality Measures

As shown, it is possible that the outcome of position correction is not the correct position of the robot. To provide for this, we need a function that will indicate if the final calculated pose is probable; in short, to estimate the extent to which correspondence between the measurements and the map exceeds chance. We would also like to be able to compare multiple solutions in terms of explanatory power.

There are three basic estimators used here: the *mean-squared error* measure, the *classification factor*, and the *comparative quality measure* (which is a combination of the other two).

The mean-squared error measure is straightforward:

$$E_{mse} = \frac{1}{n}\sum_{i=1}^{n}(\mathrm{dist}(p_i, \ell_i))^2 \tag{4}$$

where p_i is i^{th} of n range data points, ℓ_i is p_i's target line segment model, and $\mathrm{dist}(p, \ell)$ is the distance from a point p to the closest point on the line segment ℓ.

This function is locally suitable since its global minimum indeed occurs at the true pose of the robot (this is akin to not providing false negatives). At this true pose, it is assumed that all or at least the vast majority of range data points are very close to their target models, thus yielding a low value for a correct solution.

There is a difficulty with this function used in isolation: it is susceptible to outliers, and these will certainly effect the results even if the pose estimate is very accurate. Since it is not known how many outliers are in the sonar data set, the possible range of values is very broad, and this makes deciding whether a single pose is valid is very difficult based on this function alone. However, it is useful when used to compare two alternative solutions.

The Classification Factor is a quantity based on the fraction of all data points that are *well explained*. This is obtained by computing the fraction of all data points that are within some fixed distance threshold x of their associated model. Under the assumption that models occupy only a small fraction of the environment, close associations between measurements and models will occur only rarely by chance. This measure can this indicate how good our pose estimate is. The only way to obtain a value approaching unity is when the pose estimate is very close to the actual one, or to be in part of the environment that is *very* similar to the one in which the robot is located. Ignoring the latter, this measure should then give a value close to unity when the position estimate is very close to the true position, and near zero when the error of the estimate is large.

Using an abrupt, step-like neighborhood threshold has several shortcomings, in particular unstable behavior as errors exceed the threshold value. To address this, a "soft" sigmoid non-linearity is used for the threshold function assuring graceful degradation of the solution as a function of the input error. The *classification factor* is thus defined as (see figure 4):

$$E_{cf} = \frac{1}{n}\sum_{i=1}^{n}(1 - \frac{d^m}{d^m + c^m}) \tag{5}$$

where:			
	d	$=$	$\mathrm{dist}(p_i, \ell_i)$
	c	$=$	neighborhood size
	m	$=$	sigmoid steepness

The neighborhood should not be too small to accomodate innacuracies in the models and should depend on sensor error (in practice, we use a value of about 5 cm.). Figure 4 illustrates E_{cf} in the same environment as the previous figures.

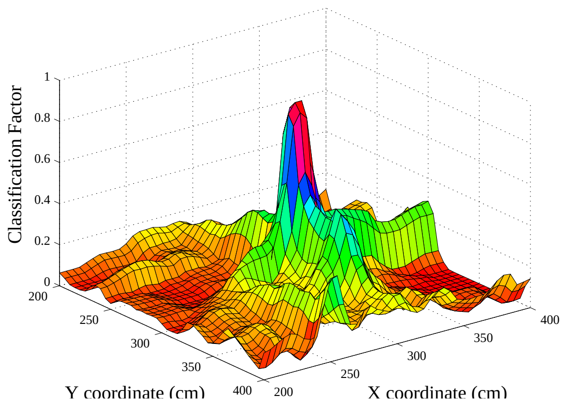


Figure 4: The Classification Factor E_{cf}

At the actual robot position (again, (x,y,θ)=(300,300,0)), we see the global maximum that approaches very closely to unity, and is much less everywhere else. The very useful property of E_{cf} is that the range is limited to a value between 0 and 1, and this allows for a simple test to be made for the likelihood of a convergence being good or bad, such as a threshold. Through experimentation, a good working threshold was found to be about 0.6; so we expect at least 60% of all points should be within some distance of their targets (for us, about 10cm).

One problem with E_{cf} is that it is not as useful as E_{mse} when dealing with very fine differences in robot position. Since it in essence just counts the number of points within a neighborhood, it cannot give precise detail within that neighborhood. For this reason, E_{cf} alone is not used as a comparative measure. E_{mse} does not suffer from this, as it deals with actual distances.

