EPiC Series in Computing
Volume 89, 2022, Pages 81–91
Proceedings of 35th International Conference on Computer Applications in Industry and Engineering

Multi-Agent Polygon Formation via Circle Formation

Rui Yang, Azad Azadmanesh and Hassan Farhat

University of Nebraska, Omaha, USA
ryang@unomaha.edu, azad@unomaha.edu, hfarhat@unomaha.edu

Abstract

This study considers the convex formation of polygons via a two-phase procedure. The approach infuses features from the behavioral and virtual structure methodologies. In the first phase, the agents form a circle. In the second phase, the agents are reconfigured into a polygon formation. Since the reconfiguration of virtual structures are often faced with challenges, the circle formation is adopted as the regrouping feature of agents before reconfiguration into a different polygon formation. No distinction is made among the agents, which simplifies formations. In addition, the agents can avoid collision during the formation process. Simulation results show precise formation of agents into different polygons. The results further indicate that the proposed approach has the potential to maintain the formation whilst the formation is rotating or changes location.

1 Introduction

Networked multi-agent systems (MASs) have been used in several disciplines to address various research problems. Some examples are cooperative control of unmanned aerial vehicles (UAVs), autonomous underwater vehicles (AUVs), and surveillance and reconnaissance missions to accomplish a common goal [1, 5, 11, 13]. A mainstream of research in MASs is on the structure (formation) of agents as the result of their collective interactions with the neighbors. In other words, each agent following a control mechanism and allowing a distributed control in which each agent communicates with its neighbors only, which is called local interaction, the agents can position themselves to form a particular geometric formation.

The three common approaches to formation controls are: leader-follower, behavioral, and virtual structure. These approaches may also be combined with artificial intelligence approaches for improving flexibility and performance. In the leader-follower approach [1, 10], some agents act as leaders and the rest as following the leaders. The leaders' trajectories (e.g., a formation pattern) are transformed by the followers into coordinates under local control laws (e.g., keeping a certain distance from the leaders). Although the salient feature of this approach is its simplicity, the leader-follower approach suffers from having single points of failures, e.g., if a leader fails, the formation becomes difficult if there are substantial number of agents. In addition, there are no feedback from the followers to the leaders, e.g., if a follower runs into an obstacle and is not able to inform the leader (s). In the behavioral approach [6, 13], the agents follow a set of predefined control laws to control their behavior in certain conditions such as avoiding collision, avoiding obstacles, and keeping a certain distance from the neighbors. An advantage of this approach is its flexibility in systems with a substantial number of agents. A disadvantage is the complexity of mathematical analysis to create control laws to guarantee precise formation.

In the virtual structure formation [7, 13], the structure is treated as a virtual rigid structure, e.g., a circle. The agents' positions are determined according to reference points on the rigid structure. If the agents can track the reference points, the formation can be kept. In addition, if the agents can follow their own specific reference points, the precise formation of the structure can be maintained. Several studies have focused on the same formation since it becomes difficult to reconfigure the formation into a different virtual structure if reconfiguration is needed often.

This study borrows elements from the behavioral and virtual structures to form polygons. The behavioral approach is utilized to formulate control laws for traveling and avoiding collisions based on local interaction with the neighbors. The virtual structures are embedded by points that are formed into polygons. Agents travel to the coordinates of these points. The agents determine the coordinates on the polygons individually and in a distributed manner.

To form a polygon, a two-phase approach is employed. In the first phase, the agents are randomly distributed in a field. Using the local control laws, the agents will then form a circle. The agents will also have the option to uniformly distribute themselves on the circle. In the second phase, each agent determines the coordinates of its assigned point on the polygon. An agent's assigned point on the polygon is the shortest trajectory from its location on the circle to the polygon. Taking advantage of linear function properties, the intersection of the trajectory and the polygon side will determine the coordinates of the point.

The two-phase approach is adopted to make the reconfiguration into polygons and other non-polygons simpler. The circle formation is used since it is the pre-requisite to a number of symmetric formations, as will be seen later in the study. Noteworthy is that if the agents are uniformly distributed on the circle, they will also be uniformly distributed on the polygons.

This study is restricted to two-dimensional space and is the continuation of the work done in [12]. The study is mostly concerned with the second phase since the first phase has already been accomplished in [12]. Thus, Section 2 addresses some background information about circle formation in addition to some research works on polygon formations. Section 3, which is the focus of this study, deals with the geometric calculations for determining various parameters and expressions needed in forming polygons. Section 4 is about the simulation of the results in Section 3. The section provides some formation examples along with reconfiguration of a polygon into other polygons. Section 5 concludes the study with a summary and some avenues for future studies.

