MRBUG: A Competitive Multi-Robot Path Finding Algorithm

Shahar Sarid, Amir Shapiro and Yoav Gabriely

Abstract— **We explore an on-line problem where a group of robots has to reach a target whose position is known in an unknown planar environment whose geometry is acquired by the robots during task execution. The critical parameter in such a problem is the physical motion time, which, under the assumption of uniform velocity of all the robots, corresponds to length or cost of the path traveled by the robot which reached the target. The *Competitiveness* of an on-line algorithm measures its performance relative to the optimal off-line solution to the problem. While competitiveness usually means constant relative performance, this paper uses generalized competitiveness, i.e. any functional relationship between on-line performance and optimal off-line solution. Given an on-line task, its *Competitive Complexity Class* is a pair of lower and upper bounds on the competitive performance of all on-line algorithms for the task, such that the two bounds satisfy the same functional relationship. We prove that in general any on-line navigation algorithm must have at least a quadratic competitive performance. This paper describes a new on-line navigation algorithm, called *MRBUG* (short for *Multi-Robot BUG*), which requires constant memory and has a quadratic competitive performance. Thus, the above mentioned problem is classified into a quadratic competitive class. Moreover, since *MRBUG* achieves the quadratic lower bound, it has optimal competitiveness. The algorithm performance is illustrated in office-like environments.**

I. INTRODUCTION

The Problem of reaching a target whose position is known in an unknown planar environment is very important in many practical and academic research fields, the most significant are search and rescue, industry and military robotics. Area coverage is in the worst case a corresponding task, since the searching unit will cover a certain area before finding the path to the target. Examples for the problems above are demining, cleaning supermarkets and train stations, detecting contaminated or radioactive substances in factories, nuclear reactors or in the open field, planetary exploration and sample acquisition. This paper is concerned with the aforementioned problem solved by multiple mobile robots.

Using a group of robots can have many advantages, the most important are shortening of the total path or mission time and increased robustness, since the multitude of robots can easily overcome a malfunction in one or more of the units, an issue associated with redundancy. The decrease of the individual mechanical wear and power consumption per mission maximizes the life span of each robot and prolongs the whole mission duration and range. Other advantages concerns maintaining radio connectivity between the robots and the base station and a decreased sensor uncertainty due to merging of overlapping information, it has been shown that multiple robots localize themselves more efficiently [1], especially when they have different sensor capabilities.

The most critical parameter in mobile robot motion tasks is the physical travel time. Under a uniform velocity assumption, travel time corresponds to path length. We denote the distance traveled by each robot, l, and the optimal off-line solution, l_{opt}. Hence, the algorithm discussed in this paper is classified in terms of length or cost of the path traveled by one robot during algorithm execution. The notion of competitiveness compares the performance of an on-line algorithm to the optimal off-line solution for the same problem. In particular, an algorithm for a task P is said to be competitive if its solution to every instance of P is bounded by a constant time l_{opt} [2]. Generalized *Competitive Complexity* and *Competitive Complexity Classes* are introduced and discussed in [3], however, most of the papers dealing with competitiveness strive to identify specific classes of environments in which constant competitiveness can be achieved. In contrast, our objective is to identify the competitive relationship governing the fully general on-line navigation problem for multiple robots.

Recent works related to the subject of mobile multi-robot motion planning deals with various aspects of the problem. A major issue is whether the group architecture is centralized, i.e., there is only one control agent or decentralized, where each robot is autonomous and no global coordination needed. Communication is a very close subject, since, the system cannot be centralized when it lack thereof. Intermediate systems represent real-world setups better, for example, the semi-decentralized approach in [4], where robot teams cover the space independent of each other, but robots within a team communicate state and share information. Limited communication plays an important role when dealing with ant-like robots, where messages between the robots are passed mainly or only through markings they leave on the terrain [5]. A solution to a problem can change according to the availability of information on the environment prior to algorithm execution. On-line solutions assume no knowledge of the environment when the algorithm starts, while off-line solutions rely on a priori knowledge. An off-line algorithm is presented in the notable early paper [6]. A new on-line algorithm [7] focuses on robustness, and completeness. Robustness measures the performance in case of failures and an algorithm is considered complete if for any input it correctly reports whether or not there is a solution in a finite amount of time. Our solution is complete and robust

S. Sarid and A. Shapiro are with the Department of Mechanical Engineering, Ben-Gurion University of the Negev, P.O. Box 653, Beer-Sheva 84105, Israel [sarids,ashapiro]@bgu.ac.il

Y. Gabriely is with the Department of Mechanical Engineering, Technion - Israel Institute of Technology, Technion City, Haifa 32000, Israel meeryg@tx.technion.ac.il

and can be either decentralized or centralized.

