A Bounded Uncertainty Approach to Multi-Robot Localization

John R. Spletzer Camillo J. Taylor

GRASP Laboratory – University of Pennsylvania
Philadelphia, PA, 19104, USA
{spletzer, cjtaylor}@grasp.cis.upenn.edu

Abstract

We offer a new approach to the multi-robot localization problem. Using an unknown-but-bounded model for sensor error, we are able to define convex polytopes in the configuration space of the robot team that represent the set of configurations consistent with all sensor measurements. Estimates for the uncertainty in various parameters of the team's configuration such as the absolute position of a single robot, or the relative positions of two or more nodes can be obtained by projecting this polytope onto appropriately chosen subspaces of the configuration space. We propose a novel approach to approximating these projections using linear programming techniques.

The approach can handle both bearing and range measurements with a computational complexity scaling polynomially in the number of robots. Finally, the workload is readily distributed - requiring only the communication of sensor measurements between robots. We provide simulation results for this approach implemented on a multi-robot team.

1 Introduction

Localization is a critical base level capability for mobile robots, enabling numerous other technologies including mapping, manipulation, and target tracking. It is not surprising then that considerable research effort has been directed at this problem [1, 2, 3, 4, 5]. Within this realm of research, there is a narrower yet still significant focus on *cooperative localization* (*CL*) for multi-robot teams. In this paradigm, groups of robots combine sensor measurements to improve localization performance. This approach is motivated by the fact that robots within a team can often identify one another and communicate sensor measurements, such as relative range and bearing.

In this paper, we offer an alternative approach to *CL* . Conceptually, the idea is that bearing and range measurements induce linear constraints on the configuration space of the robot team. Merging these constraints induces a convex polytope $\mathbf{P}$ on this configuration space that represents the set of all configurations consistent with sensor measurements. Estimates for the uncertainty in various parameters of the team's configuration such as the absolute position of a single robot, or the relative positions of two or more nodes can be obtained by projecting this polytope onto appropriately chosen subspaces of the configuration space. Unfortunately, recovering the exact projection is quite cumbersome. The number of vertices in $\mathbf{P}$ can be extremely large, and determining the projection $\mathcal{P}$ for a single robot can require exponential time. We propose a novel approach to approximating these projections using linear programming techniques.

2 Related Work

Our research relates to cooperative localization techniques in which sensor measurements from multiple robots are integrated to estimate uncertainties in absolute or relative position. Roumeliotis and Bekey [4] offered a Kalman filter based approach for addressing interdependencies in uncertainty propagation for multi-robot localization. In their representation, the uncertainty was described in $m \times n$ dimensional space, where $m = 3$ corresponded to the position and orientation, and n the number of robots. With the higher dimensional space representation, cross correlation terms were maintained in a global state covariance matrix, and the Kalman gain adjusted to account for these during future pose updates.

More recently, Howard *et al* [5] also employed a Bayesian approach to cooperative localization. In this work, a robot maintained an estimate for the relative position of each of its $n-1$ teammates as a particle set. Interdependencies between distributions were approximated by maintaining a *dependency tree* for each particle set. This assumed that a given uncertainty distribution was only dependent on the last distribution from which it was updated. The authors noted the potential weaknesses of this approximation, and also offered alternative approaches for more accurately maintaining distribution dependencies: maintaining $O(n^2)$ particle sets each, or generating a similar number of communication broadcasts.

However, most related to our *CL* approach is work by Doherty & El Ghaoui [6]. In this, a similar method relying upon linear and semi-definite programming (SDP)

techniques was used to estimate the positions of nodes in a wireless network from sensor measurements when the positions of some of the nodes were known *a priori*. In contrast to our approach, the objective was not recovering the position uncertainty region, but obtaining point estimates for the node positions. This was accomplished by bounding the feasible set with a rectangle. Such an approach can yield an arbitrarily bad estimate of the uncertainty region. Our *CL* approach augments this work by offering a method to recover the true uncertainty region arbitrarily well. Additionally, through the use of our inside-outside (IO) approximation techniques we are able to obtain a tight convergence bound.

3 Generating Sensor Constraints

The proposed approach relies upon modeling the bearing and range measurements obtained by the robots as linear constraints on the configuration of the robot team. This section focuses on the transformations required to realize this. It should be noted that while the methods presented below emphasize constraint generation in $\mathbb{R}^2$, they are extendable to $\mathbb{R}^3$.

Let $x_i, x_j \in \mathbb{R}^2$ represent the positions of robots i, j relative to some common reference frame. Again in a slight abuse of notation, we also let x_i denote the ith robot itself. We assume each of our n robots is equipped with sensors allowing it to measure bearing and possibly range to another robot with bounded error. We let α_{ij} and r_{ij} represent the bearing and range measurements taken by x_i to x_j. We further assume that each agent is able to infer its orientation θ with respect to a specified reference direction with bounded error. This can be accomplished in a distributed fashion via a local sensor (*e.g.* a compass) or cooperatively by propagating orientation uncertainty across coordinate frames as in [7]. No assumptions are made regarding the homogeneity of sensors used or of the sensors' performance.

