# New Optimality Criteria for Movement Planning in a Group of Mobile Robots Navigated via Trilateration

**S. Genchev**

***Key Words:*** *Motion planning; trilateration; cooperative robotics*

***Abstract.*** *This paper presents an algorithm for motion planning in a team of cooperating mobile robots navigated using time-of-flight trilateration. The method uses some of the robots as stationary beacons guiding the robots in motion, thus enabling long-term working in unstructured environments. The main purpose of the planning is not building collision-free paths, but maintaining the positioning accuracy during the motion. Two important optimality criteria are considered, related to specific aspects of the common motion – how to plan trajectories with good movement precision, how to choose which robots to use as beacons and how to position them, in order to form appropriate geometrical arrangements.*

## 1. Introduction

The planning algorithms, and more precisely the motion planning, form an important field of studies in robotics. The main idea of planning a continuous trajectory is to avoid collisions in the working area while the robot is in motion [1,2,3]. But there is yet another important aspect which is the motion performance. Without such algorithms, a robot would not be able to interact properly with its environment and its movements would not be optimally calculated to save power and time. The background of the planning algorithms is very rich. It is a subject that has been studied since 18'th century with the beginning of the graph theory and the piano mover problem [4,5]. Basically, depending on the available sensor information, there are global and local planning techniques. The global methods are more optimal in terms of accuracy but require a priori full knowledge of the working area and obstacles [7,8]. The local ones compute the robot's sensor information in real time, but are subject of errors related to their global „blindness" [9,10]. Combining both local and global information is therefore the best choice for building good motion strategies [6,11,12].

The optimality criteria are conditions that should be satisfied in order to obtain the desired optimal trajectories. There are classical criteria like reaching minimum consumed energy, minimum working time or minimum travelled distance, which are all applicable in every situation. Besides these, there are also other very important criteria related to the precision, which vary largely depending on the task and the robot itself. No matter if it is a wheeled platform or a complex redundant manipulator - such criteria will be certainly imposed by limitations in the mathematical model and singularities in the inverse kinematics [13,14]. Planning a path which involves the robot in a state close to its limits or a singularity, would result in abnormal task execution or even damaging the robot. The task becomes a lot more complicated in the context of cooperative robotics because several robots are involved to work together [12,15,16].

In this work, a mobile robot, working in cooperation in a team, has to reach a distant target, using information about its location, acquired from other robots. Thus, the optimal motion of the considered robot involves not only finding the shortest path around obstacles, but also building trajectories with minimum localization error and maintaining appropriate robot formations during the team movement.

## 2. Cooperative Trilateration

Consider the case where there is a robot performing a task in an unknown environment. To make the situation worse, let us assume that there is no GPS accessibility in the area. Using its sensors, the robot may remain operational for some time tracking its position by dead-reckoning estimation techniques, but finally it will get lost without any external information. Now let us imagine the same case but this time considering a team of robots performing a common task. If a robot in the group is able to track the positions of other robots around it, then each member of the team could acquire external information for its position, complementing its own estimation by sensor fusion techniques. That would allow maintaining accurate position information. The trilateration [17,18] is a good choice as a localization technique in this case. The unknown robot position is calculated via several distance measurements between the robot and other points in the space with known coordinates called *beacons* or *stations*. Normally the required distances are measured using the time-of-flight (ToF) of a radio or acoustic wave, depending on the range requirements. The trilateration computation becomes complex problem when the positions of the stations are not known with certitude, but with some degree of uncertainty. Unfortunately this is exactly our case of cooperative robots, because some of the members in the team play the role of beacons, while others are in motion. On the other hand, one of the most important things when performing trilateration is to arrange the beacons in such configuration that would maximize the localization precision. In other words, when planning the motion of the group, one has to think not only for the optimality of the motion, but also when to move a robot and when to use it as a beacon. It is a global, long-term strategy in which a robot travels as long as its position could be localized precisely enough, given the present beacon topology, and when it stops to be in good position as a station. In order to evaluate these conditions, we need quantitative criteria.

If there are $n$ beacons and their positions are denoted as $R_i$, with $i$ being the index, then a point $R_0$ will be located at distance $l_i$ from each beacon. Measuring $l_i$ enable the determination of $R_0$ after resolving the following system of equations:

(1) $\|R_0 - R_i\| = l_i; \quad i = 1, 2, ..., n$

(2) $R_0 = \begin{bmatrix} x_0 & y_0 & z_0 \end{bmatrix}^T$

where $||R_0 - R_i||$ is an Euclidean norm. There are several techniques for resolving (1) when $R_i$ and $l_i$ are random variables with known statistical behavior [19]. We are going to use the least square method (LSM) because it yields an estimate for $R_0$ and its precision in real-time, without requiring any iterations. For the trajectory generation, the planning process involves computation of $R_0$ and its statistical properties in thousand of points. Therefore we are more interested in achieving rapidity than accuracy.

## 3. Real-time LSM Estimation

The method and its characteristics are explained in details in [20]. Firstly, system (1) is transformed in linear matrix form (3), which is then resolved by linear LSM (6):

(3) $\|R_0 - R_i\|^2 - \|R_0 - R_j\|^2 = l_i^2 - l_j^2$

(4) $\Rightarrow A_{n\times 3} R_0 = B_{n\times 1}$

(5) $A = \begin{bmatrix} x_1 - x_2 & y_1 - y_2 & z_1 - z_2 \\ x_2 - x_3 & y_2 - y_3 & z_2 - z_3 \\ \square & \square & \square \\ x_n - x_1 & y_n - y_1 & z_n - z_1 \end{bmatrix}; \quad B = \frac{1}{2}\begin{bmatrix} r_1^2 - l_1^2 - r_2^2 + l_2^2 \\ r_2^2 - l_2^2 - r_3^2 + l_3^2 \\ \square \\ r_n^2 - l_n^2 - r_1^2 + l_1^2 \end{bmatrix}$

(6) $\Rightarrow R_0 = A^* B$

(7) $A^* = \left(A^T A\right)^{-1} A^T$

where the matrix $A$ contains information about the relative position of the robots, and the matrix $B$ has the measured distances. The covariance matrix $P_0$, characterizing the precision of $R_0$, is computed by the linear error propagation law (8). It uses the covariance matrices of all beacons containing information about position and measurement errors (11), modified by the associated Jacobian matrices containing the topological information (9):

(8) $P_0 = \sum_{i=1}^{n} J_i P_i J_i^T$

(9) $J_i = (C_{i-1} - C_i) L_i$

(10) $L_i = \begin{bmatrix} x_0 - x_i & y_0 - y_i & z_0 - z_i & l_i \end{bmatrix}; \quad C_i = \begin{bmatrix} a_{1i}^* & a_{2i}^* & a_{3i}^* \end{bmatrix}^T$

(11) $P_i = \begin{bmatrix} \sigma_x^2 & \rho_{xy}\sigma_x\sigma_y & \rho_{xz}\sigma_x\sigma_z & 0 \\ \rho_{xy}\sigma_x\sigma_y & \sigma_y^2 & \rho_{yz}\sigma_y\sigma_z & 0 \\ \rho_{xz}\sigma_x\sigma_z & \rho_{yz}\sigma_y\sigma_z & \sigma_z^2 & 0 \\ 0 & 0 & 0 & \sigma_l^2 \end{bmatrix}_i$

where $C_i$ is the *i*-th column in the matrix $A^*$, and $L_i$ is a quantity containing information about both the measured distance $l_i$ and its calculated value. The terms $\sigma_x$, $\sigma_y$, $\sigma_z$ and $\sigma_l$ in the matrix $P$ describe the standard deviations of the beacon coordinates and of the measured distance to the robot, while $\rho$ denotes the correlation coefficient. Once we have a solution for $R_0$ after resolving (6), then both sets of vectors $C$ and $L$ (10) are already known.

