Polygon Formation in Distributed Multi-Agent Systems

Rui Yang*, Azad Azadmanesh*, and Hassan Farhat*
University of Nebraska, Omaha, NE USA

Abstract

A primary challenge to form a geometric pattern for a fleet of randomly distributed agents is the determination of their locations on the boundary of a formation. The challenge is more acute if the agents need to be uniformly distributed on the formation. This study addresses this challenge for polygon formations via a two-phase procedure. In the first phase, the agents form an enclosing circle to the location of the formation. This prevents the agents from location-conflicts on the polygon and hence each agent can uniquely ascertain its projection point on the formation. In the second phase, the agents establish their projected points and move toward their locations on the polygon, while mitigating collisions. The circle formation is also used as a regrouping feature before the agents reconfigure themselves into a different polygon formation. The formation control laws developed are verified through simulation for circle formation, convex polygons, and some categories of concave polygons. The control laws include the cases for rotation, translation (relocation), and scaling of polygons as well.

Key Words: Consensus, control systems, cyber physical systems, formation control, multi-robot systems, peer-to-peer networks.

1 Introduction

The research in networked multi-agent systems (MASs) has flourished in recent years due to their potential in various applications [2, 3, 12, 19, 20]. Some examples are cooperative control of unmanned aerial vehicles (UAVs), autonomous underwater vehicles (AUVs), and surveillance and reconnaissance missions to accomplish a common goal [4, 9, 18, 22, 24]. A mainstream of research in MASs is on the structure (formation) of agents in a distributed manner. More specifically, each agent following a control mechanism communicates with its neighbors only. Therefore, the view of the field by each agent is local not global. Through the local interactions, the agents can form a particular geometric pattern [16, 23].

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 [4, 15], 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. For example, if a leader fails, the formation becomes difficult. In addition, there is no feedback from the followers to the leaders, e.g., a follower is not able to inform the leader (s) if it runs into an obstacle. In the behavioral approach [10, 24], 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 [11, 24], 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 preserved. 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, particularly 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 neighbors. 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 circumscribing the location of the polygon structure. The agents will also have the option to uniformly distribute themselves on the circle. In the second phase, each agent identifies the coordinates of its projected point on the polygon. Taking advantage of linear function properties, an agent's projected point is the intersection of its closest trajectory toward the circle center and the polygon.

The two-phase approach is adopted to make the reconfiguration into polygons simpler. The enclosing circle formation provides conflict free projected points. Besides, the

*E-mail: ryang@unomaha.edu, azad@unomaha.edu, hfarhat@unomaha.edu.

circle formation can be used as the pre-requisite to a number of symmetric formations and in general regular polygons, as will be seen later in the study. Noteworthy is that if the agents are uniformly distributed on the circle and the circle passes through the polygon vertices, the agents, can be uniformly distributed on the polygons as well.

This study is restricted to two-dimensional space and is the continuation of the work done in [23]. The focus in [23] was on the first phase, i.e., on circle formation, whereas this study is mostly concerned with the second phase. Section 2 sets the network model and provides the background information on circle formation based on [23], in addition to some research works on polygon formations. Section 3 introduces some perspectives on polygons and the approaches defining the circle that circumscribes a polygon. Section 4 lays out the foundation on polygon formations. It discusses the general approach for various convex and concave polygon formations, which can be applied to the classic regular polygons such as squares and equilateral triangles as well. Section 5 discusses polygon transformations such as rotation, relocation (translation), and scale adjustment of polygons while maintaining their formation. Section 6 is about the simulation of the results in Sections 4 and 5. The section includes some examples of reconfiguration and transformation of polygons. Section 7 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, [7] 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 [13] 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 [5] 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 [13], 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 [8], 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 the vertices of the polygons.

Like the work in [5], the authors in [25] 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 their approach, some external control input is injected to specific agents (vertex agents) in accordance with the desired formation. 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 vertices, and includes collision avoidance. In addition, the work herein specifically discusses regular versus irregular polygons, convex versus concave, and provides tangible insight into rotation, translation, and scaling of polygons. An advantage of the work is that most of the delay and collision avoidance operations take place during the first phase. Once the circle is formed, the delay and collisions in the polygon formation including polygon reconfigurations, rotation, translation, and scaling are negligible since the agents' transitions occur with the same relative distance from their neighbors. However, the overhead delay in the first phase could increase, e.g., more agents translates into more collision avoidance operations.

2.1 Circle Formation

Studies on circle formation make various modeling assumptions such as global observation versus limited observation and collision free versus collision avoidance. The approach in [23] investigates the circle formation in a more formal fashion with realistic assumptions in mind. These assumptions include limited visibility, autonomous behavior, 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 positioning system.

In [23], 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 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 A_i is (x_{a_i}, y_{a_i}) and its mobility at any time t is shown by its coordinates as $x_{a_i}(t)$ and $y_{a_i}(t)$. For simplicity, when there is no confusion, the time is dropped but assumed, e.g., as x_{a_i} and y_{a_i}.

The location of the circle center, as (x_c, y_c), needs to be agreed upon by an appropriate consensus protocol among agents. The circle center (x_c, y_c) might also be shown as (x_0, y_0). Once agreed on a circle center, no message communication to form the circle takes place. An agent A_i continuously travels toward the circle boundary on the shortest path if it is not in the collision path with any of its neighbors. The agent moves toward the circle by changing its coordinates according to the following:

$$x_{a_i}(t + dt) = x_{a_i}(t) + \Delta_d(r - D_{i0}) \cos(Angle_{i0}(t)) \quad (1)$$

$$y_{a_i}(t + dt) = y_{a_i}(t) + \Delta_d(r - D_{i0}) \sin(Angle_{i0}(t)) \quad (2)$$

where dt is a small-time value, Δ_d is a small positive value, r is the circle radius, and $Angle_{i0}$ is the angle formed at the circle center (x_0, y_0) with respect to agent A_i. In other words, the angle $Angle_{i0}$ has the vertex at (x_0, y_0) with the end-legs of the angle at the coordinates $(x_c + r, y_c)$ and (x_{a_i}, y_{a_i}). If an agent A_i is in collision with some neighbors A_j as it moves toward the circle, it repels from those agents by changing its coordinates according to the following:

$$x_{a_i}(t + dt) = x_{a_i}(t) - x_{a_i}^{dif}(t) \quad (3)$$

$$y_{a_i}(t + dt) = y_{a_i}(t) - y_{a_i}^{dif}(t) \quad (4)$$

where $x_{a_i}^{dif}$ is the sum of $(x_{a_j} - x_{a_i})/D_{ij}$ with respect to each agent A_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 change the scale of the circle. An agent A_i moves counterclockwise on the circle according to the following:

$$x_{a_i}(t + dt) = x_c(t) + r \cos\big(Angle_{i0}(t) + \Delta_\phi \phi\big) \quad (5)$$

$$y_{a_i}(t + dt) = y_c(t) + r \sin\big(Angle_{i0}(t) + \Delta_\phi \phi\big) \quad (6)$$

Since the circle center does not change with time, $x_c(t)$ and $y_c(t)$ can be replaced with x_c and y_c, respectively. The Δ_ϕ is a small percentage of ϕ, where $\phi = 360°$.

