Generating Formations with a Template based Multi-Robot System

Jan Carlo Barca
Department of Electrical and Computer Systems Engineering, Monash University, Clayton, Victoria 3800, Australia.
Jan_carlobarca@hotmail.com
Ahmet Sekercioglu
Department of Electrical and Computer Systems Engineering, Monash University, Clayton, Victoria 3800, Australia.

Abstract

A template based technique that generates two dimensional formations in leader-follower type multi-robot systems is presented in this paper. How this technique: i) selects a leader robot that is placed at the front of formations and ii) enables six randomly positioned identical non-holonomic robots to compose triangular, circular, linear and rectangular formations is also demonstrated. The template based technique has been compared with a state-of-the-art potential field type formation generation technique and simulation results show that the template based technique can form circular formations 77.35% faster and with 51.43% shorter travel distance. The presented work is an important contribution to multi-robot control theory as history shows that the success of groups often depend on the ability to efficiently generate appropriate formations.

1 Introduction

Multi-robot (MR) systems consist of a team of individual robots that usually collaborate to achieve common high level goals. The number of robots used in these systems range from a few to several thousand. Examples are flying robots [1-3], ground based mobile robots [4-8], robots that operate in water [9, 10] or a mixture of these [3, 11].
MR systems research is motivated by these systems ability to:

- exploit the sensing capabilities of large groups to achieve superior situational awareness [12-15]
- support a high robustness towards mission failures [14, 16-18]
- distribute workloads among its members in order to carry out large numbers of tasks simultaneously [15, 19-21]

These abilities are highly sought after in a range of application areas such as search and rescue [22-24], exploration [25, 26] and defence [1, 11, 25, 27-29], and MR systems research can therefore, have a significant economical and social impact.

The ability to generate various formations is important for MR systems as history has shown that the success of groups often depend on the ability to generate and maintain particular formations [30-32]. For example, losing a rear guard can result in a whole group being annihilated, encirclement of an enemy target can lead to quick victory, and a dispersed formation has a greater chance of surviving heavy attacks [15].

To address this issue we have devised a Template-based Formation Generation (TFG) technique, which enables six randomly distributed identical non-holonomic ground moving robots to generate triangular, circular, rectangular and linear formations in "leader/follower" type multi-robot systems. (The TFG technique is not confined to these four selected formations and it is easy to incorporate more templates with specifications for additional formation shapes.) We focus on these particular formations due to their wide applicability. E.g. triangular formations are ideal in unknown environments as few individuals are exposed to the area initially when the formation moves forward, thereby reducing the chances of exposing many robots to possible threats. Circular formations can be used to surround enemy targets, or to set up perimeters around military bases or atom reactors where it is important to detect intruders or radioactive leaks. Rectangular formations on the other hand, can act as building blocks in grids that can be used to cover down on areas when high situational awareness is required. Finally, linear formations can be used to assist in border monitoring in safety critical areas.

We focus on "leader/follower" type multi-robot systems rather than pure decentralised swarm robotics systems, as it is hard and maybe even impossible to accurately predict behaviours that emerge from the local interaction rules used in purely decentralised systems [33]. It is easier to achieve predictable outcomes with "leader/follower" type systems (which use distributed rather than decentralised control principles) as one can guide the general behaviour of the robots through the leader while still making use of swarm robotics principles to control the follower robots.

Several formation control techniques are described in literature [12, 32-42] however, a wide range of differences exist between these earlier techniques and the TFG technique. One is that the TFG technique makes robots avoid both obstacles and other robots as they move into formation, which is not the case in [34, 42]. Another difference is that the TFG technique makes use of an automated leader selection mechanism to choose a leader, rather than simply using a pre-selected leader, which is the case in [35]. The TFG technique is therefore, more robust than [35] when a leader is lost, as a new leader can be selected automatically. The technique presented in [35] furthermore, relies on signals from coloured lights that are

mounted on all robots in their system for navigation, while the TFG technique makes use of Global Positioning Systems (GPS) and Laser Range Finders (LRF) to support navigational tasks. A drawback with the approach used in [35] is that light changes appearance under varying environmental conditions, and their technique may therefore not be able to operate robustly in uncontrolled environments. The robots in [35] are furthermore forced to walk randomly around in the environment when they lose visual contact with other robots and it is therefore likely that a number of robots are lost when they become "visually disconnected" from other robots. A similar problem is observed in [36], where robots are regarded as being lost if they are unable to directly sense other robots. This is a significant drawback as new robots frequently will have to be introduced into the system to maintain the required number of robots. On the other hand, when the TFG technique is used, robots are not lost in this manner as they make use of GPS data to determine where they are in relation to their assigned positions in the formation.

The remaining parts of the paper are organized as follows. The particulars of the TFG technique are described in Section 2. A series of simulations that have been designed to measure the performance of the TFG technique and the results from these simulations are presented in Section 3. Final conclusions and future directions of research are presented in Section 4.

2 Template-based Formation Generation

An overview of the TFG technique is presented in Figure 1. One can observe that the technique is composed of six main processing steps, which interact to move randomly distributed robots into formations. A detailed description of each processing step is provided below.