The structure and contributions of the paper are as follows. In the next section we state a key assumption that the robot has a physical size D such that $D > 0$. While this assumption may seem obvious, only few papers make use of this assumption (e.g. [8], [3]). We also present some definitions regarding competitiveness. In Section III we show that for every on-line algorithm, there is a worst case path that yields a cost which is constant times l_{opt}^2. Two new algorithms, *MRBUG* and *PBUG1*, which are based on the area bounding and doubling strategy of *CBUG* [3] are introduced in Section IV. The competitiveness of *MRBUG* is analyzed in Section V. It is shown that the length of the path traveled by each robot during execution of *MRBUG* is at most quadratic in l_{opt}, implying that up to the constant coefficients *MRBUG* has optimal competitiveness. In the same Section *MRBUG* is proved to be complete. *MRBUG* simulation is presented in Section VI. Finally, we conclude and discuss additional research directions and future work.

II. Basic Setup and Definition of Competitiveness

Our basic assumptions are as follows. Each mobile robot is a freely moving planar body of size D, e.g. a disc, where $D > 0$ is a given constant. Each robot is equipped with two sensors which are assumed ideal. The first sensor measures the robot position with respect to a fixed reference frame. The second sensor is an obstacle detection tactile or short range sensor which allows tracing of an obstacle boundary. The robots use onboard or central calculation unit which have enough memory for the needed calculations. The robots should communicate with each other or with a central base station, at least upon initialization and at termination. All the robots move in the same uniform velocity.

The three most significant parameters governing the performance of mobile robot are physical travel time, on-board computation time, and on-board memory. We associate physical travel time with l. The time required for a physical motion step is typically several orders of magnitudes longer than the execution time of an on-board computation step. Since we limit our discussion to algorithms that take polynomial time to compute each physical motion step of the robot, we focus on l as the main performance parameter. Last, we limit the discussion to algorithms which require constant storage. The following definition generalizes the traditional notion of linear competitiveness to any functional relationship between l and l_{opt}.

Definition 1 (Generalized Competitiveness [3]): An on-line algorithm solving a task P is $f(l_{opt})$-*competitive* when l is bounded from above by a scalable function $f(l_{opt})$ over all instances of P. In particular, $l \leq c_1 l_{opt} + c_0$ is the traditional linear competitiveness, while $l \leq c_2 l_{opt}^2 + c_1 l_{opt} + c_0$ is quadratic competitiveness, where the c_i's are positive constant coefficients that depend on the robot size D.

Note that the definition of $f(l_{opt})$-*competitiveness* focuses on a particular algorithm solving the task P. However, our objective is to characterize the lowest upper bound that can be achieved over all on-line algorithms for P. This objective

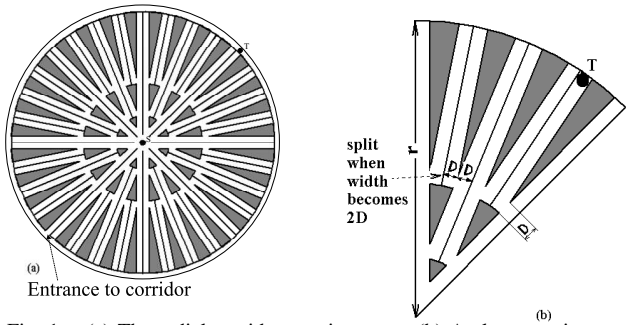

Fig. 1. (a) The radial corridors environment. (b) A close up view .

requires a universal lower bound on the performance of all on-line algorithms for P. If the lower and the upper bounds satisfy the same functional relationship, we associate that functional relationship with P itself. This notion is made formal in the following definition.

Definition 2 (Competitive Complexity Class [3]): A universal lower bound on the competitiveness of a task P is a lower bound $l \geq g(l_{opt})$ over all on-line algorithms for P. If a competitive upper bound $f(l_{opt})$ and a universal lower bound $g(l_{opt})$ for P are the same function up to constant, this function is the competitive complexity class of P.

The competitive complexity class of a task P is thus a pair of lower and upper bounds on the competitive performance of all on-line algorithms for P, such that the two bounds are identical up to constant coefficients. Note that competitive complexity characterizes the task P itself, not any specific algorithm for P. The remainder of the paper characterizes the competitive complexity class of the multi-robot on-line navigation problem.

III. Universal Lower Bound

In this section we establish a universal lower bound on the competitive complexity of a navigating group of robots. We show that in the worst case scenario the robots will cover a certain area prior to finding the path to the target. Hence, we use the environment depicted in Fig. 1 which is built from radial corridors having the width of the robots, D, and one circular corridor containing the target with only one entrance.

Theorem 1 (Quadratic Lower Bound): Let $\mathcal{A}$ be any navigation algorithm for n robot pairs of size D in an unknown planar environment to a target whose position is known. Let l be the length of the path generated by $\mathcal{A}$ for one of the robots, and let l_{opt} be the length of the optimal off-line path. Then l satisfies the quadratic lower bound,

$$l \geq \frac{4\pi}{3nD(1+\pi)^2}(1-\epsilon)l_{opt}^2$$

where ϵ is an arbitrary small positive parameter.