Because of the importance of (5) and (6) in the movement and rotation of agents discussed later, Figure 1 shows the process of obtaining (5) and (6) for better insight. The figure shows the situation where A_i at time t with its coordinates at $(x_{a_i}(t), y_{a_i}(t))$ would like to move to a new location $(x_{a_i}(t + dt), y_{a_i}(t + dt))$ on the circle at time $(t + dt)$ and make a change in its angle by $\Delta_\phi \phi$ (angular speed). At time t, the agent has the angle of $Angle_{i0}$ with respect to 0°. From the figure, since $r \cos\big(Angle_{i0}(t) + \Delta_\phi \phi\big)$ is the x-coordinate offset of the agent at time $(t + dt)$, adding the x-coordinate of the circle center results in the equation (5). Similarly, $r \sin\big(Angle_{i0}(t) + \Delta_\phi \phi\big)$ is the y-coordinate offset from the y-coordinate of the circle center, and so adding the offset to the circle center y-coordinate yields (6). It is important to realize that smaller values of $\Delta_\phi \phi$ provides for smoother transitions. Similarly, the transitions become smoother as dt is reduced, as opposed to discrete changes.

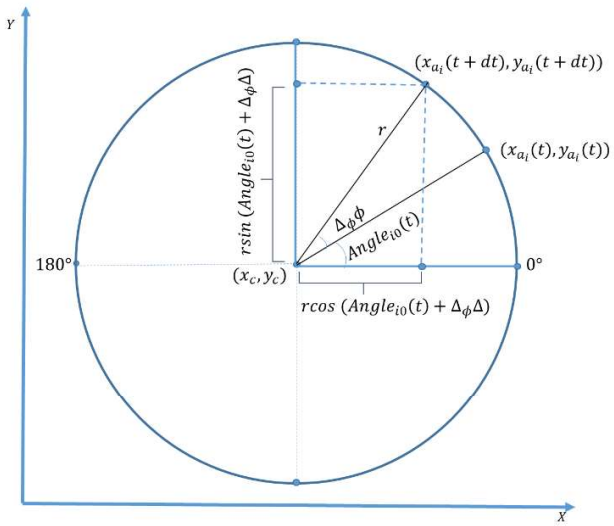


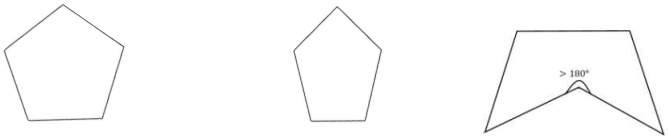
Figure 1: Travel of an agent A_i on the circle

For further detail on circle formation, the reader is referred to [23].

3 Preliminary Perspectives on Polygons

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.

A concave polygon can never be regular because the interior angles are of different measures. Another property of a concave polygon is that a line passing through the polygon can touch the polygon more than twice. In contrast, a line can cross a convex polygon at most twice. Figure 2 shows an example of a five-sided (pentagon) polygon. Figure 2a and 2b are convex polygons, whereas Figure 2c is a concave polygon because one of its internal degrees is greater than 180°.



a) Convex regular b) Convex irregular c) Concave

Figure 2: Examples of polygon structures

3.1 Circle Enclosed Polygons

Since this study employs a two-stage approach to polygon formations, it is necessary to discuss the notion of a circle enclosing a polygon. The circle enclosing a polygon can be the Smallest Enclosing Circle (SEC) that covers the entire polygon in the field [17, 21]. A polygon is inscribed in the SEC if all the vertices of the polygon are on the circle. It can also be said that the SEC circumscribes a polygon if the circle passes through the vertices of the polygon.

An inscribed polygon is called a *cyclic polygon*. Not all polygons are cyclic. All regular polygons can be inscribed and are thus cyclic. Some examples of regular polygons are squares, equilateral triangles, and equilateral pentagons. An irregular polygon can be cyclic as well. For example, all rectangles are cyclic.

For a MAS, if the polygon formation is irregular, obtaining the SEC may not be a feasible solution due to its involved mathematical algorithm. Even if the SEC can be obtained, the polygon may not be cyclic. Besides, obtaining the SEC may not be desirable as its circumference size may depend on the number of agents and the minimum distance between each pair of agents on the circle. In addition, as mentioned, concave polygons cannot be inscribed and thus not cyclic.

Consequently, a feasible approach would be to find a circle that may not be the SEC but reasonably covers the entire fleet of agents. Figure 3 illustrates how the circle can be obtained. The figure is a heptagon (7-sided polygon) that cannot be inscribed. The reason is that this heptagon is formed by replacing one of the regular (cyclic) hexagon sides by two outward sides. For the purpose of visualization, the regular hexagon is shown inside of the heptagon by thin broken lines. Therefore, this heptagon is not cyclic.

Since the coordinates of the heptagon needs to be known a-priori, the maximum and minimum x- and y-coordinates are used to form a rectangle. In the figure, the maximum x- and y-coordinates are x_3 and y_5, respectively. Similarly, the minimum x- and y-coordinates are x_6 and y_1, respectively. Using these

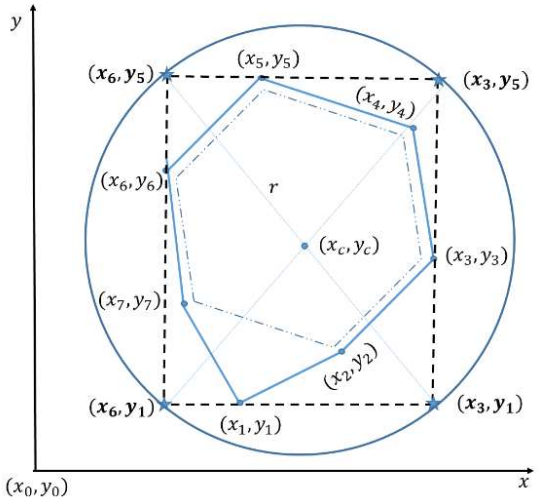


Figure 3: Example for finding a circle that covers a non-regular polygon

figure, the rectangle is shown as a thick dashed box. Its vertices are marked with stars along with their coordinates in bold. Generalizing this approach and using the subscripts of max and min in representing the maximum and minimum x- and y-coordinates, the vertices of the rectangle, starting from the lower left and moving counterclockwise, become: (x_{min}, y_{min}), (x_{max}, y_{min}), (x_{max}, y_{max}), and (x_{min}, y_{max}).

As the figure shows, since any rectangle is cyclic, a circle circumscribing the rectangle can be formed. Obviously, the circle covers all the agents as well. In addition, the rectangle can determine the coordinates of the circle center and the radius of the circle. As the example in the figure shows, the location of the circle center is halfway between the maximum and minimum x- and y-coordinates, In general,

$$(x_c, y_c) = \left(\frac{x_{min} + x_{max}}{2}, \frac{y_{min} + y_{max}}{2}\right)$$

Since the rectangle diagonal passes through the circle center, the circle radius is half the rectangle diagonal, i.e.,

$$r = \frac{\sqrt{(x_{max} - x_{min})^2 + (y_{max} - y_{min})^2}}{2}$$

It should be noted that, although the example in Figure 3 shows a convex polygon, the same approach can be used for concave polygons as well for obtaining the enclosed circle. Also, as shown in Figure 3, the formation field is assumed to be in the positive quadrant of the xy-plane.

4 Polygon Reconfiguration Phase

This section applies the formation approach discussed in the previous section to regular and non-regular polygons including concave polygons.