Because of the errors in the beacon positions, the method needs more beacons than necessary in order to extract the useful information and diminish the output error. But their number is not important if the stations are in bad configuration. The quantitative value that gives the importance of a single beacon in the current topology is the geometrical weight $m_i$, computed by:

(12) $m_i = \|C_{i-1} - C_i\|$

The weight is a function of the actual geometrical arrangement only, and shall be constant until the beacons relocate. If that arrangement is good, the common weight of all beacons (full geometrical weight) would be smaller. It would be smaller also if more beacons are used to perform the trilateration. Another important property of the weight values is that they are used to analytically find the point in the space with minimum trilateration error, i.e. the best place for a robot to be localized:

(13) $R_{\min} = \left(\sum_{i=1}^{n} m_i \overline{P}_i\right)^{-1} \sum_{i=1}^{n} m_i \overline{P}_i R_i$

(14) $\overline{P}_i = \begin{bmatrix} \sigma_x^2 + \sigma_l^2 & \rho_{xy}\sigma_x\sigma_y & \rho_{xz}\sigma_x\sigma_z \\ \rho_{xy}\sigma_x\sigma_y & \sigma_y^2 + \sigma_l^2 & \rho_{yz}\sigma_y\sigma_z \\ \rho_{xz}\sigma_x\sigma_z & \rho_{yz}\sigma_y\sigma_z & \sigma_z^2 + \sigma_l^2 \end{bmatrix}$

As error estimate for motion planning we are using the trace of the output covariance matrix $P_0$ (8), also referred here as *full variance*. It has only one minimum $\sigma_{min}$ located in (13) and it is being governed by a very simple law of the form:

(15) $trace(P_0) = \sigma^2 = \sigma_{\min}^2 + (R - R_{\min})^T U (R - R_{\min})$

where $U$ is a *symmetric*, *positive-definite* matrix, depending only on the geometrical and statistical properties of the beacons. Thus being a constant, $U$ is easily derived using (8) in several points. The main advantage from (15) is the possibility to calculate an estimate for the localization precision in any point of the space around the robots, just by computing a simple quadratic equation. For a fixed value of the variance, the possible solutions for $R$ form an ellipsoid with center $R_{min}$.

## 4. Motion Planning

As we have seen in the previous section, the precision of the ToF trilateration is characterized by one maximum and degrade at a parabolic rate around it. Thus we are interested to navigate the robots close to that minimum. Therefore if their final target is far away, it is clear that we have to move the minimum position with the robots. It could be done if the beacons and the mobile robots are exchanging dynamically theirs roles.On the other hand, the precision is improved by placing the stations in

a favourable arrangement, which involves geometrical weight analysis. The question is how to plan the motion process in order to build good beacon configurations with low common weight and in the same time keep the trajectories precise, close to the error minimum. Avoiding collisions with other robots and obstacles is naturally required, but it is not the main idea and will not be discussed in details. To summarize, we have two main criteria to respect, one related to the movement precision and one topological. A robot has to be in motion as long as it is well localized by the stations, and when it stops and becomes a beacon, to find itself in a favourable position to localize other robots.

The planning algorithm that best fits our needs is the potential field method. In this planning model, there are attraction points such as the final motion objective and the precision maximum, and repulsing points as the other robots and the full geometric weight maximum. The sum of all repulsive and attractive forces $f_i$ at step $k$ forms the resulting force $f$ (16) moving the object in motion in the optimal direction $u_k$ (17). The optimal step length $h_k$ (18) is adaptive to the desired trajectory smoothness, given by the scalar product of two succeeding steps. The step reduction $s$ (19) is triggered by reaching a threshold value $T_{smooth}$ for the smoothness (which also implies recalculation of the last step being incorrect). Thus it is possible to choose a large initial step size and speed up the algorithm.

$$(16)\quad f(R_k) = \sum_i f_i(R_k)$$

$$(17)\quad u_k = \frac{f(R_k)}{\|f(R_k)\|}$$

$$(18)\quad h_k = sh_{k-1}$$

$$(19)\quad s\begin{cases} = 1, & u_k u_{k-1} \geq T_{smooth} \\ < 1, & u_k u_{k-1} < T_{smooth} \end{cases}$$

The main drawback of the potential field method is the local potential minimums that trap the trajectories. In our case, these minimums are not a problem, but contrary they are secondary targets that trigger the changing of beacon roles. This is the first phase of the motion algorithm – to retrieve a potential minimum where to switch the roles of the moving robot and a chosen station. It is a point where the precision and topological criteria are in balance. Once we have it, we run a standard path finding algorithm to reach it, while avoiding obstacles in the way.

In our scenario, there are three concurrent forces $f_i$ driven by different optimality criteria – a target force (shortest path criterion), a topological force (good beacon configuration criterion) and a precision force (low movement error criterion).

## 4.1. Target Force

The final point to be reached is associated with a force field with direction pointing at it and a constant magnitude (*figure 1*). We will denote this force as $f_T$ and the target point as $R_T$. If there are no constraints, a robot in motion will travel to the target in a straight line. The target force magnitude is a reference for all other forces. Therefore its value is of no importance since we are interested only in the direction of the resulting force. For computational convenience we will use unity (20). The directional vector for $f_T$ at point $R$ is given at (21):

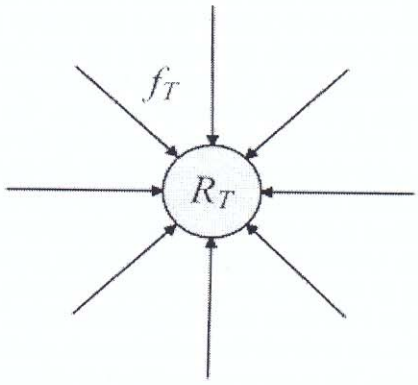



**Figure 1.** Direction of the target force

$$(20)\quad \|f_T\| = 1$$

$$(21)\quad \vec{u}_T = \frac{R_T - R}{\|R_T - R\|} = -\frac{R - R_T}{\|R - R_T\|}$$

## 4.2. Topological Force

Before running the first phase of the planning algorithm, we have to decide which of the beacons will be the next moving robot, because the topological analysis is performed regarding that robot. When the moving robot $i$ becomes a station it has to form a good configuration to navigate the new moving robot $j$. The full geometrical weight (FGW) analysis regarding robot $j$ gives us a way to resolve that task. The direction opposite to the FGW gradient is the direction of the decreasing FGW. Therefore, the force $f_M$ associated with it will try to improve the beacon arrangement.