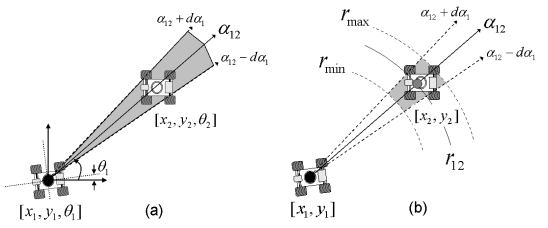

Figure 1: Bearing (left) and range measurements on the plane. Assuming bounded error models, these define two and four constraints, respectively. While bearing measurements translate directly, range measurements must be linearized over the bounds of bearing uncertainty.

Bearing measurements translate directly into linear constraints. Consider Figure 1a. Assuming bounded error in bearing measurements, α_{12} defines an uncertainty sector formed by the rays $[x_1, \alpha_{12} + d\alpha_1]$ and $[x_1, \alpha_{12} - d\alpha_1]$. Here $d\alpha_1$ reflects the uncertainty in both bearing and orientation (θ) measurements. Note that the bounds on bearing error are simply inflated to account for uncertainty in orientation. Denoting $\alpha_{max} = \alpha_{12} + d\alpha_1, \alpha_{min} = \alpha_{12} - d\alpha_1$, the uncertainty region is defined by the linear inequality constraints

$$\begin{aligned}(x_2 - x_1)\sin\alpha_{min} + (y_1 - y_2)\cos\alpha_{min} &\leq 0\\ (x_1 - x_2)\sin\alpha_{max} + (y_2 - y_1)\cos\alpha_{max} &\leq 0\end{aligned}$$

For range measurements, the procedure is only slightly more complicated. We retain the same two inequality constraints for the bearing measurements, and add two additional constraints associated with the bounds on our ranging error. These range constraints are inherently non-linear, however in many situations they can be adequately approximated by linear inequalities. Assume that the true range measurement can be bounded by $r_{min} \leq r_{12} \leq r_{max}$. Combining this with the bearing constraints results in an annular sector of uncertainty U, as illustrated in Figure 1b. By choosing two linear constraints orthogonal to α_{12} and supporting to the extreme points of U, we generate a new uncertainty region $\hat{U}|U \subset \hat{U}$. This is reflected in Figure 2. The corresponding constraints can then be written as

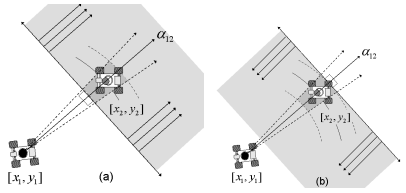

Figure 2: Range constraints are approximated using linear constraints orthogonal to α_{12} and supporting to the original uncertainty region.

$$\begin{aligned}(x_2 - x_1)\cos\alpha_{12} + (y_2 - y_1)\sin\alpha_{12} &\leq r_{max}\\ (x_1 - x_2)\cos\alpha_{12} + (y_1 - y_2)\sin\alpha_{12} &\leq r_{min}\cos d\alpha_1\end{aligned}$$

This bounding via linear constraints has the effect of overestimating the original uncertainty region. This overestimation is a function of the original shape of U, and is governed by the following relationship

$$\frac{Area(\hat{U})}{Area(U)} = \frac{r_{max}^2\tan(d\alpha) - r_{min}^2\sin(d\alpha)\cos(d\alpha)}{(r_{max}^2 - r_{min}^2)d\alpha}$$

which indicates that as $d\alpha$ becomes large or when $r_{min} \approx r_{max}$, our overapproximation can become significant. Several examples can be found in Figure 3. However, for a wide range of measurements this overapproximation amounts to only several percent. For

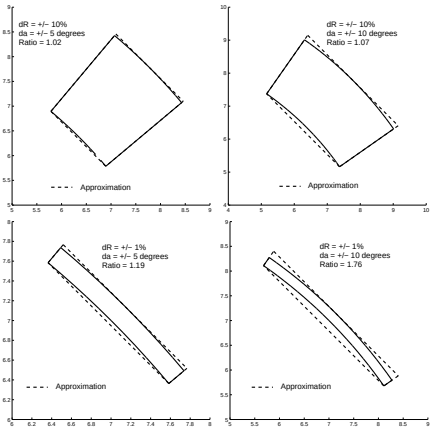

Figure 3: Approximating range-bearing constraints. The examples reflect range uncertainties of $\pm$5-10%, with bearing uncertainties of $\pm$5-10°. Low range and high bearing uncertainties admit greater overestimation of the uncertainty region.