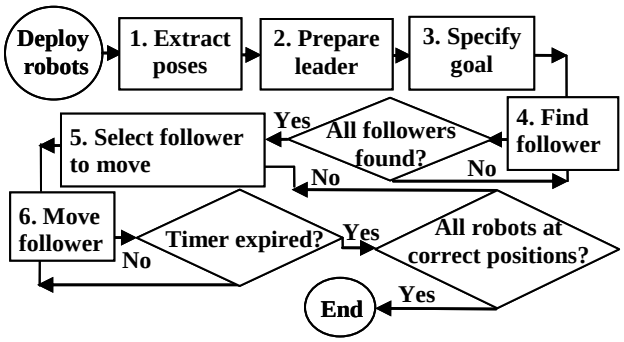

Figure 1 Overview of the Template-based Formation Generation technique.

2.1 Extract Pose Information

Processing Step 1 is initiated by a human controller once the robots have been deployed into an environment of interest. Any random distribution is acceptable as long as the distribution allows the robots to rendezvous. When processing Step 1 commences the Cartesian (*x*,*y*) coordinates and the vertical rotations of all robots are determined by analysing GPS data from each robot. Each GPS is polled every P_{t1}second during this process. The exact value for $P_{t1}$is implementation dependent but it is often desirable to use a small value as this ensures that the poses of the robots can be quickly determined. The values that were used for the implementation dependent variables in our system are presented in Section 3. Once the poses of all robots have been determined processing Step 2 is initiated.

2.2. Select and Prepare a Leader

In the beginning of processing Step 2 a Leader Robot (LR) which will head the formation is selected. To make it easy for the follower robots to position themselves behind the LR, the robot that is located furthest away from the centre of gravity of the robots is selected. This way the LR is located at the periphery of the group from the outset, which makes it easy for the other robots to position themselves behind the LR (as long as the robots are initially clustered and the LR faces away from the average position of the robots). The idea is illustrated in Figure 2.

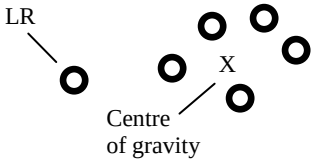

Figure 2 The LR is furthest away from the centre of gravity.

The Euclidean distance measure shown in Equation 1 is used to calculate the centre of gravity and to find the robot located furthest away from the centre of gravity.

$$d = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \quad \textbf{(1)}$$

where:

d : Euclidean distance between robot *i* and robot *j*
(x_i, y_i) : coordinates of robot *i*
(x_j, y_j) : coordinates of robot *j*

Once the LR has been selected, the polling interval of its GPS is set to P_{tlr} seconds. The value for P_{tlr} should be smaller than P_{t1} to ensure that the pose of this particular robot is monitored more frequently. The LR is then rotated away from the centre of gravity to make it easy for the other robots to position themselves behind the leader. This is done by first translating the centre of gravity from the world coordinate system into a local coordinate system with origin in the LR according to Equation 2, generating a Local Centre of Gravity (LCG).

$$x^i = x^j - x \quad \textbf{(2a)}$$

$$y^i = y^j - y \quad \textbf{(2b)}$$

where:

(x^j, y^j) : world centre of gravity
(x, y) : world coordinates of LR
(x^i, y^i) : LCG coordinates

This LCG is then mirrored over the LR by multiplying the

coordinate values with -1 if they do not equal zero (the values are left unchanged if they are zero). This mirrored LCG coordinate is then finally converted back into the world coordinate system according to Equation 3, generating a mirrored World Centre of Gravity (WCG).

$$x^{"} = x + x' \quad \textbf{(3}a\textbf{)}$$

$$y^{"} = y + y' \quad \textbf{(3}b\textbf{)}$$

where:

(x, y) : world coordinates of leader robot
(x', y') : mirrored LCG
$(x^{"}, y^{"})$: mirrored WCG

The mirrored WCG is the point the LR will be rotated towards. However, to minimise the energy consumption associated with rotating the LR towards this point, the shortest direction of rotation is calculated before the robot is rotated. This is done by first calculating θ using Equation 4.

$$\theta = \frac{\text{atan}(y_{ma} - y)}{|x_{ma} - x|} \quad \textbf{(4)}$$

where:

(x_{ma}, y_{ma}) : mirrored WCG coordinates
(x, y) : world coordinates of the LR
θ : angle between mirrored WCG and world coordinates of the LR when zero degrees is aligned with the local negative *x*-axis of the LR

Correlations between θ, the mirrored WCG and the local and world coordinate systems are illustrated in Figure 3.

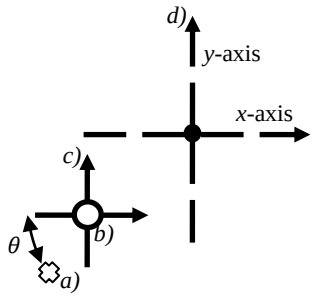

Figure 3 The θ value represents the angle between the negative *x*-axis of the local coordinate system with origin in the LR and the mirrored WCG. *a)* mirrored WCG, *b)* LR, *c*) local coordinate system, and *d*) world coordinate system.

Angle θ cannot be used directly unless the mirrored WCG is located in Quad 1 of the local coordinate system and θ must therefore be transformed into θ' to ensure that the correct results are calculated. The correlation between θ and θ' when zero degrees rotation is aligned with the negative *x*-axis and rotation increases counter-clockwise, is illustrated in Figure 4. One can observe that increasing numbers of degrees must be added to θ to get θ' as the mirrored WCG is moved counter-clockwise beyond Quad 1.