Proof: Consider the corridor environment with the target $\mathcal{T}$ placed at the end of a distal corridor, at a distance r from $\mathcal{S}$. Since the robots have no knowledge of the environment and has no information where the entrance to the outer corridor might be, they must in worst case inspect all the corridors. The total area of the obstacles in the corridor environment is almost one third of the disc area, with the approximation becoming arbitrary close to one third as the disc's radius increases [3]. The total area inspected by the robots is therefore $2\pi r^2/3$. The robots must move twice

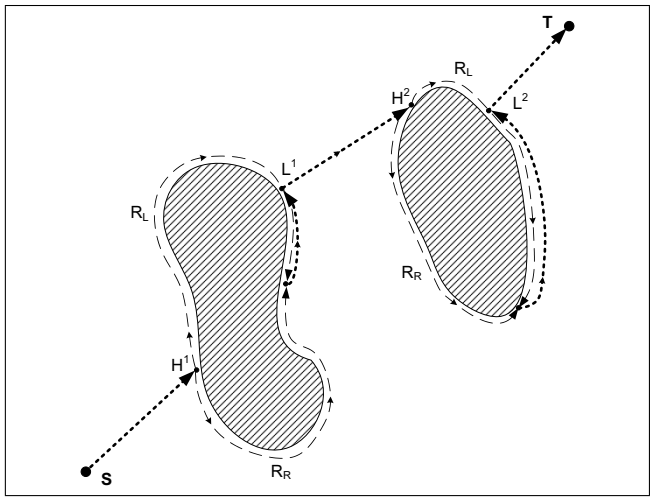

Fig. 2. A pair of two robots, R_L and R_R executing *PBUG1*. The dense dashed lines mark a mutual path of the two robots

through every corridor of the environment, once in order to inspect it once in order to exit, and since all corridors have a width D which is identical to the robot's size, the total length of the path traveled by the robots satisfies $l_{tot} \geq 4\pi r^2/3D - r$, where the substraction of r is due to the last corridor which need not be traced backward. A good algorithm will not generate overlapping paths for its robots, thus, the total length of the path traveled by one robot satisfies $l \geq 4\pi r^2/3nD - r/n$. Since $\mathcal{T}$ is placed in the circular outer corridor, we have in worst case that $l_{opt} \leq (1+\epsilon')(1+\pi)r$, where ϵ' is an arbitrary small positive parameter. It follows that $r \geq l_{opt}/(1+\epsilon')(1+\pi)$. On the other hand, $r \leq l_{opt}$ in the above mentioned environment. Substituting the last two inequalities into the lower bound on l gives $l \geq \frac{4\pi}{3nD(1+\epsilon')^2(1+\pi)^2} l_{opt}^2 - \frac{1}{n} l_{opt}$. We can write the last inequality as $l \geq c l_{opt}^2 (1/(1+\epsilon')^2) - 1/cnl_{opt}) = c l_{opt}^2 (1 - \epsilon'' - 1/cnl_{opt})$, where $c = 4\pi/3nD(1+\pi)^2$ and $1/(1+\epsilon')^2 = 1 - \epsilon''$. Since the quantity $\epsilon = \epsilon'' + 1/cnl_{opt}$ contains the quotient D/l_{opt} which can be made arbitrarily small for sufficiently large environments, we obtain the lower bound $l \geq c(1-\epsilon) l_{opt}^2$. ■

IV. *MRBUG* Motion Algorithm to a Known Target

We now introduce *MRBUG*, a multi-robot algorithm which uses as a sub procedure *PBUG1* algorithm, a new version of *BUG1* algorithm [9] for a pair of robots.

A. PBUG1 *Motion Algorithm for a Pair of Robots*

PBUG1 deploys a pair of robots that start from a common start point $\mathcal{S}$ and needs to reach a target $\mathcal{T}$ whose position is known in an unknown planar environment. The pair of robots will move together toward the target in a straight line until they hit an i^{th} obstacle at a point marked as *Hit point* H^i, $i = 1, 2, ...$ (Fig. 2). At that point they split, robot R_L turns left and robot R_R turns right, and they circumnavigate the obstacle from different directions. On that account, each robot encircle half of the obstacle perimeter. While moving, each robot calculates and remembers the closest point on the obstacle's boundary to the target. Upon meeting, the robots compare the recorded information, decide which point is the closest to the target, join and move together to that closest point which they mark as *Leave point* L^i, $i = 1, 2, ...$, from which they continue toward the target.