4.1 Regular Convex Polygons

To have a firm understanding of the approach, an example using a regular pentagon polygon is shown in Figure 4a. Since the polygon is regular, it is cyclic and thus can be inscribed by the SEC, or the approach described above may be used to obtain the circle for covering the polygon.

It is assumed that the agents have already formed a circle. Recall that a polygon formation with s sides is a two-phase process. In the first phase, the agents form a circle. The second phase is about the reconfiguration in which an agent, let's say with the coordinates (x_a, y_a), obtains its reconfigured location $(x_?, y_?)$ on the polygon. 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 [23], but the agents end up forming a polygon without the need to perform any redistribution once they reach their specified locations $(x_?, y_?)$.

In the reconfiguration process for regular polygons, it is assumed that the agents are aware of an anchor point with degree α. In Figure 4a below, $\alpha = 270°$, and the location of the anchor point is shown as (x_{anchor}, y_{anchor}). Furthermore, assume the polygon vertices are defined as v_k, $1 \le k \le s$, numbered counterclockwise, with the corresponding coordinates (x_k, y_k).

The figure shows an agent between v_5 and v_1 with the corresponding coordinates (x_5, y_5) and (x_1, y_1). For the sake of generality, let's refer to them as v_i and v_{i+1} with the corresponding coordinates (x_i, y_i) and (x_{i+1}, y_{i+1}), where $(i+1)$ is calculated in modulo s. For an agent to attain the coordinates of the end vertices v_i and v_{i+1}, the agent needs to compare its degree on the circle against the degree of each pair of adjacent vertices until a match is found that satisfies $deg_{v_i} \le deg_a \le deg_{v_{i+1}}$, $deg_{v_i} \le deg_a \ge deg_{v_{i+1}}$, or $deg_{v_i} \ge deg_a \le deg_{v_{i+1}}$, 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.

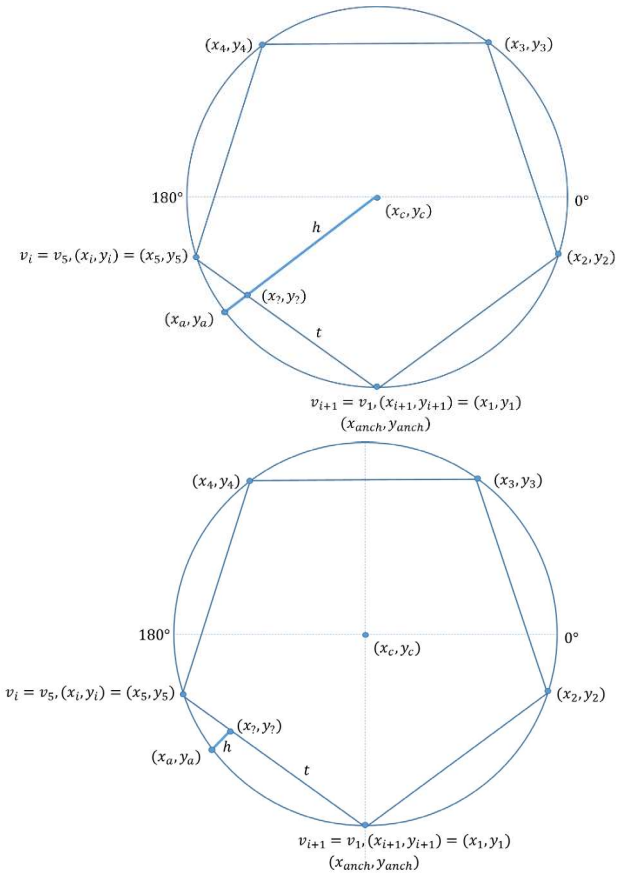
The following shows how to obtain the corresponding coordinates of a vertex v_i with degree deg_{v_i}:

$$(x_i, y_i) = (x_c + r\cos(deg_{v_i}), y_c + r\sin(deg_{v_i}))$$

where r is the circle radius. As an example, in Figure 4a, since the anchor point is at $\alpha = 270°$ and the adjacent vertices are $360°/s = 360°/5 = 72°$ degrees apart from each other,

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

To show the general approach in obtaining $(x_?, y_?)$ shown in Figure 4a, let the equation for the line t formed between these two vertices v_i and v_{i+1} be: $y_t = m_t x_t + b_t$. Similarly, let the



a) Using the trajectory to circle center
b) Using the perpendicular trajectory to the polygon

Figure 4: The approach for obtaining the reconfigured point $(x_?, y_?)$

equation for the line h between the circle center and the agent be: $y_h = m_h + b_h$.

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

$$y_t = m_t x_t + b_t \Rightarrow y_? = m_t x_? + b_t$$

$$y_h = m_h x_h + b_h \Rightarrow y_? = m_h x_? + b_h$$

This leads to:

$$x_? = \frac{b_h - b_t}{m_t - m_h} \quad (7)$$

$$y_? = \frac{m_t b_h - m_h b_t}{m_t - m_h} \quad (8)$$

Furthermore, the slopes of the lines are:

$$m_t = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} \quad (9)$$

$$m_h = \frac{y_c - y_a}{x_c - x_a} \quad (10)$$

And the y-intercepts of the lines are:

$$b_t = y_i - m_t x_i \quad (11)$$

$$b_h = y_a - m_h x_a \quad (12)$$

In (7) and (8), b_h, b_t, m_t, and m_h are known, as shown in (9) – (12). Also, (x_a, y_a), (x_i, y_i), and (x_c, y_c) are known. Consequently, $(x_?, y_?)$ in (7) and (8) can be calculated easily.

An interesting, revised version of the approach for obtaining $(x_?, y_?)$ is for an agent to continuously travel to the polygon on the shortest path. This implies that the trajectory h would be perpendicular to the polygon side t. This is shown in Figure 4b. The slope of y_t is:

$$m_t = \frac{y_i - y_{i+1}}{x_i - x_{i+1}} \quad (13)$$

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

$$y_i = m_t\, x_i + b_t \Rightarrow b_t = y_i - \frac{y_i - y_{i+1}}{x_i - x_{i+1}} x_i \quad (14)$$

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

$$m_h = -m_t^{-1} = -\frac{x_i - x_{i+1}}{y_i - y_{i+1}} \quad (15)$$

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

$$y_h = m_h x_h + b_h \Rightarrow y_a = -m_t^{-1} x_a + b_h \Rightarrow b_h = y_a + m_t^{-1} x_a \quad (16)$$

Thus,

$$y_h = -m_t^{-1} x_h + (y_a + m_t^{-1} x_a) \quad (17)$$

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

$$y_t = m_t x_t + b_t \Rightarrow y_? = m_t x_? + b_t \quad (18)$$

$$y_h = m_h x_h + b_h \Rightarrow y_? = m_h x_? + b_h \quad (19)$$

This leads to:

$$x_? = \frac{b_h - b_t}{m_t - m_h} \quad (20)$$

$$y_? = m_t \frac{b_h - b_t}{m_t - m_h} + b_t \quad (21)$$

Since b_t, b_h, m_t, and m_h are all known, $x_?$ and $y_?$ can be calculated easily as well. However, two special cases need to be checked for when a line segment happens to be horizontal or vertical. The line is horizontal if $y_i = y_{i+1}$. In that case, the travel path h for the agent is vertical, so the x-coordinate of the agent stays the same. Therefore,

$$x_? = x_a$$

$$y_? = y_i$$

If the line segment is vertical, i.e., $x_i = x_{i+1}$, the y-coordinate does not change as the agent travels since the travel is horizontal. Thus,

$$x_? = x_i$$

$$y_? = y_a$$

The discussion for Figure 4a and its revised version in Figure 4b are also applicable to cyclic non-regular polygons if the angles of the vertices are known to the agents. A notable ramification to the revised version is that if the agents are uniformly distributed on the circle, they would be distributed uniformly on the polygon structure as well.