One must therefore first determine where the mirrored WCG is located in relation to the LR to transform θ into θ'. This can be done by testing the statements in the left column of Table 1, observing the corresponding result in the middle column, and transforming θ into θ' using the matching equation in the right column. (x_{ma}, y_{ma}) are the mirrored WCG and (x, y) are the world coordinates of the LR.

Once angle θ' is found it is transformed into angle θ'' in Rad. This transformation is conducted according to Equation 5.

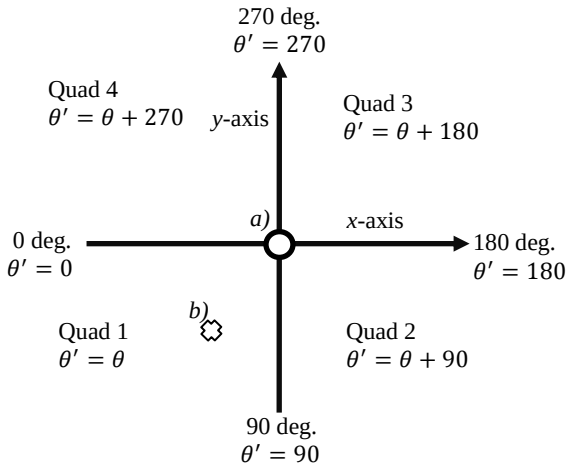

Figure 4 Correlation between θ and θ' when zero degrees rotation is aligned with the negative *x*-axis and rotation increases counter-clockwise. *a)*LR and *b)*mirrored WCG.

If statements	Goal in quad or on axis	Calculating θ'
If $x_{ma} < x$ and $y_{ma} < y$	Quad 1	$\theta' = \theta$
If $x_{ma} > x$ and $y_{ma} < y$	Quad 2	$\theta' = \theta + 90$
If $x_{ma} > x$ and $y_{ma} > y$	Quad 3	$\theta' = \theta + 180$
If $x_{ma} < x$ and $y_{ma} > y$	Quad 4	$\theta' = \theta + 270$
If $x_{ma} > x$ and $y_{ma} = y$	Positive x axis	$\theta' = 180$
If $x_{ma} < x$ and $y_{ma} = y$	Negative x axis	$\theta' = 0$
If $x_{ma} = x$ and $y_{ma} > y$	Positive y axis	$\theta' = 270$
If $x_{ma} = x$ and $y_{ma} < y$	Negative y axis	$\theta' = 90$

Table 1 Determining shortest rotation towards the mirrored WCG.

$$\theta'' = \left|\frac{\theta' \times \pi}{180}\right| - \left|\frac{r_{lr} \times \pi}{180}\right| \quad \textbf{(5)}$$

where:

θ' : angle between mirrored WCG and world coordinates of the LR in any quad
r_{lr} : rotation of the LR

θ'' : angle between rotation of the LR and the mirrored WCG in Rad.

Angle θ'' is then finally transformed into normalized angle θ''', which always is between 1 and -1 and can be used to determine the shortest direction of rotation for the LR. This transformation can be conducted through the use of the statements shown in Figure 5. The LR is rotated clockwise (direction of decreasing angles) if θ''' is smaller than R_n and counter-clockwise if θ''' is greater than R_p. Where R_n is a negative value and R_p is a positive value.

```
If θ'' >= π then
    subtract 2π from θ'' to get θ'''
    if θ'' <= −π then
        add 2π to θ'' to get θ'''
        if θ'' < π and θ'' > −π then
            θ''' = θ''
        end if
    end if
end if
```

Figure 5 Transforming θ into θ'''.

The process of calculating the shortest direction of rotation is repeated continuously while the LR rotates, until θ''' is greater than $R_n$or smaller than R_p. When this criterion is satisfied the correct rotation has been reached. R_n and R_p should ideally be close to zero as this produces the most accurate results. However, the values must be closer to -1 and 1 respectively, if the robot performs fast rotations or slow processors are used. The reason for this is that the robot in these cases, often is unable to detect that the criterion has been satisfied and thus, rotates too far creating an oscillating behaviour which rotates the robot repeatedly beyond the ideal rotational state.

2.3 Specify Goal Coordinates

In Processing Step 3 it is determined where the follower robots should go in order to generate a desired formation. These destinations, which from now on are referred to as goal coordinates, are determined according to the: i) formation to be generated, ii) position of the LR, and iii) rotation of the LR.

One can currently choose to generate triangular, circular, linear and rectangular formations with the TFG technique. (These four formations are illustrated in Figure 6.) Once a desired formation is selected a corresponding template, which contains the goal destinations for all follower robots, is automatically chosen from a library of templates. The goal coordinates refer to the positions of each follower robot with respect to a LR with zero degrees rotation located at the origin of a two dimensional coordinate system. These goal coordinates will therefore be referred to as Local Goals (LG).

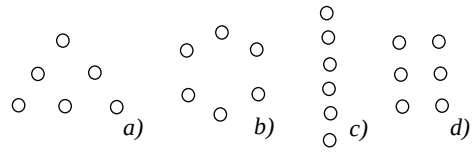

Figure 6 The TFG technique can currently generate: *a)* triangular, *b)* circular, *c)* linear and *d)* rectangular formations.