For this purpose we need the FGW gradient regarding the position variations of $j$. If we note by $M$ the FGW value then:

$$(22)\quad \nabla_j M = \sum_{i=1}^{n} \nabla_j m_i$$

$$(23)\quad \nabla_j m_i = \left(\delta_{ij} I_{3\times 3} + \varepsilon_{ij} D_{3\times 3}\right)\left(C_{i-1} - C_i\right)$$

$$(24)\quad \delta_{ij} = \left(C_{i-1} - C_i\right)^T \left(C_{j-1} - C_j\right)$$

$$(25)\quad \varepsilon_{ij} = e_{ij} - e_{ij-1} - e_{i-1j} + e_{i-1j-1}$$

$$(26)\quad E^A_{n\times n} = AA^* - I$$

$$(27)\quad D^A_{3\times 3} = \left(A^T A\right)^{-1}$$

where $e_{ij}$ are the elements of the matrix $E$ (26). The matrices $E$ and $D$ are both symmetric and depending only on the station arrangement. The inverse direction of the FGM gradient (22) gives us the required directional vector:

$$(28)\quad \vec{u}_M = -\frac{\nabla_k M}{\|\nabla_k M\|}$$

Using the above directional vector satisfy the condition to have good beacon configuration for the next beacon triggering. But in terms of optimality the solution is not good enough because the resulting force will certainly draw a longer trajectory trying to pass around the full mass maximum. The important thing here is not to seek lower full mass during the search, but in the potential minimum at the end. That's why we will rather choose the same direction as the target force while climbing the FGW maximum curve and the force will help the target force to pull strongly when in bad beacon configuration, preventing it to trigger beacon changing in that moment. Afterwards, when descending the FGW slope down towards the target, switch to gradient direction to deviate the motion towards a better configuration location. The scalar product of the two directions is used to give a criterion for choosing the moment of the changing. After retrieving the force direction, we need its magnitude.

We suppose that the initial beacon configuration is good enough, or at least it should not get any worse if possible. That is why we choose the force to be equal to the target force for the FGW of the starting configuration, which we shall denote it as $M_0$. On the other hand, for each beacon there is a position where the full geometrical weight is at its maximum level $M_{max}$. That means the worst possible configuration. Therefore we have to avoid it by giving infinite power to the force in that point and swift rate of increasing when close to it. Given that information, we shall define the magnitude of $f_M$ as follows:

$$(29)\quad \|f_M(M)\| = \|f_T\|\left[\frac{M}{M_0}\left(\frac{M_{\max}-M_0}{M_{\max}-M}\right)\right]^p$$

The tuning parameter $p$ is added only to modify the rate of change of the topological force if needed. The derivative of $f_M$ at $M_0$ is thus a simple function of $p$:

$$(30)\quad \left.\frac{\partial\|f_M\|}{\partial M}\right|_{M_0} = \frac{p\|f_T\|M_{\max}}{M_0(M_{\max}-M_0)} \xrightarrow{M_{max} \gg M_0} \frac{p\|f_T\|}{M_0}$$

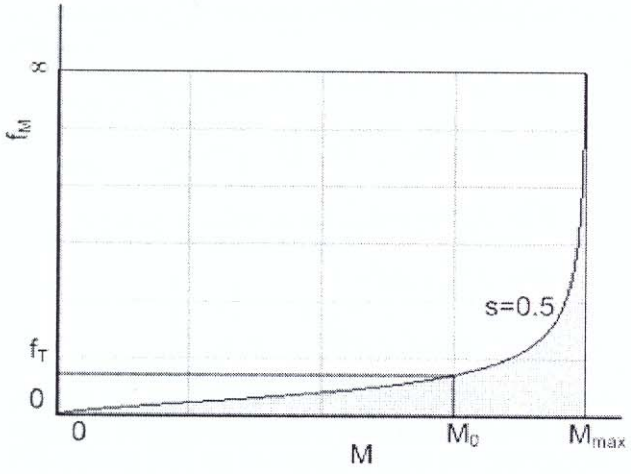



**Figure 2.** Magnitude of the topological force

The FGW has no minimum (the best configuration is infinitely large) but normally it has only one maximum, the position of which could be numerically identified. This maximum may be found by searching for the zero of the FGW gradient, but its expression is too complex to be resolved directly. The computation of the Hessian matrix is also very computationally costly, and the fast Newton method cannot be used here. However the gradient may be a valuable tool for an iterative search of the maximum. As a vector, the gradient points at the steepest slope. So we need a starting position from where we are going in the direction of the gradient some distance $h$. From the newly obtained position, we recalculate the gradient and perform a new step:

$$(31)\quad R_{k+1} = R_k + h_k \frac{\nabla_j M(R_k)}{\|\nabla_j M(R_k)\|}$$

The length of $h$ has to decrease as $R$ gets close to the solution. A possible method to do that, is by modifying the step size by a factor:

$$(32)\quad h_{k+1} = a_k h_k$$

$$(33)\quad a_k \begin{cases} 1, & M(R_{k+1}) > M(R_k) \\ a_r < 1, & M(R_{k+1}) \le M(R_k) \end{cases}$$

The iterations continue until some convergence criterion is reached. It may be translated by reaching a minimal step size or gradient value. The minimal number of iterations for reaching a minimal step $h_{min}$ is:

$$(34)\quad k_{min} = \frac{\log(h_0/h_{\min})}{\log(1/a_r)} \Rightarrow k > k_{min}$$

The size of the initial step is important. If it is not enough large, the algorithm will proceed slowly, performing a lot of unnecessary iterations. To avoid this effect, a possible improvement would be to increase the step if the first condition above is held for several iterations. So a third condition for the step growing factor may be added:

$$(35)\quad a_k = a_g \ge 1, \quad \underbrace{M(R_{k+1}) > M(R_k) > \ldots > M(R_{k-m+1})}_{m \text{ conditions}}$$

The step reduction and growing factors $a_r$ and $a_g$ have to be adjusted according to the application. The reduction occurs when the algorithm misses to perform a step, while the growing is applied when it scores for several succeeding iterations. The required number of succeeding iterations before step growing is important for optimal performance. After a miss and a step reduction, it is probable that the solution lies in the close vicinity. Therefore the next growth has to occur after travelling at least the distance of the miscalculated step:

$$(36)\quad h_k \le n h_{k+1} = n a_r h_k \Rightarrow n \ge \frac{1}{a_r} \Rightarrow n = 1 + \text{int}\left(\frac{1}{a_r}\right)$$

