# Towards Self-organized Large-Scale Shape Formation: A Cognitive Agent-Based Computing Approach

Yasir R. Darr[a], Muaz A. Niazi[*a]

[a]*Computer Science Department,*
*COMSATS Institute of Information Technology,*
*Islamabad, Pakistan*
[*]*Corresponding author*

## Abstract

Swarm robotic systems are currently being used to address many real-world problems. One interesting application of swarm robotics is the self-organized formation of structures and shapes. Some of the key challenges in the swarm robotic systems include swarm size constraint, random motion, coordination among robots, localization, and adaptability in a decentralized environment. Rubenstein et al. presented a system ("Programmable self-assembly in a thousand-robot swarm", Science, 2014) for thousand-robot swarm able to form only solid shapes with the robots in aggregated form by applying the collective behavior algorithm. Even though agent-based approaches have been presented in various studies for self-organized formation, however these studies lack agent-based modeling (ABM) approach along with the constraints in term of structure complexity and heterogeneity in large swarms with dynamic localization. The cognitive agent-based computing (CABC) approach is capable of modeling such self-organization based multi-agents systems (MAS). In this paper, we develop a simulation model using ABM under CABC approach for self-organized shape formation in swarm robots. We propose a shape formation algorithm for validating our model and perform simulation-based experiments for six different shapes including hole-based shapes. We also demonstrate the formal specification for our model. The simulation result shows the robustness of the proposed approach having the emergent behavior of robots for the self-organized shape formation. The performance of the proposed approach is evaluated by robots convergence rate.

*Keywords:* Swarm robotics, Self-organization, Agent-based modeling, Cognitive agent-based computing, Robustness, Emergent behavior

## 1. Introduction

A swarm robotics based systems have received a significant level of interest from the research community. These systems are inspired by the biological swarm, having social and collective behavior for achieving the task much bigger than their individual working capacity as discussed by Oh et al. in [1]. The same collective phenomenon is being used in the robotics field for performing everyday tasks in which group of robots interacts with each other to achieve global behavior as mentioned by Schmickl and Crailsheim in [2]. In this regard, much focus is being paid to the

formation of structures and shapes using the self-organization process in swarm robotics.

Self-organization in a swarm robotics for the formation of diverse structures and patterns in a sustainable assembly is a key challenge for the researchers. It is purely based on the coordination and cooperation between robots as presented by Haghighi and Cheah in [3], to produce and maintain the formation of a specific shape without any centralized controller. The aggregation and segregation of agents [4] along with their adaptive behavior are usually involved in achieving complex symmetries through the organization process in swarm robotics.

A number of challenges are faced in the swarm robotic system for the self-organized shape formation. These challenges include the heterogeneous behavior of robots in a swarm, localization schemes described by deŚa et al. in [5], adaptive behavior of robots in a decentralized environment, coordination among robots, and controlling the self-organized formation as presented by Cheng et al. in [6]. Numerous studies show various techniques for addressing mentioned challenges in the context of implementation and control of the self-organized formation process in swarm robotics system.

The self-assembly based system proposed for the formation of shapes by Rubenstein et al. in [7], in which experiment carried out for the formation process by applying the collective behavior algorithm. However, system is limited only to solid shapes with no holes, thus unable to make complex and diverse symmetries. The robots placed in an aggregated form having no dynamic localization and they move sequentially with no randomness. These issues are reported by Niazi in [8].

Another work regarding self-organization of agents presented by Copenhagen et al. for the group of robots in [9]. In this work, agents in a swarm made clusters through aggregation and segregation based on their alignments. While the proposed approach is dependent on the number of agents in a swarm and formed cluster contain both aligned and non-aligned agents in a two-dimensional space. In table 1, we analyze the mentioned systems regarding technique used, robustness of the system, and emergent behavior of swarm for the self-organization process.

**Table 1:** Analysis for self-organization process achieved by the swarm robotics systems presented in [7] and [9].

| Ref. | ABM | Swarm limit | Dynamic localization | Random motion | Complex shapes | Obstacle-avoidance | Heterogeneous swarm |
|------|-----|-------------|----------------------|---------------|----------------|--------------------|---------------------|
| [7] | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✓ |
| [9] | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ |

Most of the issues presented in the previous work regarding self-organized shape formation can be resolved by developing ABM for MAS in the simulation-based environment. MAS is much helpful for observing behavioral variability in swarm robotic system for the formation of structures and shapes. The complex model-based systems developed in previous study using CABC in MAS [10], can portray the self-organization process of swarm robots. These systems are capable of making robust and sustainable self-organization having the dynamic and adaptive behavior of

robots in a swarm for the complex structures.