2 Background

The subject of pattern formations and in general formation control has been investigated in numerous studies. A number of these studies have been devoted to polygon formations. Among these, [3] proposes an algorithm for a group of homogenous robots that gradually form into patterns such as polygons and circles. Through local interactions and differentiating tasks that each robot plays, the authors have shown that the robots can start making simple patterns such as a line that gradually turns into more complex patterns like a circle and then into polygon patterns. Although the authors claim that it is theoretically possible to generate more complex patterns, the approach is a time-consuming process, e.g., allowing the robots to find each other and forming themselves into a line from randomly distributed robots.

The focus of [8] is on pattern formation via machine learning, specifically using the Q-learning algorithm. In this process, the agents are rewarded based on their actions. An agent is rewarded with the highest reward once it reaches its final target, and it is penalized if it moves away from the target. The authors use six agents to form the vertices of a hexagon. For polygons with lower sides such as a square or a triangle, some agents are removed. So, the number of agents must represent the number of vertex points, and not able to form polygons with the number of agents higher than the sides. The study does not allude to any discussion on collision or obstacle avoidance.

The study in [2] proposes a distributed control strategy to form regular polygons with a specified scale and with arbitrary number of agents. Polygon formations are achieved using local measurements. These measurements are relative and cyclic in the sense that an agent k's neighbors are agents $k-1$ and $k+1$. Under this sensing strategy, each agent moves toward the end point of a vector that is perpendicular to the midpoint of the line segment that is connecting the agent with its neighbor. Like [8], the agents forming the polygons are stationed only at the vertices of the polygons. So, if there are n agents, the formation will be a n-sided polygon. Additionally, the agents are treated as point agents, so collision avoidance is not considered.

In [4], a two-stage process for forming static polygons is presented. The approach is based on the leader-follower principle. In the first stage, the leader agent provides the orientation and the distance information for each agent with respect to itself (the leader). In the second stage, the robots are moved in circulation motion until they are uniformly distributed. In this stage, the distance between any agent and the leader, and the angle formed between any two neighbors with respect to the leader are updated repeatedly until the formation stabilizes. Like the previous discussions, the agents' final positions are at vertices of the polygons.

Similar to the work in [2], the authors in [14] assume the sensing topology is cyclic, so that each agent has only two neighbors. No discussion is made as to how the sensing topology is established or can be established. In this study, some external control input is injected to specific agents (vertex agents) in accordance with the desired formation. In their approach, since local measurements are used, the agents only need to keep their orientations with respect to their nearest neighbors.

Contrary to these studies, this study does not assume any predefined relationship among the agents, does not use any leader-follower strategy, does not assign any agents to the polygon corners (i.e., angles), and includes collision avoidance. In addition, the polygon formations are scalable as the agents projected to any polygon side travel in parallel to the side. However, the overhead delay in the first phase could increase as the number of agents increases since more agents translates into more collision avoidance operations. As indicated, a two-phase process is conducted. In the first phase, the agents form a circle[12]. In the second phase, they reconfigure themselves into a polygon. The Subsection 2.1 provides some background to form a circle based on the approach in [12].

2.1 Circle Formation

A number of studies exist for forming circles with various modeling assumptions such as global observation versus limited observation and collision free versus collision avoidance. The approach in [12] investigates the circle formation in a more formal fashion with realistic assumptions in mind. These assumptions include limited visibility, fully distributed, and collision avoidance while traveling to the final destinations. The only requirement, which is beyond the scope of this study, is to assume the agents can compute their positions using a coordinate position system.

The multi-agent system is a dynamic network in which the changes in the topology are caused by the mobility of the agents noted as $\{A_0, A_1, \dots, A_n\}$, where A_0 is the virtual agent representing the center of the circle and n is the number of real agents. Two agents A_i and A_j are neighbors if they are within the sensing range of each other. The distance between the two agents is denoted as D_{ij}. Any pair of agents needs to keep a minimum distance from each other to avoid collisions, which is called the collision distance (CD).

A two-dimensional coordinate system is used, in which the location of an agent i is (x_{a_i}, y_{a_i}). An agent i's mobility is considered continuous and its coordinates at any time t are $x_{a_i}(t)$ and $y_{a_i}(t)$. For simplicity, when there is no confusion, the coordinates are often shown as x_i and y_i.