***PBUG1* Algorithm**

Sensors: A position and orientation sensors.
An obstacle detection sensor.
Input: Position of a start $\mathcal{S}$ and a target $\mathcal{T}$.
A pair of robots: R_L, R_R.
Initialization: For each of the robots in the pair R_L, R_R:
Define local direction: $Left$ for R_L, $Right$ for R_R.
Set i=1.
Set initial leave point L^0 = $\mathcal{S}$.
For each of the two robots R_L, R_R, Repeat:
From the point L^{i-1}, move toward the target along a straight line until one of the following occurs:
(1) The target is reached: STOP.
(2) An obstacle is encountered: Define a hit point H^i.
Turn in the direction of the predefined local direction and follow the obstacle boundary according to that direction. While circumnavigating the obstacle, calculate and record the coordinates of the closest point to the target, Q_{m_L} and Q_{m_R} for R_L and R_R respectively, until one of the following occurs:
(a) The target is reached: STOP.
(b) Upon meeting each other, exchange information and calculate the closest point to $\mathcal{T}$: $Q_m = min(Q_{m_L}, Q_{m_R})$.
Define a new leave point $L^i = Q_m$.
Apply the test for target reachability:
(i) If the target is not reachable: STOP.
(ii) Else, move to L^i: If $Q_m = Q_{m_L}$, trace back R_L path. Otherwise, trace back R_R path.
Set $i = i + 1$.
End of Repeat loop

The basic setup and definitions of *BUG1* in [9] for one robot are applicable in *PBUG1* for two robots along with a few modifications as follows. First, *PBUG1* needs only one register for each robot, to store the coordinates of the current point, Q_m, of the minimum distance between the obstacle boundary and the target. Second, *PBUG1* determines that the target is unreachable and trapped inside an obstacle using *BUG1* method [9]: If after circumnavigating an obstacle, the leaving direction toward the target points into the last obstacle, the target is surrounded by that obstacle.

B. MRBUG *Algorithm for a Group of Robots*

MRBUG algorithm launches n pairs of robots from a common starting point $\mathcal{S}$ and assigns each pair j' to a different ellipse to search for a path to the target $\mathcal{T}$ in it, each ellipse's focal points are $\mathcal{S}$ and $\mathcal{T}$.

The first pair of robots (j'=1) is designated to the initial ellipse of area A_0, and each of the following robots starts its search executing *PBUG1* in an ellipse of area larger than the previous ellipse's area by a factor of $\alpha > 1$, namely, the areas of the ellipses will be $A_0, \alpha A_0, \alpha^2 A_0 ...$. The execution of *PBUG1* regards the ellipse as a virtual obstacle boundary. The pair of robots repeats the process on the next

unassigned ellipse in the series until the target is detected.

***MRBUG* Algorithm:**

Sensors: A position and orientation sensors.
An obstacle detection sensor.
Input: Position of a start $\mathcal{S}$ and a target $\mathcal{T}$ points.
An initial ellipse with focal points $\mathcal{S}$ and $\mathcal{T}$ and area A_0.
n pairs of robots $\{1_L, 1_R, 2_L, 2_R, \ldots, n_L, n_R\}$.
Initialization: For each robot pair j', $j' = 1, \ldots, n$:
Set current search ellipse $e_{j'} = j'$,
Set initial leave point $L^0_{e_{j'}} = \mathcal{S}$,
Set multiplication factor $\alpha = (n+1)^{1/n}$,
Set initial search area $A_{j'}(e_{j'}) = \alpha^{e_{j'}-1} A_0$.
For each robot pair j', Repeat:
Initialize *PBUG1* with the following parameters:
Create an outer virtual obstacle boundary with an ellipse of area $A_{j'}(e_{j'})$ having focal points $\mathcal{S}$ and $\mathcal{T}$.
Start point is $\mathcal{S}$, target is $\mathcal{T}$.
Set $i = 1$.
Leave point is $L^0_{e_{j'}}$.
Execute *PBUG1* until one of the following occurs:
(1) *PBUG1* terminates at $\mathcal{T}$: STOP, target is found.
(2) $\mathcal{T}$ is trapped inside an obstacle:
(a) If the obstacle does not intersect the $e_{j'}$ ellipse: STOP, the target is unreachable.
(b) Else, move to the next unoccupied ellipse:
Set $e_{j'} = e_{j'} + n$.
Set $L^0_{e_{j'}}$ at *PBUG1* termination point.
Set $A_{j'}(e_{j'}) = \alpha^{e_{j'}-1} A_0$.
End of Repeat loop

Rather than give a formal proof of correctness, we make some informal remarks on the algorithm. First, during initialization, after getting the values of n and A_0, each robot is assigned to a number j' and to a local direction, $Left$ or $Right$ and thus can calculate its future search ellipses and corresponding areas, which means that after a pair of robots has finished searching for a path in an ellipse, it can immediately continue to search in the next unassigned ellipse regardless of the state of the other robots. Second, in step (2), *PBUG1* determines that the target is unreachable and trapped inside an obstacle (Subsect. IV-A). *MRBUG* assures in step (2.a) that the robots were not bounded by an ellipse and thus guarantee that the target is unreachable. Third, *PBUG1* uses constant memory, and *MRBUG* needs additional constant memory to remember the start position and the current ellipse area, hence, *MRBUG* require constant memory, too.