4.2 Irregular Convex Polygons

The process for forming irregular convex polygons is like that of regular polygons. Figure 5 below shows a non-regular polygon inscribed in a circle. In this example, the circle circumscribing the polygon is the SEC. Compared to Figure 4, an agent (x_a, y_a) can easily apply the approach discussed in the previous subsection to determine its reconfiguration point $(x_?, y_?)$. However, in terms of input to the agents, since the polygon is irregular, having the location of the anchor point to determine the entire structure of the polygon is no longer sufficient. More specifically, since the polygon is not regular,

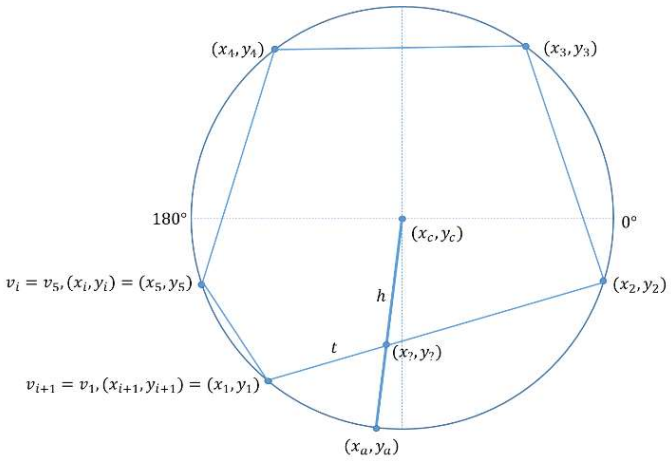


Figure 5: Example showing the approach for obtaining the reconfigured point $(x_?, y_?)$ for irregular polygons

the locations of the vertices need to be known before the reconfiguration can take place.

The approach for non-regular convex polygons that cannot be inscribed is the same, except the circle is not necessarily a SEC (see Figure 3). It should be noted that as the fleet of agents becomes larger the polygon formations become more pronounced.

The next section discusses the case for some categories of concave polygons.

4.3 Concave Polygons

Since concave polygons are not inscribable, a process similar to what was described in Section 3 needs to be applied to create a reasonable circle that covers the polygon vertices. Consider the two concave polygon examples in Figure 6 that have used the process in Section 3 for creating the circles around the vertices.

In Figure 6a, the trajectory between the agent location (x_a, y_b) and the circle center (x_c, y_c) intersects the polygon in one location. Therefore, the agent is able to determine its reconfigured point $(x_?, y_?)$ uniquely. However, in Figure 6b, the trajectory cuts through the polygon in multiple locations and thus it is not clear which intersection point to use as the reconfigured point $(x_?, y_?)$. The reason is that the degree of the polygon vertices in Figure 6b are not in increasing order. Even if the agent is able to randomly decide on one of the intersection points, the formation of the polygon would not be deterministic. Consequently, the formation of concave polygons in this research is limited to cases such as the one in Figure 6a. Specifically, the concave polygons considered are limited to those with $deg_{v_{i+1}} \geq deg_{v_i}$, where $1 \leq i \leq s$ and $(i+1)$ is in module s.

5 Polygon Transformations

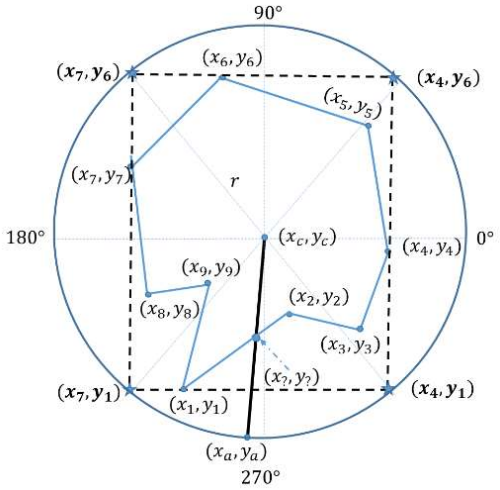
This section illustrates how the agents, once located on the polygon, would maintain the polygon structure while rotating in place, translating (moving) to a different location, rotating and translating simultaneously, or scaling in its final location of rotation and translation.

5.1 Polygon Rotation

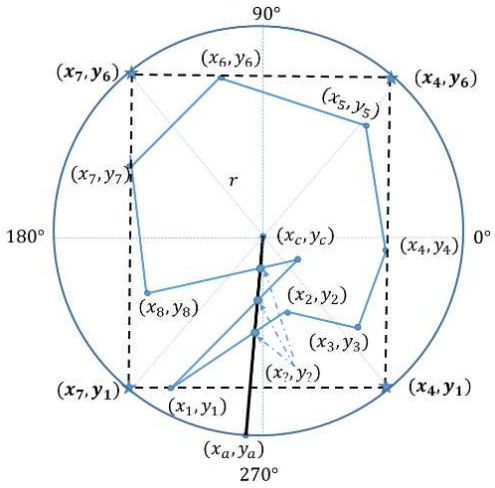
The algorithm for rotating the polygon structure in place is relatively simple since (5) and (6) have shown how to move an agent from one location to another on the circle boundary. These equations are modified slightly to account for incremental rotations. From Figure 1, the location of A_i at time t before being relocated is:

$$x_{a_i}(t) = x_c(t) + r\cos\big(Angle_{i0}(t)\big) \quad (22)$$

$$y_{a_i}(t) = y_c(t) + r\ \sin\big(Angle_{i0}(t)\big) \quad (23)$$



a)



b)

Figure 6: Agent trajectory for two examples of a concave polygon: a) cutting only one side of the polygon, b) cutting multiple sides of the polygon

After dt time, the agent's new location, as shown in Figure 1, is:

$$x_{a_i}(t+dt) = x_c(t) + r\cos\big(Angle_{i0}(t) + \Delta_\phi \phi\big) \quad (24)$$

$$y_{a_i}(t+dt) = y_c(t) + r\sin\big(Angle_{i0}(t) + \Delta_\phi \phi\big) \quad (25)$$

Taking advantage of cosine and sine properties, the cosine and sine portions of (24) and (25) can be expanded as:

$$\mathrm{Cos}\big(Angle_{i0}(t) + \Delta_\phi \phi\big) = \cos\big(\ Angle_{i0}(t)\big)\cos\big(\Delta_\phi \phi\big) - \sin\big(Angle_{i0}(t)\big)\sin\big(\Delta_\phi \phi\big) \quad (26)$$

$$\sin\big(Angle_{i0}(t) + \Delta_\phi \phi\big) = \cos\big(\ Angle_{i0}(t)\big)\sin\big(\Delta_\phi \phi\big) + \sin\big(Angle_{i0}(t)\big)\cos\big(\Delta_\phi \phi\big) \quad (27)$$