The location of the circle center, as (x_0, y_0), is not fixed and is agreed upon by a consensus protocol among agents, by repeatedly exchanging and voting on the collected estimated circle centers from their immediate neighbors. Once decided on a circle center, no message communication to form a circle takes place. An agent A_i continuously travels toward the circle on the shortest path if it is not in the collision path with any of its neighbors. An agent i moves toward the circle by changing its coordinates according to the following:

$$x_{a_i}(t + dt) = x_{a_i}(t) + \Delta(r - D_{i0}) \cos \phi_i$$
$$y_{a_i}(t + dt) = y_{a_i}(t) + \Delta(r - D_{i0}) \sin \phi_i$$

where dt is a small-time value, Δ is a small positive value, r is the circle radius, and ϕ_i is the angle at the circle center formed by the agent with respect to 0°. If an agent i is in collision with some neighbors j as it moves toward the circle, it repels from those agents by changing its coordinates according to the following:

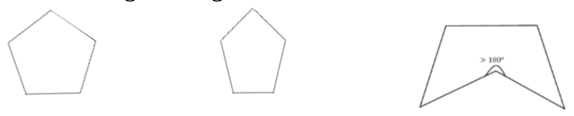
$$x_{a_i}(t + dt) = x_{a_i}(t) - x_{a_i}^{dif}(t)$$
$$y_{a_i}(t + dt) = y_{a_i}(t) - y_{a_i}^{dif}(t)$$

where $x_{a_i}^{dif}$ is the sum of $\left(x_{a_j} - x_{a_i}\right)/D_{ij}$ with respect to each agent j that is in collision course with. The $y_{a_i}^{dif}$ is defined similarly.

Once the agents are on the circle, they move counterclockwise on the circle keeping a distance called segment distance (SD) from each other. The SD is the size of the circle perimeter divided by the number of agents. Thus, the agents will be uniformly distributed on the circle. The SD can also be manually set if one wishes to form a larger circle. For further detail, the reader is referred to [12].

2.2 Polygon Formation

A polygon has a number of line segments that are connected to form a closed region. Polygons can be regular or irregular. In a regular polygon, every internal angle is of the same degree and every side is of the same length. Otherwise, the polygon is irregular. On the other hand, a polygon can be concave or convex. A concave polygon has at least one vertex pointed inward. In other words, it has an internal angle with a degree larger than 180°. Otherwise, the polygon is convex. Figure 1 shows an example of a five-sided (pentagon) polygon. Figure 1a and Figure 1b are convex, whereas Figure 1c is a concave polygon because one of its internal degrees is greater than 180°.



a) Convex regular b) Convex irregular c) Concave

Figure 1: Examples of polygon structures.

3 Regular Convex Formation

To have a firm understanding of how regular polygon formations with different number of sides can be formed, this section starts with the polygons that have the least number of sides. These are equilateral triangles, herein referred to as triangles for ease of reference. To generalize the approach for forming polygons with different sides, a pentagon will then be discussed as an example. Toward the end of the section, the discussion will turn to quadrilateral (or tetragon) polygons, which are polygons with four sides but are irregular. Specifically, the rectangle formation as a non-regular polygon will be discussed. This is to illustrate that the approach in this study has the potential to be applied to some non-regular polygons as well. Additionally, it will be shown that the rectangle formation can easily be reformed into a square, which is a special case of a rectangle formation, but it is a regular polygon.

The triangle formation lays out the groundwork for determining other regular formations. As indicated, to form a polygon formation, it is assumed that the agents have already formed a circle. In other words, a triangle formation is a two-step process. In the first step, the agents form a circle. The second step is about reconfiguration in which an agent, let's say with the coordinates (x_a, y_a), obtains its reconfigured location $(x_?, y_?)$ on the shortest path to the triangle. Once $(x_?, y_?)$ is determined, the agent moves to its new reconfigured location by assuming $(x_?, y_?)$ is the center of a circle with radius set to 0. This allows the agent to follow the same procedure for forming a circle, discussed in [12], but the agents end up forming a triangle without the need to perform any redistribution once they reach their specified locations $(x_?, y_?)$.