Our main contributions are as follows:

1. We develop a simulation model for the self-organized shape formation task of a swarm robotic system using ABM under CABC approach.
2. We propose a self-organized shape formation algorithm to validate our model.
3. We also demonstrate the formal specifications of our model.

The remaining structure of the paper is organized as follows: The section 2 shows the background and basic knowledge. In section 3, we include our research methodology for this study. The section 4 presents results and discussion for simulation-based experiments along with previous studies and their comparison. A conclusion and the future work are described in the section 5.

## 2. Background

In this section, we discuss some of the concepts like swarm intelligence, swarm robotics using swarm intelligence, self-organization in swarm robots, ABM, and CABC.

### 2.1. Swarm intelligence

Swarm intelligence refers to the collective behavior of decentralized artificial system having a bio-inspired approach as mentioned in [11] by Karaboga et al. It is the soft form of a biological swarm exists in nature. Besides the structure and capabilities of the individual entity in a swarm, the global behavior is achieved through the social interactions among local individuals. It is not essential that a unique agent is useless but much simple as compared to the resultant intelligence or outcome behavior. The swarm intelligence allows the system to get the complete benefit from the swarm without having any sort of centralized control as discussed by Tan and Zheng in [12].

### 2.2. Swarm robotics

A swarm robotics field follows the swarm intelligence approach in which multiple robots interact with each other to achieve collective behavior. The robots in a swarm have autonomous behavior and they act in a cooperative manner to achieve tremendously complex tasks as compared to their physical structure and capacity as suggested by Barca and Sekercioglu in [13]. A design for coordination of the heterogeneous robots swarm is presented by Nishikawa et al. in [14]. The swarm robotics based systems are being used to perform several tasks such as partitioning of work, aggregation, pattern formation [15], object sorting, navigation or path planning [16], localization, and collective manipulation. These sort of tasks are also given in a comprehensive study by Bayindir in [17].

### 2.3. Self-organization

Self-organization process is used to attain overall assembly, which comes after local interaction among initially disordered components of the system [18]. It is a spontaneous process and is not organized by any agent externally. The resultant organization is completely decentralized and spread over all the components of the system, thus making it sustainable and robust [19]. The

strategies for the self-organization are discussed by OǴrady et al. in [20] for the group of mobile robots.

The process of organization for the swarm intelligence based swarm robotic system is being used for several purposes including the formation of structures. Such self-organized formation includes the aggregation or segregation of swarm robots by using odometry [21] by Vardy and cue-based aggregation by Arvin and Turgut in [22]. The self-organized path formation discussed in [23] by Sperati et al. and by Nouyan et al. in [24]. While producing patterns through a group of robots mentioned in the review by Bahceci et al. in [25].

### 2.4. Agent-based modeling

An ABM involves actions, characteristics of distinct agents and their interactions for describing real-world scenarios or any complex systems. An agent is a unique entity of the simulation having autonomous behavior, properties, and states. Agent-based representations using ABM can more easily be understood than mathematical models, it is because ABM created by simple rules regarding change in behavior and individual entities of the system. An ABM can also be used to represent the self-organization process by making a computational model of the MAS [26]. Modeling of the self-organization in complex systems can be possible by using flexible, generic and effective ABM tools [27].

### 2.5. Cognitive agent-based computing

The CABC framework is capable of making complex computational models based on the autonomous behavior of multi-agents. This involves the interaction and cooperation among agents based on learning techniques so to relate real-world complex systems. This was proposed and presented by Niazi in [28]. It is capable of validating large-scale MAS by developing ABM before developing the actual system. It follows the different levels of modeling approaches such as descriptive or exploratory ABM, model validation, and complex network model. Laghari and Niazi used this framework in [29] for modeling the self-organizing communication network for the Internet of Things (IoTs).

