Multi-Robot Collaboration for Robust Exploration

Ioannis Rekleitis [a] Gregory Dudek [a] Evangelos Milios [b]

[a] *Centre for Intelligent Machines, McGill University, 3480 University St., Montreal, Québec, Canada H3A 2A7*

E-mail: yiannis,dudek@cim.mcgill.ca

[b] *Faculty of Computer Science, Dalhousie University, 6050 University Avenue, Halifax, Nova Scotia, Canada, B3H 1W5*

E-mail: eem@cs.dal.ca

This paper presents a new sensing modality for multirobot exploration. The approach is based on using a pair of robots that observe each other, and act in concert to reduce odometry errors. We assume the robots can both directly sense nearby obstacles and see each other. The proposed approach improves the quality of the map by reducing the inaccuracies that occur over time from dead reckoning errors. Furthermore, by exploiting the ability of the robots to see each other, we can detect opaque obstacles in the environment independently of their surface reflectance properties. Two different algorithms, based on the size of the environment, are introduced, with a complexity analysis, and experimental results in simulation and with real robots.

Keywords: Exploration, Mapping, Multiple Robots, Cooperative Localization.

1. Introduction

In this paper we discuss the benefits of cooperative localization during the exploration of a large environment. A new sensing strategy is used in order to improve the accuracy of the position estimation of each robot and hence the accuracy of the ensuing map. The robots explore the environment in teams of two; each robot is equipped with a *robot tracker* sensor that observes the other robot and reports its relative pose. The observing robot is using the position of its partner in order to update the estimate of its position. Our approach is sufficiently robust to be able to cope with environments that may have uneven or slippery terrains, or whose surface reflectance properties are not well suited to conventional sensors.

Conventional approaches to robotic mapping and navigation typically assume environments of rather limited size. Most existing approaches that function with real robots neglect issues like optimality and computational complexity. Further, the sensing techniques used to both explore the environment and position the robot often make rather optimistic assumptions about the environment such as: diffuse visual reflectors, substantial reflectivity. In practice, surfaces may either be specular (mirror-like) reflectors or be hard to detect due to low reflectance. Furthermore real terrains may have frictional properties that make large-scale odometry unreliable.

We deal with these issues in two ways, based on a polygonal approximation to the environment and the detection of convex (reflex) vertices. First, the two robots always move in such a way that they can see each other. More precisely, while one robot stays still the other robot moves, hence mapping the area swept by the line connecting the two robots as an area of free space. If an obstacle is located between the two robots, they can not see each other, and thus detect the obstacle. Second, the use of the other robot as an intelligent landmark decouples the odometry error from the environment. In other words, the moving robot localizes itself with respect to the stationary robot, thus improving its pose estimate independently of the floor conditions in the environment. The presence of reflex vertices is critical since it is these reflex vertices that determine the occlusion of regions of the environment. We use a pair of robots observing each other to build a map and circumvent problems of object visibility. The exploration strategy depends on the scale of the environment. When areas of free space are larger than the range of the robot tracker then a trapezoidal decomposition is used in order to guide the exploration. If the environment is small enough that it can be covered by the robot tracker then a triangulation of the environment is used.

In practice, a non-polygonal environment can always be described using a polygonal approximation where the number of segments depends on the fidelity of the approximation. Such an approximation can be readily computed so that it is either conservative in the sense that the interior of the approximated free space is assured to be free, or it can be designed to be accurate in a least-squared sense, so that for a given number of vertices in the approximation the discrepancy between the polygonal model and the actual environment is minimized [34,28].

The paper is structured as follows. In Section 2 we present an overview of previous work in mapping, localization and multi-robot applications. In Section

3 we present the fundamental ideas in our approach of cooperative localization. In Section 4 we present an outline of the exploration algorithm used for mapping environments with large areas of free space. Section 5 contains the outline of the algorithm used in environments where the two robots could stay in visual contact if an obstacle is not interfering. Section 6 covers the complexity analysis for the two algorithms, both the mechanical complexity and the computational complexity are examined. In Section 7 we examine experimental results from simulation and from laboratory experiments with real robots. Section 8 deals with extensions of the previous algorithms to more than two robots. Finally, Section 9 presents our conclusions.

2. Background

2.1. Mapping

Mapping via exploration is a fundamental problem in mobile robotics. The different approaches to mapping could be roughly divided into two categories: theoretical approaches that assume idealized robots and environments without uncertainty, and practical approaches that contend with issues of a real environment. The theoretical approaches provide lower bounds for the exploration problem while the practical approaches produce algorithms that operate in environments under uncertainty. Many algorithms have been proposed that explore the interior of a polygon or a collection of polygons, under the assumption of perfect sensing and dead reckoning: the resulting map consists of a collection of linked lines [47,46,56,51,13,14,36]. Another approach is to construct a graph like map that would encode the topological structure of the explored environment [23,19,20,17,38,53]. Real world applications have also been proposed that take into account the uncertainty of the sensors. The first approaches centered on the exploration of an unknown world using a single sensor such as vision, sonar or a laser range finder [11,27,5,67,41]. Subsequently data from different sensors were fused into a map in order to improve the efficiency and the accuracy of the map [21,1,66,12]. Thrun et al. proposed a novel approach combining an occupancy grid with a topological map in order to construct a reliable map for a mobile robot exploring an office like environment. Thrun's algorithm is based computationally on a Partially Observable Markov Model [64,63].

2.2. Estimation Theory

During the exploration of the unknown environment, the robots maintain a set of hypotheses with regard to their position and the position of the different objects around them. The input for updating these beliefs comes from the various sensors the robots posses. An " optimal estimator" [31] needs to be employed in order for the mobile robots to update their beliefs as accurately as possible. More precisely, the position of an obstacle observed in the past should be updated every time more data become available (a process called smoothing). Moreover, after an action the estimate of the pose of the robot should be updated, based on the data collected up to that point in time (a process called filtering).

Kalman filtering [31,10] is a standard approach for reducing the error, in a least squares sense, in measurements from different sources. In particular, in mobile robotics, Smith, Self and Cheeseman provided a framework for estimating the statistical properties of the error in robot positioning given different sets of sensor data [60,61]. A variation is based on Extended Kalman filtering (EKF), where a nonlinear model of the motion and measurement equations is used [42, 16]. A least squares fit provides a faster alternative to EKF but with less precise modeling of the kinematics and sensing [7,35]. Kurazume proposed the use of multiple robots, equipped with a sophisticated laser range finder, in order to localize, using some of them as movable landmarks [40,39]. The team of mobile robots was implementing a swarm behavior, using each other for localization. The fact that two robots could see each other was not used to infer that the space between them was empty.

2.3. Dead Reckoning

Dead reckoning is the procedure of modeling the pose (position and heading) of a robot by updating an ongoing pose estimate through some internal measures of velocity acceleration and time [9,22]. In most mobile robots this is achieved with the use of optical encoders on the wheels and is called odometric estimation. The estimate of the pose of the robot is usually corrupted with errors resulting from conditions such as: unequal wheel diameters, misalignment of wheels, finite encoder resolution (both space and time), wheel-slippage, travel over uneven surfaces [9]. The process of correcting the pose estimate is referred to as *localization*.

2.4. Localization

There are two major approaches to localization of a mobile robot. For both approaches a variety of sensing methodologies can be used including computational vision, sonar or laser range finding [22].The first approach is to use landmarks in the environment in order to localize frequently and thus reduce the odometry error [9]. A common technique is to select a collection of landmarks in known positions and inform the robot beforehand [30,43,32]. Another technique is to let the robot select its own landmarks according to a set of criteria that optimize its ability to localize, and then use those landmarks to correct its position [6]. The second approach to localization is to perform a matching of the sensor data collected at the current location to an existing model of the environment. Sonar, and laser range finder data have been matched to geometrical models [42,44,45,68,48,50], and images have been matched to higher order configuration space models [3,26] in order to extract the position of the robot. Borenstein suggested a two-part robot that would measure more accurately its position by moving one part at a time [8]. Also, Markov models have been used in order to describe the state of the robots during navigation [37].

The existence of clearly identifiable landmarks is an optimistic assumption for an unknown environment. Even in man-made environments, the cost of maintaining labels in prearranged positions may be prohibitive. Moreover, in large-scale explorations the robot may have to travel a large distance (larger than its sensor range) before being able to locate a distinct landmark.

2.5. Multiple robots

As mentioned earlier the multi-robot approach has both advantages and disadvantages over a single robot approach. Motion planning [62,2,69,24,33] and performing simple tasks such as box pushing and parcel delivery [49,18,65,29] with multiple robots have been studied extensively. In general, most prior methods assume complete information or neglect mapping.

Exploration using multiple robots is characterized by techniques that avoid tightly coordinated behavior [4,55,15]. In earlier work multiple robots used each other to localize when the luck of land marks made it otherwise impossible [25]. No previous work has considered the use of localization among the group members using each robot's neighbors to correct the pose estimate during mapping in order to remove uncertainty from the resulting map.

3. Cooperative Localization

Since sensing is being used to correct pose estimation errors, the major source of error in the localization of the robots is the inaccuracy of the "robot tracker" sensor that is used to update/correct the position of the moving robot relative to the position of the stationary one. Therefore, if the two robots start with one stationary robot in an initial position P_{origin} then the moving robot could localize itself with respect to that position, (see figure 2a). Note that, in practice, information from both sensing and odometry is combined using a probabilistic framework.

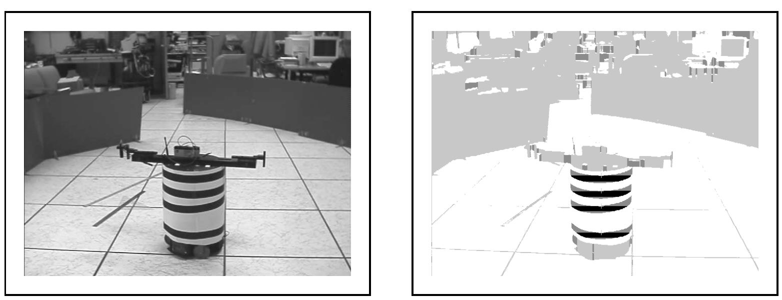
There are three potential sources of information for the localization of the moving robot. First, the odometry measurements $\hat{X}_{odom}(t)$ provide a base estimate of the moving robot's position (with high uncertainty σ_o). Second, the different objects in the environment, when sensed from different positions, could provide updates in the robot position [48,64]. Finally, the robot tracker provides pose measurements $\hat{X}_{track}(t)$ relative to the position of the stationary robot $\hat{X}_{stat}(t)$. In practice, over large scale environments, the position of moving objects changes over time and they cannot provide safe position updates. On the other hand the estimate of the robot tracker is influenced by the uncertainty in the position of the stationary robot σ_s plus the error of the tracker measurement $\hat{X}_{track}(t)$. The accumulation of uncertainty in the position of the stationary robot depends only on the number of role exchanges the two robots had. Consequently, over large open spaces where the odometry error grows without bound the moving robot could always reference back to a stationary landmark (a role that can be played by the second robot). In general the optimal way to (linearly) combine two information sources is to weight them as a function of their standard deviations. This is, in fact, the essence of the Kalman filter (an optimal estimator under appropriate conditions). Thus we compute a net pose estimate as in Equation 1.

$$\hat{X}(t) = \frac{\sigma_s(\hat{X}_{track}(t) + \hat{X}_{stat}(t))}{\sigma_s + \sigma_o} + \frac{\sigma_o \hat{X}_{odom}(t)}{\sigma_s + \sigma_o} \tag{1}$$

3.1. Tracker implementation

A variety of sensing technologies could be used for the robot tracker. Our implementation of the robot tracker is based on visual observation of a geometric target on the robot [24]. Alternative implementations use retroreflectors or laser light striping. Our actual robots are also equipped with such alternative technologies.