Substituting (26) in (24) yields:

$$x_{a_i}(t + dt) = x_c(t) + r\big(\cos\big(\ Angle_{i0}(t)\big)\cos\big(\Delta_\phi \phi\big) - \sin\big(Angle_{i0}(t)\big)\sin\big(\Delta_\phi \phi\big)\big) \quad (28)$$

From (22) and (23):

$$\Big(x_{a_i}(t) - x_c(t)\Big) = r\cos\big(Angle_{i0}(t)\big),$$

$$(y_{a_i}(t) - y_c(t)) = r\ \sin\big(Angle_{i0}(t)\big) \quad (29)$$

Substituting (29) in (28) yields:

$$x_{a_i}(t + dt) = x_c(t) + (x_{a_i}(t) - x_c(t))\cos(\Delta_\phi \phi) - (y_{a_i}(t) - y_c(t))sin(\Delta_\phi \phi) \quad (30)$$

Using (27) and taking advantage of (29), a similar substitution along the y-coordinate in (25) results in:

$$y_{a_i}(t + dt) = y_c(t) + (x_{a_i}(t) - x_c(t))\sin(\Delta_\phi \phi) + (y_{a_i}(t) - y_c(t))cos(\Delta_\phi \phi) \quad (31)$$

Note that the center of the formation is invariant under the rotation transformation equations (30) and (31). Additionally, the equations can be used to prove the transformation is a rigid-body transformation. That is, the agents on the same line of the polygon will be on the rotated line with the same relative locations and the same distances from each other with all the corresponding angles preserved.

5.2 Polygon Translation

Translation is the changes in x- and y-coordinate of a polygon location, which translates to the same changes of location for every agent on the polygon. In other words, translation is a displacement of location added to each of the agents' location. As a result, the formation topology is preserved as in the case of rotation. For instance, a translation by (x_{trns}, y_{trns}) for an agent A_i currently at location (x_{a_i}, y_{a_i}) results in the agent's new location at $(x_{a_i} + x_{trns}, y_{a_i} + y_{trns})$.

Consequently, an agent A_i with the coordinates $(x_{a_i}(t), y_{a_i}(t))$ and a desired translation by (x_{trns}, y_{trns}), at time $(t + dt)$, will have the new coordinates:

$$x_{a_i}(t + dt) = x_{a_i}(t) + x_{trns},\ y_{a_i}(t + dt) = y_{a_i}(t) + y_{trns} \quad (32)$$

Using (32) and the previous equations (30) and (31), the polygon formations can be rotated and translated incrementally until the desired rotation and translation are achieved. The order of transformation is application dependent. If translation by (x_{trns}, y_{trns}) is performed first, then the invariant point (the modified circle center) in (30) and (31) is adjusted first. The new updated (x_c, y_c) will be:

$$x_c(t + dt)x_c(t) + x_{trns} \quad , \quad y_c(t + dt) = y_c(t) + y_{trns} \quad (33)$$

5.3 Polygon Scaling

Scaling is a feature that enhances the flexibility of the polygon formation, by expanding or shrinking the area coverage of the formation about the center of the formation. To keep the topology of the formation unchanged, uniform scaling can be applied to each agent's location.

By fixing the center of the formation as the invariant location under uniform scaling, the shape of the formation is maintained. However, the agents' distances from the formation center and the relative distances among agents need to be adjusted appropriately.

If the agents' coordinates are measured relative to the origin coordinates $(0, 0)$, then uniform scaling is the process of multiplying the xy coordinates by a scale factor s. Under this transformation, $(x_{a_i}(t), y_{a_i}(t))$ is transformed at time $(t + dt)$ to:

$$\Big(x_{a_i}(t + dt), y_{a_i}(t + dt)\Big) = \Big(sx_{a_i}(t), sy_{a_i}(t)\Big) \quad (34)$$

For s between 0 and 1, the formation area is decreased. For $s > 1$, the formation area is increased.

In (34), the location (0, 0) is the invariant point under scaling. To make the center of the formation $(x_c(t), y_c(t))$ as the invariant point, (34) should be adjusted to:

$$\Big(x_{a_i}(t + dt), y_{a_i}(t + dt)\Big) = \big(x_c(t), y_c(t)\big) + s\big(x_{a_i}(t) - x_c(t), y_{a_i}(t) - y_c(t)\big) \quad (35)$$

$$= \big((sx_{a_i}(t) + (1 - s)x_c(t)), (sy_{a_i}(t) + (1 - s)y_c(t))\big) \quad (36)$$

Equations (35) and (36) are the concatenation of three transformations: a translation by $-(x_c(t), y_c(t))$, so that the scaling transformation by s in (34) can be applied, followed by a translation of $(x_c(t), y_c(t))$.

5.4 Incremental Polygon Transformation

The equations for the three primary transformations described above can be revised to achieve incremental transformations.

This allows the agents to move at a desired rate before settling on their final transformation. In addition, the incremental transformation can be applied to accomplish a composite transformation. That is, the formation structure can go under a sequence of rotation, translation, scaling, or any predefined order.

According to the translation properties, an incremental translation by (dx_{trns}, dy_{trns}) of duration dt followed by another incremental translation (dx'_{trns}, dy'_{trns}) of duration dt', and applied to the center of the formation $(x_c(t), y_c(t))$, produces a translation with the new center of formation as:

$$(x_c(t+dt+dt'), y_c(t+dt+dt') = (x_c(t) + dx_{trns} + dx'_{trns},, y_c(t) + dy_{trns} + dy'_{trns}) \quad (37)$$

Equation (37) shows that the concatenation of translations is additive. This result, and other similar results, can be proved using matrix representations of transformations applied to homogeneous coordinate system. A similar analysis can be applied to rotation, to show that a sequence of rotations about the same invariant point, results in a rotation by a degree that is the sum of the degrees used for the individual rotations. Unlike translation and rotation, composite scaling is not additive [1, 6].

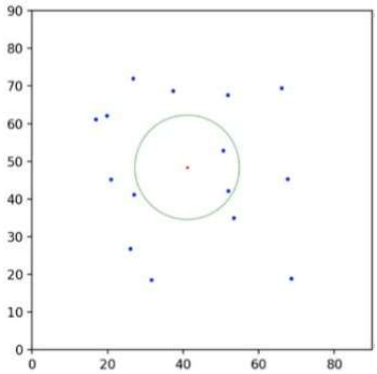
6 Simulation

This section provides some simulation experiments in accordance with the discussion in the previous section.

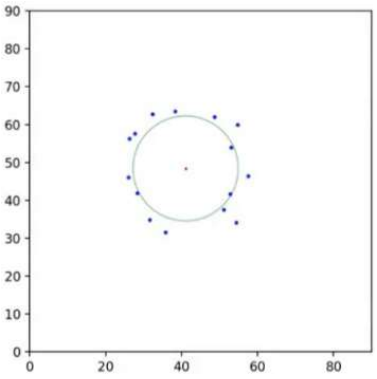
6.1 Two-Phase Formation

The Python simulations illustrate the two-phase approach of circle formation followed by the reconfiguration process for cyclic polygons to achieve a triangle formation, a rectangle, and then a pentagon. Figure 7 displays four snapshots for achieving a circle formation. Figure 7a shows the initial, random distribution of the agents. Figure 7b shows the agents moving toward the circle while avoiding collisions. In Figure 7c, the agents have almost reached the circle. In Figure 7d, the agents have formed the circle and repositioned themselves into a uniformly distributed formation, while avoiding collisions.