To take the actual rotation of the LR into consideration these LG must be transformed into Actual Local Goals (ALG). This transformation is conducted according to Equation 6 and is repeated for all LG in the appropriate template.

$$x^{ii} = x^{i}\cos(\theta) - y^{i}\sin(\theta) \quad \textbf{(6a)}$$

$$y^{ii} = x^{i}\sin(\theta) + y^{i}\cos(\theta) \quad \textbf{(6b)}$$

where:

(x^{i}, y^{i}) : LG
(x^{ii}, y^{ii}) : ALG
θ : rotation of the LR

Once the ALG for all followers have been found, the coordinates must be translated into Actual World Goals (AWG), which describes the goal coordinates for the follower robots in world space. This transformation is conducted using Equation 7 and is repeated for all ALG in the relevant template. Once all relevant AWG have been found Processing Step 4 is initiated.

$$x^{iii} = x^{ii} + x'_0 \quad \textbf{(7a)}$$

$$y^{iii} = y^{ii} + y'_0 \quad \textbf{(7b)}$$

where:

(x^{iii}, y^{iii}) : AWG
(x^{ii}, y^{ii}) : ALG
(x'_0, y'_0) : world coordinates of the LR

2.4 Find Followers

In Processing Step 4 the follower robots are labelled. This is done by first searching for the robot that: i) is located closest to the AWG for follower one, ii) is not a LR and, iii) has not been given a follower label, and assigning it the label Follower 1. The idea is illustrated in Figure 7. This process is repeated with the consecutive AWG until all the remaining robots have been assigned a follower label.

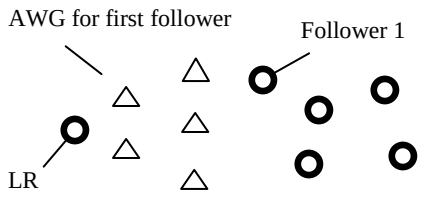

Figure 7 Process of labeling Follower 1.

2.5 Selecting Followers to Move

Processing Step 5 selects robots that are not the LR and have not reached their respective AWG, so that they can be moved incrementally towards their AWG in a pseudo parallel manner. This processing step is introduced to make it possible to run the TFG technique in a simulator with limited centralized processing power, and it is used for simulation purposes only.

The process is conducted by selecting Follower 1 if it is not located at its AWG, and setting a timer that expires after two seconds. If Follower 1 is at its associated AWG then the consecutive follower (e.g. Follower 2) is selected. The polling interval of the GPS on the selected robot is then set to P_{tf} seconds. P_{tf} should be smaller than $P_{t1}$to ensure that this robot's pose data is updated more frequently. (The GPS on the un-selected robots is not polled in order to conserve energy.) Follower 1 is then moved according to Processing Step 6. When the timer expires the consecutive follower is selected if it is not at its AWG and the timer is reset. This process is repeated for all followers until the process restarts with Follower 1. The overall selection process ends when all followers have reached their AWG. We move the follower robots for two seconds at a time, as it has been empirically found that this time period enables the robots to move in a smooth and efficient manner towards their AWG.

2.6 Move Followers

In processing Step 6 selected robots are moved towards their AWG. To enable the robots to move towards their AWG without crashing, we use a blend between an attraction force generated by a Go-to-Goal (GG) behaviour, and a repulsion force generated by an Obstacle Avoidance (OA) behaviour to move the robots. We will first describe the GG behaviour and then the OA behaviour.

2.6.1 Go-to-Goal

The GG behaviour initiates by measuring the distance between the position of the follower and its associated AWG. If the distance is greater than D_i centimeters, then the robot is rotated toward the AWG in the same manner as the LR was rotated toward the mirrored WCG in Processing Step 2. Once a satisfactory rotation is reached, the robot is moved forwards with a motor power of M_p on both wheels while the rotation of the robot is continuously monitored and corrected on the basis of incoming GPS data.

If the distance between the robot and the AWG is less than D_i centimetres, then the robot is forced to assume the same rotation as the LR with an error bound of E_bRad. The values for D_i and $E_b$are strongly dependent on the speed of the processor at hand and should be close to zero if fast processors are used, as values close to zero increases the accuracy of the results. Higher values must be employed when slower processors are used. The particular values that were used for D_i and E_b in this research project are shown in Section 3.

The process of rotating the LR is conducted by first calculating angle θ_d with Equation 8. If θ_d is greater than π, then 2π is subtracted from θ_d, otherwise 2π is added to θ_d. The result is angle θ_{sd}, which is used to determine the shortest direction of rotation. If θ_{sd} is smaller than zero then the robot turns clockwise to match the rotation of the LR, otherwise the robot turns anti-clockwise.

$$\theta_d = LR_{rrad} - F_{rrad} \quad \textbf{(8)}$$

where:

F_{rrad} : rotation of follower in Rad.
LR_{rrad} : rotation of the LR in Rad.
θ_d : angle used in the process of calculating shortest direction of rotation

2.6.2 Obstacle Avoidance

The LRF on the selected robot is polled every P_l second while the robot moves towards its associated AWG. P_l should be close to zero when fast moving robots are used as this will enable the robots to detect and avoid obstacles quickly. If the robots are relatively slow then P_l should be closer to one as less frequent polling reduces energy consumption. The data from the LRF is continuously analysed to determine if any obstacles are located in front of the robot. Each data sample contains 360 distance measurements, which have been captured at evenly spaced intervals from right to left across an angle of 180° in front of the robot. To facilitate the overall OA behaviour the data sample is split into three 60° sectors that are labelled right, front, and left sectors respectively. The idea is illustrated in Figure 8. To determine if an obstacle is located directly in front of the robot, the data in the front sector is analysed in 36 step increments, meaning that only every 36th distance measurement is processed. Thirty six step increments are used as this allows us to reduce the processing load by 90%, and at the same time detect the smallest obstacles in our domain when they are located D_m meters in front of a moving robot. We use D_m meters as this distance empirically has been found to enable our robots to react in time to avoid obstacles when the robots drive at a speed of T_s meters per second, which is the maximum speed used by our robots.