While it quite possible to use E_{cf} for pass/fail decisions, and E_{mse} for comparative quality, using a combination of the two can make the indication of the true pose more pronounced. The *comparative quality measure (E_{cqm})* is such a combination, defined as follows:

$$E_{cqm} = \frac{(E_{cf})^a}{(E_{mse})^b} \qquad (6)$$

where a and b are factors which weight E_{cf} and E_{mse} relative to each other. In this way, E_{cf} acts as a non-linear scaling factor applied to the inverse of E_{mse}. Figure 5 shows E_{cqm} in the same environment as figures 1 and 4. This time the true robot position at the central peak is more pronounced with respect to the surrounding positions, and is in fact decades higher in magnitude than neighboring regions.

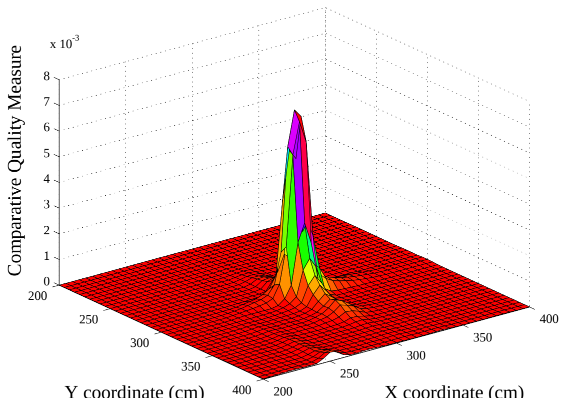
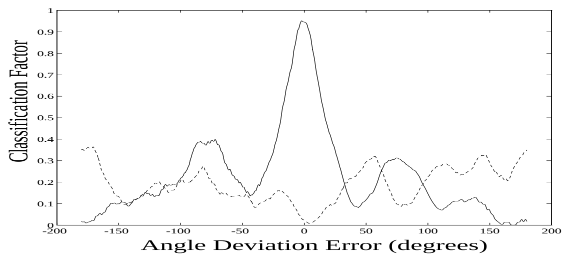


Figure 5: The Comparative Quality Measure (E_{cqm}); a=2, b=1

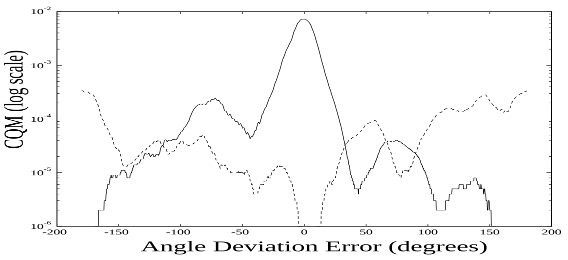
6 Orientation Correction

The approach to orientation correction is based on the two quality measures, the classification factor E_{cf} and the comparative quality measure E_{cqm}. Consider figure 6: Near the true orientation (an angle deviation/error of 0°), E_{cqm} and E_{cf} are rather well-behaved convex functions of angle deviation θ_d. Therefore, if the estimate of the robot's position is close enough, then the proper orientation of the robot can be found by maximizing E_{cqm}. Since we know a given pose estimate is good if E_{cf} exceeds some threshold, we can tell when the estimate is good enough to ensure that E_{cqm} is a well-defined, single-peak function. Figure 6 shows that both orientation **and** position must be correct in order to optimize the quality measures. It also shows that they are convex functions only for pose estimates local to the true pose, so gradient searches to find a global maximum will only work in this region. However, the angle domain that bounds this convexity are not constant for all poses in all environments. Therefore, it is useful to use a modified E_{cqm}, which we call $\hat{E}_{cqm}$, defined as:

$$\hat{E}_{cqm} = \begin{cases} E_{cqm} & \text{if } E_{cf} \geq \text{Acceptance Threshold} \\ 0 & \text{otherwise} \end{cases} \qquad (7)$$



(a) Angle Deviation vs. E_{cf}



(b) Angle Deviation vs. E_{cqm} (log scale)

Figure 6: Variations in Quality Measures as functions of Angle Deviation: the true robot orientation is at an angle deviation of 0°. The solid line represents the quality measures at true robot position and varying orientation, while the broken line represents the quality at an incorrect position (about 100 cm away).

Thus, once we have a good position estimate (from position correction) and a relatively small error in orientation, we can correct the angle error by optimizing $\hat{E}_{cqm}$: if $\hat{E}_{cqm} = 0$, then we know right away that our local pose estimate is too poor to correct orientation.

Now we have the tools to formulate a complete localization algorithm when given a pose estimate:

1. Do position correction as before, except that each iteration, check E_{cf} to see if it exceeds the acceptance threshold.