In the reconfiguration step, for the sake of simplicity, it is always assumed that the polygon formation is anchored at its lowest y coordinate, aligned with x_c coordinate value of the circle center, when rotated at its centroid. Figure 2 displays the structure of a triangle embedded in a circle with radius r. One of the agents with the coordinates (x_a, y_a) is shown on the circle. The lowest point (x_{anch}, y_{anch}) is the anchor point, which is at 270°. However, it will be shown shortly that the choice of the anchor coordinates (x_{anch}, y_{anch}) will have no bearing on the approach in acquiring $(x_?, y_?)$. In reality, the choice of the anchor is application dependent.

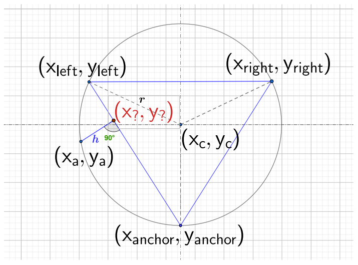


Figure 2: Structure of triangle formation.

For an equilateral triangle, once the coordinates or the angle for one of the vertices is determined, the coordinates for the other two vertices can be obtained. Since the angle of the anchor point is assumed at 270° and the vertices are 360°/ 3 = 120° apart from each other,

$$x_{right} = x_c + r\cos(x_{anch} + 120°) = x_c + rcos(30°) = x_c + 0.5r\sqrt{3} \tag{1}$$

$$y_{right} = y_c + r\sin(y_{anch} + 120°) = \mathrm{y_c} + r\sin(30°) = y_c + 0.5r \tag{2}$$

Similarly,

$$x_{left} = x_c + r\cos(x_{anch} + 240°) = x_c + r\cos(150°) = x_c - 0.5r\sqrt{3} \tag{3}$$

$$y_{left} = y_c + r\sin(y_{anch} + 240°) = \mathrm{y_c} + r\sin(150°) = y_c + 0.5r \tag{4}$$

At this point, it should be clear from (1) – (4) that regardless of where the anchor point is (x_{right}, y_{right}) and (x_{left}, y_{left}) can be determined easily.

To show the general approach in obtaining $(x_?, y_?)$, as shown in Figure 2, assume the agent is located on the circle between the left and the bottom vertices and let the equation for the triangle line formed between these two vertices be: $y_t = m_t x_t + b_t$. Similarly, let the equation for the line h on which the agent travels to reach $(x_?, y_?)$ be: $y_h = m_h + b_h$. The slope of y_t is:

$$m_t = \frac{y_{left} - y_{anch}}{x_{left} - x_{anch}} \tag{5}$$

On the other hand, the y-intercept of y_t is b_t, which can be obtained as:

$$y_{left} = m_t\, x_{left} + b_t \Rightarrow b_t = y_{left} - \frac{y_{left} - y_{anch}}{x_{left} - x_{anch}} x_{left} \tag{6}$$

Since the slope of y_h is the negative inverse of m_t,

$$m_h = -m_t^{-1} = -\frac{x_{left} - x_{anch}}{y_{left} - y_{anch}} \tag{7}$$

Since the location of the agent is known as (x_a, y_a),

$$y_h = m_h x_h + b_h \Rightarrow y_a = -m_t^{-1} x_a + b_h \Rightarrow b_h = y_a + m_t^{-1} x_a \tag{8}$$

Thus,

$$y_h = -m_t^{-1} x_h + (y_a + m_t^{-1} x_a) \tag{9}$$

Since $(x_?, y_?)$ is at the intersection of both lines y_t and y_h, $(x_?, y_?)$ should be valid for both lines. Thus, two equations with two unknowns are formed that can be solved to obtain $(x_?, y_?)$:

$$y_t = m_t x_t + b_t \Rightarrow y_? = m_t x_? + b_t \tag{10}$$

$$y_h = m_h x_h + b_h \Rightarrow y_? = m_h x_? + b_h \tag{11}$$

This leads to:

$$x_? = \frac{b_h - b_t}{m_t - m_h} \tag{12}$$

$$y_? = m_t \frac{b_h - b_t}{m_t - m_h} + b_t \tag{13}$$

Since b_t, b_h, m_t, and m_h are all known, $x_?$ and $y_?$ can be easily calculated. However, there are two special cases for when a line segment happens to be horizontal or vertical. The line is horizontal if $y_{left} = y_{anch}$. In that case, the travel path h for the agent is vertical, so the x coordinate of the agent stays the same. Therefore,

$$x_? = x_a$$
$$y_? = y_{left}$$

If the line segment is vertical, i.e., $x_{left} = x_{anch}$, the y coordinate does not change as the agent travels since the travel is horizontal. Thus,

$$x_? = x_{left}$$
$$y_? = y_a$$

From Figure 2, depending on where (x_a, y_a) is located, three cases are differentiated. Once it is determined on which side of the triangle the agent is located, equations (5)-(13) can be applied to obtain the reconfiguration point $(x_?, y_?)$. With this discussion, the following section generalizes the approach for any regular convex polygon.