Because the algorithm has to be performed repeatedly, at every beacon role change, the initial step size is supposed to be the maximum successful step from the previous realization. A good choice for the starting position is the error minimum location $R_{min}$ because in most configurations it is relatively close

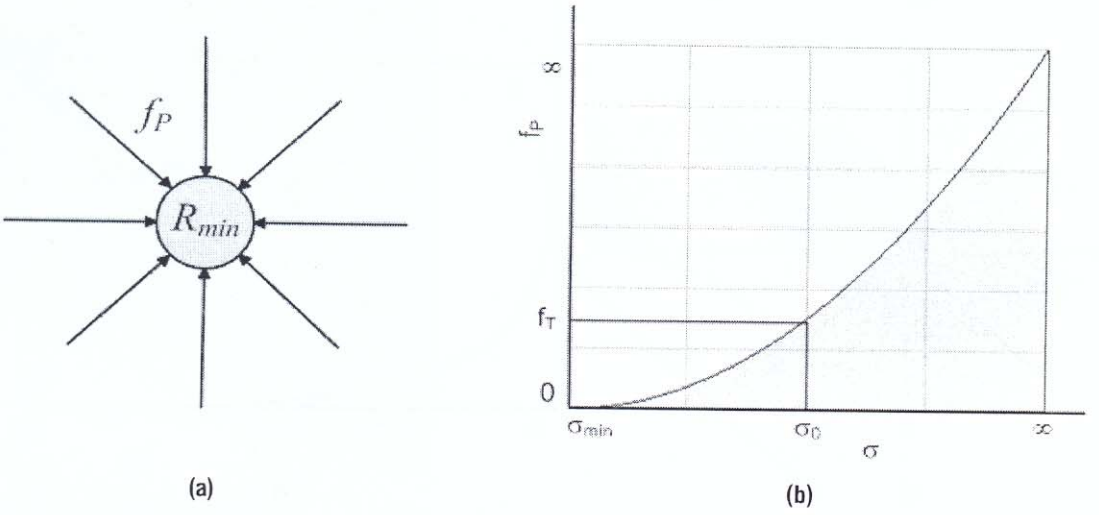



**Figure 3.** Direction (a) and magnitude (b) of the resulting force

to the full mass maximum location. Unfortunately, for some configurations of few beacons the FGM slope may not be steep and the number of iterations for reaching convergence could be considerable. But in fact we do not need the exact full mass maximum, but only a good idea for it. To do that simplification, the final step should not be taken too small. Instead, the convergence criterion has to cover the gradient size also, and thus to abort the iterations once the slope is flat enough. In this case however it is possible while navigating the moving robot to have full mass values greater than the computed maximum. These locations would be considered worst configurations and the optimal direction will be given entirely to the FGW force without any module calculations.

### 4.3. Precision Force

From (15) we have that the precision distribution in the space is parabolic, with one maximum location $R_{min}$ computed by (13). The associated force $f_p$ will try to push the moving robot towards that point (*figure 3*), in order to minimize the trilateration localization errors:

$$(37)\quad \vec{u}_P = \frac{R_{\min} - R}{\|R_{\min} - R\|} = -\frac{R - R_{\min}}{\|R - R_{\min}\|}$$

Its magnitude, similarly to the topological force $f_M$, will start with a value equal to the target force $f_T$ at the starting position, where the variance level is $\sigma_0$. The rate of change of the force magnitude will then follow the variance level:

$$(38)\quad \|f_P(\sigma)\| = \|f_T\| \frac{\sigma^2 - \sigma^2_{\min}}{\sigma_0^2 - \sigma^2_{\min}}$$

### 4.4. Resulting Force

The forces of the target point, the station topology and the trajectory precision form together the resulting force of the first phase of the method.

$$(39)\quad f_1(R) = f_T + f_M + f_P$$

Its direction is used as direction for the iterative search of the potential minimum, while the magnitude is being monitored for detection of the minimum when decreasing below some threshold value. Eventually the target location will be reached during some of these searches, and the method will reach its goal.

If the starting point is $R_S$ then the initial resulting force would be:

$$(40)\quad f(R_S) = -\|f_T\| \left( 2\frac{R_S - R_T}{\|R_S - R_T\|} + \frac{R_S - R_{\min}}{\|R_S - R_{\min}\|} \right)$$

The size of this initial resulting force may be used for determining which robots have to be in motion and which ones to serve as beacons. Obviously the robots that are located at the far side of the error minimum will have priority. Another simpler way of classification is to prioritize these robots which are localized the most far away from the target, but in most cases this will yield the same result.

### 4.5. Obstacle Force

In the second phase, after locating the potential minimum, a direct movement trajectory has to be constructed for reaching that new target. Here we are going to implement a simple potential field algorithm with no obstacles other than the robots themselves. For that purpose the obstacle robots will be represented by spheres with a certain radius $r_c$ and a centre $R_c$. The radius has to include the obstacle dimensions, the position uncertainties and the size of the moving robot which will be modelled as a point. We shall refer to these spheres as critic regions. The associated obstacle force $f_c$ has a direction going out of the sphere, trying to move away the passing robots in motion (*figure 4*). Therefore its direction at point $R$ is:

$$(41)\quad \vec{u}_c = \frac{R - R_c}{\|R - R_c\|}$$

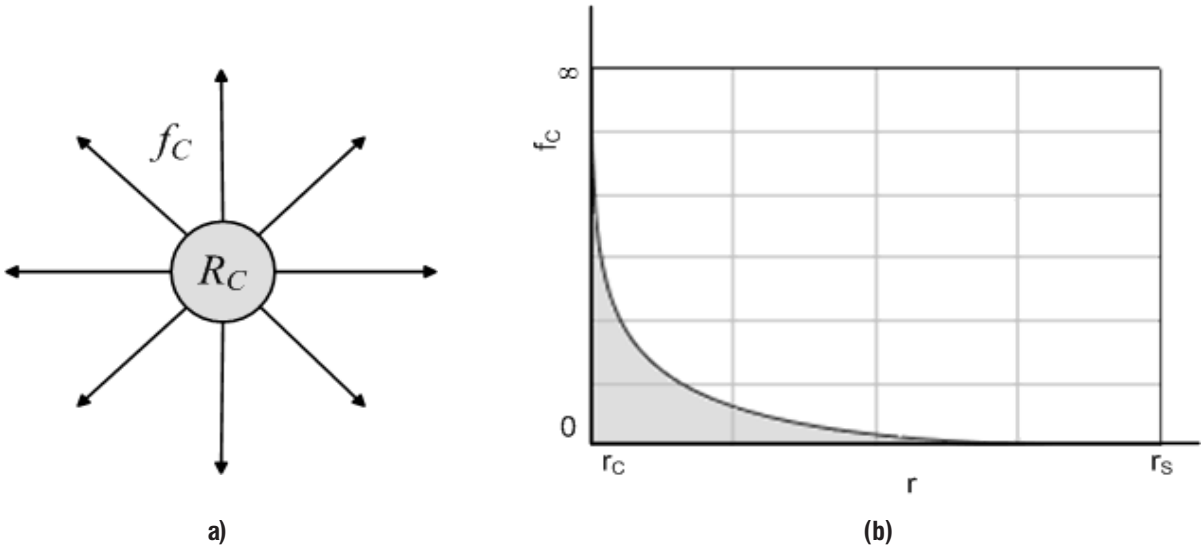



**Figure 4.** Direction (a) and magnitude (b) of the obstacle force

In any case, a robot must not enter in collision with the spheres, so the magnitude at the sphere edges should be infinity. On the other hand, as the robot is at a safe distance from the obstacle, the force should be null. So we define a safe boundary, a second sphere with radius $r_s$ around the obstacle, where the force is active. The space between the two spheres is the zone where the force is active and thus we will refer to it as *active region*, while the outside where the force is null – *safe region* (*figure 5*).

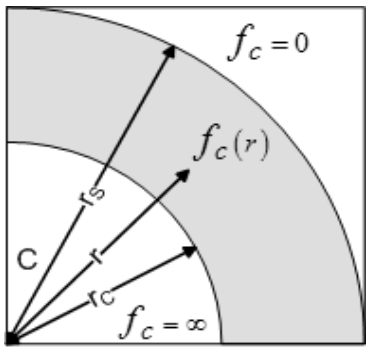



**Figure 5.** Critic, active and safe regions of the obstacle force

We have to choose the force magnitude in this active region in such a way that at $r_c$ it is infinity and increasing at infinite rate, while at $r_s$ it is zero and increasing at zero rate. The rate conditions are important in order to obtain smooth transition between the regions. Since the planning process is a discrete process, it is possible to enter in collision with an obstacle if the step size is considerably large compared to the active region. To avoid this, a third distance $r_t$ is defined between $r_c$ and $r_s$, where the obstacle force is equal to the target force. Its value is chosen to be bigger than the critic radius with at least one step, say $N_c$ steps. On the other hand, to avoid sharp directional changes in the motion plan, $r_t$ has to be smaller than the safe radius with several steps, say $N_s$. However if in some realizations we obtain a collision event, the step size $h$ (18) has to be reduced.

Taking into consideration the above conditions, a possible solution for the magnitude of $f_c$ at radius $r$ is:

$$r_t - r_c = N_c h, \quad N_c \geq 1 \tag{42}$$

$$r_s - r_t = N_s h, \quad N_s \geq N_c \tag{43}$$

$$\|f_c(r)\| = \|f_T\| \frac{r_t - r_c}{(r_s - r_t)^2} \frac{(r_s - r)^2}{r - r_c} = \begin{cases} +\infty, & r = r_c \\ \|f_T\|, & r = r_t \\ 0, & r = r_s \end{cases} \tag{44}$$

$$\frac{\partial \|f_c\|}{\partial r} = \|f_T\| \frac{r_t - r_c}{(r_s - r_t)^2} \left[ 1 - \left( \frac{r_s - r_c}{r - r_c} \right)^2 \right] = \begin{cases} -\infty, & r = r_c \\ 0, & r = r_s \end{cases} \tag{45}$$

$$r = \|R - R_c\| \tag{46}$$

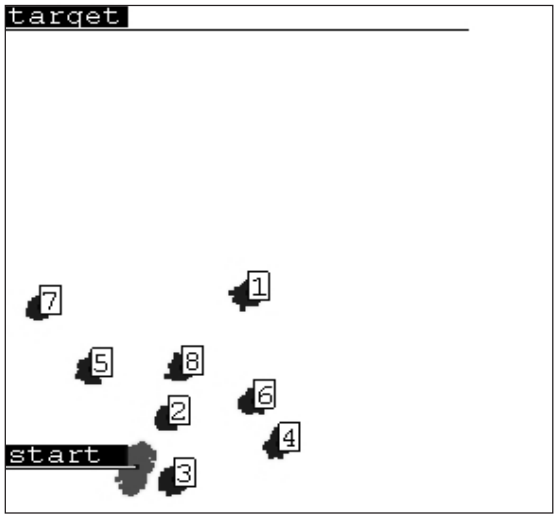
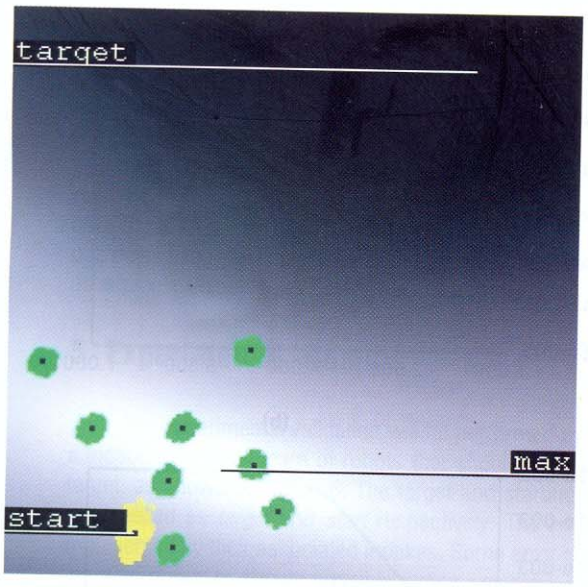
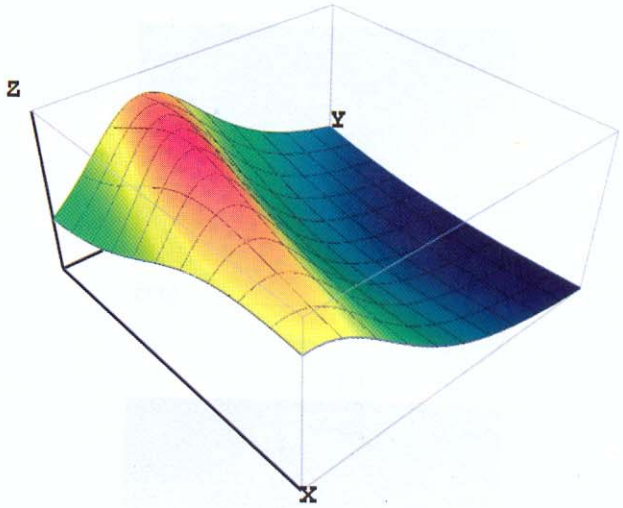