If an obstacle is detected closer than D_m meters away from the robot in the front sector, then a more comprehensive search process is initiated. The aims of this new search process is to find the shortest distance to the obstacle in the front sector and the shortest distance to any object located in the right or left sectors. The search process is performed in 10 step increments. Increments of this size are used as the smallest obstacles in our domain have been empirically found to generate data samples where the smallest distance values are spread over a minimum of 11 consecutive measurements, when the robots are located D_m meters in front of a robot. Ten step increments are therefore, regarded as providing a good

balance between reduction in processing load and accuracy.

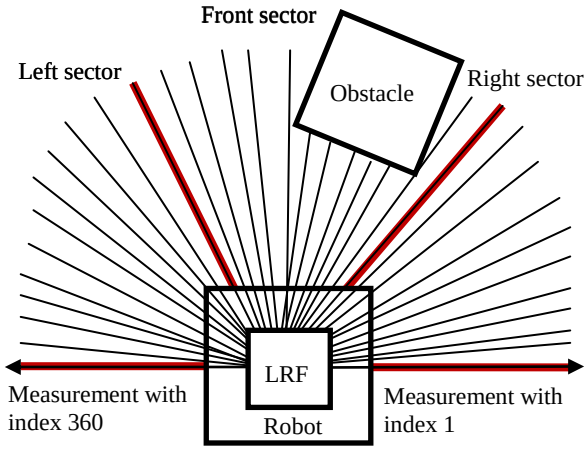

Figure 8 The LRF generates a data sample from 360 distance measurements captured at evenly spaced intervals from right to left over a 180° sensing angle. This data sample is split up into three 60° sectors to facilitate the overall OA behavior.

The shortest distance to the obstacle in the front sector is used to calculate the magnitude of a repulsion force that "pushes" the robot away from the obstacle. This repulsion force is calculated using the Inverse Square Distance Law in Equation 9.

$$R = \frac{M_i \times (D_{mi})^2}{(D_r)^2} \quad \textbf{(9)}$$

where:

R : repulsion force
M_i : maximum intensity of the force
D_r : distance between robot and obstacle
D_{mi} : distance where maximum intensity is reached

The value for M_i should be high enough to enable the robot to smoothly steer around obstacles and low enough to prevent the robot from bouncing directly away from obstacles as they get closer. D_{mi} is specified in millimeters and is found by calculating the distance between the LRF (which in our case is centered over the robot) and the furthest point of the robot. Adding an error margin E_m, compensates for the reaction time of the robot when it travels at maximum speed. Heuristics that can be employed when specifying E_m are that: i) the value must be large enough to prevent collisions and ii) small enough to enable the robots to generate the desired formations without being pushed away by the repulsion force exerted by other robots.

Repulsion force R is left unchanged for the left wheel and made negative for the right wheel if an object is located closer to the robot in the left sector than in the right. If an object is closer to the robot in the right sector than in the left then R is left unchanged for the right wheel and made negative for the left wheel. The R values for the left and the right wheels are then blended with the forward movement generated by the GG behaviour according to Equation 10. Blended forces B_r are calculated separately for the left and the right wheels.

$$B_r = \frac{M_p}{100} \times R \quad \textbf{(10)}$$

where:

B_r : blended wheel power
M_p : motor power from GG behavior
R : repulsion force

When a follower robot has reached its AWG the consecutive follower robot is selected and moved if it satisfies the criterions of the selection process. When all follower robots have reached their AWG, the TFG technique ends.

3 Simulation Design and Results

Simulations that compare the performance of the TFG technique with that of a state-of-the-art Potential Field (PF) technique [43] is presented in this section. This particular PF technique is used as a comparison as it is a recent advancement on a class of techniques that have been widely used to control multi-robots systems [42, 44-47].

The two techniques are compared in terms of the overall travel distance of the robots and the convergence time when circular formations are generated. The comparison is conducted on circular formations because such shapes have a wide applicability. E.g. circular formations can be used to protect moving convoys [43], surround enemy targets [15] or to set up perimeters around sites such as atom reactors and chemical plants to detect pollution [34].

We regard a circle as having been formed when all robots are located between: i) 1 and 3.6 meters from the centre of the formation, and ii) 1.35 and 2.4 meters from their closest neighbours. We use these particular constraints as they have been empirically found to be valid when circular shapes have been formed with the two formation generation techniques specified above. In this comparison we assume that the LR satisfies the first constraint at the beginning of each simulation, as the TFG technique does not move the LR but rather generates formations on the basis of the initial position of this particular robot, and therefore would be unable to satisfy both constraints unless the LR is located between 1 and 3.6 meters from the centre of the formation.

The comparison is conducted across two sets of 25 simulations, whilst systematically increasing the radius of a bounding area that the robots are randomly distributed within from 1 to 4.75 meters in 15 centimetre increments. To reduce the effect of outliers, corresponding results from the two sets of simulations are averaged prior to outcomes being graphed and analysed.

If a circular formation is not generated after 2000 seconds in a particular simulation, then the robots are regarded as being trapped in a local minimum and the simulation is restarted.