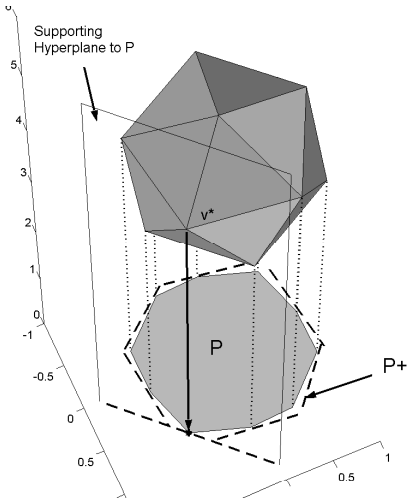

Figure 4: A polytope $\mathbf{P}$ in $\mathbb{R}^3$ and its $\mathbb{R}^2$ projection $\mathcal{P}$ (shaded polygon). $\mathcal{P}$ can be approximated by solving multiple linear programming problems - each yielding an extreme point v^* on the polytope and its supporting hyperplane. The projections of these correspond to half-spaces in $\mathbb{R}^2$. Intersecting these generates a bounding approximation $\mathcal{P}^+$ to the true projection.

4 The Localization Approach

Let $\mathcal{C} \subset \mathbb{R}^{(m \cdot n)}$ denote the configuration space of our robot team. Let $\vec{x} = [x_1, ..., x_n] \in \mathcal{C}$ denote the current configuration of the ensemble. Note that $\vec{x}$ is simply the concatenation of the position vectors of the n robots, and $m \in \{2, 3\}$ denotes the dimension of these position vectors. With this representation, bearing and range sensor measurements can be modeled as linear constraints on $\vec{x}$, as described in the previous section. By concatenating these constraints, we obtain a system of linear inequalities of the form $\boldsymbol{A}\vec{x} \leq b$, where $A \in \mathbb{R}^{l \times (m \cdot n)}$ represents the sensor constraints.

Implicitly, the matrix $\boldsymbol{A}$ induces a polytope $\mathbf{P} \subset \mathcal{C}$ on the configuration space of the formation which corresponds to the set of all configurations consistent with the sensor measurements.

Key to our approach is the concept of projection. The projection $\pi : \mathbb{R}^{(m \cdot n)} \rightarrow \mathbb{R}^k$ is defined as

$$\pi(\vec{x}) = \boldsymbol{D}\vec{x} \tag{1}$$

where the projection matrix $\boldsymbol{D}$ maps the configuration vector $\vec{x}$ from $\mathbb{R}^{m \cdot n}$ to $\mathbb{R}^k$. Typically the dimension k will be relatively small. For our purposes, we are interested in projections of the form

$$\pi_i(\vec{x}) = x_i \tag{2}$$

$$\pi_{ij}(\vec{x}) = x_i - x_j \tag{3}$$

which correspond to the position of x_i, and the relative positions of x_i and x_j, respectively. Other useful projections can be imagined.

This notion of projection can be extended to describe the process of projecting a region $\mathbf{P}$ onto a k-dimensional subspace.

$$\mathcal{P} = \Pi(\mathbf{P}) = \{\pi(\vec{x}) | \vec{x} \in \mathbf{P}\} \tag{4}$$

The exact projection $\mathcal{P}$ can actually be obtained using general algorithms for projecting a d dimensional polytope onto a k dimensional subspace, such as [8]. Unfortunately, such approaches run in exponential time in the number of variables for the worst case. We offer our own extreme point/convex hull based approach which exploits linear programming techniques and offers a similar running time. However, it also has the very beneficial attribute that at each iteration of the algorithm, upper and lower bound estimates for the true projection are recovered. These estimates are refined after each iteration, allowing it to be employed as an approximation algorithm which runs in polynomial time.

5 Approximating the Projection

In outlining the concepts behind our approximation algorithm, we refer to Figure 4. In this example, the polytope $\mathbf{P} \in \mathbb{R}^3$ is projected onto $\mathbb{R}^2$ generating the projection $\mathcal{P}$ (shaded polygon). For our purposes, generating the true projection $\mathcal{P}$ is computationally too expensive. Instead, our approximation approach chooses a search direction and finds the maximum extent of $\mathcal{P}$ in this direction.

This amounts to solving the linear programming problem

$$\max_{\vec{x}} c_i^T \vec{x} \quad \boldsymbol{A}\vec{x} \leq b \tag{5}$$

where c_i corresponds to the search direction, and is chosen parallel to the projection subspace (*i.e.* $c_i \in Span(\boldsymbol{D}^T)$) .

The solution to this linear programming problem yields a vertex $v_i^* \in \mathcal{V}$ in direction c_i and its supporting hyperplane H_i, where $\mathcal{V} \subseteq \mathbf{P}$ are the extreme points of $\mathcal{P}$. The projection of H_i induces a half-space constraint on the subspace under consideration. By repeating this process for several search directions, we obtain an approximation $\mathcal{P}^+$ for the true projection by intersecting these half-spaces. This is denoted by the dashed polygon in Figure 4. Note that $\mathcal{P}^+$ is guaranteed to bound the true projection $\mathcal{P}$.

