A New Decentralized Planning Strategy for Flocking of Swarm Robots

Ximing Liang, Xiang Li
School of Information Science and Engineering, Central South University, Changsha 410083, China

Abstract—In this paper, we present a new decentralized planning strategy, named Triangular Formation Algorithm (TFA), for swarm robots. This strategy is expanded to large scale swarm by utilizing an appropriate neighbour selection method. For swarm obstacle avoidance, a simplified artificial physical model is introduced to work with the TFA. Simulation results showed that the swarm behaviours such as aggregation, flocking and obstacle avoidance in an unknown environment are achieved effectively.

***Index Terms*—swarm robotics; decentralized planning; virtual physics; geometric approach**

I. INTRODUCTION

Swarm robotic studies deal with the problem of coordinating large number of simple robots to collectively perform a task. The system functional properties appearing from social bio-systems such as robustness, flexibility and scalability are one of the most important objectives to achieve in swarm robotics [1]. During the last decade, the development in hardware components in robotics such as sensors, microcontrollers, actuators, communication devices etc., made the robots affordable. The idea of deploying large number of low cost robots for a particular task began to appear in various engineering applications [2], [3], [4], [5].

Flocking behavior, which can be commonly seen in nature, is an important swarm behavior as the behavior has many potential applications in practice of engineering such as mobile sensor networking, surveillance, and search and rescue [1]. Flocking strategies can be classified into centralized and decentralized strategies. The latter is more attractive to research communities since this kind of strategies is scalable to swarms. We also concern about decentralized strategy in this study. Recently, various decentralized strategies were published in order to achieve the desired flocking/formation behavior. These approaches are typically inspired from natural phenomena related to physical systems or social animals. Balch and Hybinette [6], [7] presented a behavior-based approach to scalable robot formation which is primarily inspired from the way of molecules forming crystals. Robots only rely on local sensing, and there is no communication between them. Kim et al. [8] presented a set of analytical guidelines for designing potential functions to avoid local minima for a number of representative scenarios. Spears et al. [9], [10] defined a distributed framework, "Artificial Physics (AP)", for controlling large numbers of robots using an artificial attractive/repulsive force. Shucker and Bennett [11], [12] developed a decentralized control mechanism based on virtual spring mesh for the deployment of distributed robotic macrosensor (DRM). Each robot in the macrosensor interacts with the neighbors relying on a physics model of virtual spring mesh abstraction. However, all existing approaches for flocking have at least one of the following problems: (1) either the local formation is not considered or the local formation is not unique. (2) Individual's behavior in the next time is almost dependent on its all visible/sensible neighbors, which usually causes high uncertainty of the behavior. (3) Complex parameter setting is needed. (4) Local minimal trap has to deal with, particularly, in potential methods. To circumvent these problems, we propose a new decentralized method, Triangular Formation Algorithm (TFA) strategy, which basically allows three neighboring robots form isosceles. As each robot's decision making only relies on the local position information of two selected robots in each time step, individual's behavior decision is less interrupted by other robots. Principle of TFA strategy is simple and minimal conditions for robots is required. The amount of computation of each robot is independent from scale of swarm.

Manuscript received February 8, 2009; revised August 20, 2009; accepted October 1, 2009.
Corresponding Author: Xiang Li. His current research interests include swarm intelligent, bio-inspired system and evolutionary robotics. Email: XL_HUSE@126.COM

doi:10.4304/jcp.5.6.914-921

The rest of this paper is organized as follows. Section 2 states the robot model and the problem. Section 3 describes the TFA algorithm and its convergence. Section 4 demonstrates the aggregation behaviour of the typical sized swarms. Section 5 introduces an obstacle avoidance mechanism and demonstrates the flocking behaviour of swarms in environments with or without obstacles. Finally, conclusions are given in section 6.

II. PROBLEM STATEMENT

We consider a swarm system consisting of n homogenous and autonomous robots. Individual robots are denoted by r_1, r_2, …, r_n respectively and they are modeled as mobile points in two-dimensional space. Each robot contains a compass, limited range sensors, and simple actuators. The robots are able to detect the positions of neighboring robots within their sensing range with respect to their local frame using a digital compass. Direct communication between the robots does not exist. All the robots execute the same algorithm TFA and their acts are independent and asynchronous from each other.