3.1 Pentagon Formation

The process for forming a polygon is similar to what has been discussed for triangles. What follows shows an example for 5-sided polygons called regular pentagon (herein referred to as pentagon). The example will be used to generalize the approach to other regular, convex polygons with lower sides (e.g., triangles) or higher sides (e.g., octagon).

Like the discussion in the triangle formation, since there are five vertices, there are five xy coordinates that need to be determined. However, depending on where an agent is located on the circle, the agent evaluates and uses a pair of adjacent coordinates only.

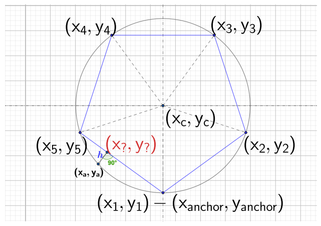


Figure 3: Structure of pentagon formation.

Figure 3 captures the structure of the pentagon along with the anchor point at 270°. Given a polygon with s sides, the adjacent vertices are $360°/s$ degrees apart from each other. So, in the case of the pentagon, the adjacent vertices are apart from each other by 72°. By considering the anchor point as vertex v_1 with coordinates (x_1, y_1) and numbering the other vertices counterclockwise, it follows that:

$$v_1: (x_1,\ y_1) = (x_{anchor},\ y_{anchor}) = (x_c + r\cos(270°)\ ,\ y_c + r\sin(270°))$$
$$v_2: (x_2,\ y_2) = (x_c + r\cos(270° + 72°)\ ,\ y_c + r\sin(270° + 72°))$$
$$v_3: (x_3,\ y_3) = (x_c + r\cos(270° + 2 \times 72°)\ ,\ y_c + r\sin(270° + 2 \times 72°))$$
$$v_4: (x_4,\ y_4) = (x_c + r\cos(270° + 3 \times 72°)\ ,\ y_c + r\sin(270° + 3 \times 72°))$$
$$v_5: (x_5,\ y_5) = (x_c + r\cos(270° + 4 \times 72°)\ ,\ y_c + r\sin(270° + 4 \times 72°))$$

Consequently, the vertex v_i for any polygon with α as the degree of the anchor point has the coordinates:

$$(x_i, y_i) = (x_c + r\cos(\alpha + (i-1) \times 360°/s)\, , y_c + r\sin(\alpha + (i-1) \times 360°/s)$$

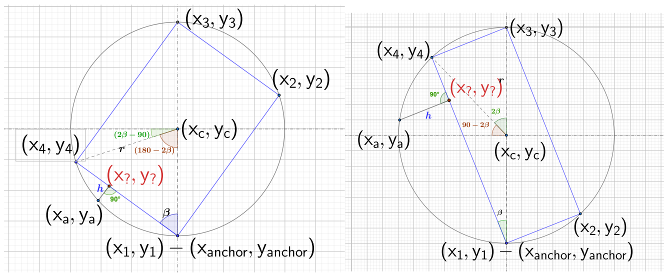
The coordinates can then be converted into their corresponding angle degrees using the inverse of cosine function:

$$deg_a = \arccos(x_a)$$
$$deg_{v_i} = \arccos\,(x_i)$$

To use (1) - (13), the agent needs to attain the coordinates of the end vertices of the polygon side that the agent will travel to. This is done by comparing the agent's degree against the degree of each pair of adjacent vertices until a match is found that satisfies $deg_{v_i} \leq deg_a \leq deg_{v_{i+1}}$, $deg_{v_i} \leq deg_a \geq deg_{v_{i+1}}$, or $deg_{v_i} \geq deg_a \leq deg_{v_{i+1}}$, where $i+1$ is done in modulo s. The latter two cases might happen because of the rotation from 360° back to 0°. Once a match is found on deg_{v_i} and $deg_{v_{i+1}}$, the agent will use the corresponding coordinates and apply them to (1)-(13).