## 3. Methodology

In this section, we discuss the approach used in this study regarding development of ABM under CABC approach for self-organized shape formation. It starts from the understanding of ABM and development of the simulation model. To provide informal and formal specification of the model along with the pseudo-code specifications for the proposed shape formation algorithm. After that, applying of the algorithm on six different shapes to perform simulation-based experiments. At the end, analyze the results obtained from the simulation experiments. Figure 1 shows method block diagram of our research.
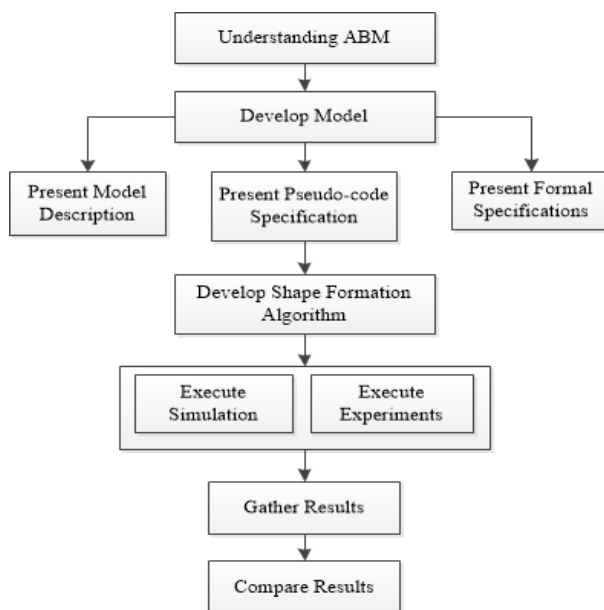
**Figure 1:** Method block diagram for our research.

### 3.1. Model description

We present a model using ABM for self-organization of swarm robots at large-scale for shape formation task. In this model, agents act as a small programmable mobile robot initially placed at random locations in NetLogo environment before starting the self-organization process. These agents have heterogeneous behavior as very few are specially programmed to act as a seed robots. All other agents have the same behavior to follow the process and achieve collective behavior. The seed robots remain stationary throughout the organization process while other robots move randomly inside and outside shape coordinates. There is a deterministic type of environment for all the agents and predefined information given to all the agents in a swarm, about the shape to be formed in the coordinate system.

Agents interact with each other as well as with the environment while moving randomly. The local interaction takes place among neighboring agents. The states of the agents are "stationary", "localize", and "un-localize", that changes during the motion. Agents while moving enter the desired shape area and approach toward the seeds, or already localized robot to become stationary and localize. The formation process continues in this way until all the moving agents become stationary and localize in the desired shape area to form the shape completely.

There are three cases of termination of the formation process. Firstly, if the shape is complete, i.e. all the agents filled the shape and became stationary and localize. While there are no moving robots left, the process stops. Secondly, the shape is incomplete if there are few robots to fill the shape and no more robots present outside the shape area, the process stops again. Lastly, if the shape is formed completely while some robots still move outside the shape wait to join the formation process. These robots stop and separate themselves from the self-organization and the process

stops. The resultant organization is completely distributed which is dependent upon all the agents as well as is sustainable and robust.

### 3.1.1. States of formation process

The proposed shape formation algorithm comprises of different states of the robots during the organization process. Before the process start, the system is in the initial state and robots are in the idle or stationary state. At the start of the process, all the robots move randomly outside or inside shape coordinates. Only four seed robots placed inside shape are stationary and localized throughout the process. A robot with the move-inside state changes its state to stop-in and joins the shape. Move-outside state changes to the move-inside state, when robot enters in the shape-area coordinates while moving randomly. The states transitions are shown in figure 2.
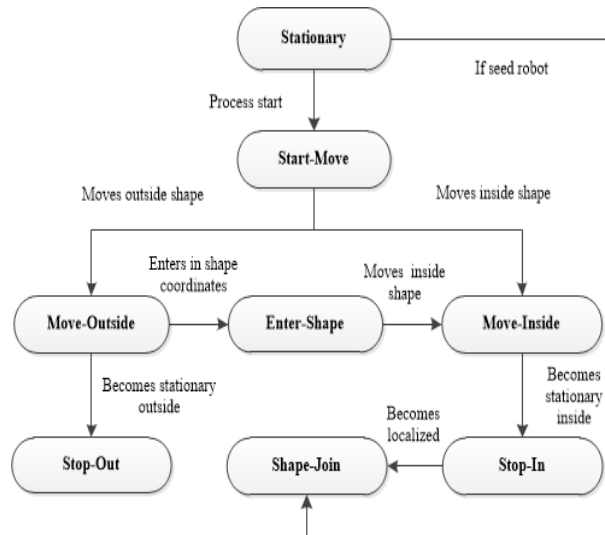


**Figure 2:** State transition diagram for shape formation algorithm.

### 3.1.2. Evaluation metric

We consider number of parameters along with their values as evaluation metric in our proposed self-organized shape formation model. The "region" parameter defines the dimension of the environment world in simulation tool. A "number of robots" parameter is the dynamic user choice for the six different shapes, the given values are associated with one of the shape respectively. The communication or sense range is given in "vision-radius" parameter having three values. A "shapes" parameter consists of all the desired shapes. The "performance evaluation" parameter has the average convergence rate of robots for the shape formation. A "number of runs" parameter is for the experiment iterations. Table 2 shows all these parameters along with there values.