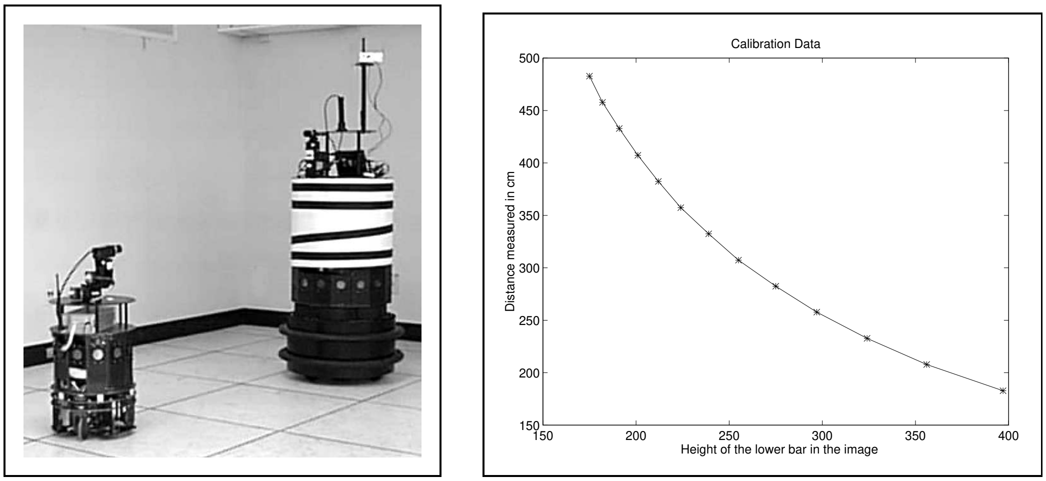
Each robot is equipped with a camera that allows it to observe its partner. The robots are both marked with a special pattern for pose estimation. The first part of the pattern is a series of horizontal circles (in fact these are cylinders and they project into almost linear pattern in the image). This allows the robot to be easily discriminated from background objects: the ratio of spacing between the circles is extremely unlikely to occur in the background by chance. Thus, the presence of the robot is established by a set of lines (curves) with the appropriate length-to-width ratio, and the appropriate inter-line ratios. The second component of the pattern is a helix that wraps once around the robot. The elevation of the center of the helix allows the relative orientation of the robot to be inferred (see Figure 2a, 1). In practice, this allows the robot's pose to be inferred with an accuracy of a few centimeters in position and 3 to 5 degrees in heading.



(a) (b)

Figure 1. Robot Tracker: (a) The raw image of the moving robot as observed by the robot tracker. (b) The helical and cylindrical pattern detected in the image.

By mounting the observing camera above (or below) the striped pattern of the other robot, the distance from one robot to the other can be inferred from the height of the stripe pattern in the image, due to perspective projection (scaling of the pattern could also be used). The difference in height between the observing camera and the target can be selected to provide the desired tradeoff between range of operation and accuracy. One advantage of the helical target for orientation estimation is that it functions correctly even at very large distances (although with reduced accuracy, of course) [1].



(a) (b)

Figure 2. (a) The visual robot tracker system with the camera mounted on one robot and the helical target pattern mounted on the second robot. (b) Calibration data for the distance estimation relating observed image position to actual distances.

3.2. Experimental results for the Robot Tracker

The accuracy of the visual tracker is shown in Table 1. While the relationship between the appearance of the target and the actual distance can be computed analytically, this would presuppose an accurate knowledge of the camera parameters. In order to relax this requirement, as well as to accommodate potential deviations from our ideal camera model, we use experimental calibration data to relate observed target positions with actual ranges. Calibration data relating the projected image and the distance estimates is shown in Figure 2b. It is possible to estimate distances between roughly 180 and 450 cm with the camera configuration used in this experiment, although accuracy degrades with increasing distance (due to decreased image resolution).

[1] Note that constraints due to specific task (such as mine sweeping) can sometime introduce additional constraints on the maximum inter-robot separation or optimal sensor geometry.

Distance Accuracy	1.5cm/pixel
Orientation Accuracy	1.3°

Table 1
Accuracy of simple visual tracker

3.3. Exploration with the Robot Tracker

Our mapping strategy exploits standard sensing technology in a new way. The two robots maintain an uninterrupted line of visual contact between them. When the moving robot proceeds along a trajectory, the line of visual contact sweeps a wedge defined by the lines connecting the stationary robot position to the initial and final positions of the moving robot (see Figure 3) and the trajectory of the moving robot. If an obstacle obstructs the line of visual contact the moving robot backtracks and then proceeds to map around the interfering obstacle. This permits the robots to measure objects with reflectance properties that would be unmanageable with traditional sensors (like laser range finders).

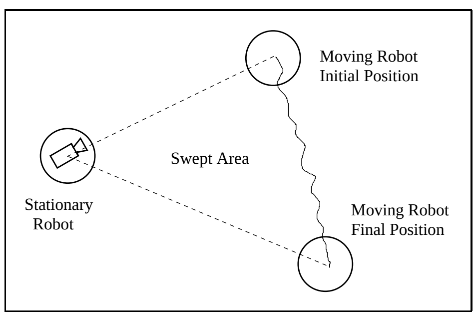


Figure 3. Area covered when one robot moves and the other one is stationary.

4. Exploration of large areas

In [57] we introduced an algorithm for exploring an area of size much larger than the sensing range of the robots. In environments consisting of large areas of open space (eg. warehouses, docking areas, open fields) it is quite common for the robots to be unable to follow a wall or to detect any landmarks. In such environments the moving robot is using the stationary robot as a portable marker for relocalizing and mapping. Different motion strategies are examined for the complete mapping of the environment. The core idea of the algorithms is the mapping of an area of free space by one moving robot while the other robot is stationary. The purpose of the algorithm described below is to provide the order, in which the free areas are going to be explored, without duplication and ensuring full coverage of free space. In a bottom up description of the algorithm there are the following operations. One robot moves and sweeps the line of visual contact across the free space, thus mapping a single region of free space. Then the two robots exchange roles in order to explore a chain of free-space areas which forms a stripe; a series of stripes are connected together to form a trapezoid. Finally, the entire collection of the trapezoids provides the trapezoidal decomposition of the entire free space – a complete spatial decomposition of the interior of the environment.

4.1. Outline of the trapezoidation algorithm

The proposed algorithm is based on the trapezoid spatial decomposition of a polygon [52,54]. A top down description of this algorithm is illustrated in Figure 4a-d. More specifically, the two robots explore the world using a trapezoid decomposition of the free space as their guide, as can be seen in Figure 4a. Each trapezoid is mapped completely before the two robots proceed to the next one. The order in which the trapezoids are mapped is given by a depth-first traversal of the embedded graph (see Figure 4b). Every trapezoid corresponds to a vertex in the graph; vertices corresponding to adjacent trapezoids are connected with an edge in the graph. The sensing range of the robot tracker provides a limit on the space that can be explored at any single time. Consequently if a trapezoid is larger than the range of the robot tracker then it is broken down into stripes with a width that depends on the sensing range **R** (see Figure 4c).

The exploration of a single stripe can be accomplished using different motion strategies. At the top of Figure 4d, two different motion strategies are displayed. One obvious approach (Strategy **A**) is, in each exchange, for one of the robots to move on a straight line (dotted line in figure 4d) sweeping (and hence mapping) a triangular region. The optimal strategy (Strategy **B**) is, in each exchange, for one of the robots to traverse the two chords shown as dashed lines in figure 4d, sweeping a diamond shaped area. The moving robot travels across the first chord and at an angle $\theta/2$ changes heading and follows the second chord (see Figure 4d).

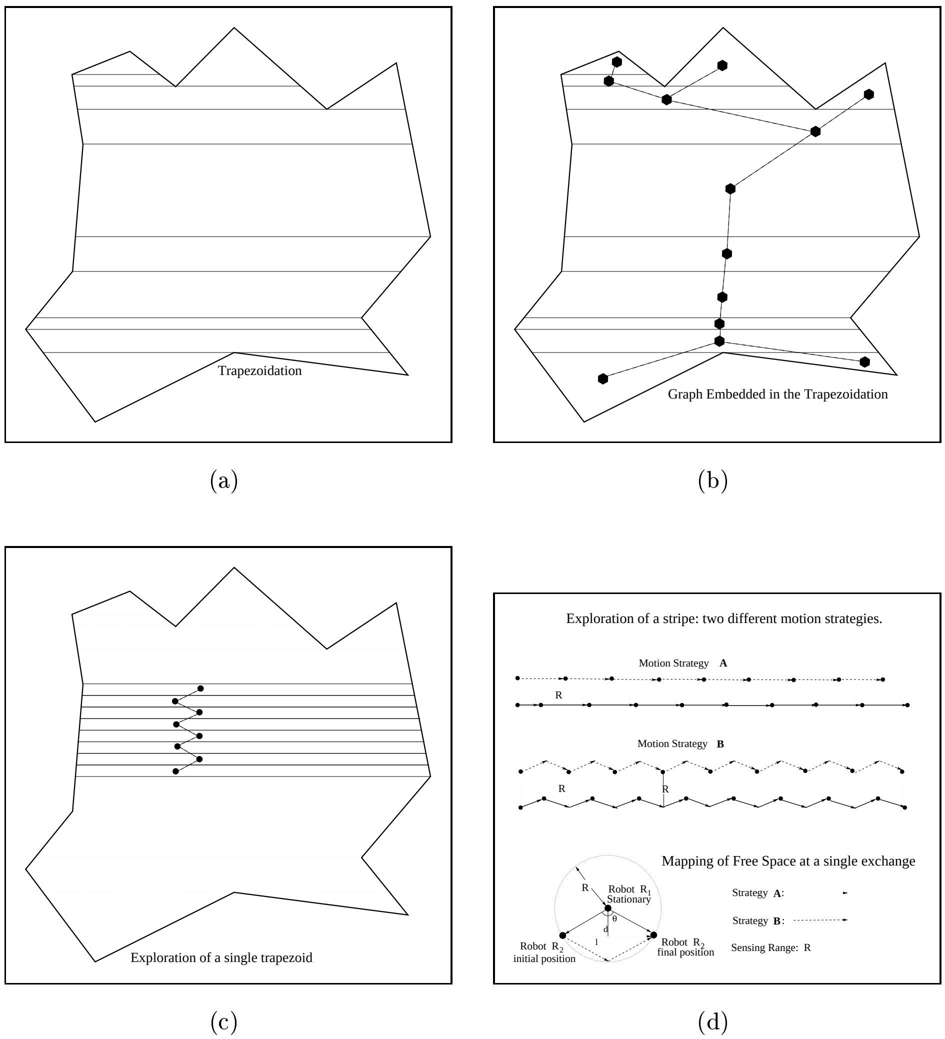


Figure 4. A top down description of the Trapezoidation algorithm. (a) The environment is divided into trapezoids. (b) The order in which the trapezoids are mapped is given by a traversal of the *Dual Graph*. (c) Each trapezoid is further divided into stripes with a width proportional to the sensing range R. (d) Each stripe is covered by areas of free space one next to the other. Each area of free space is explored by the motion of a single robot. Different motion strategies can be used, and the size of the area is controlled by the angle θ.

Strategy **A** is simpler, and requires a smaller number of changes in direction, but unfortunately, the width of the stripe (d) produced is suboptimal ($d < R$), and thus a larger number of stripes is needed in order to cover the same area.

Strategy **B** is optimal in terms of path traveled over area covered (see Appendix A) because at any single time the width of the stripe covered (d) is the maximum possible ($d = R$). At the bottom of Figure 4d, the mapping of free space is presented over a single exchange. Angle θ is an input parameter, that can be chosen to minimize a cost as a function of θ. In the case of reflex corners one trapezoid splits into two new trapezoids, and the two agents decide which branch of the embedded graph to follow.

When a series of explored regions are linked to each other as the exploration progresses, different types of stripes are created. In the case of the coverage of a triangular area, the two robots travel in parallel lines separated by d, and the stripe mapped has the same width d. In the case where each robot covers a diamond area, the trajectory of each robot would be a zig-zag line creating a stripe with width R (equal to the sensing range of the robot tracker). A series of stripes connected together (lengthwise) map a single trapezoid. At the end of each stripe the two robots follow the walls and reposition themselves to explore the next stripe.

5. Exploration of small areas

In environments where the two robots can maintain visual contact and effectively track each other across any open space a different strategy is employed. In [58] we presented an algorithm for mapping the interior of such an environment. The size of the area should be small enough to be covered by the range of the tracker sensor. Both robots use a traditional range finder in order to detect and circumnavigate obstacles during the exploration. In addition, each robot has a robot tracker sensor that is used to detect interfering obstacles when the line of visual contact is broken. The exploration strategy is based on a triangulation of the free space.