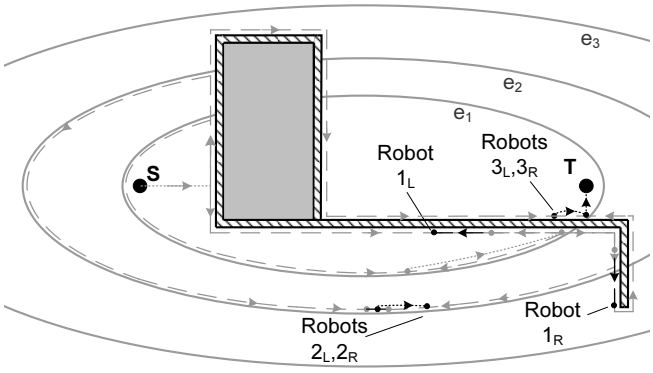

Fig. 3. Last step of execution example of *MRBUG* with 3 pairs of robots. The dense dashed lines indicates a mutual path of a pair. The path traversed in the previous step is colored in grey.

In the following example depicted in Fig. 3, *MRBUG* launches three pairs of robots to search for the target in a simple environment. Each robot pair is initially assigned to a bounding ellipse, e_1, e_2, e_3 to execute *PBUG1* in it, and each robot in a pair is assigned to a different local direction, $Left, Right$: $1_L, 1_R, 2_L, 2_R, 3_L, 3_R$. At first, the robots move directly toward the target, and as they encounter the obstacle they split, and each robot moves in its local direction. Note that pair no. 3 does not have to traverse ellipse no. 3 in this example, since it does not intersects the obstacle. The robots positions in the last step the moment pair no. 3 reached the target are depicted in Fig. 3.

V. Competitive Complexity Analysis of *MRBUG*

We now establish an upper bound on the path length of the robot that reached the target while executing *MRBUG* in terms of the optimal off-line solution, l_{opt}. Since the robot tracing an obstacle boundary is of size D, we first substitute the obstacle into a special object in a way that changes its area and perimeter, but preserves the path length property. Thus, in the following lemmas we use terms related to *Configuration Space* (or $\mathcal{C}$-space). The $\mathcal{C}$-space of a disc shaped robot is $\mathbb{R}^2$, and the $\mathcal{C}$-space obstacle $\mathcal{CB}_i$ consist of all robot configurations where it intersects $\mathcal{B}_i$.

Definition 3 ([3]): Let $\mathcal{CB}_i$ be the $\mathcal{C}$-space obstacle induced by an obstacle $\mathcal{B}_i$ for a disc robot of size D. The *traceable obstacle* induced by $\mathcal{B}_i$, denoted $\boldsymbol{\mathcal{B}_i}$, is obtained by filling any internal holes in $\mathcal{CB}_i$ and then shrinking $\mathcal{CB}_i$ inward by a distance of $D/2$.

Lemma 5.1 ([3]): Let a planar environment contain z disjoint traceable obstacles $\boldsymbol{\mathcal{B}_i}$, $i = 1, \ldots, z$. Let a disc robot of size D trace the i^{th} obstacle boundary, and let q_i be the total area swept by the robot during tracing of the i^{th} boundary. let $\mathcal{C}$ be any simple closed curve which surrounds the z regions swept by the robot. Then $\sum_{i=1}^{z} q_i \leq 4\mathcal{A}(\mathcal{C})$, where $\mathcal{A}(\mathcal{C})$ is the area of the traceable obstacle-free points enclosed by $\mathcal{C}$.

Note that the regions swept during tracing of the individual boundaries may overlap, so that in general the sum $\sum_{i=1}^{z} q_i$ may be larger than $\mathcal{A}(\mathcal{C})$. In the following two lemmas, the complete proofs had to be relegated to [10] due to lack of space. Thus, only the last results are presented.

Lemma 5.2: The length l_i of the path traveled by each robot of the pair traversing the i^{th} ellipse is bounded by $l_i \leq 4\frac{A(i)}{D} + (||L^0_i - \mathcal{T}|| - ||L^0_{i+n} - \mathcal{T}||)$, where $A(i)$ is the area of the i^{th} ellipse, D is the size of each robot, n is the number of robots, L^0_i and L^0_{i+n} are the start points at the i^{th} and at the next ellipse respectively, and $||\beta - \gamma||$ denotes the Euclidean distance between β and γ .

Proof: From [10], [3], the total length of the robot's path during circumnavigation of obstacles in the i^{th} ellipse is at most $4A(i)/D$. Since in *MRBUG* each robot in a pair travels exactly half of the way, the path length of one robot is

not more than $2A(i)/D$. Adding the path to the leave point which does not exceed half of the obstacle's perimeter, leads to the total path length bound of $4A(i)/D$ for each robot. The total length of the motion between obstacles equals to the net decrease of the distance of the robot from $\mathcal{T}$, which is $||L_i^0 - \mathcal{T}|| - ||L_{i+n}^0 - \mathcal{T}||$. ■

Lemma 5.3: Let $\mathcal{T}$ be reachable from $\mathcal{S}$. If the initial ellipse contains no path from $\mathcal{S}$ to $\mathcal{T}$, *MRBUG* reaches the target in an ellipse whose area $A(i)$ is bounded by $A(i) < \frac{\pi\alpha}{4} l_{opt} \sqrt{l_{opt}^2 - ||\mathcal{S}-\mathcal{T}||^2}$, where l_{opt} is the length of the optimal off-line path, and α is the multiplication factor.