The simulations are conducted using the Microsoft Robotics Developer Studio simulator and six simulated Pioneer 3DX robots placed in an urban 3D

environment. Each robot is equipped with a differential drive system, a GPS and an LRF with a sensing angle of 180° and a resolution of 360. The robots drive 0.3 meters per second straight forward and 0.06 meters per second during rotations.

The implementation dependent parameter settings that were used for the TFG technique throughout the simulations are shown in Table 2.

Parameters and Settings			
P_{t1}	1	M_p	0.3
P_{tlr}	0.4	P_l	0.25
R_n	0.15	D_m	2
R_p	0.15	T_s	0.3
P_{tf}	0.25	M_i	65
D_i	10	D_{mi}	520
E_b	0.1	E_m	235

Table 2 Implementation dependent parameter settings.

3.1 Travel Distance

Results from the simulations that compares the travel distance of the robots as the TFG technique and the PF technique generates circular formations are described in this section.

The travel distances of the robots produced by each technique as the radius of the bounding area increases are shown in Figure 9. The mean travel distance across all simulations is 71.26 meters for the TFG technique and 146.71 meters for the PF technique. This shows that the TFG technique generates circular formations with 51.43% shorter travel distance across runs.

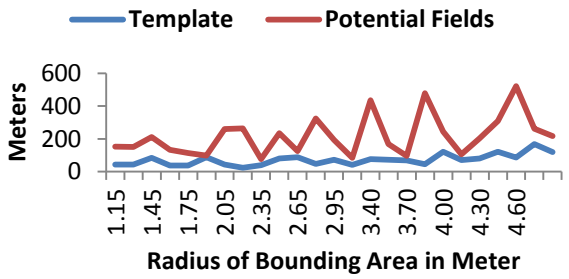

Figure 9 Mileage when circular formations are generated.

By studying the graphs, one can observe that the travel distance increases in a more stable manner when the TFG technique is used. The outliers that can be observed in the performance of the PF technique are generated because the robots occasionally are trapped temporarily in a local minimum, which makes them oscillate in a small area until a more globally optimal solution is found. This problem is not as extensive when using the TFG technique.

The difference in performance appears to arise from the fact that the TFG technique navigates each robot towards specific evenly distributed goal positions, while the PF technique navigates robots towards the closest point on a general circular shape. The drawback with the latter approach is that several robots easily move towards the same region if they are closely separated from the outset, thereby blocking each other's travel path and generating local minimums. The benefit is that this approach makes the PF technique scalable and robust at runtime as no robot is "linked" to a specific area in the formation. Hence, these results also indicate that there is a trade-off between scalability and robustness on the one hand and efficiency on the other.

3.2 Convergence Time

Results from the simulations that compares the convergence time of the robots when circular formations are generated are described in this section.

The convergence time for both techniques as the initial dispersion of the robots increases are shown in Figure 10. The mean convergence time for the TFG technique across runs is 80.94 seconds, and 357.48 seconds for the PF technique. By comparing these mean results one find that the TFG technique generates circular formations 77.35% faster than the PF technique across runs hence, outperforming the latter technique also on this performance criterion.

By comparing the graphs produced in this simulation with those from the previous, one can observe that there are close correlations between travel distance and convergence time. However, the difference in mean performance across runs is still much larger in this latter simulation. This can be explained by the fact that the PF technique does not increase the travel distance of the robots considerably when it makes the robots oscillate in a small area, however the travel time is still greatly affected by this oscillatory behaviour. The performance discrepancy arises from the variations in system design as specified in the previous section.

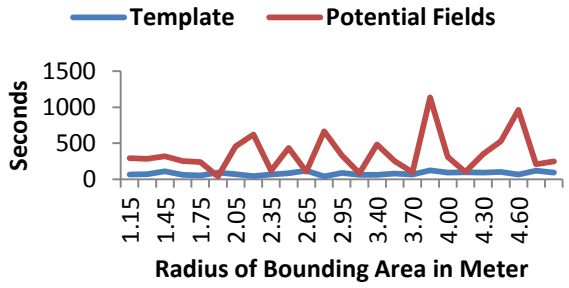

Figure 10 Convergence time when circular formations are generated.

4 Conclusions

A TFG technique that generates formations in leader-follower type multi-robot systems has been presented in this paper. We have also shown how this technique: i) performs automatic leader selection and ii) allows randomly positioned robots to compose triangular, circular, linear and rectangular formations. The proposed technique was compared with a state-of-the-art PF type formation generation technique, and simulation results show that the TFG technique is able to generate circular formations 77.35% faster and with 51.43% shorter travel distance. An additional strength of the proposed technique is that it is not confined to the four particular formations

that are described in this paper, and that it is easy to incorporate more templates with specifications for additional formation shapes. None of the techniques we have found in literature makes use of such a "template" based approach. Limitations of this research include that we: i) have not incorporated techniques that can guarantee that the robots move to the closest position in the selected formation, ii) have not investigated how varying numbers of robots affect system performance and iii) have not performed experiments on real world robots. These issues will be addressed in future research.

Acknowledgements

This research was supported in part by Lise and Arnfinn Hejes Grant for Education and Research. I would like to thank Adv. Harald Røer for the way he administered the grant. We would also like to thank Dimitra Bowen for her comments and insightful suggestions for improving the paper.