6 Observations on the Approximation

Recall that the true projection $\mathcal{P}$ will typically not be recovered using our approach, and must be approximated from a limited number of searches on **P**. We can generate an upper bound estimate $\mathcal{P}^+$ from the projected hyperplanes. Let $\mathcal{V}^-$ denote the set of projected extreme points corresponding to different solutions of the linear programming problem in Equation 5. These are represented by the grey vertices in Figure 5. Similarly let $\mathcal{V}$ and $\mathcal{V}^+$ represent the vertices of the true projection $\mathcal{P}$ and the bounding approximation $\mathcal{P}^+$, respectively.

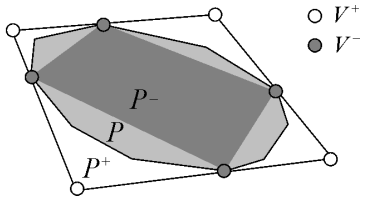

Figure 5: Set hierarchy generated through projection estimates. $\mathcal{P}^+$ and $\mathcal{P}^-$ correspond to upper and lower bounds to the true set projection $\mathcal{P}$.

Note that $\mathcal{V}^- \subseteq \mathcal{V}$, *i.e.* the projected vertices $\mathcal{V}^-$ are a subset of the true projection's vertex set $\mathcal{V}$. This means we can also construct a lower bound estimate $\mathcal{P}^-$ for $\mathcal{P}$ from the convex hull of $\mathcal{V}^-$. This formalizes a set hierarchy $\mathcal{P}^- \subseteq \mathcal{P} \subseteq \mathcal{P}^+$ from which we define the following performance metrics to quantify the closeness of our approximation.

$$\Delta\mathcal{P} = Area(\mathcal{P}^+ \setminus \mathcal{P}^-) \tag{6}$$

$$PR = \frac{Area(\mathcal{P}^-)}{Area(\mathcal{P}^+)} \tag{7}$$

Here $\Delta\mathcal{P}$ corresponds to the absolute difference in the area of the two convex sets, whereas the *performance ratio* PR corresponds to the relative difference in projection area. Either metric can be used as a termination criterion on the number of search directions. Since we generate upper ($\mathcal{P}^+$) and lower ($\mathcal{P}^-$) bounds of the projection estimate at each iteration, we can think of the two as corresponding to primal-dual problems, and where our $\Delta\mathcal{P}$ corresponds to the duality gap of the solution.

7 Choosing Search Directions

Each iteration of Equation 5 runs in polynomial time in the number of robots and sensor measurements. As a consequence, the number of search directions used to approximate $\mathcal{P}$ should be minimized. A straightforward approach would be to choose a maximum allowable number of iterations, and uniformly discretize the search space. We offer an alternative maximin approach for our $\Delta\mathcal{P}$ metric. Successive search directions are chosen to maximize the minimum possible reduction in $\Delta\mathcal{P}$. This will also allow us to characterize the convergence performance for our $\Delta\mathcal{P}$ metric.

7.1 Initialization

Convergence characteristics for our performance metrics will be strongly dependent on the initial estimates for P^+ and P^-. Our initialization scheme provides the following performance bounds after only four search iterations:

1. A 2-approximation for the performance ratio ($PR_4 = 0.5$)

2. $\Delta\mathcal{P}_4 \leq Area(\mathcal{P})$

Again, let $\mathcal{V}, \mathcal{V}^+, \mathcal{V}^-$ denote the vertex sets for $\mathcal{P}, \mathcal{P}^+, \mathcal{P}^-$, respectively. The initialization process is then described in Figure 6.

Our initial search direction $c_1 \in Span(\boldsymbol{D}^T)$ can be chosen randomly without affecting the convergence analysis. The second search direction c_2 is then chosen opposite to c_1. As a result, we are guaranteed to recover two vertices in the true projection $\mathcal{V}$. This is reflected in Figure 6(a-b). At this point, the set $\mathcal{P}_2^-$ will be the line segment $\overline{v_1 v_2}$, and $\mathcal{P}_2^+$ a band formed from the two projected hyperplanes. The next two search directions c_3, c_4 are chosen orthogonal to the feasible set $\mathcal{P}_2^-$ and with $c_4 = -c_3$, as shown in Figure 6(c-d). Such a strategy essentially ensures that the first four searches will recover four vertices in $\mathcal{V}$. There is a "degenerate" case where only three vertices will be recovered. This occurs when one side of the true projection corresponds to a side of $\mathcal{P}^+$ as described in Figure 7. Such an occurrence is only possible for projections where the vertices of one edge are both minimal and maximal for a given search direction. Regardless, we show that the convergence bounds hold for the degenerate case as well.