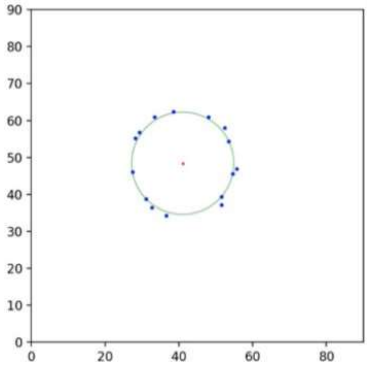
During the entire process of formation, the agents are entirely distributed with no assistance from any external entity. The only external input received is the number of agents and the minimum collision distance for avoiding collision with their neighbors. The agents are optionally able to receive input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



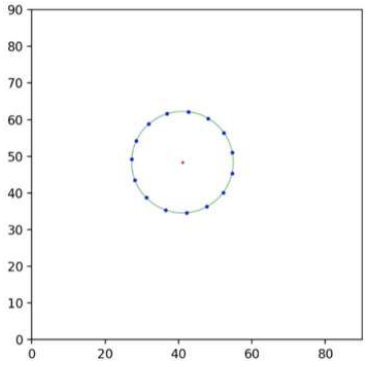
a) Initial distribution



b) Reaching the circle



c) Circle formation



d) Uniformly distributed

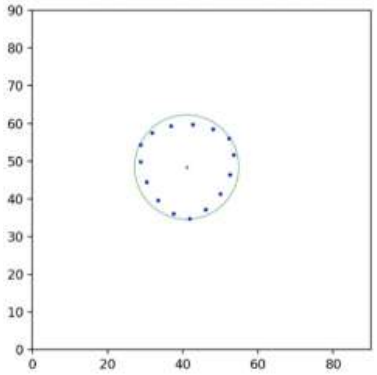
Figure 7: Snapshots of circle formation

necessary. It is merely drawn as a reference to assist in better observation.

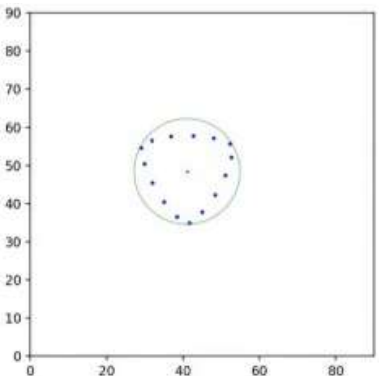
Figure 8 shows the snapshots of agents reconfiguring themselves into a triangle immediately following the Figure 7d circle formation. In Figure 8a, the agents have determined their reconfigured locations $(x_?, y_?)$ and about to move toward those locations. In Figure 8b, the triangle formation is clearly visible. Figure 8c 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 and move toward the polygon sides perpendicularly. If they were not, then their distribution on the triangle would not be uniform either.

Figure 9 shows the continuation of Figure 8c, where the agents continuously reconfigure themselves to a rectangle, to a square, and finally to a pentagon.

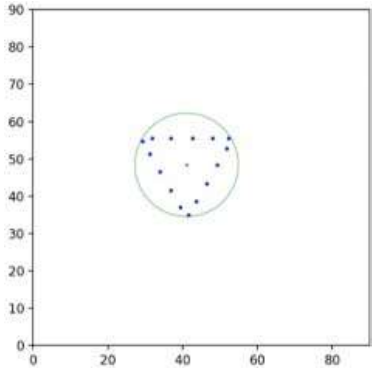
These simulations followed the process shown in Section 4. However, the study revealed other alternatives for obtaining $(x_?, y_?)$. Although the calculations are more involved, one approach that we have developed and simulated with success is taking advantage of the triangulation process. More specifically, once the coordinates of v_i and v_{i+1} are determined,



a) Starting to move

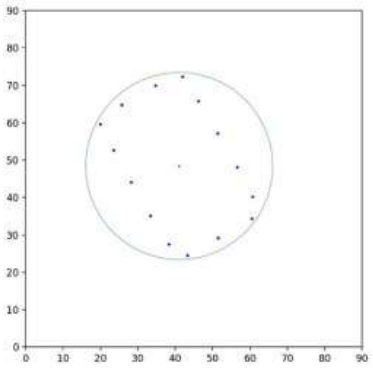


b) Close to completion

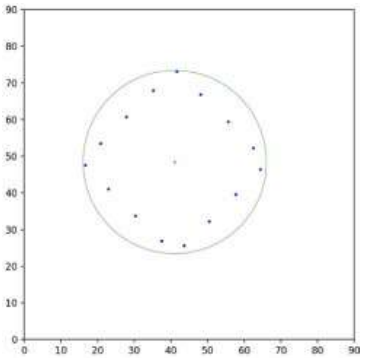


c) Triangle formed

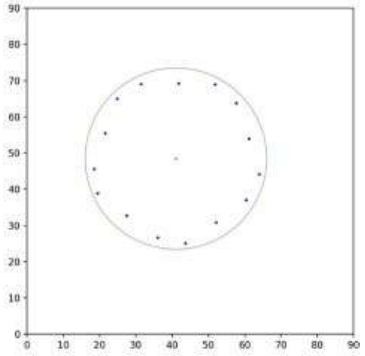
Figure 8 : Snapshots of triangle reconfiguration



a) Rectangle



b) Square



c) Octagon

Figure 9: Snapshots of changing reconfiguration

Heron's formula [14] is applied to find the distance h (see Figure 4b). The location $(x_?, y_?)$ is then triangulated using the three circles centered at the agent, v_i and v_{i+1}.

6.2 Transition of Formation through Transformations

Following the two-phase approach to the polygon formation, incremental transformation can be applied in the form of translation, rotation, or scaling of the entire formation, while preserving the formation structure. For a better insight and before providing some snapshots of a pentagon transformation in Subsection 6.3, the following illustrates the Excel implementation of rotation with translation using the incremental procedure described in Subsection 5.4.

Suppose it is desired to translate a circle formation from $(x_c(t), y_c(t))$ to the new location centered at $(x_c'(t'), y_c'(t'))$, for some time $t' > t$, and rotated by deg degrees. For this to happen, the formation is incrementally rotated by a small degree $d\phi$ followed by a small translation displacement (dx_{trns}, dy_{trns}). Suppose these steps are repeated n times to reach the desired transformation. Accordingly, each incremental transition uses:

$$d\phi = \frac{deg}{n}, \quad dx_{trns} = \frac{x_c(t')-x_c(t)}{n}, \quad dy_{trns} = \frac{y_c(t')-y_c(t)}{n},$$

$$dt = \frac{t'-t}{n} \tag{38}$$

In (38), $deg = \Delta_\phi \phi$ (See Figure 1). In addition, the number of steps n does not have to be the same for rotation, translation, and time. For example, if the number of steps for rotation and translation are n and m, respectively, where $n > m$, the rotation will continue without any translation for the remaining $(n - m)$ steps. The following shows the steps in carrying out the task:

For $k = 1$ to n do {

// Apply incremental rotation
For each A_i do {
$x'_{a_i}(t + dt) = (x_{a_i}(t) - x_c(t))\cos(d\phi) - (y_{a_i}(t) - y_c(t))sin(d\phi) + x_c(t)$ // See (30)
$y'_{a_i}(t + dt) = (x_{a_i}(t) - x_c(t))sin(d\phi) + (y_{a_i}(t) - y_c(t))cos(d\phi)$ // See (31)
}

// Apply incremental translation by updating the coordinates
For each A_i do {
$(x_{a_i}(t), (y_{a_i}(t)) = (x'_{a_i}(t + dt), y'_{a_i}(t + dt))$
}
$(x_c(t), y_c(t) = (x_c(t) + dx_{trns}, y_c(t) + dy_{trns})$

}

In the code above, it is important to update the center of the formation so as to maintain the correct rotation about the updated center of formation.