Proof: Let A_{min} denote the area of the smallest ellipse with focal points $\mathcal{S}$ and $\mathcal{T}$ which contains the optimal off-line path. From [10], [3], $A_{min} \leq \frac{\pi}{4} l_{opt} \sqrt{l_{opt}^2 - ||\mathcal{S} - \mathcal{T}||^2}$. By assumption the initial ellipse contain no path from $\mathcal{S}$ to $\mathcal{T}$. Hence, *MRBUG* multiplies the area of the search ellipse by a factor of α at least once. The area $A(i)$ of the last ellipse searched by *MRBUG* satisfies the inequality $A(i) < \alpha A_{min}$. Otherwise the previous ellipse has an area $A(i-1) \geq A_{min}$, which implies that *MRBUG* terminated while inspecting the previous ellipse. Substituting for A_{min} in the inequality $A(i) < \alpha A_{min}$ gives the result. ■

The following proposition establishes a quadratic competitive upper bound on *MRBUG*.

Proposition 5.4: If the target $\mathcal{T}$ is reachable from $\mathcal{S}$, *MRBUG* finds the target using n robots of size D and the path length l traveled by the robot which reached the target satisfies the quadratic inequality,

$$l \leq \frac{\pi}{D} \frac{\alpha^{n+1}}{\alpha^n - 1} l_{opt}^2 + ||\mathcal{S} - \mathcal{T}|| + 4\frac{A_0}{D}$$

where l_{opt} is the length of the optimal off-line path from $\mathcal{S}$ to $\mathcal{T}$. Note that the upper bound is scalable.

Proof: First consider the case where the initial search ellipse area is expanded at least once before the target is found. Suppose the search ellipse is expanded $i-1$ times until the target is reached (in ellipse i). The ellipses areas increase in each step by a factor of α. Let $A_{j'}$ denote the total area of the regions inspected by a robot from pair j' which reached the target after searching in k ellipses[1], then $A_{j'}$ is bounded by: $A_{j'} \leq A(j') + A(j'+n) + A(j'+2n) + ... + A(i)$
$A_{j'} \leq \alpha^{j'-1} A_0 + \alpha^{j'-1+n} A_0 + \alpha^{j'-1+2n} A_0 + ... + \alpha^{(i-1)} A_0$.
Substituting $i = j' + n(k-1)$ yields
$A_{j'} \leq A_0 \frac{\alpha^{j'-1}(\alpha^n)^k}{\alpha^n - 1}$. According to Lemma 5.3,
$\alpha^{(i-1)} A_0 < \frac{\pi\alpha}{4} l_{opt} \sqrt{l_{opt}^2 - ||\mathcal{S} - \mathcal{T}||^2} \leq \frac{\pi\alpha}{4} l_{opt}^2$.
Substituting this into the bound on $A_{j'}$ gives $A_{j'} < \frac{\pi}{4} \frac{\alpha^{n+1}}{\alpha^n - 1} l_{opt}^2$. Hence, the total length of the path traveled by the robot that reached the target is bounded by $l = l_{j'} + l_{j'+n} + l_{j'+2n} + ... + l_{j'+(k-1)n} = \sum_{i=1}^{k} l_{j'+(i-1)n}$
$\leq \frac{4}{D} A_{j'} + \sum_{i=1}^{k} \left(\left| \left| L_{e_{j'+(i-1)n}}^0 - \mathcal{T} \right| \right| - \left| \left| L_{e_{j'+in}}^0 - \mathcal{T} \right| \right| \right)$.
Substituting $L_{e_{j'}}^0 = \mathcal{S}$ and $L_{e_{j'+kn}}^0 = \mathcal{T}$ in the last results we get: $l < \frac{\pi}{D} \frac{\alpha^{n+1}}{\alpha^n - 1} l_{opt}^2 + ||\mathcal{S} - \mathcal{T}||$. Finally, the constant additive term $4A_0/D$ bounds the path length in case the target is reached from within the first search ellipse. ■

[1] k can be considered a global step. In the initial global step the n robots search in the first n consequent ellipses, in the next global step the robots search in the next n consequent ellipses and so on.

No. of robots $2n$	**Multiplication factor** α	**Relative performance** B_n
2	2	4
4	1.732	2.598
8	1.495	1.869
26	1.225	1.319
200	1.04	1.058

TABLE I

SOME α AND B_n VALUES CORRESPONDING TO n ENTRIES

The following lemma, inspired by [11], asserts that search area multiplying is indeed an optimal strategy.

Lemma 5.5: The competitive complexity of *MRBUG* is minimal when the multiplication factor is $\alpha = (n+1)^{1/n}$.