The distance, between any pair of robots r_i and r_j located at p_i and p_j respectively, is denoted as $dist(p_i, p_j)$. The sensing radius of a robot is denoted as SR and the corresponding sensing area as SA. d_u is a constant distance greater than zero and less than sensing radium SR. Given robot, r_i, can detect the robots within its SA and pick two specific robots, r_{s1} and r_{s2}, among them as its neighbors. It is therefore required that robot r_i has at least two robots located within its SA. For robot r_i, the positions of its two neighbors r_{s1} and r_{s2} are denoted as p_{s1} and p_{s2} respectively and defined as neighbor position set $\{p_{s1}, p_{s2}\}$. Given p_i and $\{p_{s1}, p_{s2}\}$, C_i is defined as *a certain configuration* determined by position set $\{p_i, p_{s1}, p_{s2}\}$ denoted as S_i. The configuration C_i is possibly an acute triangle, line segment or obtuse triangle. Configuration C_i is defined as *Equilateral Triangle* (E_i), if and only if all the possible distance permutations $dist(p_{(i)}, p_{(j)})$ in S_i are equal to d_u.

Each robot r_i in the swarm attempts to generate E_i together with its two neighbors r_{s1} and r_{s2}. Consequently, the whole swarm can form a network with uniform distance d_u between members. Obviously, individual robot's behavior is local. For forming configuration E_i, robot r_i needs to calculate a goal position p_g to move. The robot, r_i, runs the algorithm TFA, which will be explained in the following section, to calculate the goal position p_g using the local position information of its two neighbors. We start with the basic behavior, that is, three neighboring robots form E_i configuration starting from any arbitrary and distinct initial distribution.

III. TRIANGULAR FORMATION ALGORITHM (TFA)

The Triangular Formation Algorithm (TFA) proposed here is a foundational mechanism, resulting in the collective behaviors of swarm, such as aggregation, flocking and obstacle avoidance. TFA algorithm is scalable and practical as the basic requirements to implement on a robot is minimal. The basic aim of TFA is to make three neighboring robots to form a dynamic E_i configuration, regardless of their arbitrary initial distribution. Therefore, each robot attempts to maintain isosceles with other two neighbors. For a given robot r_i in the neighborhood, a goal position p_g, depending on the local position of neighbors r_{s1} and r_{s2}, needs to be calculated in order to derive direction of the motion at the next step. Robot r_i can calculate the position p_g which has the same distance of d_u from other two robots, as shown in Fig. 1 and Fig. 2. The constant distance, d_u, is predefined and it is equal to the side length of the E_i configuration. As shown in Fig. 1 (a), robot r_i calculates the goal position p_g which is at the equal distance of d_u to its neighboring robots r_{s1} and r_{s2}. After the goal position is known, r_i, moves towards this position. Since each robot has the same motivation of forming isosceles with other two neighbors, as a result, they can build the configuration, E, eventually.

The basic structure of the algorithm requires robots to perform only three local behaviors and execute them periodically. These local behaviors are:

Step 1. Detecting other robots' positions;

Step 2. Calculating a goal position;

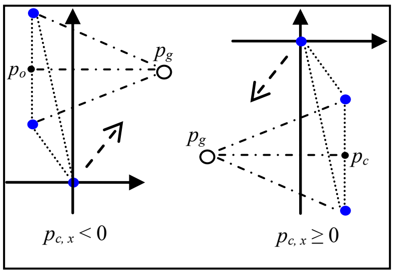
Step 3. Moving towards the goal position;

The algorithm TFA running on robot r_i takes the position information p_{s1} and p_{s2} of other two neighboring robots r_{s1} and r_{s2} as input, and the goal position p_g calculated as output. In actual robot, the input may directly come from proximity sensors, and the output can be used to tune the motion with actuators.

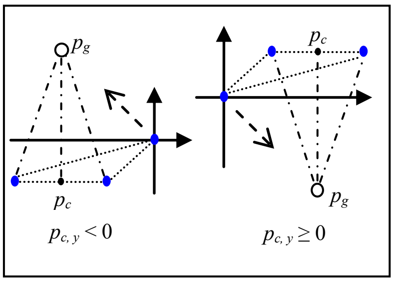
The configuration C_i is possible a triangle or a line segment, depending on the specific position set S_i, i.e. the relative position relationship of three neighboring robots at current time. Apparently, C_i is variable as S_i will change with time t. That is, C_i is a *configuration function* of time t, $C_i(t)$. According to the possible distance permutations $dist(p_{(i)}, p_{(j)})$ in S_i , at current time t , and the constant distance, d_u, the possible configurations, $C_i(t)$, may be classified into two types to calculate p_g for robot r_i: the regular and irregular configurations.