Figure 10 below shows the results of the above code execution. In the simulation, four agents are distributed around a circle of radius 3 centered at $(4, 6)$. The formation was rotated by 30° and translated by $(5, 3)$, with $n = 30$. The circles on the graph are not part of the simulation. They are drawn to enhance visualization.

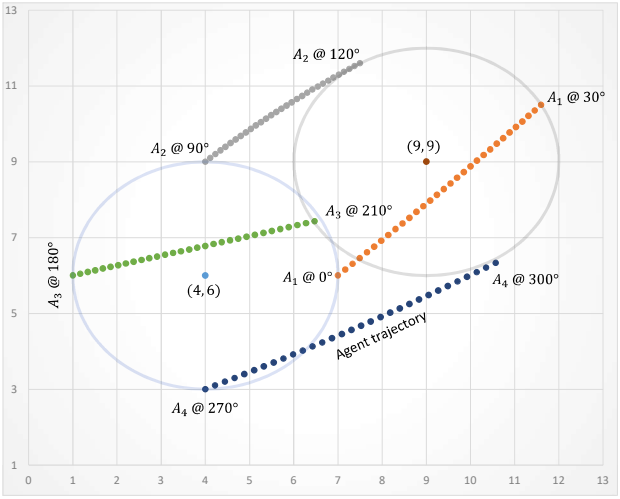
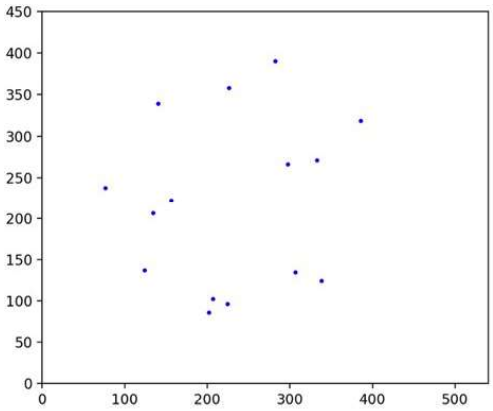


Figure 10: Rotation and translation of four agents

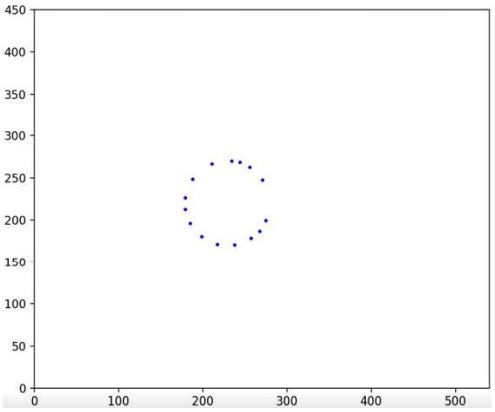
6.3 Pentagon Transformation

Having a better understanding of the discussion in the previous subsection, the following illustrates the formation of a pentagon followed by its transformation of rotation and translation as described in Section 5.

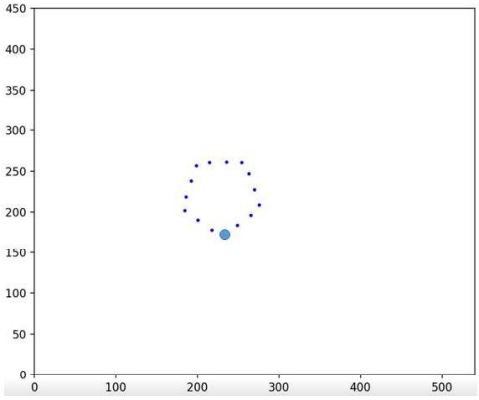
Figure 11a shows the initial distribution of some agents in a field. Figure 11b shows that the agents have formed a circle in phase 1 and attempting to distribute themselves on the circle boundary uniformly. In Figure 11c, the agents have formed a pentagon in phase 2. For better visualization in the follow up figures, the bottom agent is shown thicker. Figure 11d displays the fact that the pentagon has rotated to the left and moved to a different place while maintaining its shape. In Figure 11e, the pentagon has made about half a turn compared to Figure 11c. In Figure 11f, the pentagon has made a full turn, while moving to a different location. Once reaching its position in Figure 11f, the pentagon keeps rotating in place, but its rotation is not shown in this figure.



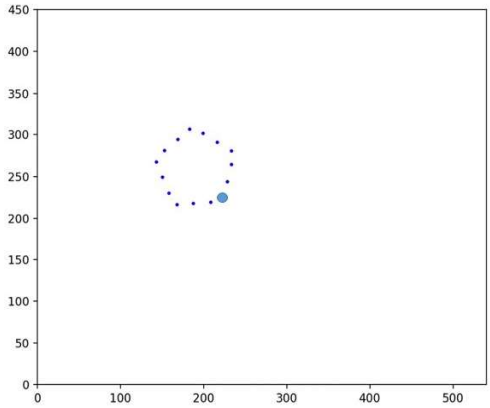
a) Initial distribution of agents



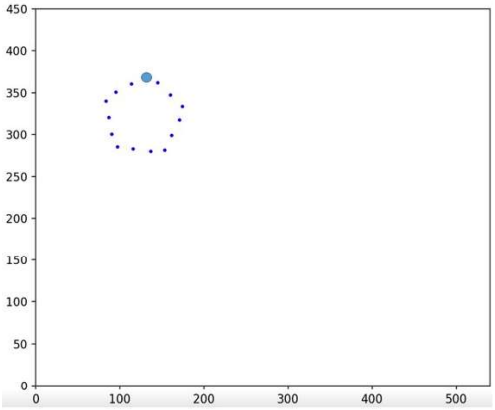
b) Agents attempting to distribute uniformly



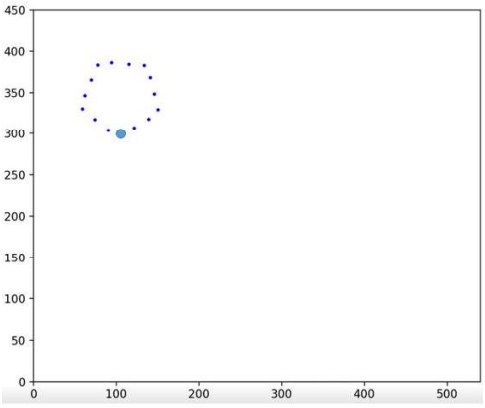
c) Agents forming the pentagon



d) The pentagon rotating while translating



e) The pentagon almost at half turn while translating



f) The pentagon making a full turn while translating

Figure 11: Transformation of a pentagon through rotation and translation

7 Conclusion

A major challenge in the design of multi-agent formations is the identification of the agents' positions on the formation structure. The proposed research offers a two-phase approach to polygon formations, which borrows elements from the behavioral and virtual structures principles. The approach has played a fundamental role in improving performance and mitigating the impractical assumptions. It enables the agents to identify their positions on a polygon uniquely, autonomously, and avoid collisions during the reconfiguration phase. In the first phase, the agents form an enclosing circle over the

formation structure. For better and more concise polygon formations, the agents have the option of uniformly distribute themselves on the circle. During the second phase, the agents reconfigure themselves into the desired polygon formation. In addition to polygon formations, the two-phase approach simplified the design for rotation, translation, and scaling of polygons.