5.1. *Outline of the triangulation algorithm*

The exploration algorithm is based on the following idea. At any single time one robot is positioned at a vertex (corner) of the environment operating as an intelligent landmark, while the other robot moves along the perimeter of the environment maintaining visual contact with the stationary robot (see Fig. 5). More precisely, as the moving robot follows one wall of the environment, it “sweeps” the line of visual contact across the triangle defined by the corner where the stationary robot is positioned and the two ends of the wall. Thus, the robot establishes the position of the wall and the occupancy status of the swept free space inside the triangle. The two robots progressively map the environment by dividing it into triangles of free space, thus constructing a triangulation of the environment. Both robots run the same exploration algorithm, taking turns between (a) moving, thus mapping the free space, and (b) being stationary, thus providing a fixed localization reference for the moving robot [2]. First, a few definitions are presented, then, the major operations of the algorithm are discussed and finally, an outline of the exploration algorithm is presented.

Interior of the Polygon The free space of the environment where the robots explore.

Polygon Vertex The corner where two walls meet.

Reflex Vertex A *Polygon Vertex* with its internal angle (the angle in the interior of the polygon) strictly greater than 180 degrees.

Internal Diagonal A line segment connecting two polygon vertices completely contained in the interior of the polygon.

Map A set of triangles residing exclusively in the polygon, which cover completely the interior of the environment (polygon) without overlaps (*Triangulation*).

Unfinished Triangle is a triangle that is not completely mapped, in other words, one of the wall sides is not fully explored.

Internal Triangle is a triangle constructed by three internal diagonals of the polygon. The corresponding *Dual Graph Vertex* has degree three.

Dual Graph A graph $(\mathcal{V}, \mathcal{E})$ such that every vertex $v_i \in \mathcal{V}$ corresponds to a triangle T_i, and an edge $e_{ij} \in \mathcal{E}$ between two vertices v_i, v_j exists iff their triangles T_i, T_j share an internal diagonal.

Open Edge is an edge in the dual graph that connects a mapped triangle with an *Unfinished Triangle*.

Degree of a triangle is the degree of the corresponding vertex in the dual graph, and is equal to the number of triangle sides that are internal diagonals of the polygon.

[2] In the following we assume no three vertices are collinear. If not, it would involve a minor but tedious change to the algorithm.

As mentioned earlier the basic operation is the mapping of a triangle of free space as the moving robot travels along one (or two) sides of each triangle. After the triangle is mapped it is included in the map and the corresponding node is added to the *dual graph*. *Internal diagonals* that separate fully mapped triangles from unexplored (or partially explored) areas (unfinished triangles) correspond to *open edges* in the *dual graph*; these *open edges* guide the next step of the exploration (which triangle to map next). The exploration continues as long as there are areas of free space to be mapped and the line of visual contact between the robots is uninterrupted. Adjacent mapped triangles form a chain of nodes in the *dual graph* which ends either in an *internal triangle* (where there is a bifurcation), or, at a triangle with degree one (where two of its sides are walls, as in the top left triangle in Fig. 5f). One instance where the exploration halts is when a triangle of degree one is fully mapped. In that case, the two robots search the *dual graph* and select the closest *open edge*, then they travel to the two ends of the corresponding diagonal and resume the exploration.

The second instance where the exploration stops is when the line of visual contact is interrupted. There are four distinct cases where the line of visual contact is interrupted (see Figs. 5a,b,c,e). In these cases the moving robot cannot continue its previous course and it has to make a decision where to move next in order to maintain visual contact with the stationary robot.

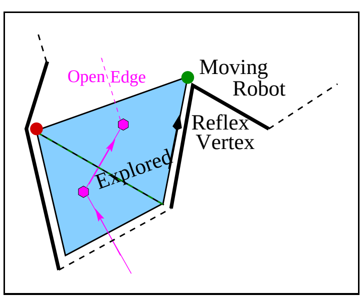
Case 1: The stationary robot is located at a non reflex vertex while the moving robot reaches a reflex vertex. If the moving robot continues to follow the next wall then the line of visual contact is interrupted (see Fig. 5a). In this case the two robots simply switch roles, the moving robot sends a signal to the stationary robot to start exploring and then becomes stationary and the stationary robot (which was waiting, see Algorithm 3) continues the exploration.

Case 2: During the mapping of a triangle a reflex vertex located between the two robots interrupts the line of visual contact (see Fig. 5b). First, the partially mapped triangle is stored as an unfinished triangle. Then, the moving robot travels towards the stationary robot until the reflex vertex is encountered and mapped. Consequently, an *internal triangle* is constructed defined by the reflex vertex and the first *internal diagonal* of the unfinished triangle (see Fig. 5b, Algorithm 2). The *internal triangle* is connected with three triangles the previous fully mapped triangle, and two *unfinished triangles* located at the two sides of the reflex vertex (see Fig. 5b).

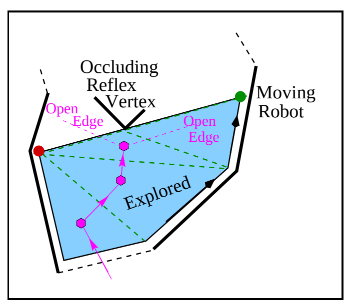
Case 3: Both robots are located next to reflex corners and any motion along the unexplored walls would result in a break of the line of visual contact (see Fig. 5c). In this case, the moving robot explores in a direction perpendicular to the *internal diagonal* between the two reflex vertices until a wall and then a vertex is encountered (see Fig. 5d). An *internal triangle* is created from the newly found vertex and the two reflex vertices, and two *unfinished triangles* are attached to it on the two sides of the newly mapped vertex (see Algorithm 2). Then the exploration continues in the *unfinished triangle* that is closer to the robots.

Case 3: During the exploration an occluding edge interrupts the line of visual contact (see Fig. 5e). This is a subcase of the *occluding reflex vertex* case where the stationary robot is placed next to the edge adjacent to the occluding vertex. It is treated differently in order to eliminate redundant traveling. The two robots exchange roles and the previously stationary robot receives a command to explore only up to the *occluding reflex vertex* and add the triangle to the map. Then the two robots exchange roles again and continue the exploration.

These four cases cover all possible configurations of interruptions in the line of visual contact. An outline of the algorithm is presented in Algorithm 1.



(a)



(b)

Figure 5. Thick line represent walls, dashed lines represent unexplored walls, grey area is explored free space, dashed lines inside the grey are are internal diagonals. Line of Visual Contact interrupted: (a) **Case 1:** The stationary robot is at a non-reflex vertex and the moving robot encounters a reflex vertex that would interrupt the line of visual contact (b) **Case 2:** Occluding Vertex between the two robots.

```
while Dual Graph contains Open Edges do
  while (No Occlusion) AND (Dual Graph has Open Edges) do
    Assign closest Open Edge to Unfinished Triangle
    Explore Unfinished triangle
  end while{Occlusion has occurred Or branch of Dual Graph is completed}
  if Occlusion is of Case 1 then {One Robot at reflex vertex, Fig. 5a}
    SIGNAL(OtherRobot,Continue){ Exchange role with the other Robot.}
    WAIT {see Algorithm 3}
  else if Occlusion is of Case 2 then {Occluding Vertex between the two
  Robots, Fig. 5b}
    Mark current position as a temporary Polygon Vertex
    repeat
      Go Towards the Stationary Robot
    until Occluding Polygon Vertex Encountered
    Map the Occluding Polygon Vertex
    CreateInternalTriangle() {see Algorithm 2}
  else if Occlusion is of Case 3 then {Both Robots at Reflex Vertices, Fig.
  5c,d}
    while New Polygon Vertex Not Found do
      Explore perpendicular to the line of visual contact
    end while
    Map New Polygon Vertex
    CreateInternalTriangle() {see Algorithm 2}
  else if Occlusion is of Case 4 then {Occluding Edge, Fig. 5e}
    SIGNAL(OtherRobot,ExploreOccludingEdge)
    WAIT {see Algorithm 3}
  end if
  if Current branch of Dual Graph ends then {see Fig. 5f}
    Traverse the Dual Graph towards the closest Open Edge
  end if
end while{The Map is complete}
```

Algorithm 1: Triangulation Algorithm; functions are noted as underlined text, comments are inside curly brackets "{}".

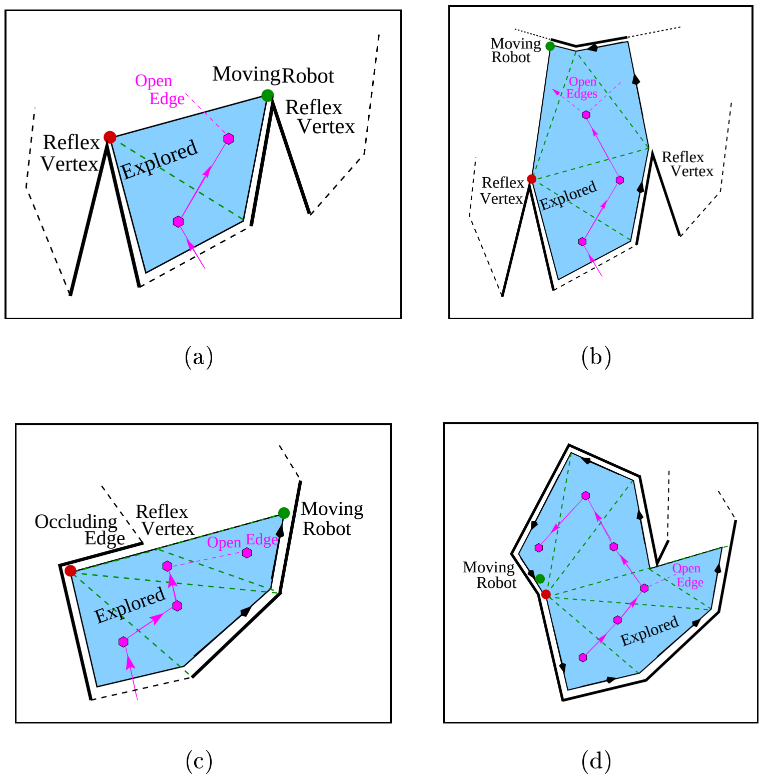


Figure 6. Same notation as in Fig. 5. Line of Visual Contact interrupted: (a) **Case 3:** Both robots are placed at reflex vertex such that any further exploration would break the line of visual contact (b) The moving robot explores and creates an internal triangle. (c) **Case 4:** Occluding Edge next to the stationary robot.(d) One branch of the dual graph is completely mapped, the robots would proceed to the nearest open edge of the dual graph.

```
function CreateInternalTriangle()
Create an Internal Triangle with two Open Edges
Add node to the Dual Graph
Connect the Unfinished Triangle to the Internal Triangle
     via the first Open Edge
Continue the exploration following the second Open Edge
```

Algorithm 2: Create Internal Triangle

6. Complexity Analysis

In order to analyze the complexity of the exploration we need to distinguish between two qualitatively different stages of exploration, the *local* and the *global*

```
function Wait()
repeat
   Check Condition from Signal
until Condition is set
if Condition=ExploreOccludingEdge then
   Map the Occluding Edge up to the next corner
   Create Triangle containing the occluding edge
   Add Triangle to the Map.
   SIGNAL(OtherRobot,Continue)
   WAIT()
end if
```

Algorithm 3: Wait

exploration phases. The local exploration strategy guides the path traveled for the mapping of a convex area of free space (a triangle, or a trapezoid). The global exploration strategy provides the order in which these areas are explored.

6.1. Complexity of Global Exploration

As noted earlier, the exploration strategy is guided by the dual graph of the spatial decomposition used. More specifically, during the trapezoidation algorithm the two robots explore one trapezoid at a time and then proceed to map the next trapezoid by following the dual graph in a depth first traversal. Every trapezoid is "*traversed*" twice, a first time when is being mapped and a second time when the two robots pass through in a shortest path traversal to visit the rest of the graph. When the triangulation algorithm is used, the dual graph is attached to the triangles. The two robots visit every triangle in a depth first traversal, thus passing through at most twice (the first time exploring, the second moving through towards the unmapped parts of the environment). In general the complexity is proportional to the number of edges of the polygon, also the size of the environment and the range of the tracking sensor.