References

1. Hauert, S., Zufferey, J., Floreano, D.: Evolved swarming without positioning information: an application in aerial communication relay. Autonomous Robots 26(1), 21-32 (2008).
2. Holland, O., Woods, J., Nardi, R., Clark, A.: Beyond Swarm Intelligence: The Ultraswarm. Paper presented at the IEEE Swarm Intelligence Symposium, 2005, pp. 217-224
3. Pinciroli, C., O'Grady, R., Christensen, A., Dorigo, M.: Self-Organised Recruitment in a Heteregeneous Swarm. In: Proc. of 14th International Conference on Advanced Robotics, Munich, 22-26 June 2009
4. Trianni, V., Ampatzis, C., Christensen, A., Tuci, E., Dorigo, M., Nolfi, S.: From solitary to collective behaviours: Decision making and cooperation. In: Proc. of the 9th European Conference on Artificial Life 2007. Lecture Notes in Artificial Intelligence, pp. 575-584. Springer Verlag, Berlin
5. Groß, R., Dorigo, M.: Fifty Years of Self-Assembly Experimentation. In: Proc. of the Workshop on Self-Reconfigurable Robots/Systems and Applications 2007, pp. 1-6, USC Information Science Institute
6. Fukuda, T., Nakagawa, S.: A dynamically reconfigurable robotic system (concept of a system and optimal configurations). Paper presented at the International Conference on Industrial Electronics, Control, and Instrumentation, 1987, pp. 588-595
7. Asama, H., Matsumoto, A., Ishida, Y.: Design of an autonomous and distributed robot system: ACTRESS. In: IEEE/RSJ IROS, Tokyo 1989, pp. 283-290
8. Caloud, P., Choi, W., Latombe, J., Le Pape, C., Yin, M.: Indoor automation with many mobile robots. IEEE/RSJ IROS, 67-72 (1990).
9. Esposito, J., Feemster, M., Smith, E.: Cooperative manipulation on the water using a swarm of autonomous tugboats. In: IEEE International Conference on Robotics and Automation 2008, pp. 1501-1506
10. Shaneyfelt, T., Joordens, M., Nagothu, K., Jamshidi, M.: RF communication between surface and underwater robotic swarms. In: World Automation Congress, 2008, Waikoloa 2008, pp. 1 - 6
11. Birk, A., Schwertfeger, S., Pathak, K.: A networking Framework for Teleoperation in Safety, Security, and Rescue Robotics. IEEE Wirel. Commun. in Networked Robotics(February), 6-13 (2009).
12. Anderson, B., Yu, C., Hendrickx, J.: Rigid Graph Control Architectures for Autonomous Formations. IEEE Control Systems Magazine 28(6), 48-63 (2008).
13. Hart, D.: Reducing Swarming Theory to Practice for UAV Control. In: IEEE Aerospace Conference Proceedings 2004, pp. 3050-3063
14. Vincent, P., Rubin, I.: A Framework and Analysis for Cooperative Search Using UAV Swarms. Paper presented at the ACM Symposium on Applied Computing, Nicosia, March, 2004, pp. 79-86
15. Edwards, S.: Swarming on the Battlefield. RAND, Santa Monica, 1-88 (2000).
16. Pugh, J., Martinoli, A.: Distributed Adaptation in Multi-Robot Search using Particle Swarm Optimization. In: 10th International Conference on the Simulation of Adaptive Behaviour 2008, pp. 393-402
17. Cortes, J., Martinez, S., Karatas, T., Bullo, F.: Coverage control for mobile sensing networks. IEEE Transactions on Robotics and Automation 20(2), 243 - 255 (2004).
18. Lerman, K., Martinoli, A., Galstyan, A.: A review of probabilistic macroscopic models for swarm robotic systems. In: Lecture notes in computer science 3342. pp. 143-152. (2005)
19. Christensen, A., O'Grady, R., Dorigo, M.: Morphology Control in a Multirobot System. IEEE Robotics & Automation Magazine 14(4), 18-25 (2007).
20. Cao, Y., Fukunaga, S., Kahng, A.: Cooperative mobile robotics: Antecedents and directions. Autonomous robots 4(1), 7-27 (1997).
21. Barca, J.C., Rumantir, G., Li, R.: A Concept for Optimizing Behavioural Effectiveness & Efficiency. In: Machado, T., Patkai, B., Rudas, J. (eds.) Intelligent Engineering Systems and Computational Cybernetics. pp. 449-458. Springer Verlag, (2008)
22. Martinez, S., Cortes, J., Bullo, F.: Motion Coordination with Distributed Information IEEE Control Systems Magazine 27(4), 75-88 (2007).
23. Heo, N., Varshney, P.: Energy-Efficient Deployment of Intelligent Mobile Sensor Networks. IEEE Transactions on Systems, Man and Cybernetics 35(1), 78-92 (2005).
24. Jung, J., Park, S., Kim, S.-L.: Multi-Robot Path Finding with Wireless Multihop Communications. IEEE Communications Magazine, 126-132 (2010).
25. Hinchey, G., Sterritt, R., Rouff, C.: Swarms and Swarm Intelligence. Computer 40(4), 111-113 (2007).
26. Ma, M., Yang, Y.: Adaptive Triangular Deployment Algorithm for Unattended Mobile Sensor Networks. IEEE Transactions on Computers 56(7), 946-958 (2007).
27. Elston, J., Frew, E.: Hierarchical Distributed Control for Search and Tracking by Heterogeneous Aerial Robot Networks. In: Automation, I.I.C.o.R.a. (ed.), May 2008, pp. 170-175
28. LaValle, S., Lin, D., Guibas, J., J., L., Motwani, R.: Finding an unpredictable target in a workspace with obstacles. In: International Conference on Robotics and Automation, Albuquerque 1997, pp. 737-742
29. Parker, L.: Multi-robot learning in a cooperative observetion task. In: Pakers, L., Bekey, G. (ed.)