Regular configuration $C_i(t)$: The characteristic of this type of configurations is that, all the possible distance permutations $dist(p_{(i)}, p_{(j)})$ in S_i are not greater than $2\times d_u$. In this case, each robot of the three neighboring robots $\{r_i, r_{s1}, r_{s2}\}$ can follow a simple rule to maintain an isosceles with the other two neighboring robots which is described earlier.

All the possible methods for calculating the goal position, p_g, of robot r_i, are illustrated in Fig. 1 and Fig. 2. In these Figures, a hollow circle indicates the goal position and full circle indicates robot positions. The arrowhead, pointing towards the goal position p_g from robot r_i at position p_i, indicates the expected direction of motion for r_i. The center of the line segment $p_{s1}p_{s2}$ is denoted with p_c. The point p_g locates on the line which is perpendicular to $p_{s1}p_{s2}$ and passes through p_c, and is also required to have the same distance d_u from p_{s1} and p_{s2}. Fig. 1 also shows the calculation of p_g where the line segment $p_{s1}p_{s2}$ is parallel to one of the axis. In addition, Fig. 2 shows the calculation of p_g where the line segment $p_{s1}p_{s2}$ is not parallel to any axis. Here, l represents the line, which passes through the origin of the local coordinate frame of r_i and it is also parallel to the line segment $p_{s1}p_{s2}$. Parameter k indicates the slope of the line l with respect to the local frame of r_i, and θ is the included angle between the line l and the horizontal axis.

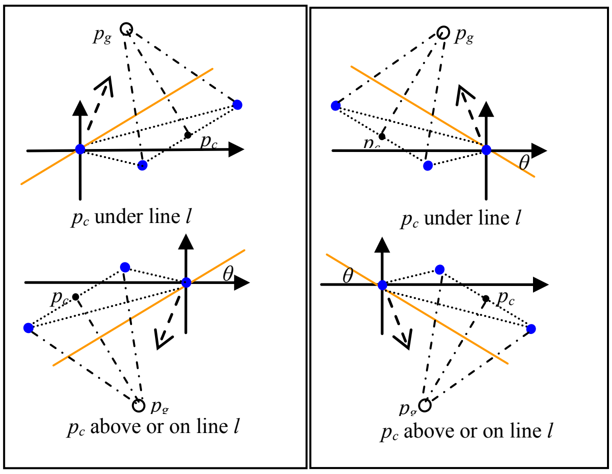


(a) $p_{s1}p_{s2}$ is parallel to axis oy



(b) $p_{s1}p_{s2}$ is parallel to axis ox

Figure 1. Calculations of p_g where $p_{s1}p_{s2}$ are parallel to the axis.



(a) k>0 (b) k<0

Figure 2. Calculations of p_g where $p_{s1}p_{s2}$ are not parallel to the axis.

Irregular configuration $C_i(t)$**:** The characteristic of this type of configurations is that, there exists a possible distance permutation, $dist(p_{(i)}, p_{(j)})$ in S_i , which is greater than $2\times d_u$ which may often happen in practice. As an example, the moment r_i derives that the distance between r_{s1} and r_{s2} is greater than $2\times d_u$, it cannot calculate the goal position p_g because it does not exist based on the above analysis. Similar situation may be encountered by any one of the neighbors, r_{s1} and r_{s2}, at the same time. In other words, in this case, there is at least one robot in the neighborhood of three robots, which can not obey the rule and calculate its goal position p_g accurately. When this case happens to any of the neighboring robots $\{r_i, r_{s1}, r_{s2}\}$, they should first cluster together before the calculations of p_g. Therefore, we defined the following strategy: when any robot detects the distance between the two neighboring robots or between itself and any one of the two neighbors greater than $2\times d_u$, the robot will take the average position of the position set S_i as the goal position p_g. In particular, if the set S_i can form a triangle, p_g will be the centroid position of the triangle $p_ip_{s1}p_{s2}$.