6.2. Complexity of the exploration over a single exchange

During the exploration of a triangle (triangulation algorithm) the stationary robot is located at a corner of the environment while the moving robot is constrained to move along the wall it is mapping. If it moves in a different trajectory then some areas would be unmapped. When the moving robot has completely

Covering	Triangle Area	Diamond Area
Path length P_θ	$2Y + \frac{XY}{R}\frac{2}{\cos\frac{\theta}{2}}$	$2Y + \frac{XY}{R}\frac{2}{\cos\frac{\theta}{4}}$
# of exchanges E_θ	$\frac{XY}{R^2}\frac{2}{\sin\theta}$	$\frac{XY}{R^2}\frac{1}{\sin\frac{\theta}{2}}$
# of turns R_θ	$2\frac{Y}{R\cos\frac{\theta}{2}}$	$2\frac{Y}{R} + 2\frac{XY}{R^2\sin\frac{\theta}{2}}$

Table 2

Analytical complexity of two different path curves.

map one wall, the triangle defined by the location of the stationary robot and the two corners of the wall is added to the map of the environment.

In contrast, when the trapezoidation algorithm is used, the moving robot could move through different trajectories as long as it stays inside the sensing range of the stationary robots. Different motion strategies present certain advantages and disadvantages. More precisely, there are different factors that affect the cost of the exploration depending on the configuration of the different robots. Every time the two robots exchange roles, the moving robot uses the stationary one to correct its position and then the stationary one starts exploring. Each of these operations introduces a time delay, therefore the number of exchanges increases the cost. In addition, every time one of the robots has to change directions the rotation adds to the total cost. Finally, the total path traveled has to be taken into consideration. For the two different motion strategies (diamond area covered, and triangular area covered) examined earlier, the total mechanical effort can be computed as shown in Table 2. The cost is calculated for the exploration of a rectangle X by Y, when the robot tracker sensor range is R.

6.2.1. Cost Analysis

The factors that affect the cost of the exploration are: the number of exchanges, the total path traveled and the number of rotations. For a specific team of robots the cost of the above factors could be determined beforehand. Specifically, the total cost of the exploration could be computed as the weighted sum of:

the total path traveled (P_θ) multiplied by the cost of path traveled (C_p in sec/m), the total number of exchanges (E_θ) multiplied by the cost for an exchange (C_e in sec/exchange), and the total number of rotations (R_θ) multiplied by the cost of rotation (C_r in sec/rotation). The factors (C_p, C_e, C_r) could be determined before the exploration, while the sensing range (R) of the robot tracker is known. Equations 2 and 3 provide the total cost $C_{total}(\theta)$ as a function of angle θ for the exploration, using diamond area and triangular area covering respectively (Fig. 4d), of a rectangle $X \times Y$ as a function of θ, using the cost estimates and the analytical results from table 2. The optimal θ for the exploration is the one that minimizes $C_{total}(\theta)$.

$$\begin{aligned} C_{total,diamond}(\theta) &= C_p P_\theta + C_e E_\theta + C_r R_\theta = \\ &= C_p(2Y + \tfrac{2XY}{R\cos\frac{\theta}{4}}) + (\tfrac{C_e XY}{R^2 \sin\frac{\theta}{2}}) + C_r\theta(\tfrac{2Y}{R} + \tfrac{2XY}{R^2\sin\frac{\theta}{2}}) \end{aligned} \tag{2}$$

$$\begin{aligned} C_{total,triangle}(\theta) &= C_p P_\theta + C_e E_\theta + C_r R_\theta = \\ &= C_p(2Y + \tfrac{2XY}{R\cos\frac{\theta}{2}}) + C_e(\tfrac{2XY}{R^2 \sin\theta}) + C_r\theta(\tfrac{2Y}{R\cos\frac{\theta}{2}}) \end{aligned} \tag{3}$$

For one of our robots, a Nomad 200, the cost factors, for a typical experimental arrangement are: $C_p = 4.1$ sec/m, $C_e = 7$ sec/exchange, $C_r = 4.65$ sec/rad. The optimal angle θ is $180°$, for the diamond area motion strategy. For the same costs the optimal angle θ is $90°$ for the triangular area motion strategy. As expected the total cost is lower for the motion strategy that covers the diamond area than that which covers a single triangular area.

7. Experimental Results

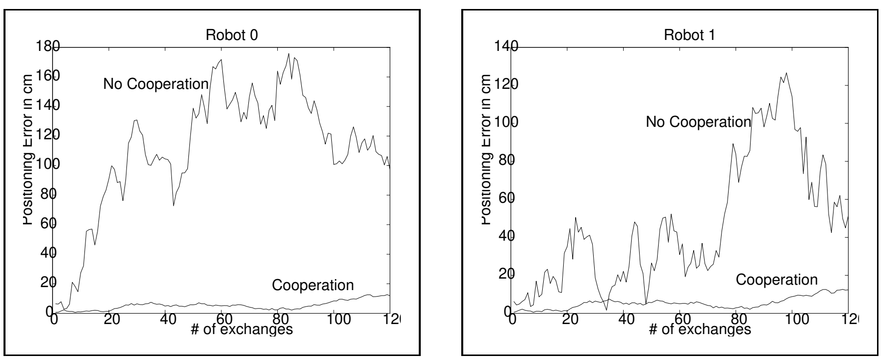
Experiments were conducted in simulation, using the robotic simulation package *RoboDaemon* [3] and in the lab. Experiments in the lab were used in order to validate the improvement in the localization based on the robot tracker over a pure odometry approach. In simulation a variety of odometry error models were applied in order to simulate different surfaces as well as different model worlds were explored.

[3] RoboDaemon is robot control software employed at McGill University. It allows us to control the robots in the lab and also to perform experiments with simulated robots. The main advantage is that the simulated robots can be replaced by the real ones with no overhead.

7.1. Trapezoidation Algorithm

7.1.1. Simulation

In order to estimate the improvement in position estimation when the two robots collaborate a series of experiments were performed. The two robots explored a single stripe, exchanging roles 120 times. Odometry error estimates gathered during experiments with an RWI robot were used to model the error in dead reckoning. The accuracy of the helix vision tracker was used to model the accuracy and the range of the robot tracker. The same path was traveled twice, and the error in positioning was measured. The first time no cooperation took place between the two robots, while the second time every time the two robots exchanged roles they corrected their position estimates. In the case of no cooperation the two robots are following the same trajectory as before but without correcting their position estimate.



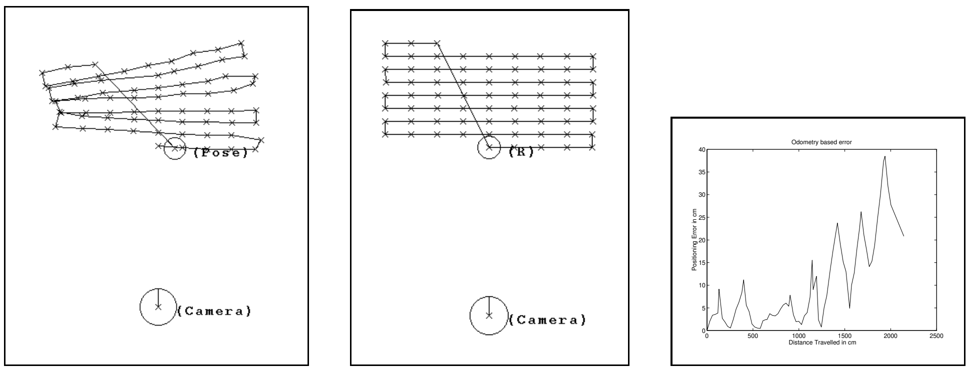
(a) (b)

Figure 7. Exploration of one stripe (120 exchanges). The results are from a single run.

From the results shown in figures 7(a,b), it is clear that the cooperative exploration strategy improves performance substantially over the non-cooperative strategy. It is worth pointing out that in the previous experiment no systematic error was included in the model, such as would occur on an inclined floor where with every translation a small amount of slippage would occur. It is clear that the cooperation of the two robots helps to maintain reduced localization error and improves mapping robustness.

7.1.2. Real Robots

A pattern similar to the one traveled by the two robots was traced by one robot while the other one was stationary.



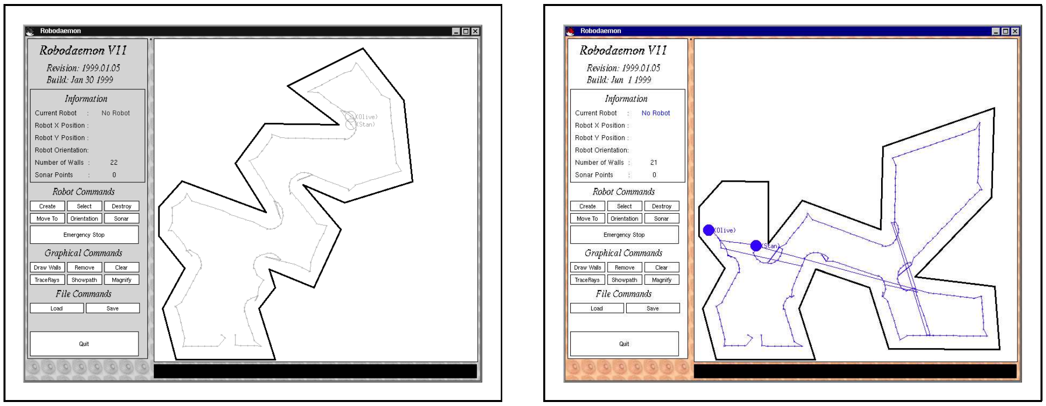
(a) (b) (c)

Figure 8. (a) Path of the moving robot as estimated by the visual tracker. Measurements taken in different position validate the accuracy with precision of roughly 2cm. (b) The desired path of the moving robot. Although the robot can driven along this path using open-loop control, dead reckoning error lead to a substantial discrepancy. (c) The error in positioning from the odometry estimations.

In order to measure the accuracy of the map, a few locations along the path were selected and the position of the robot was estimated relative to the stationary camera. The accuracy of the positions estimated by the camera-based tracker was between 1.0 and 2.3 cm. As can be seen in Figure 8a,b the inaccuracy is largely due to rotational error and thus it is more evident near the sides of the rectangle. Figure 8c presents the absolute odometry error as it accumulates over the distance traveled by the robot.

7.2. Triangulation Algorithm

Different sets of experiments have been conducted in order to validate our approach. Experiments in simulated environments (using the RoboDaemon package, see Figure 9) provided verification in a variety of model worlds. In addition, laboratory experiments with the real robots helped us estimate realistic values for the uncertainty of the sensors and the odometry.



(a) (b)

Figure 9. The paths of the two robots after the completion of the exploration.

7.2.1. Simulation

Extensive experiments have been contacted using the robotic simulation package RoboDaemon. The simulations allowed us to specify different parameters such as odometry error, robot-tracker uncertainty and the complexity of the explored environments. Figure 9 presents two typical environments used in the simulations (approximate area 144 m^2) and the path the two robots followed. Figures 10 and 11 present the exploration of two model environments; these examples illustrate different aspects of the triangulation algorithm. Figures 10 (a-i) present snapshots of the exploration as perceived by *Robot 0*, *Robot 1*, and the resulting map at different instances of the exploration. The two robots exchange roles when the line of visual contact breaks. In the first row an early phase of the exploration is presented. The two robots have exchanged roles twice and *Robot 0* explores five new triangles. Consequently, in the second row *Robot 1* is exploring again (Figure 10d) and then the two robots exchange roles and *Robot 0* explores three additional triangles. The third row illustrates the final stages of the exploration where *Robot 1* explores the final parts of the environment using *Robot 0* as a reference.

In Figure 10, in the last row, the early phase of the exploration is presented, using pure odometry for positioning. The dashed line depicts the real path of the robot and the solid line the odometry based paths. In small worlds and/or cluttered environments multiple observations of the same object could be used in order to correct the positioning of the moving robot. As can be seen in Figure 10(j,k), the accumulation of uncertainty causes the map to be distorted although local consistency is maintained. These distortions could lead over time to a map that is not even topologically sound.