The approach identified the types of polygons that can be formed. Specifically, the proposed approach handles convex and concave polygons under certation constraints. For example, concave polygons can be formed if the degrees of the vertices are in ascending order. As the number of agents increases, the structure of polygon formations become more pronounced, especially for concave polygons. In contrast to some studies, the number of agents deployed does not depend on that of polygon vertices. Furthermore, no distinction is made between agents for conducting special tasks or allocating special agents to form the polygon vertices.

Several future studies are anticipated. One is to remedy or reduce the current restrictions to include a wider range of concave polygons. Also, early results indicate that the current approach can be adjusted to better distribute the agents on non-cyclic polygons. Another avenue of research is to modify the control laws to handle faults, e.g., if an agent does not adhere to the consensus protocol or to the collision avoidance operations. In addition, the current research is testing the proposed approach in Robotic Operating System (ROS) as the stepping-stone to prototyping the control laws.

References

[1] E. Angle, *Interactive Computer Graphics*, 5th Ed., Addison Wesley, 2009.

[2] F. Dignum, P. Mathieu, J. M. Corchado, and F Prieta, "Advances in Practical Applications of Agents, Complex Systems Simulation. The PAAMS Collection," 20th Int'l Conf., PAAMS 2022, Springer, *LNCS*, 13616, 2022.

[3] A. Dorri, S.S. Kanhere, and R. Jurdak, "Multi-Agent Systems: A Survey," *IEEE Access*, 2018.

[4] X. Chen, A. Serrani, and H. Ozboy, "Control of Leader-Follower Formation of Terrestrial UAVs," *IEEE Conf. on Decision and Control*, pp. 498-503, 2003.

[5] K. Fathian, N. R. Gans, W. Krawcewics, and D. Rachinskii, "Regular Polygon Formations with Fixed Size and Cyclic Sensing Constraint," *IEEE Transactions on Automatic Control*, 64(12):5156-5163, 2019.

[6] D. Hearn and, M. P. Baker, *Computer Graphics*, 2nd Ed., Prentice Hall, 1994.

[7] Ikemoto Y. Y. Hasegawa, T. Fukuda, and K. Matsuda, "Gradual Spatial Pattern Formation of Homogenous Robot Group," *Information Sciences: An Int'l J.*, 17(14):431-445, 2005

[8] B. A. Issa, A. T. Rashid, and M. T. Rashid, "Leader-Neighbor Algorithm for Polygon Static Formation Control," *Int'l Conf. on Electrical Communication, and Computer Eng.*, 2020.

[9] T. Kopfstedt, M Mukai, M. Fuita, and C. Ament, "Control of Formations of UAVs for Surveillance and Reconnaissance Missions," *IFAC Proceedings Volumes*, 41(2):5161-5166, 2008.

[10] J. R. Lawton, R. W. Beard, and B. Young, "A Decentralized Approach to Formation Control Maneuvers," *IEEE Transactions on Robotics and Automation*, 19(6):933-941, 2004.

[11] C. B Low., "A Dynamic Virtual Structure Formation Control for Fixed Wing UAVs," *IEEE Int'l Conf. on Control and automation*, 2011.

[12] M. Oprea, "Applications of Multi-Agent Systems," *IFIP Congress Tutorials*, 2004.

[13] B. K. S. Prasad, A. G. Manjunath, and H. Ramasangu, "Multi-Agent Polygon Formation Using Reinforcement Learning," *Int'l Conf. on Agents and Artificial Intelligence*, 2017.

[14] C. H. Raifaizen, "A Simple Proof of Heron's Formula," *Mathematics Magazine*, 44(1):27-28, 1971.

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

[16] S. B. Sarsilmaz and T. Yucelen, "Control of Multiagent Systems with Local and Global Objectives," *IEEE Conf. on Decision and Control*, 2018.

[17] S. Skyum, "A Simple Algorithm for Computing the Smallest Enclosing Circle," *Information Processing Letters*, 37(3):121-125, 1991.

[18] Y. H. Tan, S. Lai, K. Wang, and B. M. Chen, "Cooperative Control of Multiple Unmanned Aerial Systems for Heavy Duty Carrying," *Annual Reviews in Control*, 46:44-57, 2018.

[19] A. Uhrmacher, D. Weyns, and P.J. Mosterman, "Multi-Agent Systems: Simulation and Applications," CRC Press, 2009.

[20] X. K. Wang, X. Li, and Z .Q. Zheng, "Survey of Developments on Multi-Agent Formation Control Related Problems," *Control and Decision*, 28(11):1601-1613, 2013.

[21] E. Welzl, "Smallest Enclosing Disks (Balls and Ellipsoids)," H. Maurer (ed.), New Results and New Trends in Computer Science, *LNCS*, 555:359-370, 1991.

[22] B. Xu, J. L. Bai, Y. L. Hao, W. Gao, and YlL. Liu, "The Research Status and Progress of Cooperative Navigation for Multiple AUVs," *Acta Automatica Sinica*, 41(3):445-461, 2015.

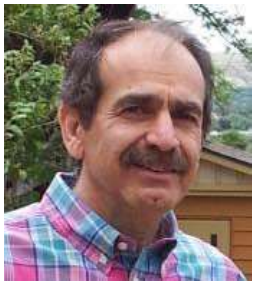
[23] R. Yang, A. Azadmanesh, and H. Farhat, "A New Approach to Circle Formation in Multi-Agent Systems," *Int'l Conference on Wireless Networks*, 2019.

[24] Ziquan Yu, Y. Zhang, B. Jiang, J. Fu, and Y. Jin, "A Review on Fault-Tolerant Cooperative Control of Multiple Unmanned Aerial Vehicles," *Chinese J. of Aeronautics*, 35(1):1-18, 2022.

[25] B. Zhou, Q. Yang, L. Dou, H. Fang, and J. Chen, "An Attempt to Self-Organized Polygon Formation Control of

Swarm Robots Under Cyclic Topologies,", *IFAC-PapersOnLine*, 53(2):11000-11005, 2020.

Rui Yang is a Ph.D. candidate in the college of IS&T at the University of Nebraska-Omaha. His research interests are in multi-agent systems with emphasis on formation control and their applications in AI and robotic related areas.



Azad Azadmanesh received the Ph.D. degree in computer science from the University of Nebraska-Lincoln, USA. He is currently a Professor in the Department of Computer Science at the University of Nebraska-Omaha. His research interests include fault-tolerance, network technologies, distributed agreement, and reliability modeling.

Hassan Farhat (photo not available) received his Ph.D. in Computer Science and Engineering from the University of Nebraska-Lincoln, USA. His research interests and publications are in the areas of computer graphics, VLSI design and testing, and recently in multi-agent systems. His current research includes comparative studies of the numerical solutions to the consensus problems in multi-agent systems. Dr. Farhat has also been involved with active learning, which has resulted in textbook publications.