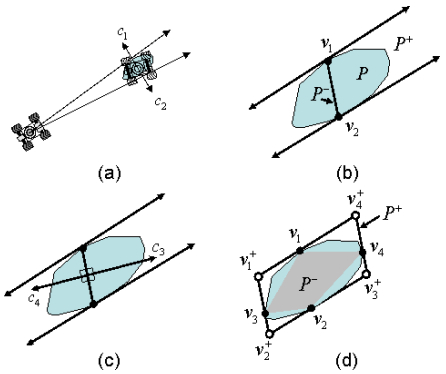

Figure 6: Initializing $\mathcal{P}^+$ and $\mathcal{P}^-$. After 4 searches, $\mathcal{P}^+ = 2\mathcal{P}^-$.

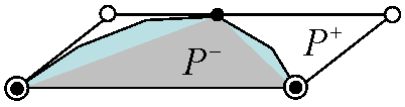

Figure 7: An example of a "degenerate" initialization, where only 3 vertices of $\mathcal{V}$ are recovered. Convergence bounds also hold for this case.

Lemma 7.1. *After 4 searches, $\mathcal{A}^+ = 2\mathcal{A}^-$ where $\mathcal{A}^+$ and $\mathcal{A}^-$ correspond to $Area(\mathcal{P}^+)$ and $Area(\mathcal{P}^-)$, respectively.*

Proof. There are two cases: the degenerate case, where three projection vertices are recovered (Figure 7), and the standard case described in Figure 6. In the former case, $\mathcal{P}^-$ is denoted by a triangle inscribed within the parallelogram $\mathcal{P}^+$. Since the base of the triangle must coincide with one of the sides of the parallelogram, the latter must have exactly twice its area. In the standard case, we have a quadrilateral $\mathcal{P}^-$ inscribed within a parallelogram $\mathcal{P}^+$. However, referring to Figure 6d we see that the line segments $\overline{v_1v_2}$, $\overline{v_1^+v_2^+}$, $\overline{v_3^+v_4^+}$ are parallel, and the standard case reduces to two subproblems equivalent to the degenerate case. □

Lemma 7.1 provides our 2-approximation bound for PR, and since our set hierarchy is governed by $\mathcal{P}^- \subseteq \mathcal{P} \subseteq \mathcal{P}^+$, we also obtain our bound for ΔP. This is a significantly better result than if a bounding rectangle were recovered, which can provide an arbitrarily bad initialization.

7.2 Choosing Subsequent Search Directions

Referring to Figures 6 and 7, we see that after initialization, the set $\Delta\mathcal{P} = \mathcal{P}^+ \setminus \mathcal{P}^-$ will correspond to disjoint triangular regions. The search strategy continues by considering each triangle that forms $\Delta\mathcal{P}$. In Figure 8(a), this would correspond to the 4 triangles ABE, ADH, BCF and CDG. Each of these can be thought of as a prospective search region where $\mathcal{P}$ is poorly defined. Our search strategy then proceeds as follows:

1. Determine the prospective search region of greatest area.
2. Choose a search direction $c^\perp$ normal to the corresponding edge of $\mathcal{P}^-$.
3. Solve Equation 5 with $c = c^\perp$.
4. Refine estimates for $\mathcal{P}^+$ and $\mathcal{P}^-$ accordingly.

The is illustrated in Figure 8. Using such a strategy allows us to bound the number of searches necessary to recover $\mathcal{P}$ as a function of the number of its vertices $\mathcal{V}$.

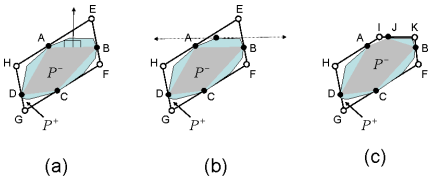

Figure 8: Generating projection approximations: after initialization, subsequent search directions explore the largest disjoint region where the projection is poorly defined (a-b), and $\mathcal{P}^+$ and $\mathcal{P}^-$ are updated accordingly(c).

Lemma 7.2. *By choosing search directions normal to the edges of search regions, all of the vertices $\mathcal{V}$ of the projection $\mathcal{P}$ can be recovered in at most $2|\mathcal{V}|$ searches.*

Proof. Referring to Figure 9, there are four possible outcomes from exploring a search region. In each search we will either:

a. Locate a new vertex in $\mathcal{V}$, and as a consequence increase the number of search regions by one

b. Locate a new vertex in $\mathcal{V}$, while maintaining the number of search search regions.

c. Locate a vertex already in $\mathcal{V}^-$, while eliminating the search region.

d. Locate a new vertex in $\mathcal{V}$ already in $\mathcal{V}^+$, while eliminating the search region.