The results from the exploration of a different environment are presented in Figure 11. The presence of reflex vertices that interrupt the line of visual contact introduce internal triangles and therefore branches in the dual graph making the exploration more complicated. The results are presented in three columns. The first column presents the trajectory of *Robot 0* and the environment as perceived by that robot. The second column presents the trajectory of *Robot 1* and its perception of the world. Finally, the third column presents the constructed map up to that point in the exploration. The light grey disk represents the position of the stationary robot and the black disk the location of the moving robot in the figures of the moving robot. In the first row, *Robot 1* is stationary (after mapping two triangle), while *Robot 0* is mapping the right branch of the first bifurcation. The line of visual contact was broken by a reflex vertex, thus, a internal triangle was built (node with degree 3), and two branches were started. Each branch consists of one open triangle with a gateway to unexplored space. In the second row, *Robot 0* is stationary (after adding two more nodes in the embedded graph, and *Robot 1* is mapping the second occluding vertex. Again an internal triangle is created (node with degree 3), and *Robot 1* is mapping the left branch of the bifurcation. In row three the environment is mapped for the area that corresponds to the branch being explored, with the last triangle having two walls and one internal diagonal (node with degree 1). Figure 11i presents the map up to that point where the last wall (not fully explored yet) is marked with a thiner line. Then the two robots proceed to the closest gate (following a depth first traversal of the embedded graph). Row four demonstrates the exploration of *Robot 0* of the final branch (right) of the second bifurcation, while *Robot 1* is stationary at the second occluding vertex. Finally, the fifth row illustrates the final step of the exploration. *Robot 0* is stationary at the first occluding vertex encountered, while *Robot 1* maps the final triangle. In Figure 11o the completed map is shown. The dual graph is presented in the figures of the third row superimposed on the metric map.

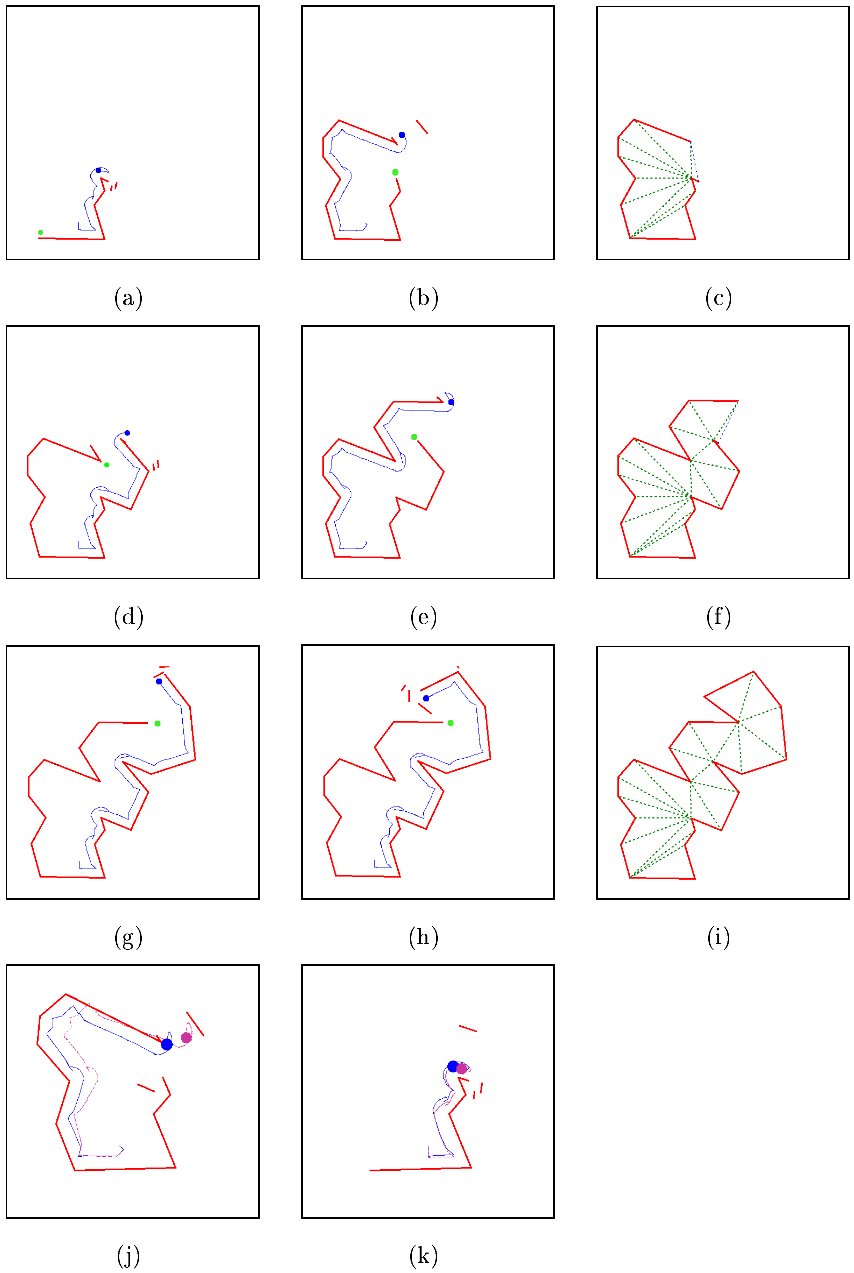
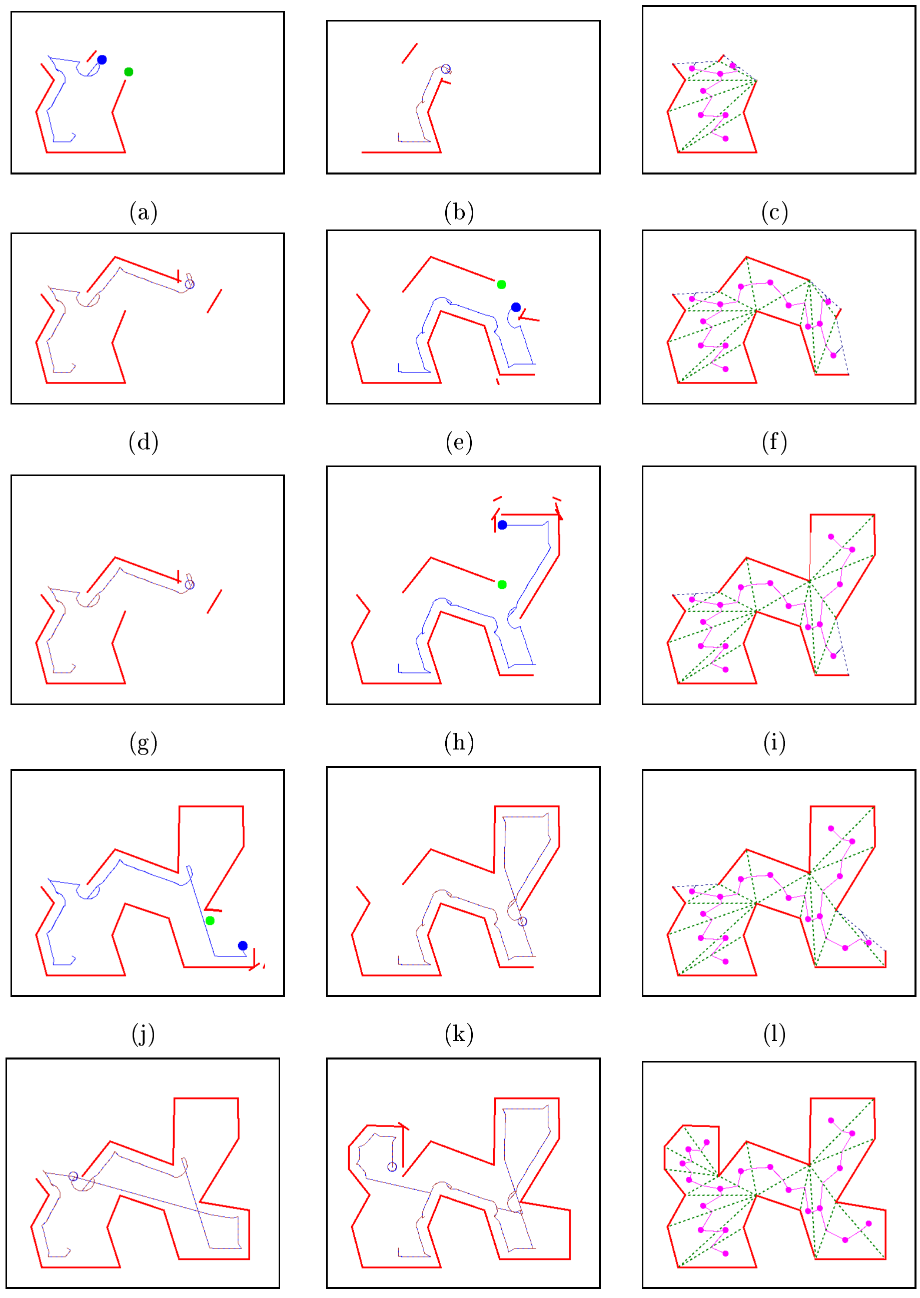


Figure 10. First three rows: Exploring an unknown environment, figures b and e illustrate the trajectory of Robot 0. Figures a,c,d,f,g,h illustrate the trajectory of Robot 1. Finally the third column (Figures c,f,i) presents the map up to that point. Last row: Close-up on the build up of the uncertainty when only odometry was used. The solid line is the odometry based estimation of the robots while the dashed line is the real position of the robots (see text Section 7.1.1).



(a) (b) (c)
(d) (e) (f)
(g) (h) (i)
(j) (k) (l)
(m) (n) (o)

Figure 11. **Exploring an unknown environment with two occluding vertices:** The first column illustrates the trajectory of **Robot 0**. (a,d,g,j,m). The second column illustrates the trajectory of **Robot 1** (b,e,h,k,n). Finally the third column presents the **map** up to that point (c,f,i,l,o). (See text Section 7.1.1).

7.2.2. Physical Validation

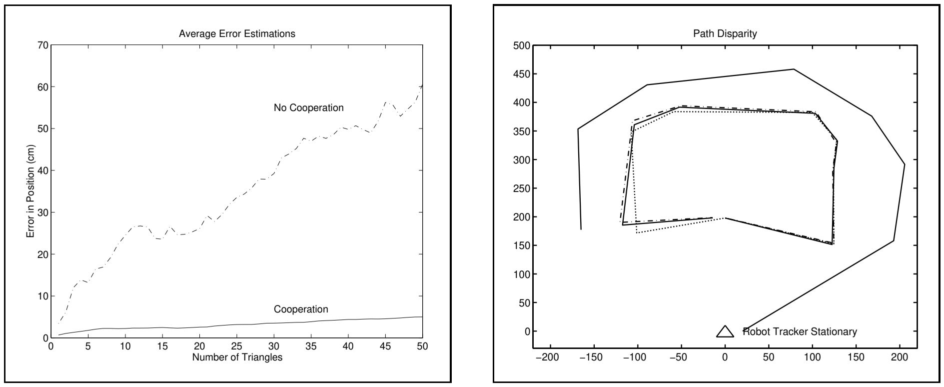
In order to demonstrate the effectiveness of the proposed approach with real robots, several preliminary exploration tests were carried out in our laboratory in workspaces of area roughly 16 m^2. This comparatively small testbed allowed us to control various factors such as inhomogeneities in the terrain as a function of trajectory and obtain ground truth data. Using this testbed we compared the time, accuracy, and robustness of different exploration strategies. In our experimental arrangement the role of the stationary robot is played by a tripod mounted camera at the same height as one of our robots. The camera was placed next to the first wall. This allowed us to more reliably and repeatably verify ground truth. It is worth noting that our strategy works equally well with homogeneous robots and with heterogeneous robots, eg. one robot has a camera the other robot has the pattern.

A laser pointer pointing straight down to the floor has been placed on top of the moving robot in order to accurately mark its current position on the floor. This setup allowed us to measure the displacement from the initial position after the completion of the tour.

Figure 12b shows the actual path of the moving robot, the odometry-based estimate of position, and the tracker-based estimate. The final displacement from pure odometry estimates is approximately 15cm with an orientation error of 15°. The tracker estimate has approximately 1.3cm error. This corroborates our assumption that joint exploration and localization using a "tracker" can lead to much more robust modeling than odometry alone.