3.2 Rectangle Formation

A rectangle is not a regular polygon because not all its sides are of the same length. But the discussion of it will lead us to the procedure for forming squares. Similar to the previous discussion, it is assumed that the reconfigured formation is anchored at its lowest point with the coordinates (x_{acnh}, y_{anch}) when rotated at its centroid that is aligned with (x_c, y_c), as shown in Figure 4.



a) Up-Left b) Up-Right

Figure 4: Rectangle formation cases.

There are two possibilities of rectangle formation as shown in Figure 4: Up-Left and Up-Right. In addition to the known parameters such as $(x_a,\ y_a)$, (x_c, y_c), α, and r, β needs to be input as well, which can vary between 0° and 90°. This degree determines how wide or how narrow the formation will be. In the case of Up-Left, the larger β becomes, the narrower the rectangle will be. In the case of Up-Right, the larger β causes the rectangle to become wider. At $\beta = 0°$ (Up-Right) or $\beta = 90°$ (Up-Left), the rectangle becomes a vertical line passing through the circle center with its length equal to $2r$. Consequently, one could argue that the rectangle formation can be used to form a straight-line formation vertically as well. However, if $\beta = 0°$ or $\beta = 90°$, there can be collisions on the line as the agents from both sides of the line approach the line. Therefore, the value of β should be such that the rectangle width stays at a value not smaller than the collision distance (*CD*). Furthermore, the value of β should never be set at either 0° or 90°.

On the other hand, if $\beta = 45°$, the rectangle turns into a square, which is a special case of a rectangle, with its four corners aligned at 0°, 90°, 180°, and 270°. Thus, UpLeft = UpRight. It should be noted that $45° \leq \beta \leq 90°$ makes the formation an Up-Left formation, whereas $0° \leq \beta \leq 45°$ leads to an Up-Right formation.

Because a rectangle is a four-sided polygon, there are four vertices to consider. Unlike the regular polygons discussed, β needs to be incorporated into the calculation since it controls the intended shape of the rectangle formation:

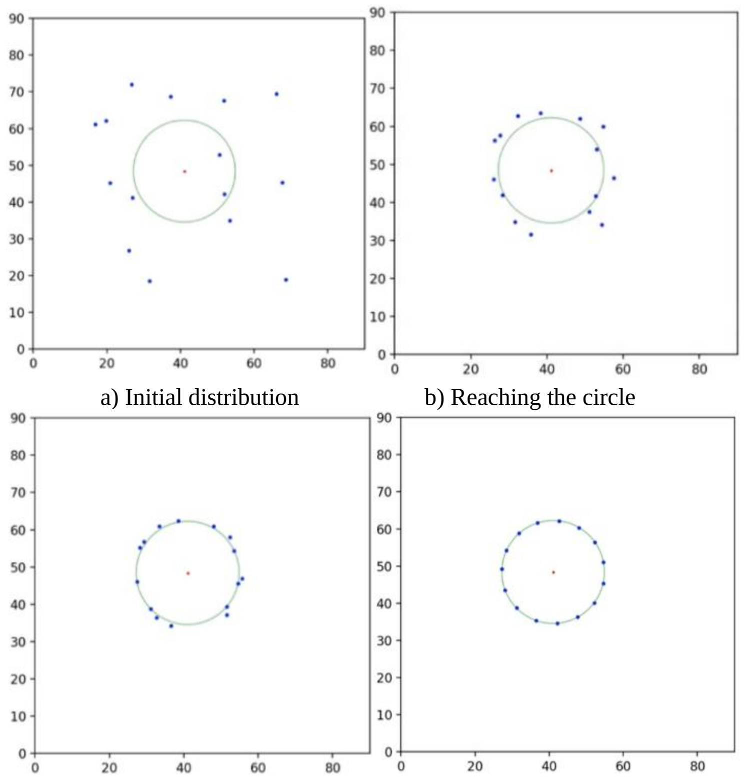
$$v_1: (x_{anchor}, y_{anchor}) = (x_1, y_1) = (x_c + r\cos\alpha\,, y_c + \sin\alpha)$$
$$v_2: (x_2, y_2) = (x_c + r\cos(\alpha + 2\beta)\,, y_c + \sin(\alpha + 2\beta))$$
$$v_3: (x_3, y_3) = (x_c + r\cos(\alpha + 180)\,, y_c + \sin(\alpha + 180°))$$
$$v_4: (x_4, y_4) = (x_c + r\cos(\alpha + 2\beta + 180°)\,, y_c + \sin(\alpha + 2\beta + 180°))$$

As mentioned in Section 3.2, deg_{v_i} and $deg_{v_{i+1}}$ control which vertices v_i and v_{i+1} to use. Once the two vertices are obtained, the process in (1)-(13) can be followed to acquire $(x_?, y_?)$.