**Table 2:** Evaluation metrics of proposed approach for the self-organized shape formation.

| Parameter | Value |
|---|---|
| *Region* | 32 x 32 |
| *No. of robots* | 1036, 1252, 566, 438, 1382,1040 |
| *Vision-radius* | 1, 2, 3 |
| *Shapes* | Starfish, k-letter, wrench, rectangle, tyre, spinner |
| *Performance measure* | Average convergence rate |
| *No. of runs* | 4 (1, 10, 20,30 runs) |

### 3.2. Pseudo-code specification

In this section, we present the pseudo-code for the proposed self-organized shape formation algorithm along with the detailed descriptions for each of the entity and behavior involved in the algorithm. We demonstrate the specifications for the agent involve in a process, all the global variables used, and all the procedures a proposed algorithm comprises of.

### 3.2.1. Agent design

There is only one agent as "Robot" used in our simulation model. A Robot act as an agent in a large-scale swarm having characteristics and states, performs various actions in order to model self-organized shape formation task.

---

Breed **Robot:** The robot agent used to represent robots in a swarm

*Internal Variables: $< seed?, Stationary?, localize?, position\text{-}inside? >$*

   **seed?:** Specialize robots as a reference for all other robot
   **stationary?:** All robots which are stationary
   **localize?:** All robots which are localize
   **position-inside?:** All robots which are inside desired shape area

---

We first stated the agent used in the model environment to represent the robots in a swarm. Then we listed four internal variables for robots as "seed?", "stationary?", "localize?", and "position-inside?". Seeds are specialized robots which remain stationary and localize throughout the self-organization process. Robots which are static and not moving have stationary state, while localize state for robots placed at the correct position and position-inside state means robots which reside inside desired shape area whether moving or static.

### 3.2.2. Global variables

In this part, we describe global variable for our simulation. Input global variables are used during the initialization of the simulation environment through setup procedure.

| Global input:*< num-robots, shape-type, vision-radius >* |
|---|
| *Slider:* |
| **num-robots:** Select number of robots in swarm |
| **vision-radius:** Communication range for robots interaction |
| *Chooser:* |
| **Shape-type:** Select desire shape for the formation |

Three global input variables used in our simulation model. Two variables "num-robots" and "vision-radius" are used through the slider, which is GUI element of NetLogo. Number of robots available at the start of the self-organization process are selected by "num-robots". The "vision-radius" used for choosing a radius for communication among neighboring robots. The "shape-type" input variable provided by the chooser, which is also a GUI element for a simulation model of NetLogo. The "shape-type" has a list of six desired shapes to be formed by robots. The list includes star, wrench, k-letter, rectangle, tyre, and spinner shape choices.

### 3.2.3. Procedures

We present all used procedures and their specifications in this section.

1. *Setup*

| Procedure **setup:** Creating simulation environment |
|---|
| *Input:* Global variables |
| *Output:* Initialize the whole environment |
| **begin** |
| 1: Clear all |
| 2: Set-patches |
| 3: Set-shape |
| 4: Set-robots |
| **End** |

The specification model for setup procedure is presented, which is used to create and initialize the main environment of the simulation model by considering all global variables. It clears all the previous states and stuffs present in the environment. After this, "set-patches", "set-shape", and "set-robots" procedures are called which we discuss next.

2. *set-patches*

Procedure **set-patches:** Creates number of patches in an environment world of NetLogo

*Input:* World size for x and y coordinate

*Output:* Generates patches with boundary wall

**begin**

  1: **for all** patches **do**

  2:      set color = white

  3:      **if** neighboring patches < 8 **then**

  4:         set color = black

**End**

In the set-patches procedure, patches are created in the world as specified in the environment setting, which is a basic element in NetLogo. The x,y coordinate values is set along with the patch size for the number of patches. The color of all patches set to white, while black color for the corner patches to consider them as a closed boundary wall for robots.