8. More than two robots

The above strategies could be extended by the addition of more robots. By forming a chain of robots that "sweeps" through the free space the range of the tracker is multiplied by the number of robots, thus, covering a much larger area in a single sweep. In addition, every robot could refer to more than one stationary robots therefore, gaining higher precision in its measurements. Two motion strategies are proposed with respective advantages. Using the first motion strategy, during the exploration only one robot moves while the stationary ones that are still visible are used as landmarks. This method ensures minimum uncertainty build up as, at any given time, the moving robot would correct its odometry error with respect to more than one landmarks. Using the second motion strategy, the



(a) (b)

Figure 12. (a) The average error during the exploration of 50 triangles (over 100 experiments) without and with cooperative localization. (b)The path of the robot after the completion of the exploration. The outside solid line marks the position of the walls the moving robot followed. The actual path of the robot is the solid line, the odometry based estimate of position is the dotted line, while the tracker estimate is the dashed-dotted line.

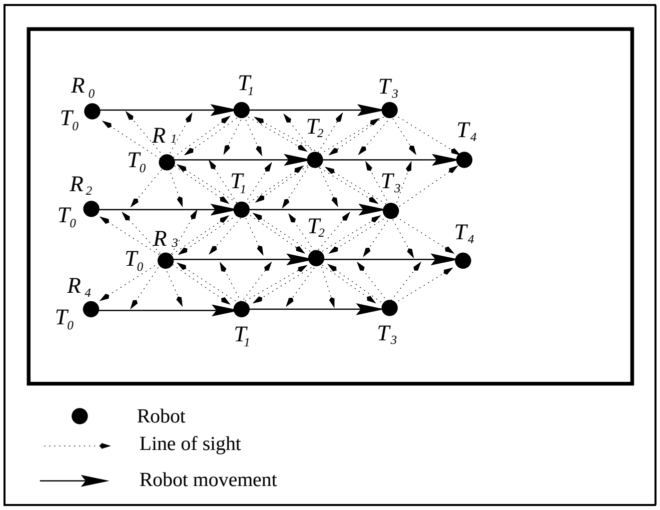
robots divide into two teams and they interchange roles: while the first team is moving the second team works as a set of landmarks. This method explores an environment in less time but less robots are available as landmarks.

8.1. Triangulation Algorithm

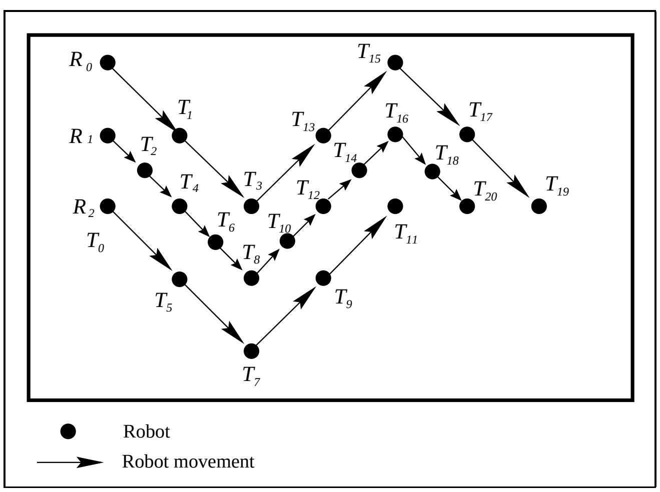
The triangulation algorithm can be extended by using more than two robots. The first robot is stationary at a corner, the N^{th} robot moves along a wall of the environment, while the other robots form a chain connecting the stationary to the moving one, thus extending the range of the robot tracker. The robots between the first and the last could move by using an odd-even strategy for minimum time, or one at a time for maximum accuracy and robustness.

8.2. Trapezoidation Algorithm

As mentioned earlier an immediate extension of the trapezoidation algorithm can be obtained by the addition of more robots. When the two robots sweep one stripe of width d then by adding an extra robot (50% increase) we could double the area swept. In the original algorithm, every robot has only one device to track the other robots; in this case a scheduling algorithm should be applied in the order the robots are moving. If we add a second tracking device, one robot could track robots on both sides, allowing a parallel cover of double the area at the same time.



(a)



(b)

Figure 13. (a) Exploration of a stripe with 5 robots. The robots move at time T_1,T_2,T_3, and T_4. (b) Exploration of a stripe with 3 robots, covering space in diamond areas. The robot move at time T_1-T_{19}.

In the example of Figure 13a we use five robots ($R_0 \dots R_4$) that are positioned in two lines at time T_0. First the robots R_0, R_2, R_4 move forward, tracked by R_1 and R_3 accordingly, mapping the four triangles as free space(time T_1), then both R_0, R_2 track R_1, which moves forward (time T_2); while R_2, R_4 track R_3, which moves forward (time T_2. Then it is time for the other column of robots (R_0, R_2, R_4) to advance marking more area as free space (time T_3). The tracking is marked with the dotted lines of sight. The same pattern is followed as the two columns alternatively advance, marking a stripe of free space much wider than that possible with only two robots.

The second part of the algorithm concerning the exploration strategy for the whole space and the order in which the trapezoids should be explored is identical to the previous algorithm where only two robots were used.[4]

Moving only one robot at a time can also be easily extended to multiple robots. The robots start exploration aligned with each other in a straight line, at distance R from each other, where R is the tracker sensor range. The first robot and the last robot in the line act out the algorithm for two robots, while the role of the other robots is simply to provide a communication path between them. As such, the first robot in the line remains stationary, and the rest of the robots are moving such that the distance between two robots is never more than R. The width of the explored stripe is $(n-1)R$, where n is the number of robots. A pictorial representation of this strategy can be seen in Figure 13b, the robots R_0 and R_2 sweep a stripe using the diamond pattern and the Robot R_1 stays between them.

9. Conclusions

In this paper, we have described an approach to exploring and navigating in *large scale spaces* where positioning and obstacle detection might be difficult using traditional methods. In fact, such difficulties are likely to arise in many real-world environments.

Our approach is based on exploiting a line-of-sight constraint between two robots to achieve exploration with reduced odometric error. This approach can also cope with obstacles with hard-to-sense reflectance characteristics. Different algorithms were proposed depending on the scale of the environment. Where the

[4] There is a possible speedup by splitting up the group in order to explore different parts in critical points, but that would in the end spread the robots too thin.

environment is small enough so that the robots can see each other from any two points on its boundary that have clear line of sight between them (i.e. they are never unable to see one another simply because they are too far away), then the triangulation algorithm is applied. If the environment is larger than the range of the robot tracker sensor then the trapezoidation algorithm is used. An open issue is how to automatically detect such situations *efficiently* during exploration and switch strategies, or switch back-and-forth between strategies based on local properties of the environment.

We are currently planning large-scale experiments of this strategy in a real physical environment. One difficulty that we must address is how to obtain accurate ground-truth to validate the performance of our approach over a large terrain. A standard practice is to simply observe the consistency and clarity of the resulting map and use this as a performance metric [64]. We are also considering combining this approach with more traditional localization methods (such as using landmarks [59]).

10. Acknowledgements

We would like to thank the Canadian Natural Science and Engineering Research Council for their funding. Also we would like to thank Marinos Rekleitis for his contributions in the optimality proof, and anonymous reviewers for important suggestions.

References

[1] Kurz Andreas. Constructing maps for mobile robot navigation based on ultrasonic range data. *IEEE Transactions on Systems, Man, & Cybernetics, Part B: Cybernetics.*, 26(2):233–242, Apr 1996.

[2] R. C. Arkin and T. Balch. Motor schema-based formation control for multiagent robot teams. In *International Conference on Multiagent Systems*, pages 10–16, San Francisco, CA, 1995.

[3] Ivan A. Bachelder and Allen M. Waxman. A view-based neurocomputational system for relational map-making and navigation in visual environments. *Robotics and Autonomous Systems*, 16:267–289, 1995.

[4] Tucker Balch and Ronald C. Arkin. Communication in reactive multiagent robotic systems. *Autonomous Robots*, 1(1):27–52, 1994.

[5] R. Bauer and W D. Rencken. Sonar feature based exploration. In *IEEE International Conference on Intelligent Robots and Systems.*, volume 1, pages 148–153. IEEE, 1995.

[6] M. Betke and L. Gurvits. Mobile robot localization using landmarks. *IEEE Trans. on Robotics and Automation*, 13(2):251–263, April 1997.

[7] Daniel L. Boley, Erik S. Steinmetz, and Karen Sutherland. Robot localization from landmarks using recursive total least squares. In *Proc. IEEE Cont. on Robotics and Automation*, pages 1381–1386, Minneapolis MN, April 1996. IEEE Computer Society Press.

[8] J. Borenstein. The clapper: a dual-drive mobile robot with internal correction of dead-reckoning errors. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 3085–3090, San Diego, CA,, May 8-13 1994.

[9] J. Borenstein, H R. Everett, L. Feng, and D. Wehe. Mobile robot positioning: sensors and techniques. *Journal of Robotic Systems*, 14(4):231–249, Apr 1997.

[10] S. M. Bozic. *Digital and Kalman filtering.* Edward Arnold, second edition, 1994.

[11] R.A Brooks. Visual map making for a mobile robot. *IEEE Trans. Robotics and Automation*, pages 824–829, 1985.

[12] H. Bulata and M.Devy. Incremental construction of a landmark-based and topological model of indoor environments by a mobile robot. *Proc. IEEE Conference on Robotics and Automation*, pages 1054–1060, April 1996.

[13] Howie Choset. Incremental construction of the generalized voronoi diagram, the generalized voronoi graph and the hierarchical generalized voronoi graph. In *FIRST CGC WORKSHOP ON COMPUTATIONAL GEOMETRY*, Baltimore, MD, Octobor 1997.

[14] Howie Choset and Philippe Pignon. Coverage path planning: The boustrophedon cellular decomposition. In *International Conference on Field and Service Robotics*, Canberra, Australia, 1997.

[15] William W. Cohen. Adaptive mapping and navigation by teams of simple robots. *Robotics & Autonomous Systems*, 18(4):411–434, Oct 1996.

[16] A. Curran and K J. Kyriakopoulos. Sensor-based self-localization for wheeled mobile robots. *Journal of Robotic Systems*, 12(3):163–176, Mar 1995.

[17] Xiaotie Deng and Andranik Mirzaian. Competitive robot mapping with homogeneous markers. *IEEE Trans. on Robotics and Automation*, 12(4):532–542, August 1996.

[18] Bruce Randall Donald, James Jennings, and Daniela Rus. Analyzing teams of cooperating mobile robots. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 3, pages 130–135. IEEE, 1994.

[19] Gregory Dudek, Paul Freedman, and Souad Hadjres. Using local information in a non-local way for mapping graph-like worlds. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence*, pages 1639–1645, Chambery, France, August 1993. Internation Joint Conf. on Artificial Intelligence Inc.

[20] Gregory Dudek, Paul Freedman, and Souad Hadjres. Mapping in unknown graph-like worlds. *Journal of Robotic Systems*, 13(8):539–559, August 1996.

[21] Gregory Dudek, Paul Freedman, and Ioannis M. Rekleitis. Just-in-time sensing: efficiently combining sonar and laser range data for exploring unknown worlds. In *International Conference in Robotics and Automation*, volume 1, pages 667–671. IEEE, April 1996.

[22] Gregory Dudek and Michael Jenkin. *Computational Principles of Mobile Robotics.* Number ISBN: 0521568765. Cambridge University Press, May 2000.

[23] Gregory Dudek, Michael Jenkin, Evangelos Milios, and David Wilkes. Robotic exploration as graph construction. *Transactions on Robotics and Automation*, 7(6):859–865, December 1991.

[24] Gregory Dudek, Michael Jenkin, Evangelos Milios, and David Wilkes. Experiments in sensing and communication for robot convoy navigation. In *Proceedings IEEE International Conference on Intelligent Robots and Systems (IROS)*, volume 2, pages 268–273, Pittsburgh, PA, August 1995.

[25] Gregory Dudek, Michael Jenkin, Evangelos Milios, and David Wilkes. A taxonomy for multiagent robotics. *Autonomous Robots*, 3:375–397, 1996.

[26] Gregory Dudek and Chi Zhang. Vision-based robot localization without explicit object models. In *Proc. International Conference of Robotics and Automation*, Minneapolis, MN, 1996. IEEE Press.