In the following, the algorithm TFA is presented in detail. The basic principle of the algorithm is to maintain a dynamic equilateral triangle configuration E_i with side length of d_u, which is specified by user, regardless of initial position distribution of robots. Once the goal position p_g is calculated, robot r_i can tune its direction of motion accordingly.

Triangular Formation Algorithm (TFA):

```
INPUT: ps1, ps2; // positions of the two
neighbors within ri's local frame.
OUTPUT: pg; // goal position of ri within
its local frame.
```

```
OUTPUT calculatePg(INPUT) {

IF d_u < p_s1p_s2/2 || d_u < p_ip_s1/2 || d_u <
p_ip_s2/2
    p_g = (p_i + p_s1 + p_s2)/3; // value of p_i
is 0 within r_i's local frame.
ELSE
    p_c = (p_s1 + p_s2)/2;
    p_cp_g = sqrt(d_u^2 - p_cp_s1(2)^2) ; // d_u is predefined by
user.
    IF p_s1, x == p_s2, x || p_s1, y == p_s2, y
        IF p_s1, x == p_s2, x
            IF p_c, x < 0
                p_g, x = p_c, x + p_cp_g;
                p_g, y = p_c, y;
            ELSE
                p_g, x = p_c, x - p_cp_g;
                p_g, y = p_c, y;
        ELSE
            IF p_c, y < 0
                p_g, x = p_c, x;
                p_g, y = p_c, y + p_cp_g;
            ELSE
                p_g, x = p_c, x;
                p_g, y = p_c, y - p_cp_g;
    ELSE
        k = (p_s1, y - p_s2, y)/(p_s1, x - p_s2, x);
        y = k*p_c, x;
        θ = |arctan(k)|;
        IF k > 0
            IF p_c, y < y
                p_g, x = p_c, x - p_cp_g*sin(θ);
                p_g, y = p_c, y + p_cp_g*cos(θ);
            ELSE
                p_g, x = p_c, x + p_cp_g*sin(θ);
                p_g, y = p_c, y - p_cp_g*cos(θ);
        ELSE
            IF p_c, y < y
                p_g, x = p_c, x + p_cp_g*sin(θ);
                p_g, y = p_c, y + p_cp_g*cos(θ);
            ELSE
                p_g, x = p_c, x - p_cp_g*sin(θ);
                p_g, y = p_c, y - p_cp_g*cos(θ);
Return p_g;
}
```

A. Convergence Analysis of TFA

For any arbitrarily distinct positions of three neighboring robots, TFA converge the initial configuration of robots to the desired configuration E_i. Four typical processes of configuration convergence are illustrated in Fig. 3. These results demonstrate the stabilization characteristic of TFA. For all the experiments, the value of d_u is fixed to 10. Fig. 3 (a) shows the case where initial configuration is a regular configuration. Fig. 3 (b), (c) and (d) show the cases where initial configurations are irregular configurations. From the results, we can observe that all the side lengths of the initial configurations gradually converge to the same value d_u. Specifically, in Fig. 3 (a), all-side lengths of the initial configuration is less than $2\times d_u$; in Fig. 3 (b), one-side length of the initial configuration is greater than $2\times d_u$; in Fig. 3 (c), two-side lengths of the initial configuration is greater than $2\times d_u$; in Fig. 3 (d), all-side lengths of the initial configuration is greater than $2\times d_u$. If the initial configuration is a regular configuration, the robots can directly converge to E_i. However, when they start from an irregular initial configuration, they first cluster to form a regular configuration, and then converge to E_i. From the results, we observe a minor oscillation at the balancing point of $d_u = 10$ after a period of time. The final configuration E_i built by the robots is at dynamic quasi-balance. The reason for this oscillatory occurrence is due to our assumption of constant velocity for all the robots.