3. *set-shape*

Procedure **set-shape:** Import selected shape on the patches of the world

*Input:* List of shapes in the chooser menu "shape-type"

*Output:* Generates selected shape information to the model

**begin**

  1: **if** shape-type = "star" **then**

  2:      import-star-shape

  3: **if** shape-type = "wrench" **then**

  4:      import-wrench-shape

  5: **if** shape-type = "k-letter" **then**

  6:      import-k-letter-shape

  7: **if** shape-type = "rectangle" **then**

  8:      import-rectangle-shape

  9: **if** shape-type = "tyre" **then**

10:      import-tyre-shape

11: **if** shape-type = "spinner" **then**

12:      import-spinner-shape

**End**

In this procedure, the shape is selected by the global variable 'shape-type". The selected shape is imported on the patches and its information is given to all robots.

4. *set-robot*

---

Procedure **set-robot:** To create and set robots

*Input:* Number of robots from "num-robots" slider

*Output:* Generates number of robots at random locations

**begin**

  1:  **if** count robots = 0  **then**

  2:      create num-robots

  3:  **if** robot-id >= 0 and <= 3  **then**

  4:      set shape = "circle"

  5:      set color = "red"

  6:      set seed? = true

  7:      set xy-coordinates near origin

  8:      set stationary? = true

  9:      set localize? = true

10:  **else**

11:      set shape = "directional-circle"

12:      set color = "magenta"

13:      set xy-coordinates random

14:      set seed? false

15:      set stationary? false

16:      set localize? false

17:  **for all** robots **do**

18:      **if** xy coordinate inside shape **then**

19:         set position-inside = true

20:      **else**

21:         set position-inside = false

**End**

---

In the set-robots procedure, existing number of robots are checked first, then number of robots selected by the "num-robots" slider are created. First four robots dedicated as seed robots. The shape for seeds set as "circle", color as "red" and location near the origin, while values for internal state variables "seed?", "stationary?", "localize?", and "position-inside?" as true. For all other robots, the shape is chosen as "directional-circle", color as "magenta" and random coordinates as location, while values for internal state variables are set false. The locations of robots in the environment world are checked to set "position-inside?" value as true or false.

5. *Procedure go*

Procedure **go:** Main shape-formation algorithm

*Input:* Number of robots from num-robots slider
*Output:* Generates shapes by convergence of robots
*Execution:* Called repeatedly during simulation
**begin**

```
 1:  while stoping condition = true do
 2:      for all non-stationary robots do
 3:          if coordinates = outside-shape then
 4:              set position-inside? = False
 5:              Move Random
 6:              if patch at vision-radius = black then
 7:                  Rotate Random 360
 8:              else
 9:                  Move Random
10:              if any moving robot at vision-radius then
11:                  Rotate Random 360
12:              else
13:                  Move Random
14:          else (inside shape coordinates)
15:              set position-inside? = True
16:              Move Random
17:              if any moving robot at vision-radius then
18:                  Rotate Random 360
19:              else
20:                  Move Random
21:              if any empty-patches at neighbors of localize robots then
22:                  face empty-patches
23:                  move-towards-location
24:                  set stationary? = true
25:                  set localize? = true
```

**End**

In this procedure, the main working for the formation process is shown. A number of robots randomly moving outside or inside shape coordinates in the cartesian coordinate system of the environment world. The "position-inside" variable is set false outside shape and true for inside shape coordinates. Non-stationary robots avoid collision with each other in "move-until-empty-ahead" procedure. While inside shape coordinates, non-stationary robots check and proceed towards empty neighbor patches of seeds or any stationary and localized non-seed robot in "move-towards-location" procedure. After reaching to the desired location, robots set their values for "stationary?" and "localize?" as true. Figure 3 shows the flow of the shape formation algorithm.
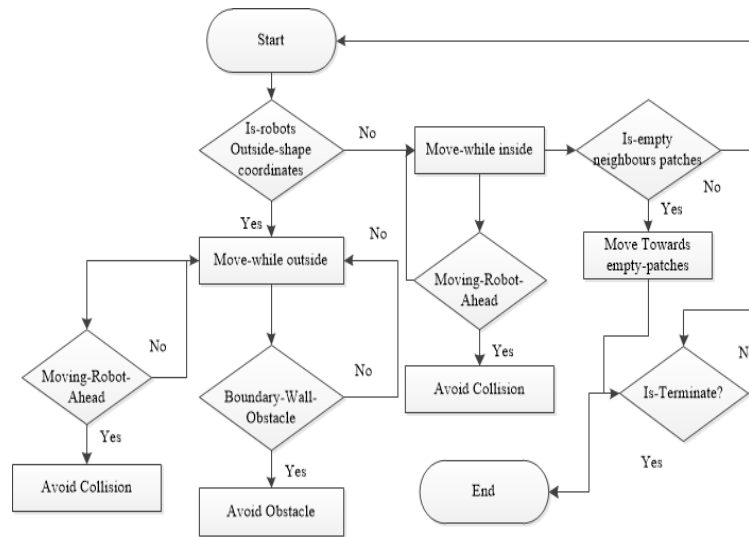
**Figure 3:** Flowchart for the proposed shape formation algorithm.