[27] A. Elfes. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation*, 3(3):249–265, June 1987.

[28] David Eu and Godfried Toussaint. On approximating polygonal curves in two and three dimensions. *CVGIP: Graphical Models and Image Processing*, 56(3):231–246, May 1994.

[29] T. Fujii, H. Asama, T. von Numers, T. Fujita, H. Kaetsu, and I. Endo. Co-evolution of a multiple autonomous robot system and its working environment via intelligent local information storage. *Robotics & Autonomous Systems*, 19(1):1–13, Nov. 1996.

[30] Toshio Fukuda, Shigenori Ito, Fumihito Arai, Yasunari Yokoyama, Yasunori Abe, Kouetsu Tanaka, and Yoshio Tanaka. Navigation system based on ceiling landmark recognition for autonomous mobile robot - landmark detection based on fuzzy template matching (ftm). In *IEEE International Conference on Intelligent Robots and Systems*, volume 2, pages 150–155, 1995.

[31] Arthur Gelb. *Applied Optimal Estimation*. MIT Press, Cambridge, Massachusetts, 1974.

[32] F. Giuffrida, C. Massucco, P. Morasso, G. Vercelli, and R. Zaccaria. Multi-level navigation using active localization system. In *IEEE International Conference on Intelligent Robots and Systems*, volume 1, pages 413–418. IEEE, 1995.

[33] Susan Hert and Vladimir Lumelsky. The ties that bind: Motion planning for multiple tethered robots. *Robotics and Automation Systems*, (17):187–215, 1996.

[34] H. Imai and Masao Iri. Polygonal approximations of a curve – formulations and algorithms. In G. T. Toussaint, editor, *Computational Morphology*, pages 71–86. Elsevier Science Publishers B. V., New York, N.Y., 1988.

[35] Daehee Kang, Hideki Hashimoto, and Fumio. Harashima. Position estimation for mobile robot using sensor fusion. In *IEEE International Conference on Intelligent Robots and Systems*, volume 1, pages 271–276. IEEE, 1995.

[36] Jon Kleinberg. On-line Search in a Simple Polygon. In *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 8–15, 1994.

[37] S. Koenig and R.G. Simmons. Unsupervised learning of probabilistic models for robot navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2301–2308. IEEE, 1996.

[38] B. Kuipers and T. Levitt. Navigation and mapping in large-scale space. *AI Magazine*, pages 25–43, summer 1988.

[39] Ryo Kurazume, Shigeo Hirose, Shigemi Nagata, and Naoki Sashida. Study on cooperative positioning system. In *International Conference in Robotics and Automation*, volume 2, pages 1421–1426. IEEE, April 1996.

[40] Ryo Kurazume and Shigemi Nagata. Cooperative positioning with multiple robots. In *International Conference in Robotics and Automation*, volume 2, pages 1250–1257. IEEE, 1994.

[41] S. Lang and A. Wong. Building geometric world models with graph synthesis for sensor fusion in mobile robots. *Computational Intelligence*, 6:91–107, 1990.

[42] 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.

[43] C. Lin and R. Tummala. Mobile robot navigation using artificial landmarks. *Journal of Robotic Systems*, 14(2):93–106, 1997.

[44] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349, 1997.

[45] Feng Lu and E. Milios. Robot pose estimation in unknown environments by matching 2d range scans. *Journal of Intelligent and Robotic Systems*, 1998.

[46] V. Lumelsky, S. Mukhopadhyay, and Kang Sun. Sensor-based terrain acquisition: The 'sightseer' strategy. In *Proceedings of the IEEE Conference on Decision and Control Including The Symposium on Adaptive Processes*, volume 2, pages 1157–1161, IEEE Service Center, Piscataway, NJ, USA, 1989. IEEE, IEEE.

[47] Vladimer Lumelsky, Snehasis Mukopadhyay, and Kang Sun. Dynamic path planning in sensor-based terrain acquisition. *IEEE Trans. on Robotics and Automation*, 6(4):462–472, 1990.

[48] Paul MacKenzie and Gregory Dudek. Precise positioning using model-based maps. In *Proceedings of the International Conference on Robotics and Automation*, San Diego, CA, 1994. IEEE Press.

[49] Maja J. Mataric, Martin Nilsson, and Kristian T. Simsarian. Cooperative multi-robot box-pushing. In *IEEE International Conference on Intelligent Robots and Systems.*, volume 3, pages 556–561. IEEE, 1995.

[50] F. Nashashibi and M. Devy. Combining terrain maps and polyhedral models for robot navigation. In *1993 International Conference on Intelligent Robots and Systems.*, pages 685–691, July 1993.

[51] B. Oommen, S. Iyegar, S. Nageswara, S. Rao, and R. Kashyap. Robot navigation in unknown terrains using learned visibility graphs, part i: The disjoint convex obstacle case. *IEEE J. of Robotics and Automation*, 3(6):672–681, 1987.

[52] J. O'Rourke. *Computational Geometry in C.* Cambridge University Press, 1994. ISBN 0-521-44592-2/Pb.

[53] David Pierce and Benjamin. Kuipers. Learning to explore and build maps. In *Proceedings of the National Conference on Artificial Intelligence.*, volume 2, pages 1264–1271, Menlo Park, CA, USA, 1994. AAAI.

[54] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction.* Springer-Verlag, New York, NY, 1985.

[55] N. S. V. Rao, V. Protopopescu, and N. Manickam. Cooperative terrain model acquisition by a team of two or three point-robots. In *International Conference in Robotics and Automation*, volume 2, pages 1427–1433. IEEE, April 1996.

[56] Nageswara S. V. Rao. Robot navigation in unknown generalized polygonal terrains using vision sensors. *IEEE Transactions on Systems, Man, and Cybernetics*, 25(6):947–962, June 1995.

[57] Ioannis Rekleitis, Gregory Dudek, and Evangelos Milios. Multi-robot exploration of an unknown environment, efficiently reducing the odometry error. In IJCAI, editor, *International Joint Conference in Artificial Intelligence*, volume 2, pages 1340–1345, Nagoya, Japan, August 1997. Morgan Kaufmann Publishers, Inc.

[58] Ioannis M. Rekleitis, Gregory Dudek, and Evangelos E. Milios. Accurate mapping of an unknown world and online landmark positioning. In *Proceedings of Vision Interface 1998*, pages 455–461, Vancouver, Canada, June 1998.

[59] Robert Sim and Gregory Dudek. Learning visual landmarks for pose estimation. In *Proc. of Int. Conf. on Robotics and Automation (ICRA)*, May 1999.

[60] R. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainty. *International Journal of Robotics Research*, 5(4):56–68, Winter 1986.

[61] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In I.J. Cox and G. T. Wilfong, editors, *Autonomous Robot Vehicles*, pages 167–193. Springer-Verlag, 1990.

[62] Kazuo Sugihara and Ichiro Suzuki. Distributed algorithms for formation of geometric patterns with many mobile robots. *Journal of Robotics Systems*, 13(3):127–139, 1996.

[63] S. Thrun. Learning metric-topological maps for indoor mobile robot navigation. *AI Journal*, 99:1:21–71, 1998.

[64] S. Thrun, D. Fox, and W. Burgard. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning and Autonomous Robots*, 31,5:29–53,253–271, 1998.

[65] Thomas von Numers, Hajime Asama, Takanoru Fujita, Shin'ya Kotosaka, Sakae Muyao, Hayato Kaetsu, and Isao Endo. An intelligent data carrier system for local communication between cooperative multiple mobile robots and environment. In *2nd IFAC Conf. on Inteligent Autonomous Vehicles 95*, pages 366–370, 1995.

[66] Ashely Walker, Hihn Hallman, and David Willshaw. Bee-havior in a mobile robot: The construction of a self-organized cognitive map and its use in robot navigation within a complex, natural environment. In *IEEE Conf. on Neural Networks*, pages 1451–1456, 1993.

[67] P. Weckesser, R. Dillmann, M. Elbs, S. Hampel, and U. Rembold. Multiple sensorprocessing for high-precision navigation and environmental modeling with a mobile robot. In *IEEE International Conference on Intelligent Robots and Systems.*, volume 1, pages 453–458. IEEE, 1995.

[68] Gerhard Weiss, Christopher Wetzler, and Ewald. von Puttkamer. Keeping track of po-

sition and orientation of moving indoor systems by correlation of range-finder scans. In *IEEE/RSJ/GI International Conference on Intelligent Robots and Systems.*, volume 1, pages 595–601. IEEE, 1994.

[69] Xiaoping Yun, Gokhan Alptekin, and Okay Albayrak. Line and circle formation of distributed physical mobile robots. *Journal of Robotic Systems*, 14(2):63–76, Feb 1997.

11. Appendix A: Optimality Proof

Every time one robot is moving it covers a small area of free space, by alternating roles the two robots try to cover the whole area of free space. Therefore, the total area covered is the sum of the areas covered every single exchange minus the overlapping covered area. This is equivalent with the tiling problem where every small area covered by the motion of one robot represents a tile, and the objective is to cover the free space with tiles without leaving any space uncovered. We present an optimal tiling of the space, under certain assumptions.

Assumptions: Every tile is contained in a wedge of radius R and angle θ.

We consider only tilings defined by the line of sight, between the two robots, sweeping across the space. One end of the line is fixed (the position of the stationary robot) and the other follows the trajectory of the second robot. The length of the line of sight is bounded by the sensing range R.

Lemma 1: A tiling in which no tile with an angle more than 180° is used, can replace any arbitrary tiling, without increasing the complexity.

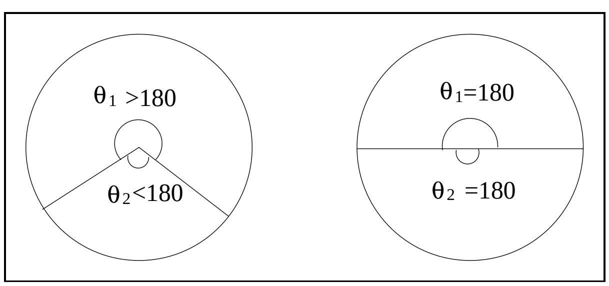


Figure 14. Equivalence among two pairs of tiles.

Proof: If a tile with an angle more than 180° is used then, at least another tile with an angle less than 180° is necessary to cover the remaining free area. The combination of the two (or more) tiles is equivalent (in terms of complexity) to two (or more) tiles of angle 180° or less (as in figure 14). The number of exchanges stays the same and the total path traveled is $2\pi R$ for a πR^2 area.

Lemma 2: When two tiles are connected along the arc part of their wedges, the most efficient curve (in terms of path length) is the common chord they share.

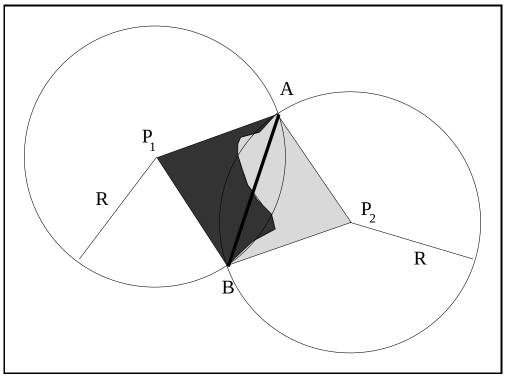


Figure 15. Shortest path along the arc connections.

Proof: Given two tiles (eg. P_1AB and P_2AB, see Figure 15), the shortest distance between A and B is given by the straight line connecting A and B. If the two robots travel in a a different path (than a straight line) then the length of the path traveled would be larger than AB.

Lemma 3: When the robots exchange roles they produce tiles (areas of free space) that are connected along the rays that specify the boundaries of the wedges (see Figure 16).

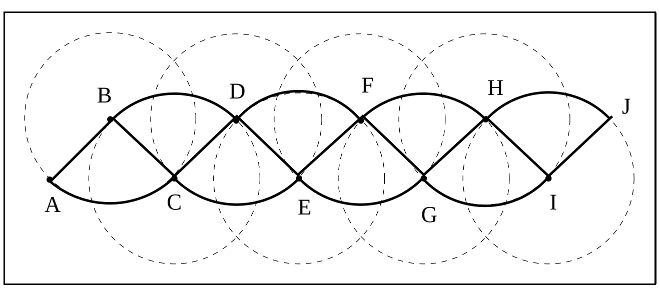


Figure 16. Sequence of adjacent tiles.