4 Simulation

Polygon and circle formation can be used in a number of applications such as unmanned aerial vehicles (UAV) for surveillance and data collection. This section provides some simulation experiments in Python for the two steps of circle formation followed by the reconfiguration process to achieve a triangle formation, a rectangle, and then a polygon. Figure 5 displays four snapshots for achieving a circle formation. Figure 5a shows the initial, random distribution of the agents. Figure 5b shows the agents moving toward the circle while avoiding collisions. In Figure 5c, the agents have almost reached the circle. In Figure 5d, the agents have formed the circle and repositioned themselves into a uniformly distributed formation, while avoiding collisions.

During the entire process of formation, the agents are entirely distributed with no assistance from any external entity. The only external input received is the number of agents and the minimum collision distance for avoiding collision with their neighbors. The agents are optionally able to receive input as to how large the circle formations should be. It should be mentioned that the drawing of the circle in the figure is not necessary. It is merely drawn to assist in better observation.


a) Initial distribution b) Reaching the circle

c) Circle formation d) Uniformly distributed

Figure 5: Snapshots of circle formation: a) Initial distribution.

Figure 6 shows the snapshots of agents reconfiguring themselves into a triangle immediately following Figure 5d circle formation. In Figure 6a, the agents have determined their reconfigured locations $(x_?, y_?)$ and about to move toward those locations. In Figure 6b, triangle formation is clearly visible. Figure 6c illustrates the complete formation of agents into a triangle. The agents on the reconfigured formation are still uniformly distributed because they were distributed uniformly on the circle. If they were not, then their distribution on the triangle would not be uniform either.

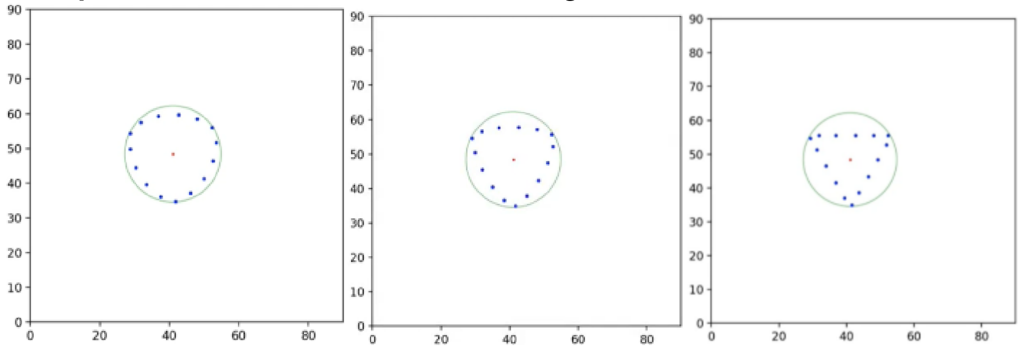
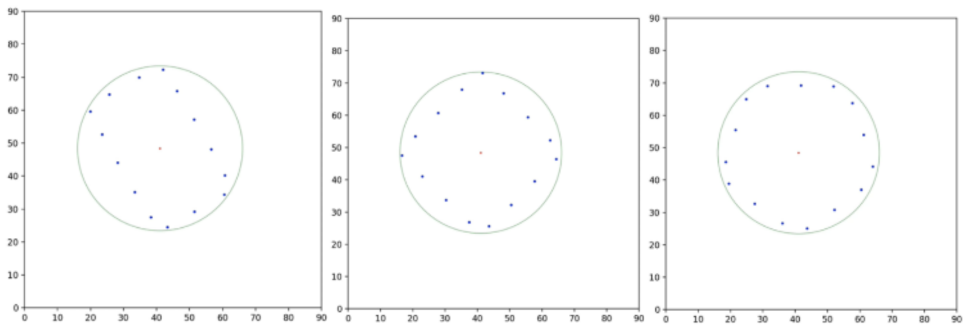

a) Starting to move b) Close to completion c) Triangle formed

Figure 6: Snapshots of triangle reconfiguration.

Figure 7 shows the continuation of Figure 6c, where the agents continuously reconfigure themselves to a rectangle, to a square, and finally to an octagon.



a) Rectangle b) Square c) Octagon

Figure 7: Snapshots of changing reconfiguration.