6. *boundary-wall*

---

Procedure **boundary-wall:** Avoiding boundary wall obstacle

---

*Input:* Moving robots and their locations
*Output:* Directions for moving robots
*Execution:* Repeatedly called by "go" procedure
**begin**

  1:  **for all** non-stationary robots **do**
  2:      **if** outside-shape coordinates **then**
  3:         **if** next patch = black **then**
  4:            Rotate Random 360
  5:         **else**
  6:            Move Random

**End**

---

In this procedure, all the moving robots outside shape coordinates check next location in their direction for the boundary wall obstacle. When reach to the boundary, robots return and divert their direction without hitting the wall obstacle.

7. *move-until-empty-ahead*

Procedure **move-until-empty-ahead:** Avoiding collision between moving robots

*Input:* Moving robots and their neighbors
*Output:* Movement and directions of moving robots
*Execution:* Repeatedly called by "go" procedure
**begin**

1: **for all** non-stationary robots **do**
2:    **if** any moving robot ahead **then**
3:        Rotate Random 360
4:    **else**
5:        Move Random

**End**

All moving robots avoid the collision with other robots by checking any robot ahead in their direction of motion. If so, robots change their motion direction.

8. *move-towards-location*

Procedure **move-towards-location:** Robots choose and move towards candidate location to converge

*Input:* Moving robots and their locations
*Output:* Robots stop adjacent to localized robot
*Execution:* Repeatedly called by "go" procedure
**begin**

1: **for all** non-stationary robots **do**
2:    **if** inside-shape coordinates **then**
3:        **if** any empty-patches at neighbors of localized **then**
4:            face empty-patches
5:            move-towards empty-patches
6:            set stationary? = true
7:            set localize? = true

**End**

Inside the desired shape coordinates, moving robots find an empty patch among neighbor patches of the seeds or other localized robots. They proceed toward that location and set values for "stationary?" and "localize?" as true.

*3.2.4. Behavior experiment*

The behavior of the moving robots along with their states is observed in the plots and experiment specification model.

1. *Produce plot*

| Procedure **produce-plot:** To plot current values of the simulation model |
|---|
| *Input:* NA |
| *Output:* Generates plot during simulation run |
| *Execution:* Generated by "go" procedure |
| **begin** |
|     Number of stationary robots |
|     Number of localize robots |
|     Robots with position-inside = true |
|     Number of empty-patches inside shape |
| **End** |

The "produce-plot" procedure generates plot during the execution of the simulation to monitor the convergence of robots for shapes formation. It monitors a number of robots along with the state variable values for "stationary", "localize", and "position-inside" as true.

2. *Experiment*

| **Experiment:** Exhibits the convergence behavior of robots regarding six shapes |
|---|
| *Input: < num-robots, shape-type >* |
| *Setup procedure: < setup >* |
| *Go procedure: < shape-formation >* |
| *Repetition:* 10 |
| |
| *Input:* |
|     **num-robots:** 1036, 566, 1352,438,1282, 1040 |
|     **shape-type:** star, wrench, k-letter, rectangle, tyre, spinner |
| *Report:* |
|     **stationary:** Number of stationary robots |
|     **localize:** Number of localized robots |
|     **empty-patches:** Number of empty-patches |
| *Stop condition:* Self-organization process stops, when all robots become stationary and localize to form the desired shape completely. |

A simulation experiment used in our model for the six shapes convergence analysis by varying number of robots. The average convergence rate calculated for the ten repetitions along with the shape completeness in term of localize, unlocalize robots, and empty-patches in the formed shape.