Proof: Let two robots explore a series of areas of free space by exchanging roles. Without loss of generality, let robot number one move inside the corresponding wedges as it covers the paths, AC, CE, EG, GI while robot number two moves across BD, DF, FH. That happens because after the motion of one robot and the mapping of free space the two robots have to exchange roles and the other robot would continue mapping the free space starting from the common line of sight. Therefore each two neighboring tiles (such as ACB, BDC) are going to be connected along the common ray / line_of_sight (BC) that connects them when they exchange roles.

Lemma 4: Assume a sequence of N tiles (a stripe) connected as in **Lemma 3** with varying angles ($\theta_i \quad 1 \leq i \leq N$) for each wedge. There exist an angle θ' such that: a sequence of N tiles with the same angle θ' is going to cover an equal or larger area for the same length of the path traveled.

Proof: The total length of the path traveled for the two stripes is a sum of the sub-paths P^{θ_i} and $P^{\theta'}$ and is given in Equation 4:

$$\sum_{i=1}^{N} P^{\theta_i} = \sum_{i=1}^{N} P^{\theta'}$$

which is equivalent to (4)

$$\sum_{i=1}^{N} 4R\sin\frac{\theta_i}{4} = \sum_{i=1}^{N} 4R\sin\frac{\theta'}{4}$$

We are going to prove that the sum of the areas covered with different angles is smaller or equal to the area covered by the same angle (see Equation 5).

$$\sum_{i=1}^{N} A^{\theta_i} \leq \sum_{i=1}^{N} A^{\theta'} \Leftrightarrow$$

$$\sum_{i=1}^{N} R^2\sin\frac{\theta_i}{2} \leq \sum_{i=1}^{N} R^2\sin\frac{\theta'}{2} \tag{5}$$

Removing the constant terms from both sides Equation 4 and Equation 5 became:

Given N angles $\theta_i \; : \; i = 1, N$. **If**

$$\sum_{i=1}^{N} \sin\frac{\theta_i}{4} = N \sin\frac{\theta'}{4} \tag{6}$$

Then

$$\sum_{i=1}^{N} \sin\frac{\theta_i}{2} \leq \sum_{i=1}^{N} \sin\frac{\theta'}{2} \tag{7}$$

- For N=2, we solve equation 6 for θ' (N=2):

$$\begin{aligned} \sin\frac{\theta_1}{4} + \sin\frac{\theta_2}{4} &= 2\sin\frac{\theta'}{4} \Leftrightarrow \\ \sin\frac{\theta'}{4} &= \frac{\sin\frac{\theta_1}{4} + \sin\frac{\theta_2}{4}}{2} \end{aligned} \tag{8}$$

For any pair (θ_1, θ_2) where $\theta_i \in [0, \pi]$ then $\Delta A = 2\sin\frac{\theta'}{2} - (\sin\frac{\theta_1}{2} + \sin\frac{\theta_2}{2}) \geq 0$ (see the graph of ΔA in figure 17).

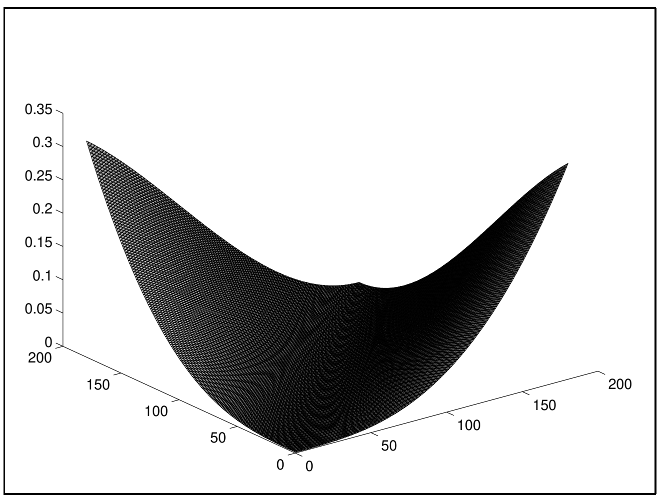


Figure 17. Graph of ΔA for $0 \leq \theta_i \leq 180^\circ$, $\Delta A \geq 0$ for any pair of angles

As $\Delta A \geq 0$ then

$$\sin\frac{\theta_1}{2} + \sin\frac{\theta_2}{2} \leq 2\sin\frac{\theta'}{2} \tag{9}$$

Q.E.D. for N=2.

- For $N > 2$ we examine two cases, first $N = 2^K$ and then the general case.

  * For $N = 2^K$:
    Let the angles be in pairs of $(\theta_{2i-1}, \theta_{2i} : 1 \leq i \leq (N/2) = 2^{K-1})$. Then for every pair $(\theta_{2i-1}, \theta_{2i})$ calculate the angle $\theta'_i : 1 \leq i \leq 2^{K-1}$ as in the case for N=2 (Equation 10).

$$\sin \frac{\theta'_i}{4} = \frac{\sin \frac{\theta_{2i-1}}{4} + \sin \frac{\theta_{2i}}{4}}{2} \tag{10}$$

From Equation 9:

$$\begin{aligned} \sum_{i=1}^{2^K} \sin \frac{\theta_i}{2} = \sum_{i=1}^{2^{K-1}} (\sin \frac{\theta_{2i-1}}{2} + \sin \frac{\theta_{2i}}{2}) \leq \\ \sum_{i=1}^{2^{K-1}} 2 \sin \frac{\theta'_i}{2} = 2 \sum_{i=1}^{2^{K-1}} \sin \frac{\theta'_i}{2} \end{aligned} \tag{11}$$

Therefore:

$$\sum_{i=1}^{2^K} \sin \frac{\theta_i}{2} \leq 2 \sum_{i=1}^{2^{K-1}} \sin \frac{\theta'_i}{2} \tag{12}$$

Now we have $M = 2^{K-1}$ angles (θ'_i). Repeat the calculations for the θ'_i angles finding $M' = 2^{K-2}$ angles (θ''_i). Solving for pairs of angles $(\theta'_{2i-1}, \theta'_{2i} : 1 \leq i \leq (N/4) = 2^{K-2})$. Calculate from Equation 13 all the angles $\theta''_i : 1 \leq i \leq 2^{K-2}$

$$\sin \frac{\theta''_i}{4} = \frac{\sin \frac{\theta'_{2i-1}}{4} + \sin \frac{\theta'_{2i}}{4}}{2} \tag{13}$$

From Equation 9:

$$\sum_{i=1}^{2^{K-1}} \sin\frac{\theta'_i}{2} = \sum_{i=1}^{2^{K-2}} (\sin\frac{\theta'_{2i-1}}{2} + \sin\frac{\theta'_{2i}}{2}) \leq$$
$$\sum_{i=1}^{2^{K-2}} 2\sin\frac{\theta''_i}{2} = 2\sum_{i=1}^{2^{K-2}} \sin\frac{\theta''_i}{2} \tag{14}$$

From Equation 11 and Equation 14

$$\sum_{i=1}^{2^K} \sin\frac{\theta_i}{2} \leq 2\sum_{i=1}^{2^{K-1}} \sin\frac{\theta'_i}{2} \leq 2\times 2\sum_{i=1}^{2^{K-2}} \sin\frac{\theta''_i}{2} \tag{15}$$

Repeating the previous steps K times. For $\theta^{(K)}$ that satisfies Equation 15 Equation 17 is true.

$$\sum_{i=1}^{2^K} \sin\frac{\theta_i}{4} = 2\sum_{i=1}^{2^{K-1}} \sin\frac{\theta'}{4} = 4\sum_{i=1}^{2^{K-2}} \sin\frac{\theta''}{4} = N\sin\frac{\theta^{(K)}}{4} \tag{16}$$

$$\sum_{i=1}^{2^K} \sin\frac{\theta_i}{2} \leq 2\sum_{i=1}^{2^{K-1}} \sin\frac{\theta'_i}{2} \leq 4\sum_{i=1}^{2^{K-2}} \sin\frac{\theta''_i}{2} \leq \ldots \leq N\sin\frac{\theta^{(K)}}{2} \tag{17}$$

Q.E.D. For any N in the form: $N = 2^K$

$*$ For $N \neq 2^K$:

Let $K \in \mathcal{N}$ such that $2^K \leq N \leq 2^{K+1}$. Then for the $\theta_i : 1 \leq i \leq N$ solve Equation 18 for the θ'. And we want to prove that the inequality in Equation 19 is true.

$$\sum_{i=1}^{N} \sin\frac{\theta_i}{4} = N\sin\frac{\theta'}{4} \tag{18}$$

$$\sum_{i=1}^{N} \sin\frac{\theta_i}{2} \leq \sum_{i=1}^{N} \sin\frac{\theta'}{2} \tag{19}$$

Consequently, by adding in both sides of Equation 18 ($M\sin\frac{\theta'}{4}$) where $M = (2^{K+1} - N)$, Equation 18 becomes Equation 20.

$$\sum_{i=1}^{N} \sin\frac{\theta_i}{4} + M \sin\frac{\theta'}{4} = N \sin\frac{\theta'}{4} + M \sin\frac{\theta'}{4} = 2^{K+1} \sin\frac{\theta'}{4} \tag{20}$$

Now the number of terms in each side of Equation 18, is 2^{K+1} and from Equation 17 we get:

$$\begin{gathered} \sum_{i=1}^{N} \sin\frac{\theta_i}{2} + M \sin\frac{\theta'}{2} \leq (N+M) \sin\frac{\theta'}{2} \Leftrightarrow \\ \sum_{i=1}^{N} \sin\frac{\theta_i}{2} \leq N \sin\frac{\theta'}{2} \end{gathered} \tag{21}$$

Q.E.D.

Theorem 1: For any tiling, and with optimality criterion the shortest path for a given number of exchanges, the optimal tile is the union of two isosceles triangles (equal edges are R), with the angle of the two equal edges equal to $\theta/2$, where θ the wedge angle (see figure 18).

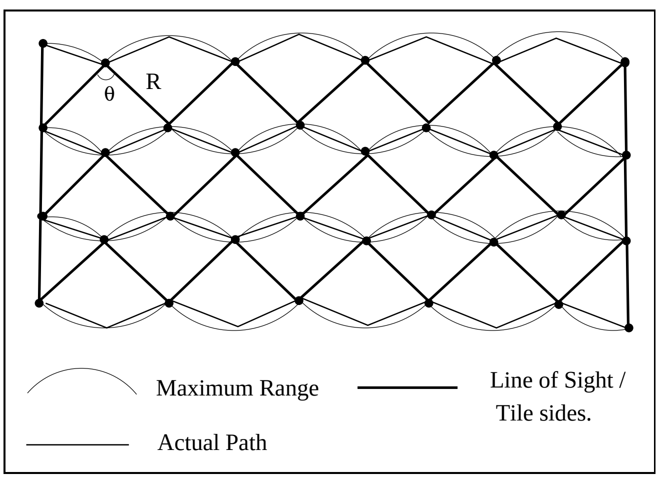


Figure 18. Positioning of the wedge stripes.

Proof: From lemma 3 and for the same angle θ for every wedge, the wedges are going to be arranged into stripes as in figure 16. Moreover given a constant number of tiles the angle θ is set. When one stripe is positioned next to its neighbor, there must be complete coverage, with minimum overlap. The optimal positioning of the wedges is displayed in figure 19, such that the curves are complement each other. From lemma 2, the optimal path is that given in figure 18, by connecting with straight lines the overlapped areas.

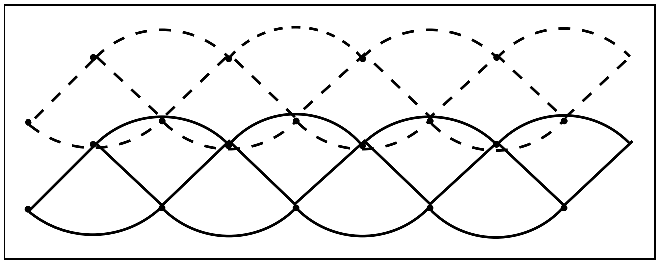


Figure 19. Optimal tiling.