Proof: Let n be the number of robot pairs and α be the area multiplying factor in *MRBUG* execution. Suppose the path to the target was found by robot pair number j'. The total area $A_{j'}$ covered by each robot as obtained in Prop. 5.4 is $A_{j'} < \frac{\pi}{4} B_n l_{opt}^2$, where $B_n = \frac{\alpha^{n+1}}{\alpha^n - 1}$. Minimizing $A_{j'}$ for α and equating with zero while taking into account $\alpha > 1$ and $n \geq 1$, yields $\alpha = (n+1)^{1/n}$. A second derivative verifies the minimality of α, and thus of l'_j. ■

Corollary 5.6: MRBUG is complete.

Proof: The first important property established in Prop. 5.4, is that if the target $\mathcal{T}$ is reachable, *MRBUG* will find a path to it. The second property is that *MRBUG* will find that path in a finite and limited time and is deduced from the bound on the path length proved in Prop. 5.4. ■

We compared the performance of *MRBUG* with the performance of *CBUG*, an optimal algorithm using one robot. We calculated the upper bound on the path length l'_j of the robot that found the target in *MRBUG* for executions with various number of robots. First, when $n \to \infty$, α goes to 1. Thus, $l \leq \frac{\pi}{D} l_{opt}^2 + ||\mathcal{S} - \mathcal{T}|| + 4\frac{A_0}{D}$. On the other hand, for *CBUG*, $l_{CBUG} \leq \frac{6\pi}{D} l_{opt}^2 + ||\mathcal{S} - \mathcal{T}|| + \frac{6A_0}{D}$, which is 6 times longer.

Some more values of α and B_n for several cases of n are shown in Table I. When using one hundred robot pairs, B_n approaches one, and *MRBUG* multiplies the performance compared to execution with one robot by a factor of 5.67. Executing 4 robot pairs, *MRBUG* triples the performance.

Using the definitions of competitive complexity and the two bounds found previously, we can now establish a competitive complexity class for the problem solved by *MRBUG*.

Theorem 2: Quadratic competitive complexity class
The on-line multi-robot navigation problem belongs to the quadratic competitive complexity class.

Proof: A competitive complexity class, as defined in Def. 2, is formed from a lower and an upper bounds on the competitiveness of a task. According to Lemma 1, the lower bound of the problem discussed above is quadratic in l_{opt}. Since the upper bound of *MRBUG*, as found in Prop. 5.4, is quadratic in l_{opt}, too, this navigation problem belongs to the quadratic competitive complexity class. ■

The last theorem exhibits the quadratic competitiveness class of the problem solved by *MRBUG*. Since *MRBUG* has a quadratic competitiveness, *MRBUG* is optimal.

VI. *MRBUG* SIMULATION

MRBUG simulation was executed in an office like environment consisting of three rooms and a desk. Three pairs of robots were launched from the same point in the left room with a mission to reach the target which lies behind a desk in the rightmost room. The start and the target points form the focal points of all the ellipses, and the distance between them equals $2c$ which defines the ellipse parameter $c = 1753mm$. The semiminor axis is defined to be $b_0 = 615mm$ and thus the ellipse is completely defined, $a_0 = 1858mm$ and $A(0) = 3560mm^2$ which are calculated from c and b_i. The multiplication factor for $n = 3$ equals $\alpha = 1.587$.

The first step for each pair is depicted in Fig. 4(a). It can be observed that none of the robot pairs can reach the target in its initial step. The first pair to continue searching in the next ellipse is evidently robot pair No. 1., which moves to the fourth ellipse (Fig. 4(b)). In this step robot pair No. 1 reach the target, and the path length of each of the robots equals $l = 16168mm$. The optimal off-line solution is depicted in Fig. 4(b) and $l_{opt} = 5459mm$. Thus, $l = 5.4 \cdot 10^{-4} l_{opt}^2$.

A *CBUG* simulation in the same environment with one robot and $\alpha = 2$ took two ellipses multiplication prior to reaching the target and produced a path length of $33108mm$, which is about 2 times longer than the path length produced by *MRBUG*. However, it should be noted that the number of robots in *MRBUG* was six times greater.

VII. CONCLUSION

The problem of multi-robot navigation to a known target belongs to the quadratic competitive complexity class, i.e. its two classifying bounds, the universal lower bound of the problem and the upper bound of *MRBUG* are both quadratic in the optimal off line solution, l_{opt}. Hence, *MRBUG* algorithm is optimal, meaning that the path length to the target it produces might be improved only by the constant coefficients of its upper bound, since linear competitiveness is not achievable for such a problem. That optimality is obtained requiring only a constant amount of memory.