### 3.3. *Formal specification*

The formal specification is a technique used for modeling different types of systems mathematically. A system developer can comment about the completeness of the model for all aspects needed in specific abstraction level. The one significant feature of formal specification is that

through modeling, it can effectively represent the real-world scenarios at various abstract levels. A formal specification language is known as "Z" developed by Oxford University in 1970. It can easily be converted into executable program code [30]. A specification using "Z" comprises of sets and predicates from the mathematical notation [31]. This specification language is being used for modeling large-scale real-world complex systems. It is also used to relate the agent logic in MAS [32].

### 3.3.1. A Formal specification for self-organized shape formation

We present the formal descriptions of our self-organized shape formation model using z-language for formal specification. The purpose of this is to validate all the informal requirements of the system by using formal specification. In this study, robot and shape are two entities involved in the formation process. We use "sets" along with operation or state "schemas", for declaring user-defined entities such as robot and shape regarding shape formation task. The states of the robot and shape are shown in figure 4.
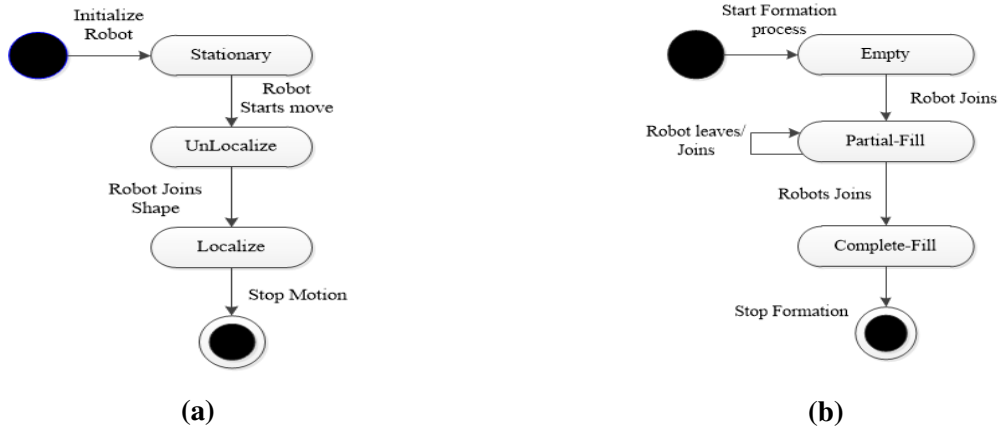


**Figure 4:** The States for the robot and shapes **(a)** Robot states comprises of "Stationary", "UnLocalize", "Localize", **(b)** Shape states comprises of "Empty", "Partial-Fill", "Complete-Fill" .

1. *Sets and schemas*

   A "ROBOT" is a set of all the robots in a swarm for the shape formation.

   [*ROBOT*]

   We defined the "max-xcor" and "min-xcor" coordinate values for x coordinate as 35 and -35 respectively and similarly for the other two coordinates y and z. While "maxRobot" is user defined global input variable for defining maximum number of robots in a swarm.

$$\begin{array}{|l}
maxRobot : \mathbb{N} \\
xcor, ycor, zcor : \mathbb{Z} \\
\hline
maxRobot \geq 0 \\
min\text{-}xcor = \text{-}32 \\
max\text{-}xcor = \text{-}32 \\
min\text{-}ycor = \text{-}32 \\
max\text{-}ycor = \text{-}32 \\
min\text{-}zcor = \text{-}32 \\
max\text{-}zcor = \text{-}32
\end{array}$$

We use a free type definition for the two types of robots in a swarm i.e. seed and non-Seed. A formal definition for "RobotType" is:

$RobotType ::= seed \mid nonSeed$

Robot schema defines a robot placed randomly having x,y,z coordinates in the cartesian coordinate system bounded between "min" and "max" values for coordinate parameters. A variable of the type "RobotType" introduced in this schema to define a robot as seed or non-seed.

$$\begin{array}{|l}
\underline{Robot} \\
type : RobotType \\
x, y, z : \mathbb{Z} \\
\hline
x \geq min\text{-}xcor \\
x \leq max\text{-}xcor \\
y \geq min\text{-}ycor \\
y \leq max\text{-}ycor \\
z \geq min\text{-}zcor \\
z \leq max\text{-}zcor
\end{array}$$

We use free type definition for states of the desired shape to be formed. Elements for this free type are "empty", "partial", and "complete" for the empty shape, partially formed shape, and completely formed shape respectively.

$ShapeState ::= empty \mid partial \mid complete$

The schema describes the shape to be formed in coordinate system and shape dimension variable value obtained from the product of coordinates value of the shape. We define a variable "sState" of the type "ShapeState" who's value can be "empty", "partial", and "complete".