2. If $E_{cf} >$ threshold, then the pose estimate is close enough to be corrected. Maximize $\hat{E}_{cqm}$ as a function of angle deviation θ_d by using a maximization technique such as Brent's method. Direct gradient descent methods may also be used if derivative information is approximated.

3. Once the maximum is found, update the orientation estimate θ, and reiterate.

4. For speed purposes, if θ changes very little over the course of a few iterations, ignore future orientation corrections and concentrate on refining position.

6.1 Global Localization

Global localization refers to the case where we have a map of the environment, but no prior estimate of the robot's pose. To use the localization algorithm developed thus far, a initial pose estimate is required. We describe a localization procedure based on the local pose estimator that does not require an initial pose estimate. This is done in a similar fashion to the manner in which position correction was incorporated into orientation correction: by optimization of the quality measures.

The pose of the robot within the environment can be considered as the domain of a quality function $E_{cqm}(x, y, \theta)$. Each pose (x, y, θ) within the map region is considered an initial pose estimate of the robot. E_{cqm} may be calculated following local pose localization at one of these initial poses. This gives a global quality function that describes the quality of localization when applied to a particular location, and the resulting pose for which E_{cqm} is a global maximum (not the initial pose) is the true pose of the robot. This results in a highly non-convex function (although convex local to the global maximum) because any estimate that converges closely to the true pose will have a high quality, and any that do not will have a much lower quality. Therefore, local gradient information in the lower valued regions may not assist in the search for the global maximum. However, maximization is still possible by exhaustive search or other non-convex maximization techniques.

7 Discussion and Conclusions

This paper presents a geometric method to generate maps from sonar data and to perform localization based on a line segment map of obstacles that need not be individually identified. Individual sonar data were classified using a weighted soft non-linearity that combined robustness with graceful degradation.

Given maps of this construction, this paper addressed localization in terms of hierarchical techniques for position-only localization, local pose localization (refining both position and orientation), and global localization. The choice of which of these is required depends on the assumptions that could be made within a given environment and with a given robot and sensor.

Even using position-only localization (which assumes no orientation error) performance was very good for the range of position errors likely to be encountered in practice, small errors in actual orientation and for office-like environments. General pose localization is more robust since orientation is also corrected – this is most appropriate to typical operations. Global localization, while more computationally costly, is required if no pose estimates are available. This can arise when a system has been powered-down or when external influence leads to a large positional errors. In areas where insufficient environmental structure is observable practical systems would normally make use of dead-reckoning information as well.

References

[1] C. Biber, S. Ellin, E. Shenk, and J. Stempeck. The polaroid ultrasonic ranging system. *Proc the 67th Convention of the Audio Engineering Society*, 1980.

[2] Ingemar J. Cox. Blanche - an experiment in guidance and navigation of an autonomous robot vehicle. *IEEE Transactions on Robotics and Automation*, 7(2):193–204, April 1991.

[3] Gregory Dudek, Michael Jenkin, Evangelos Milios, and David Wilkes. Sonar sensing and obstacle detection. In *Proc. of the Conference on Military Robotics*, Medecine Hat, Alberta, September 1991.

[4] Gregory Dudek and Paul MacKenzie. Model-based map construction for robot localization. In *Proceedings of Vision Interface 1993*, North York, Ontario, May 1993.

[5] Javier Gonzalez, Anthony Stentz, and Anibal Ollero. An iconic position estimator for a 2d laser rangefinder. In *IEEE International Conference on Robotics and Automation*, pages 2646–2651, May 1992.

[6] Alois A. Holenstein, Markus A. Müller, and Essam Badreddin. Mobile robot localization in a structured environment cluttered with obstacles. In *IEEE International Conference on Robotics and Automation*, pages 2576–2581, Nice, France, May 1992.

[7] John J. Leonard and Hugh F. Durrant-Whyte. Mobile robot localization by tracking geometric beacons. *IEEE Transactions on Robotics and Automation*, 7(3):376–382, June 1991.

[8] John J. Leonard, Hugh F. Durrant-Whyte, and Ingemar J. Cox. Dynamic map building for an autonomous mobile robot. *The International Journal of Robotics Research*, 11(4):286–298, August 1992.

[9] David Marr. *Vision.* W. H. Freeman and Co., New York, 1982.

[10] Azriel Rosenfeld and Avinash C. Kak. *Digital Picture Processing.* Academic Press, New York, 1976.

[11] Godfried Toussaint. A graph-theoretical primal sketch. *Computational Morphology*, 1988.

[12] Jiang Wang and William J. Wilson. 3d relative position and orientation estimation using kalman filter for robot control. In *IEEE International Conference on Robotics and Automation*, pages 2638–2645, Nice, France, May 1992.