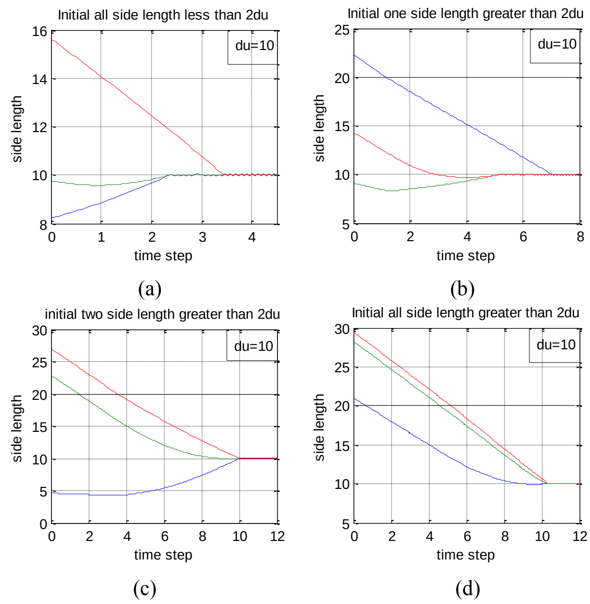


Figure 3. Illustration of the convergence of TFA.

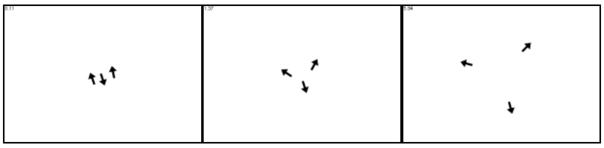
IV. AGGREGATION BEHAVIOUR OF SWARMS

The core algorithm TFA needs three robots. However, for large a swarm, that is swarm with more than three robots, we combined a neighbor selection mechanism with TFA. As mentioned earlier, we assumed each robot can only detect the robots located within its *SA*. In each time step, r_i selects the nearest robot to point p_i as the first neighbor r_{s1}. The second neighbor r_{s2} is selected such that the total distance from p_{s1} to p_i passing through p_{s2} is minimized. If there is more than one robot satisfying the above conditions, any one of them is selected as r_{s1} or r_{s2}. Once neighbors are selected, r_i then runs TFA to calculate its goal position to maintain isosceles with these two neighbors. Since each robot within the swarm attempts to maintain isosceles configuration with the two dynamically selected neighbors, whole swarm can maintain multi E_i that emerge from local interaction of individuals. Through dynamic selection of neighbors, TFA is extended to a large swarm regardless of initial distribution of the individual members of the swarm. In order to validate TFA, we performed series of aggregation experiments for swarms of various sizes.

Here, we use an open simulation platform, BREVE simulation environment [23] for simulation studies. Parameters for simulation can be changed for various experiments. These parameters are the number of robots (n), the sensing range of robot (SR), the side length of E_i (d_u) and the initial distribution space ($\Delta x,\Delta y$). Apparently, when simulating, the condition, $SR>d_u$, should be satisfied to form the desired multi E_i formation. Each robot has at least two neighbors during simulation, is assumed. Latter can easily be achieved by using a wider sensing range or alternatively, distributing robots into a narrow initial distribution space.

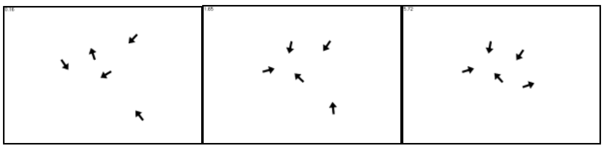
A. Aggregations of Small Swarms

For the convenience of observation during simulation, an arrowhead is used to represent a robot and also to indicate the direction of its motion.



(a) n=3, SR=20, d_u=10, ($\Delta x,\Delta y$)=(5,5).

Figure 4. Aggregations of three robots.



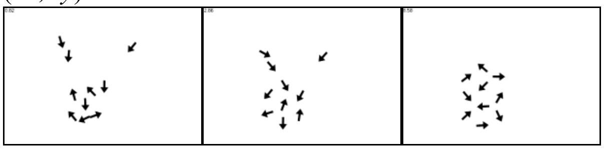
(b) n=5, SR=20, du=5, ($\Delta x,\Delta y$) = (15,15).