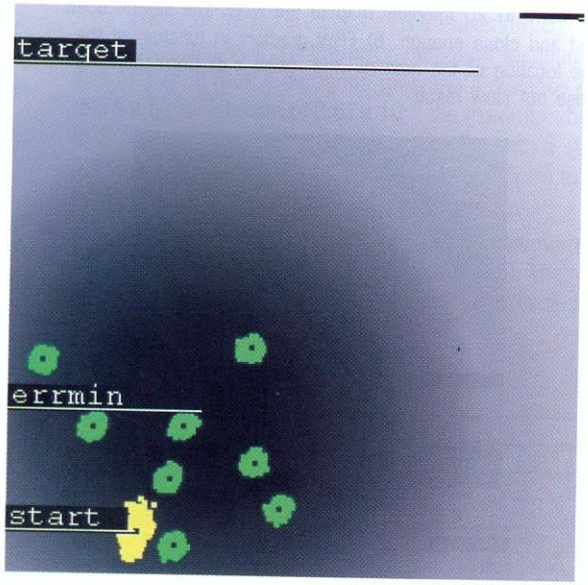

**Figure 6.** A small team of robots in movement towards a target location in a square working area



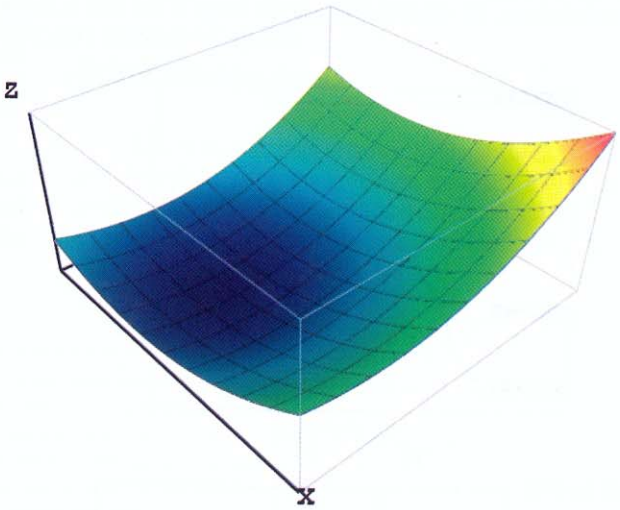

(a)




(b)

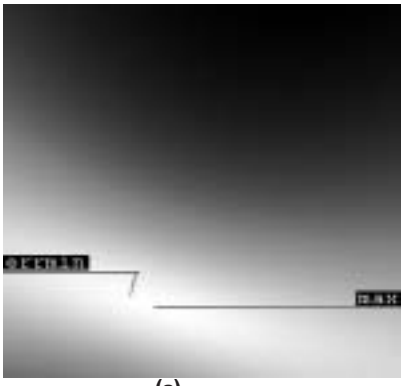



(c)

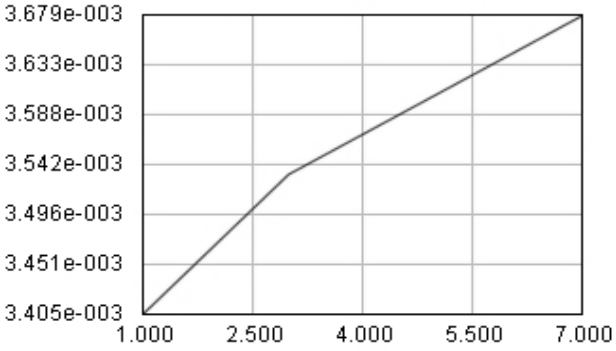



(d)

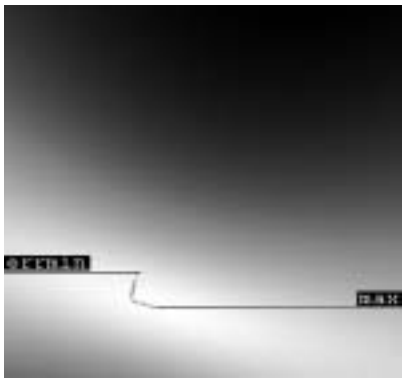
**Figure 7. a,b)** 2D intensity map and 3D image of the FGW in function of the location of the next moving robot; **c,d)** 2D intensity map and 3D image of the error levels in function of the position of the actually moving robot



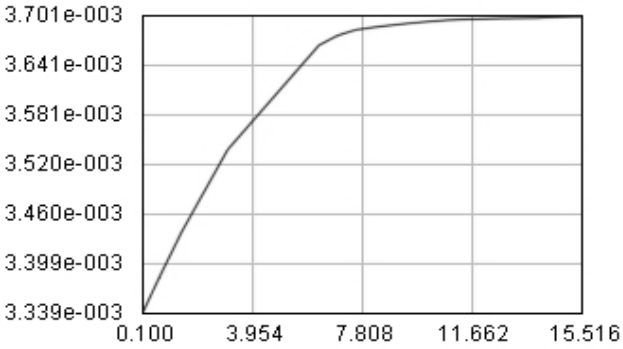
(a)




(b)

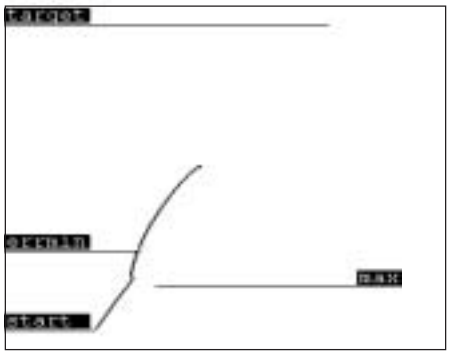


(c)

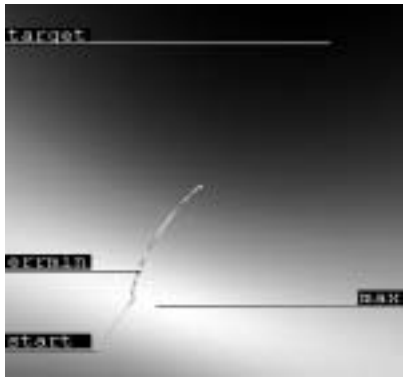



(d)

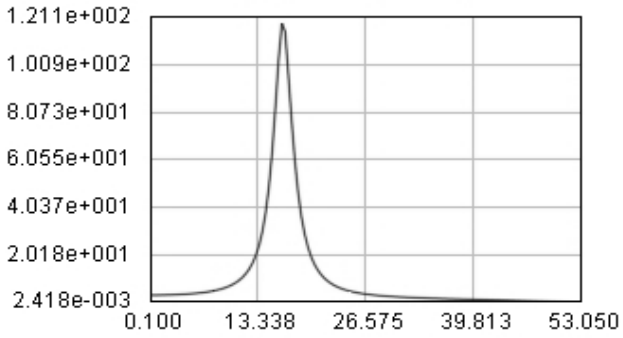
**Figure 8.** The FGW maximum is searched iteratively starting from the error minimum: **a)** 2D intensity map of the FGW. The search does not reach the location of the maximum with large starting step, but is very fast and close enough. **b)** FGW gradient in function of the search distance. 2D intensity map and 3D image of the FGW in function of the location of the next moving robot; **c,d)** Corresponding data for small starting step, yielding precise but slow result