These simulations followed the process shown in Section 3. However, the study revealed other alternatives for obtaining $(x_?, y_?)$. Although the calculations are more involved, one approach that we have developed and simulated with success is taking advantage of the triangulation process. More specifically, once the coordinates v_i and v_{i+1} are determined, Heron's formula [9] is applied to find the distance h (e.g., see Figure 3). The location $(x_?, y_?)$ is then triangulated using the three circles centered at the agent, v_i and v_{i+1}.

5 Conclusion

This study has demonstrated a simple but practical approach for accomplishing polygon formations from randomly distributed agents in a field. The proposed methodology involves elements from the behavioral and virtual structures principles. The formation follows a two-phase process, which has played a fundamental role in improving performance and mitigating impractical assumptions. For example, once the circle formation is accomplished, the chances for collisions during the reconfiguration process to form a polygon is virtually non-existent since the agents allocated to each segment line move in parallel to their new locations on the polygon. The approach taken is flexible enough to form any regular convex polygon by retracting the agents from their current polygon formation to the circle before forming a different polygon. As illustrated, the approach can also be applied to some irregular polygons such as rectangles.

In contrast to a number of studies, the number of polygon sides does not depend on the number of agents deployed. Furthermore, no distinction is made between agents for conducting special tasks or allocating some agents to form the polygon vertices.

Several future studies are anticipated. These include adding more flexibility to the current work, such as the ability to form asymmetric polygon and irregular polygons. Another avenue is to turn a static polygon into a dynamic one by allowing each agent on a polygon to change position relative to the movement of the polygon centroid.

References

[1] Chen X., A. Serrani, H. Ozboy, "Control of leader-follower formation of terrestrial UAVs", *IEEE Conf. on Decision and Control*, pp. 498-503, 2003.

[2] Fathian K., N.R. Gans, W. Krawcewics, D. Rachinskii, "Regular polygon formations with fixed size and cyclic sensing constraint", *IEEE Transactions on Automatic Control*, 64(12), pp. 5156-5163, 2019

[3] Ikemoto Y. Y. Hasegawa, T. Fukuda, K. Matsuda, "Gradual spatial pattern formation of homogenous robot group", *Information Sciences: An Int'l J.*, 17(14), pp. 431-445, 2005

[4] Issa B.A., A.T. Rashid, M.T. Rashid, "Leader-neighbor algorithm for polygon static formation control", *Int'l Conf. on Electrical Communication, and Computer Eng.*, 2020.

[5] Kopfstedt T., M Mukai, M. Fuita, C. Ament, "Control of formations of UAVs for surveillance and reconnaissance missions", *IFAC Proceedings Volumes*, 41(2), pp. 5161-5166, 2008.

[6] Lawton J.R. Beard R.W., B. Young, "A decentralized approach to formation control maneuvers", *IEEE Transactions on Robotics and Automation*, 19(6), pp. 933-941, 2004.

[7] Low C.B., "A dynamic virtual structure formation control for fixed wing UAVs", *IEEE Int'l Conf. on Control and automation*, 2011.

[8] Prasad B.K.S., A.G. Manjunath, H. Ramasangu, "Multi-agent polygon formation using reinforcement learning", *Int'l Conf. on Agents and Artificial Intelligence*, 2017.

[9] Raifaizen C.H., "A simple proof of Heron's formula", *Mathematics Magazine*, 44(1), pp. 27-28, 1971.

[10] Ren W., "Consensus strategies for cooperative control of vehicle formation", *Control Theory and Applications*, 1(2), pp. 505-512, 2015.

[11] Tan Y.H., S. Lai, K. Wang, B.M. Chen, "Cooperative control of multiple unmanned aerial systems for heavy duty carrying", *Annual Reviews in Control*, 46, pp. 44-57, 2018.

[12] Yang R., A. Azadmanesh, H. Farhat, "A new approach to circle formation in multi-agent systems", *Int'l Conference on Wireless Networks*, 2019.

[13] Yu Ziquan, Y. Zhang, B. Jiang, J. Fu, Y. Jin, "A review on fault-tolerant cooperative control of multiple unmanned aerial vehicles", *Chinese J. of Aeronautics*, 35(1) pp. 1-18, 2022.

[14] Zhou B., Q. Yang, L. Dou, H. Fang, J. Chen, "An attempt to self-organized polygon formation control of swarm robots under cyclic topologies", *IFAC-PapersOnLine*, 53(2), pp. 11000-11005, 2020.