Since we can only discover $|\mathcal{V}|$ vertices, we can create no more than $|\mathcal{V}|$ additional search regions - each of which will subsequently be eliminated with a single search, for a maximum of $2|\mathcal{V}|$ searches. □

This is of significance, as it shows that our approach is deterministic and will not create an unbounded number of

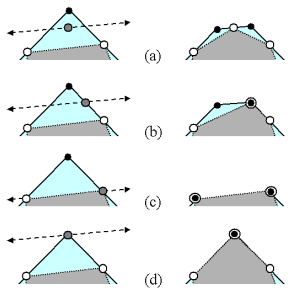

Figure 9: The possible outcomes from exploring a search region between $\mathcal{P}^+$ and $\mathcal{P}^-$. At most one additional search region is added as a consequence of locating a new vertex in the actual projection $\mathcal{P}$, or the search region is itself eliminated. This bounds the number of searches necessary to recover $\mathcal{P}$ to $2|\mathcal{V}|$.

vertices in approximating $\mathcal{P}$. At any given time in the approximation algorithm, $|\mathcal{V}^+| \leq 2|\mathcal{V}| \leq 2|\mathcal{V}^-|$. Unfortunately, $|\mathcal{V}|$ can grow exponentially with the number of constraints and problem dimension. This means that recovering all of the vertices of $\mathcal{P}$ is typically not an attractive option. However, we can characterize the performance of this search strategy against both our PR and $\Delta\mathcal{P}$ approximation metrics.

Lemma 7.3. *Let $ABE \in \mathcal{P}^+ \setminus \mathcal{P}^-$ represent a search region where $\mathcal{P}$ is poorly defined. Choosing a search direction normal to edge $\overline{AB}$ will increase $\mathcal{P}^-$ and decrease $\mathcal{P}^+$ so the area of search region ABE will be reduced by at least a factor of 4.*

Proof. Referring to Figure 10, searching normal to edge $\overline{AB}$ will yield projections of an extreme point and hyperplane parallel to $\overline{AB}$. The new projected extreme point F must lie somewhere in ABE. F then corresponds to a (potentially new) vertex in $\mathcal{V}^-$, and the intersection of the projected hyperplane with the edge segments $\overline{AE}$ and $\overline{BE}$ corresponds to new vertices in $\mathcal{V}^+$. The regions ABH and EGI then correspond to areas that will be added to $\mathcal{P}^-$ and removed from $\mathcal{P}^+$, respectively. Recall our definition of $\Delta\mathcal{P}$ and letting $h_1 = h - h_2$ we obtain

$$\begin{aligned}\Delta\mathcal{P}_{i+1} &= \mathcal{P}^+_{i+1} - \mathcal{P}^-_{i+1} \\ &= \mathcal{P}^+_i - \frac{1}{2}\overline{AB}(h - h_1) - \mathcal{P}^-_i - \frac{1}{2}\overline{AB}\frac{h_1^2}{h}\end{aligned}$$

Differentiating this with respect to h_1 and setting equal to zero, we see that the expression is *maximized* for $h_1 = h_2 = \frac{1}{2}h$, which will provide worst case convergence. The remaining area of the original ABE which remains can be determined as follows:

$$\begin{aligned}\frac{Area_{new}}{Area_{old}} &= \frac{ABE - ABH - EGI}{ABE} \\ &= \frac{b(h_1+h_2) - b(h_2) - \frac{bh_1}{(h_1+h_2)}(h_1)}{b(h_1+h_2)} \\ &= \frac{h_1 h_2}{(h_1+h_2)^2} \leq \frac{1}{4}\end{aligned}$$

□

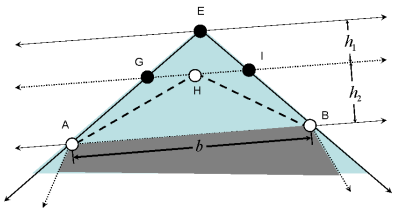

Figure 10: By projecting normal to the edge in $\mathcal{P}^-$, we create two disjoint regions ABH and EGI. These will be added to and subtracted from $\mathcal{P}^-$ and $\mathcal{P}^+$, respectively. The net result will reduce the search region area ABE by at least a factor of 4.

By using this ratio in concert with the bounds established at initialization - and always choosing the largest search region at each search iteration - we can characterize the convergence rate of our approach. For our ΔP metric, we obtain the following result:

Lemma 7.4. *The proposed search algorithm converges in the worst case at a rate of $1/s^2$ where s denotes the number of search iterations.*