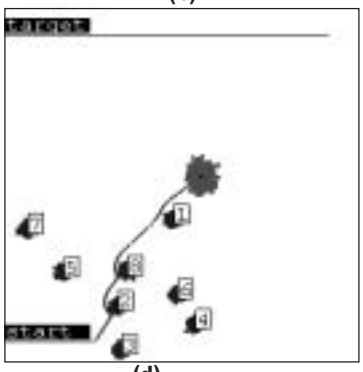

(a)




(b)




(c)




(d)

**Figure 9. a)** Searching the potential minimum where the changing of beacon take place; **b)** Corresponding intensity map of the FGW; **c)** Evolution of the resulting force in function of the searched distance. The peak value corresponds to the topological force direction changing; **d)** Building a collision-free trajectory toward the potential minimum

The active region size may be chosen accordingly to a desired rate for the force at $r_t$. Initially, the known obstacles are the other robots in the working area and the obstacles from the eventually available map information. The other obstacles are perceived by proximity sensors and are added in the algorithm data base as the robot is in motion. To avoid potential traps, obstacles that are too close to each other have to be remodeled as a single bigger sphere.

## 5 Experimental Results

The experimental validation will be performed by planning a motion of a small team of robots that should reach a distant target, as shown on *figure 6*. The target and starting locations are labelled as *target* and *start* respectively. The beacons are represented by their associated indexes. Some error realizations are also drawn to shown possible wrong positions around the real ones. All setup parameters in use are given i*n table 1*.

The first step is to define the next moving robot, which will continue the motion once the actual active robot stops and become station. In this case it is beacon 3, with the biggest initial resulting force. Now we need to find the FGW maximum location in regard to that beacon. On *figure 7* are presented the 3D image and it corresponding 2D intensity graph for the FGW and the full variance levels.

The error minimum and the FGW maximum are close, thus the search begins from the error minimum location with step size of 1 m. It took 3 iterations (passing a distance of 7 m) to reach a good estimate of $3{,}679.10^{-3}$, the maximum being equal to $3{,}701.10^{-3}$ (*figure 8 a, b*). When starting with a step size of 0.1 m, 30 iterations are necessary (passing 15 m) to reach the maximum with negligible error (*figure 8 c, d*).

When we have the value of the FGW maximum we run a search for the potential minimum (*figure 9* a, b). The topological force direction triggering is clearly from the evolution of the gradient value (*figure 9* c). Once the FGW hill climbed, it starts to follow the slope down. The precision force however is beginning to pull it back when getting too far from the error minimum until reaching the potential minimum. The final run is to build a trajectory to its location with collision detection (*figure 9 d*).

Now the first beacon changing makes beacon 3 a moving robot, to be navigated by the new arrangement of stations. The topological force guarantees that the new station forms a good arrangement for guiding the new mobile robot, while the precision force limits the errors in the position of the new station. Both The motion planning algorithm is about to begin again from the beginning. It will loop trough the robots, changing their roles until the target location is not reached by one of them. On *figure 10* are presented all the trajectories, including the first one, until the target is finally reached after several runs.

## Conclusion

This paper briefly presented an efficient and yet simple algorithm for motion planning in a team of cooperating mobile robots navigated using time-of-flight trilateration. According to simple criteria, based on properties of the linear trilateration model, the method dynamically decides when to move a robot and when it is convenient to use it as a beacon for localizing the other members of the team. The analytical work has been validated by computer simulation of a small team of mobile robots and proven to be operational.

## Acknowledgements

This work was supported by National Ministry of Science and Education of Bulgaria under Contract BY-I-302/2007: „Audio-video information and communication system for active surveillance cooperating with a Mobile Security Robot“.

**Table 1.** Setup parameters

| | |
|---|---|
| Number of robots | 9 |
| Number of beacons | 8 |
| Working area | 100x100 m |
| Position standard deviation | 1 m |
| Dimensional correlation factor | 0.2 |
| Measurement standard deviation | 1 m |
| Multi-path error | ignored |
| Minimal iteration gradient module | 0.001 |
| Starting iteration step size for FGW maximum search | 1 m |
| Reduction factor for FGW maximum search | 0.1 |
| Growing factor for FGW maximum search | 2 |
| Reduction factor for the potential minimum search | 0.5 |
| Starting iteration step the potential minimum search | 1 m |
| Minimal distance for the potential minimum search | 0.1 m |
| Critical obstacle radius | 1 m |
| Active obstacle region | 6 m |
| Equality obstacle radius | 3 m |