Figure 5. Aggregations of five robots

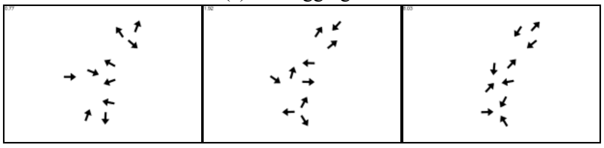
Fig. 4 illustrates the aggregation of the smallest swarm consisting of three robots. Fig. 5 demonstrates the aggregation of a swarm with five members. For these swarms with 3~5 robots, aggregation into one is always achieved successfully regardless of the parameter d_u and ($\Delta x,\Delta y$). This is because, once any of the three robots aggregate, remainder robots are not enough to form another smallest swarm.

B. Aggregations of Large Swarms

We now look into the aggregation behavior from a large swarm. In the following, we present the simulation study for this case with various settings of d_u and ($\Delta x,\Delta y$).



(a) One aggregation.



(b) Three aggregations.

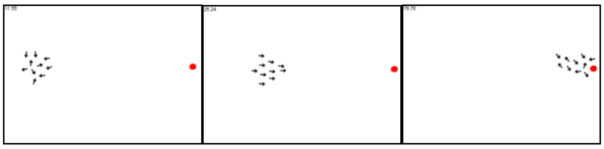
Figure 6. Aggregations of large swarm, n = 10, SR = 20, du = 3, (Δx, Δy) = (15, 15)

When initial distribution is the case of cluster, the swarm can always aggregate together as a whole. This condition is easily achieved in real application by simply setting to larger d_u and smaller ($\Delta x,\Delta y$) values. When the initial distribution is relatively scattered, multi-aggregation behavior will occur, as illustrated in Fig. 6. The number of the aggregations that will emerge is closely related to the size of the swarm and the initial distribution conditions, d_u and ($\Delta x,\Delta y$). However, this behavior might be desired for some applications such as multi-target search. If d_u and ($\Delta x,\Delta y$) are fixed, the swarm with smaller size tends to produce more multiple aggregations. For a certain swarm with size n, we can control the emergence of above various aggregation behaviors by appropriately choosing the parameters d_u and ($\Delta x,\Delta y$), according to the requirements of a real application. The swarm will have more chance to form multi-aggregation for smaller d_u and wider ($\Delta x,\Delta y$) values. From above analysis, we can deduct that the possible number of aggregations is in rang of $1\sim\lfloor n/3 \rfloor$, where n is the size of the swarm.

V. Flocking behaviours for swarms

A. Flocking behaviour in an unconstraint environment

The swarm behaviors like bird flocking and fish schooling could be seen frequently in nature. One of the main objectives of swarm robotics is to achieve similar global behaviors in an unknown environment artificially. That is, how to control a large number of robots to move towards a common target while avoiding collisions between members, and between members and obstacles, in an environment. Instead of letting robots move along a certain direction, we assume all robots known the target position. The half area of *SA* of a given robot, on the flocking vector direction, is denoted as SA_d. Firstly, r_i checks if there is any neighboring robot within SA_d. If multiple neighbors exist, r_i selects two neighbors from SA_d instead of *SA*, but the procedure of selection is same as before. If there is only one neighbor within SA_d, r_i spots a virtual point located an adequate distance away from p_i along its flocking direction and defines this point as p_{s1}, and uses the only robot's position as p_{s2}. Otherwise, it moves towards the target at a unit velocity. Fig. 7 shows the simulation results of swarm flocking in an environment without obstacle, where the robots are flocking towards a common target while maintaining uniform intervals between them.



n=10, SR=20, d_u=3, $(\Delta x,\Delta y)$=(5,5)

Figure 7. Flocking towards a target in environment without obstacle.

B. *Flocking in environments with obstacles*

(1) Obstacle avoidance for swarm

A mechanism of obstacle avoidance, which works with TFA, is introduced and it mimics the interaction effect between a motion electron and an atom nucleus as illustrated in Fig. 8. As a result, swarm can maintain a regular configuration and adapt to an environment with obstacles when flocking towards a target. If a robot approaches to an obstacle, a repulsive force f_r will exert on it. A repulsive velocity v_r is equivalent to the effect of f_r on the robot. The direction of v_r depends on the nearest point of the obstacle and directs from the point to the robot. For simplicity, an obstacle is considered as a closed disc and has a safety margin of d. The nearest point of obstacle from the robot is denoted by O. In order to avoid the obstacle, any robot r_i will adjust current velocity by adding the component v_r on the current velocity. If the distance between r_i and O is larger than d, there is no force on r_i. Otherwise, there will be a repulsive force exerting on r_i. The repulsive velocity is calculated with the following formula:

$$v_r = \frac{P_i - O}{|P_i - O|} r\left(1 - \frac{|P_i - O|}{d}\right) \tag{1}$$