Proof. Letting $\mathcal{A}$ denote $Area(\mathcal{P})$, after initialization we have:

$$\Delta\mathcal{P}_4 = Area(\mathcal{P}^+_4 \setminus \mathcal{P}^-_4) \leq \mathcal{A} \tag{8}$$

where in the worst case this area is distributed amongst 4 disjoint search regions. After searching these 4 regions - regardless of order - we obtain

$$\Delta\mathcal{P}_8 \leq \frac{\mathcal{A}}{4} \tag{9}$$

by Lemma 7.3. This remaining area will now be distributed across up to 8 search regions in accordance with Lemma 7.2. Thus, in the worst case the number of regions searched doubles in order to reduce the area by a factor of four. From this, for $s \geq 4$ we obtain the relation

$$\Delta\mathcal{P}_s \leq \frac{\mathcal{A}}{4^{k-2}} = \frac{16\mathcal{A}}{4^k} = \frac{16\mathcal{A}}{2^{2k}} = \frac{16\mathcal{A}}{2^k 2^k} \tag{10}$$

where $k = \lfloor \log_2 s \rfloor$. Noting that $2^{\lfloor \log_2 s \rfloor} > \frac{s}{2}$, we obtain

$$\Delta\mathcal{P}_s \leq \frac{64\mathcal{A}}{s^2} = O(\frac{1}{s^2})\mathcal{A} \tag{11}$$

□

Thus, given a normalized tolerance for $\Delta\mathcal{P}$, we can determine the worst case number of search iterations j to reduce it to a desired value $\Delta\mathcal{P}_j$. A normalized log-plot of the worst case convergence rate as a function of the number of search iterations is at Figure 11.

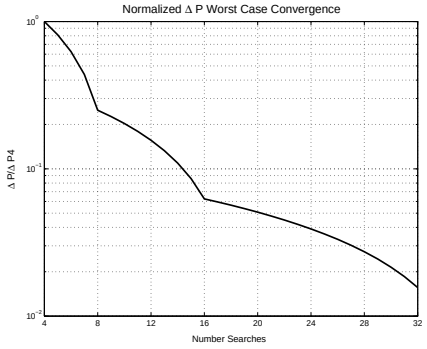

Figure 11: Worst case convergence of $\Delta\mathcal{P}$ versus the number of search iterations.

What is most significant about this result is that the convergence rate holds *regardless of the number of robots, sensor measurements, or constraints!* Thus, the complexity of the approach will only scale in accordance with the linear programming portion of the algorithm.

We have obtained similar convergence results for our PR metric. However, including these would exceed the page requirements for this document. As a summary of the results, we will achieve PR levels of 0.85, 0.9, 0.95, and 0.99 within 8,12,15, and 30 iterations, respectively.

8 Simulation Results

We implemented a bearings-only localization simulator as a proof-of-concept for our approach. For the sake of expediency, we assumed the sensors to be homogeneous (although this is *not* required). We further assumed that each robot was capable of measuring the bearing to its neighbor in a common reference frame with a tolerance of ± 5 degrees. The actual error model used in corrupting sensor measurements was a uniform distribution over this same range.

Numerous simulations were run with formation sizes ranging from 5-20 agents. In each of these, we contrasted the performance of our search approach vs. a uniformly discretized search technique. Results from a sample simulation with 12 agents can be found in Figures 12-13.

Here, the common localization frame is relative to the reference robot. Since we are localizing to a scale (bearings-only), a second robot's x coordinate was chosen to represent the formation scale. Thus, although there were 12 robots the optimization was constrained to 21 variables. Additionally, the sensor range was constrained so that only

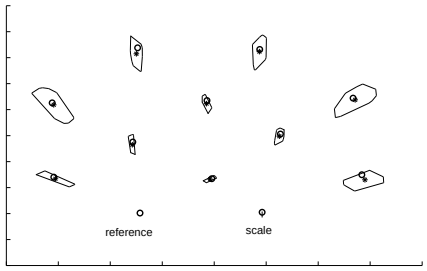

Figure 12: Localization of a robot formation. The true position of each agent is reflected by the circle. The surrounding polygons reflect the uncertainty region for each robot's position. The star reflects the centroid of this polygon, which can be used as a point estimate for robot position if necessary.

a robots immediate neighbors were visible. As a result, multiple frame (and as a result uncertainty) transformations were necessary from the reference frame to the distant edges of the formation. As expected, the uncertainty regions bound the actual position (blue circle) of each robot, and grow dramatically with the number of frame transforms from the reference/scale robots. The centroid of these uncertainty regions (red star) can be used as point estimates for robot positions if required.

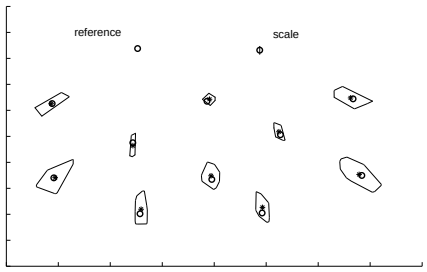

Figure 13: Localization of the same formation relative to an alternate reference frame. While localization to a common reference frame may be required for a team level task, individual robots may be more interested in position estimates of their immediate neighbors for local tasks. The choice of which positions to recover can be decided at the robot level.

The only difference between Figures 12-13 is the reference frame from which the estimates were recovered. We do this to emphasize the flexibility of our approach. For a team level task, it may be necessary for each robot to localize relative to a common reference frame (*i.e.* a team leader). In this event, the robots would communicate their sensor measurements, individually estimate their own relative position, and communicate these as necessary. Conversely, a robot may only be interested in recovering position estimates relative to *its own* reference frame.