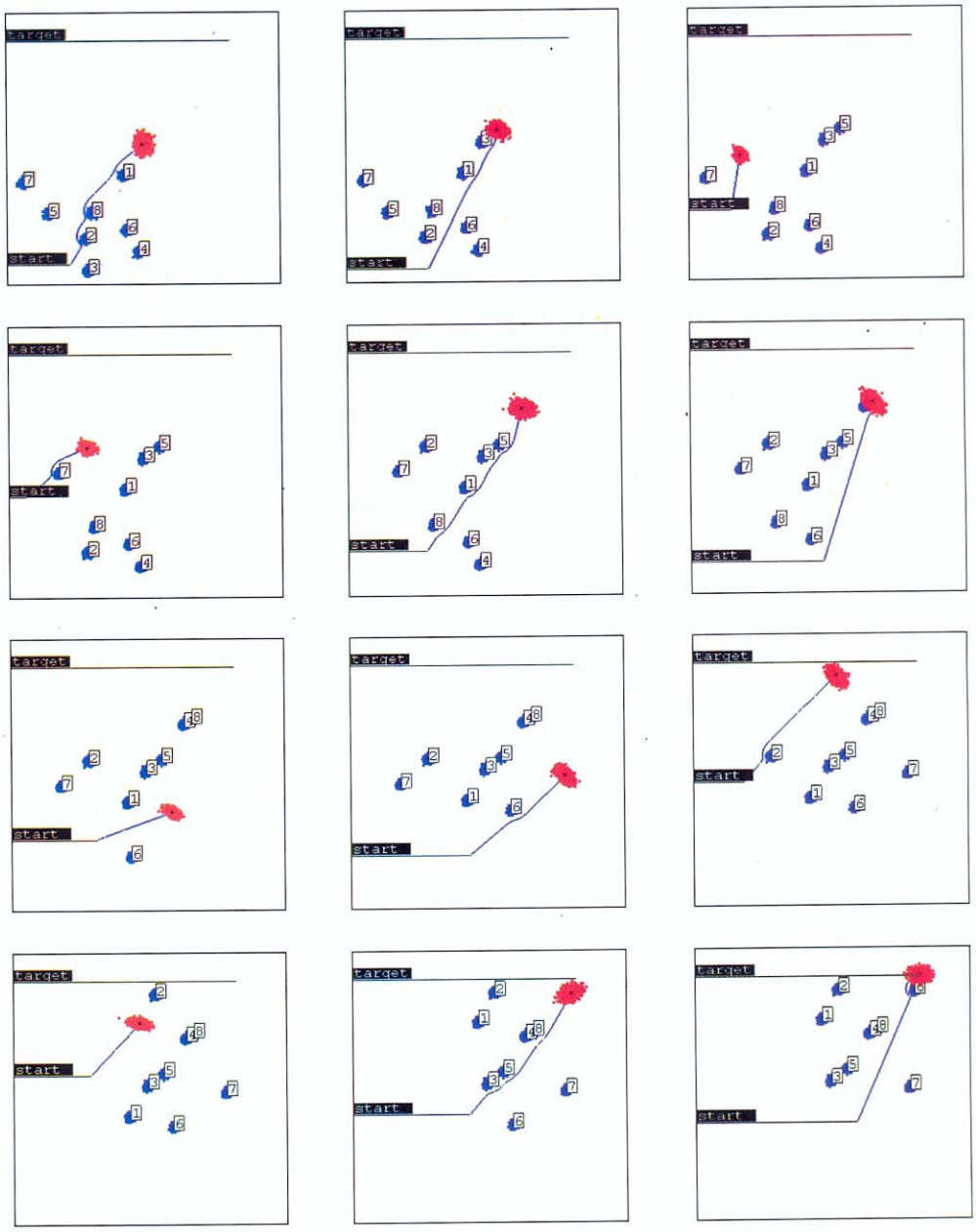



**Figure 10.** Succeeding trajectories until the target is reached

## References


1. Lumelsky, V. L. Sensing. Intelligence, Motion: How Robots and Humans Move in an Unstructured World. John Wiley & Sons, Inc, 2006, ISBN: 0-471-70740-6.

2. Schwartz, J., J. Hopcroft, M. Sharir. Planning, Geometry, and Complexity of Robot Motion. Ablex Publishing Corporation, Norwood, NJ, 1986, ISBN: 0893913618.

3. LaValle, S. M. Planning Algorithms. Cambridge University Press 2006, ISBN: 0-521-86205-1.

4. Reif, J. Complexity of the Mover's Problem and Generalizations. 20th Symposium of the Foundations of Computer Science, 29-31 Oct. 1979, 421-427, ISSN: 0272-5428.

5. Schwartz, J., M. Sharir. On the Piano Mover's Problem II. General Techniques for Computing Topological Properties of Real Algebraic Manifolds. – *Advances in Applied Mathematics*, 4, 1983, 298-351.

6. Diéguez, A. R., R. Sanz, J. López. Deliberative On-Line Local Path Planning for Autonomous Mobile Robots. – *Journal of Intelligent and Robotic Systems*, 53, October 2008, 2, 145-168, ISSN: 0921-0296.

7. Barraquand, J. , L. Kavraki, J.-C. Latombe, T.-Y. Li, R. Motwani, and P. Raghavan. A Random Sampling Scheme for Path Planning. – *International Journal of Robotics Research*, 16, 1997, 759-744.

8. Hart, P. E., N. J. Nilsson, B. Raphael. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. – I*EEE Transactions on Systems Science and Cybernetics*, 4, July 1968, 2, 100-107, ISSN: 0536-1567.

9. Ge, S. S., Y. J. Cui. Dynamic Motion Planning for Mobile Robots Using Potential Field Method. – *Autonomous Robots*, 13, November 2002, No 3, 207-222.

10. Huang, D. S., L. Heutte, M. Loog. Artificial Potential Field Based Path Planning for Mobile Robots Using Virtual Water-Flow Method. Advanced Intelligent Computing Theories and Applications. With Aspects of Contemporary Intelligent Computing Techniques, 2, 2007, 588-595, ISBN: 978-3-540-74281-4.

11. Geraerts, R., M. H. Overmars. The Corridor Map Method: Real-Time High-Quality Path Planning. IEEE International Conference on Robotics and Automation, Roma, Italy, 10-14 April 2007.

12. Kamphuis, A., M. Overmars. Finding Paths for Coherent Groups Using Clearance. Eurographics Symposium on Computer Animation, 2004, 19-28.

13. Gracia, L., J. Tornero. Optimal Trajectory Planning for Wheeled Mobile Robots Based on Kinematics Singularity. – *Journal of Intelligent and Robotic Systems*, 37, May 2003, 1, 1-19, ISSN: 0921-0296.

14. Vidolov, B., S. Genchev. Heuristic Approaches for the Coordinated Motion of a Redundant Heavy-Duty Hydraulic Excavator. International Symposium on Intelligent Control, 13th Mediterranean Conference on Control and Automation, June 27-29, 2005.

15. Correll, N., A. Martinoli. Collective Inspection of Regular Structures using a Swarm of Miniature Robots. Proc. of the Ninth Int. Symp. on Experimental Robotics ISER-04, Singapore 2006, Springer Tracts in Advanced Robotics 6, 21, 2006.

16. Spletzer, J., A. K. Das, R. Fierro, C. J. Taylor, V. Kumar, J. P. Ostrowski. Cooperative Localization and Control for Multi-Robot Manipulation. Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems 2001, 2, 2001, 631-636, ISBN: 0-7803-6612-3.

17. Thomas, F., L. Ros. Revisiting Trilateration for Robot Localization. – *IEEE Transactions on Robotics*, 21, Feb. 2005, 1, 93-101.

18. Manolakis, D. E. Efficient Solution and Performance Analysis of 3-D Position Estimation by Trilateration. – *IEEE Transactions on Aerospace and Electronic Systems*, 32, Oct. 1996, 4, 1239-1248, ISSN: 0018-9251.

19. Genchev, S. Cooperative Localization by Time-Of-Flight Trilateration – Fast, Robust or Optimal. – J*ournal of Information Technologies and Control*, 2008, No 2, ISSN: 1312-2622.

20. Genchev, S., P. Venkov, B. Vidolov. Trilateration Analysis for Movement Planning in a Group of Mobile Robots. 13th International Conference on Artificial Intelligence: Methodology, Systems, Applications AIMSA 2008, Varna, Bulgaria, September 4th - 6th, 2008, 353-64, ISSN: 0302-9743.



**Manuscript received ot 15.04.2009**



***Svetoslav Genchev*** *received his engineering degree from the Technical University of Sofia in 2005, currently is preparing for defending his PhD Thesis in navigation of cooperative robots. His work was mainly helped by a French scholarship program allowing him to perform some of his research in the Technological University of Compiegne, France.*

*Contacts:*
*Svetoslav Genchev*
*French Language Faculty of Electrical Engineering,*
*Technical University of Sofia,*
*8, Kliment Ohridski St. Sofia-1000, BULGARIA*
*Phone: +359 2 965 2379, e-mail: genchevs@abv.bg*