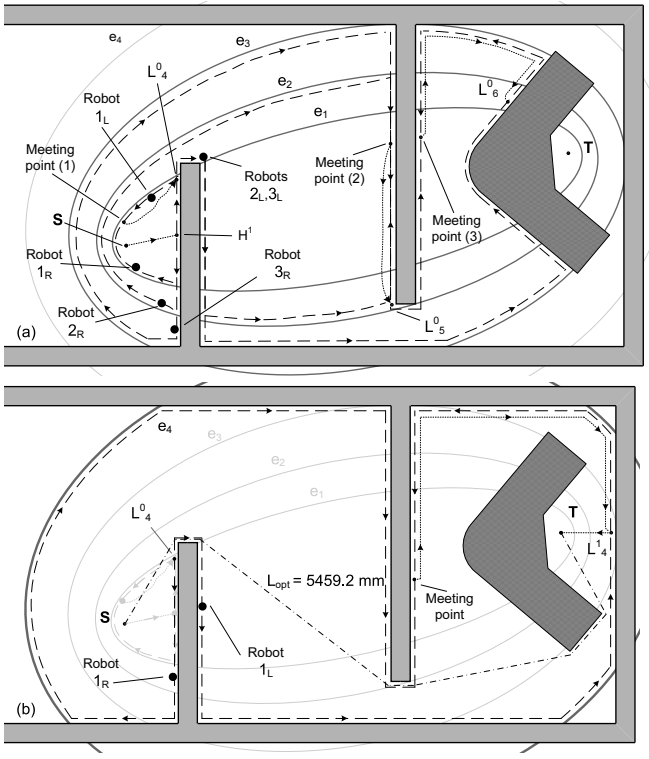

Fig. 4. The first step of the three robot pairs in *MRBUG* simulation (a). Last step of *MRBUG* simulation for pair no. 1. l_{opt} is denoted with a dash-dot line (b).The dense dashed lines denote a mutual path of a robot pair.

The following are some related open problems for further research. First, *MRBUG* uses limited vision or tactile senors. More advanced sensors such as vision and laser sensors do not have a significant advantage in highly congested environments. However, practical environments tend to be reasonably sparse, and an adaptation of *MRBUG* to such sensors is an important open problem. Second, the constant coefficients in the quadratic upper bound on *MRBUG* and in the quadratic universal lower bound differ by values of $\frac{3(1+\pi)^2}{4}\alpha^{n+1}$. Closing this gap is a major challenge that can yield new algorithms that possess the quadratic competitiveness of *MRBUG* but perform much better on average. Third, on-line disconnection [12], a criterion which judges an on-line algorithm according to the efficiency by which it determines that a solution does not exist, should be investigated for a group of robots.

REFERENCES

[1] D. Fox, W. Burgard, H. Kruppa, and S. Thrun, "Collaborative Multi-Robot Localization," in *Proc. of the 23rd German Conference on Artificial Intelligence*. Springer Verlag, 1999, pp. 325–340.

[2] C. Icking and R. Klein, "Competitive Strategies for Autonomous Systems," in *Modelling and Planning for Sensor Based Intelligent Robot Systems*, H. Bunke, T. Kanade, and H. Noltemeier, Eds. World Scientific, 1995, pp. 23–40.

[3] Y. Gabriely and E. Rimon, "CBUG: A Quadratically Competitive Mobile Robot Navigation Algorithm," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2005, pp. 2014–2019.

[4] D. T. Latimer, IV, S. Srinivasa, V. L. Shue, S. Sonne, H. Choset, and A. Hurst, "Towards Sensor Based Coverage With Robot Teams," in *Proc. IEEE Int. Conf. on Robotics and Automation*, May 2002, pp. 961–967.

[5] S. Koenig and Y. Liu, "Terrain Coverage with Ant Robots: A Simulation Study," in *AGENTS '01: Proceedings of the fifth international conference on Autonomous agents*. New York, NY, USA: ACM Press, 2001, pp. 600–607.

[6] D. Kurabayashi, J. OTA, T. Arai, and E. Yoshida, "Cooperative Sweeping by Multiple Mobile Robots," in *Proc. IEEE Int. Conf. on Robotics and Automation*, Minneapolis, Minnesota, April 1996, pp. 680–685.

[7] N. Hazon, F. Mieli, and G. A. Kaminka, "Towards Robust On-line Multi-Robot Coverage," in *Proc. IEEE Int. Conf. on Robotics and Automation*, May 2006, pp. 1710–1715.

[8] A. Datta and S. Soundaralakshmi, "Motion Planning in an Unknown Polygonal Environment with Bounded Performance Guarantee," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 1999, pp. 1032–1037.

[9] V. J. Lumelsky and A. A. Stepanov, "Path-Planning Strategies for a Point Mobile Automaton Moving Amidst Unknown Obstacles of Arbitrary Shape," *ALGORITHMICA*, vol. 2, pp. 403–430, 1987.

[10] S. Sarid, A. Shapiro, and Y. Gabriely, "MRBUG: A Competitive Multi-Robot Path Finding Algorithm," http://www.bgu.ac.il/ ashapiro/, Tech. Rep., January 2007.

[11] R. A. Baeza-Yates, J. C. Culberson, and G. J. E. Rawlins, "Searching in The Plane," *Information and Computation*, vol. 106, no. 2, pp. 234–252, 1993.

[12] Y. Gabriely and E. Rimon, "Competitive Disconnection Detection in On-Line Mobile Robot Navigation," in *Proc. Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2006.