The motivation for this is illustrated in Figure 13. Position uncertainty regions grow significantly with the number of frame transformations (although point estimates remain quite good), and may be of little interest for local tasks. The decision as to which position estimates to recover can be decided at the robot level.

We should also emphasize that the choice of which reference frame to recover to is transparent to our algorithm. *Relative* sensor measurements are used, and no frame transformations are required. Only the equality constraints of the reference frame are affected (*e.g.* $x_i = 0, y_i = 0$).

The corresponding plots of our ΔP and PR metrics as a function of the number of search iterations are at Figure 14. These reflect the mean values for the 10 free robots, and clearly shows the advantage of our search strategy over a uniform-search technique. While the former achieves a 90% PR after 7 search iterations, the latter fails to achieve this level of performance after 15 search iterations.

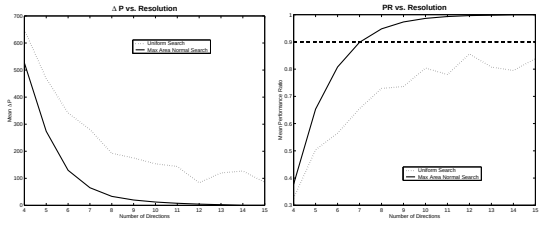

Figure 14: ΔP and PR metrics versus search iterations, clearly showing the advantage of our search strategy over a uniform search.

9 Discussion and Conclusions

To date, our results are primarily from static localization. We envision these results readily extending to dynamic operations. Recall that the position estimate $\mathcal{P}^+$ for each robot is formed through the intersection of constraints in $\mathbb{R}^m$. These constraints can be propagated using a bounded uncertainty motion model. Since interdependencies are handled by our bounded error model, these propagated constraints can be directly merged with those generated from current sensor measurements. Such a filter would propagate a limited constraint set at each time step, and as a consequence run with a computational complexity similar to the static case.

Our localization approach offers several desirable features in the context of multi-robot operations. It offers performance guarantees by providing an exact uncertainty representation guaranteed to contain the robots' positions. Interdependencies in sensor/position estimates are also handled in a very principled way as a result of employing a bounded error sensor model. Robots need only communicate sets of linear constraints derived from measurements or from projection operations. Individual robots can estimate parameters of interest through projections onto relevant subspaces.

Unlike Extended Kalman filter based approaches, there are no linearizing assumptions and as a consequence uncertainty estimates will not be dependent upon the order in which the measurements are combined. It also eliminates the need for approximating interdependencies as in [5]. This comes at a cost, as our projection procedure is more involved than Kalman filter updates. Nonetheless, from empirical evidence in [6] for a similar LP problem, we expect the approach to run in real time for large (100+) robot formations. We will be contrasting these approaches more formally in the near future.

Acknowledgments: This material is based upon work supported by the National Science Foundation under Grant Nos. 9875867 and 0130858. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] S. Majumder, S. Scheding, and H. Durrant-Whyte, "Multi-sensor data fusion for underwater navigation," *Robotics and Autonomous Systems*, vol. 35, no. 1, pp. 97–108, 2001.

[2] S. Thrun, "A probabilistic online mapping algorithm for teams of mobile robots," *International Journal of Robotics Research*, vol. 20, no. 5, pp. 335–363, 2001.

[3] D. Fox, W. Burgard, H. Kruppa, and S. Thrun, "A probablistic approach to collaborative multi-robot localization," *Autonomous Robots: Special Issue on Heterogeneous Multi-Robot Systems*, vol. 8, no. 3, pp. 325–344, 2000.

[4] S. Roumeliotis and G. Bekey, "Collective localization: a distributed kalman filter approach," *IEEE Transactions on Robotics and Autonomation*, vol. 18, no. 5, pp. 781–795, 2002.

[5] A. Howard, M. Mataric, and G. Sukhatme, "Putting the 'i' in team: an ego-centric approach to cooperative localization," in *Proceeding of the IEEE Int. Conference on Robotics and Automation, to appear*, Taipei, Taiwan, May 2003.

[6] L. Doherty, K. Pister, and L. El Ghaoui, "Convex position estimation in wireless sensor networks," in *Proceedings of IEEE Infocom*, April 2001.

[7] A. Das, J. Spletzer, V. Kumar, and C. J. Taylor, "A distributed multi-robot system for cooperative manipulation," in *Multi-Robot Systems: From Swarms to Intelligent Automata*, 2002.

[8] J. Ponce, S. Sullivan, A. Sudsang, J. Boissonnat, and J. Merlet, "On computing four-finger equilibrium and force-closure of polyhedral objects," *International Journal of Robotics Research*, vol. 16, no. 1, pp. 11–35, 1997.