Robotic Systems. pp. 391-401. Springer, (2000)
30. Fidan, B., Yu, C., Anderson, B.: Acquiring and maintaining persistence of autonomous multi-vehicle formations. IET Control Theory & Applications 1(2), 452-460 (2007).
31. Eren, T., Belhumeur, P., Anderson, B., Morse, A.: A framework for maintaining formations based on rigidity. Paper presented at the 15th Triennial IFAC World Congress Barcelona, July
32. Erent, T., Brian, D., Anderson, A., Morse, S., Whiteley, W., Belhumeur, P.: Operations on Rigid Formations of Autonomous Agents. Communications in Information and Systems 3(4), 223-258 (2004).
33. Guo, H., Meng, Y., Jin, Y.: Analysis of local communication load in shape formation of a distributed morphogenetic swarm robotic system. In: IEEE Congress on Evolutionary Computation (CEC), Barcelona 2010, pp. 1-8
34. Li, X., Frey, H., Santoro, N., Stojmenovic, I.: Localized Self-Deployment of Mobile Sensors for Optimal Focused-Coverage Formation. In., pp. 1-18. Carleton University, (2008)
35. Christensen, A., Grady, R., Dorigo, M.: A Mechanism to Self-Assemble Patterns with Autonomous Robots. In: Advances in Artificial Life. pp. 716-725. Springer, Berlin (2007)
36. Mamei, M., Vasari, M., Zambonelli, F.: Experiments of Morphogenesis in Swarms of Simple Mobile Robots. Applied Artificial Intelligence 18(9-10), 903-919 (2004).
37. Desai, J., Kumar, V., Ostrowski, J.: Control of changes in formation for a team of mobile robots. In: IEEE International Conference on Robotics and Automation, Detroit, 10-15 May 1999, pp. 1556 - 1561
38. Sandeep, S., Fidan, B., Yu, C.: Decentralized Cohesive Motion Control of Multi-Agent Formations. In: 14th Mediterranean Conference on Control and Automation, Ancona, June 2006, pp. 1-6
39. Cheng, L., Hou, Z., Tan, M.: Decentralized adaptive leader-follower control of multi-manipulator system with uncertain dynamics. In: 34th Annual Conference of IEEE Industrial Electronics, 10-13 Nov. 2008, pp. 1608 – 1613
40. Lafferriere, G., Williams, A., Caughman, J., Veermany, J.: Decentralized Control of Vehicle Formations. Systems & control letters 54(9), 899-910 (2005).
41. Desai, J., Ostrowski, P., Kumar, V.: Modeling and Control of Formations of Nonholonomic Mobile Robots. TRANSACTIONS ON ROBOTICS AND AUTOMATION 17(6), 905-908 (2001).
42. Barnes, L., Fields, M., Valavanis, K.: Unmanned ground vehicle swarm formation control using potential fields. In: Mediterranean Conference on Control & Automation, Athens 2007, pp. 1-8
43. Barnes, L., Fields, M., Valavanis, K.: Swarm Formation Control Utilizing Elliptical Surfaces and Limiting Functions. IEEE Transactions on Systems, Man and Cybernetics 39(6), 1434-1445 (2009).
44. Xue, Z., Zeng, J.: Formation Control Numerical Simulations of Geometric Patterns for Unmanned Autonomous Vehicles with Swarm Dynamical Methodologies. In: International Conference on Measuring Technology and Mechatronics Automation, Zhangjiajie 2009, pp. 477 - 482
45. Gazi, V.: Swarm aggregations using artificial potentials and sliding-mode control. IEEE Transactions on Robotics 21(6), 1208 - 1214 (2005).
46. Ge, S., Fua, C.: Quenses and Artificial Potential Trenches for Multirobot Formations. IEEE Trans. on Robotics and Automation 21(4), 646 - 656 (2005).
47. Kowalczyk, W., Kozlowski, K.: Artificial potential based control for a large scale formation of mobile robots In: Proceedings of the Fourth International Workshop on Robot Motion and Control 2004, pp. 285 - 291
48. Jin, Y., Meng, Y. anf Guo, H.: A Morphogenetic Self-Organization Algorithm for Swarm Robotic Systems using Relative Position Information. (2010).
49. Guo, H., Jin, Y., Meng, Y.: A unified Framework for Self-organized Multi-Robot Pattern Formation and Boundary Coverage Inspired from Morphogenesis. ACM Transactions on Adaptive and Autonomous Systems (2010 (submitted)).
50. Jin, Y., Meng, Y., Guo, H.: A morphogenetic self-organization algorithm for swarm robotic systems using relative position information. In: UK Workshop on Computational Intelligence (UKCI), Colchester 2010, pp. 1-6
51. Jin, Y., Meng, Y.: Morphogenetic Robotics: An Emerging New Field in Developmental Robotics. IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews, 1-16 (2010).
52. Sayama, H.: Robust Morphogenesis of Robotic Swarms. IEEE Computational Intelligence Magazine 5(3), 43-49 (2010).