Where *r(x)* is defined as a ramp function:

$$r(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \tag{2}$$

Equation (1) implies that, in the worst case, the repulsive velocity exerting on a robot increases from 0 to 1 linearly during the course that the robot enters buffer zone until it just touches the obstacle. For a robot, moving at a unit velocity, this will cause it to stop when touching the obstacle because its total velocity will be zero at that moment. In fact, as a member of the swarm, a robot will keep away from the obstacle consequently by following its neighbors before collision happens. In other words, individuals possibly will avoid the local minima with the help of their neighbors.

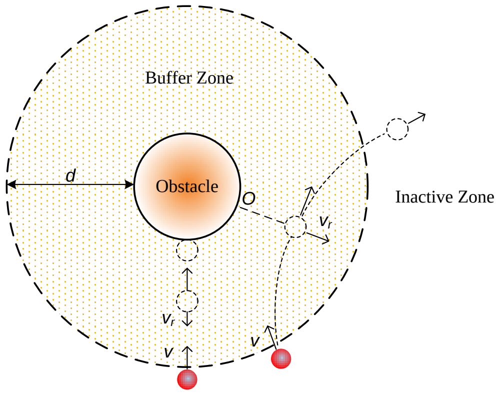
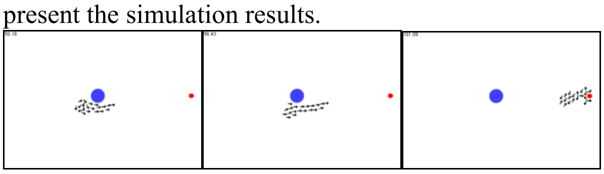


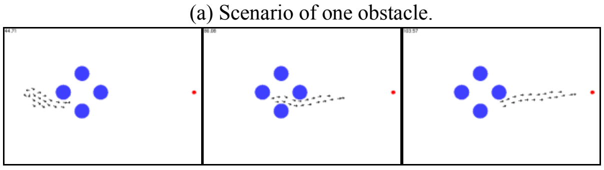
Figure 8. Illustration of obstacle avoidance of a single robot.

(2) Results for obstacle avoidance of a flocking swarm

The obstacle avoidance incorporated to TFA algorithm is experimented with various parameters for swarm and obstacle setting. The following Fig. 9 and 10 present the simulation results.

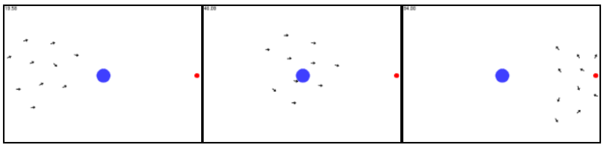


(a) Scenario of one obstacle.

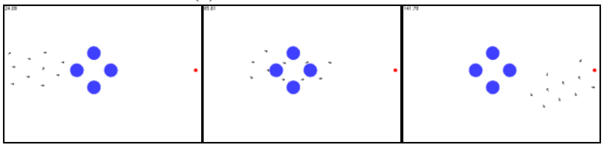


(b) Scenario of multi obstacles

Figure 9. n=20, SR=20, d_u=2, $(\Delta x,\Delta y)$=(5,5), d_o=6



(a) Scenario of one obstacle



(b) Scenario of multi obstacles

Figure 10. n=10, SR=20, du=10, $(\Delta x, \Delta y)$=(5, 5), do=6

We observed that if the size of an obstacle is bigger than d_u, the swarm has a tendency to move around the obstacle as a whole, as shown in Fig. 9 (a). Otherwise, the swarm will more likely to flock towards the destination directly while avoiding the obstacle along the way as shown in Fig. 10 (a). When a large swarm is flocking towards a target in an environment with multiple obstacles, if the size of the obstacles is larger than d_u, the swarm will possibly adapt its shape to fit the gap between the obstacles and then go through, as seen in Fig. 9 (b). If the size of the obstacle is much smaller than the distance d_u, the swarm will maintain its shape and directly flock through the obstacles seen in Fig. 10 (b). TFA rely on the fact that robot can identify the positions of other robots located at *SA* with respect to local coordinate and distinguish itself from obstacles. As calculations involve only two other neighbours' position, TFA has less computation and become less influenced by other robots

VI. Conclusions

This paper demonstrates a decentralized planning method, Triangular Formation Algorithm (TFA), for a swarm of robots to demonstrate aggregation and flocking behaviours. The principle of TFA is simple and doesn't require setting detailed parameters to model. The assumptions such as robot identifiers, common coordinate, direct communication memory to past states, are not needed in TFA. By using neighbour selection strategy, TFA is extended to a large swarm. Due to these characteristic, TFA is scalable and favourable to apply in practice. Robots can negotiate the environment and dynamically adapt a collective behaviour. For swarm obstacle avoidance, we present a simplified virtual physical mechanism. Robots may, directly, go through or move around the obstacle as a whole while maintaining a uniform distance from each other. The principle of TFA can be extensible to three-dimensional space which is our primary interest in our future study.

Acknowledgment

This work was supported by National Science Foundation of China (60874070), by National Research Foundation for the Doctoral Program of Higher Education of China (20070533131).

References

[1] L. Bayindir and E. Şahin, "A review of studies in swarm robotics," *Turk. J. Elec. Engin.*, vol. 15, no. 2, pp. 115-147, 2007.

[2] N. Correll, C. Cianci, X. Raemy, and A. Martinoli, "Self-organized embedded sensor/actuator networks for "smart" turbines," Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems Workshop on Network Robot System: Toward intelligent robotic systems integrated with environments, Beijing, China, October 2006.

[3] C. Y. Chong and S. P. Kumar, "Sensor networks: evolution, opportunities, and challenges," *Proc. of the IEEE*, vol. 91, no. 8, pp. 1247-1256, August 2003.

[4] M. F. Godwin, S. C. Spry, and J. K. Hedrick, "Distributed system for collaboration and control of UAV groups: experiments and analysis," *Lecture Notes in Economics and Mathematical Systems*, vol. 588, pp. 139-156, Springer-Verlag, Berlin Heidelberg, Germany, 2007.

[5] E. H. Gustafson, C. T. Lollini, B. E. Bishop, and C. E. Wick, "Swarm technology for search and rescue through multi-sensor multi-viewpoint target identification," *Proc. of the 37th Southeastern Symposium on System Theory*, pp. 352-356, March 2005.

[6] T. Balch and M. Hybinette, "Behavior-based coordination of large scale robot formations," *Proc. of the 4th International Conference on Multiagent Systems (ICMAS)*, pp. 363-364, Boston, MA, USA, July 2000.

[7] T. Balch and M. Hybinette, "Social potentials for scalable multi-robot formations," *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, vol. 1, pp. 73-80, San Francisco, CA, USA, April 2000.

[8] D. H. Kim, H. Wang, and S. Shin, "Decentralized control of autonomous swarm systems using artificial potential function-analytical design guidelines," *J. Intell. Robot Syst.*, vol. 45, 369-394, 2006.

[9] W. M. Spears, D. F. Spears, R. Heil, and W. Kerr, "An overview of physicomimetics," *Lecture Notes in Computer Science*, vol. 3342, pp. 84-97, Springer-Verlag, Berlin Heidelberg, Germany, 2005.

[10] W. Spears, D. Spears, J. Hamann, and R. Heil, "Distributed, physics-based control of swarms of vehicles," *Autonomous Robots*, vol. 17, no. 2-3, 137-162, 2004.

[11] B. Shucker and J. K. Bennett, "Scalable Control of Distributed Robotic Macrosensors," *Proc. of 7th*

International Symposium on Distributed Autonomous Robotic Systems (DARS '04), June 2004.

[12] B. Shucker and J. K. Bennett, “Virtual Spring Mesh Algorithms for Control of Distributed Robotic Macrosensors,” *technical report CU-CS-996-05*, Department of Computer Science, University of Colorado at Boulder, May 2005.

[13] J. Klein, “Breve: a 3D simulation environment for the simulation of decentralized systems and artificial life,” *Proc. of Artificial Life VIII, the 8th International Conference on the Simulation and Synthesis of Living Systems*, MIT Press, Cambridge, 2002.