All robots in a swarm carry characteristics like shape, color, state and position. The values for these features vary among robot, so we define the finite sets of robots as "stationary", "unLocalize" and "localize" for the states of robot.

```
┌─ Shape ────────────────────────────────────────────────────
│ p, q : ℤ
│ shapeDimention : ℕ
│ sState : ShapeState
│ stationary : 𝔽 ROBOT
│ unLocalize : 𝔽 ROBOT
│ localize : 𝔽 ROBOT
├────────────────────────────────────────────────────────────
│ p ≥ min-xcor
│ p ≤ max-xcor
│ q ≥ min-ycor
│ q ≤ max-ycor
│ shapeDimention = p * q
│ localize ≤ maxRobot
│ localize ≤ shapeDimension
└────────────────────────────────────────────────────────────
```

At the start of the formation process, shape is in empty state. That is, still no robot has converged and joined the shape while "localize" set is still a null set.

```
┌─ InitShape ────────────────────────────────────────────────
│ Shape
├────────────────────────────────────────────────────────────
│ localize = ∅
│ sState = empty
└────────────────────────────────────────────────────────────
```

When process starts, all the non-seed robots in a swarm start moving randomly and change their state from "stationary" to "unLocalize". Robots are added into the finite set of "un-Localize" and eliminated from "stationary" set as shown in the schema "StartMove" given below.

```
┌─ StartMove ────────────────────────────────────────────────
│ ΔShape
│ Robot
│ robot? : ROBOT
├────────────────────────────────────────────────────────────
│ robot? ∈ stationary
│ type = nonSeed
│ robot? ∉ unLocalize
│ unLocalize′ = unLocalize ∪ {robot?}
│ stationary′ = stationary \ {robot?}
└────────────────────────────────────────────────────────────
```

Robots moving randomly enter and joins the shape upon reaching next to one of the localized seed robot near origin or other localized non-seed robot inside the shape. A robot belongs

to "unLocalize" set, becomes localize and added to "localize" set as well as removed from "unLocalize" robot set. The state of the shape changes from "empty" to "partial".

---
*JoinShape*
$\Delta Shape$
$robot? : ROBOT$

---
$\#localize < shapeDimension$
$robot? \in unLocalize$
$robot? \notin localize$
$localize' = localize \cup \{robot?\}$
$unLocalize' = unLocalize \setminus \{robot?\}$
$sState' = partial$

---

While formation process continues for the shape, procedure given in "JoinShape" schema is repeated. When shape is completely formed by a number of localized robots, the "sState" variable changes its value from "partial" to "complete".

---
*ShapeFormation*
$\Delta Shape$
$robot? : ROBOT$

---
$\#localize < shapeDimension$
$sState = partial$
$robot? \notin localize$
$localize' : localize \cup \{robot?\}$
$unLocalize' : unLocalize \setminus \{robot?\}$
$\#localize' = shapeDimension$
$sState' = completed$

---

We consider both perspectives for working of the "JoinShape", as successful and error outcome of the schema. The success outcome involves robots entering and joining the shape formation while for the error outcome, there will be two reasons for that. The shape is already formed so that no further robot can join the shape for formation. Secondly, if a total number of the robot is less than shape dimension value i.e. all robots belong to "localize" set and shape is not completely formed. This scenario is presented in the table 3

**Table 3:** Free type definition table for "JoinShape" schema

| Schema | Pre-condition for success | Condition for error |
|---|---|---|
| JoinShape | # localize < shapeDimension robot? ∉ localize | Shape is alreadyfilled: # localize = ShapeDimension robot? ∉ localize<br><br>Shape is incomplete: maxRobot < ShapeDimension robot? ∈ localize |

We define a free type definition "REPORT" for success and error outcome for the shape formation process. The elements of this free type are "success", "alreadyComplete", and "incomplete".

$$REPORT ::= success \mid alreadyComplete \mid incomplete$$

2. *Schemas for Success and Error*

A robot joins the shape for the formation successfully in the "JoinShape" schema. The report variable gives "success" as output value.

```
┌─ Success ──────────────────────────────
│ report! : REPORT
├────────────
│ report! = success
└─────────────────────────────────────────
```

The shape is completed if the number of localized robots inside the desired shape is equal to the shape's dimension value. The report variable gives "alreadyComplete" as outpu

```
┌─ AlreadyComplete ──────────────────────
│ ΞShape
│ report! : REPORT
├────────────
│ sState = complete
│ #localize = shapeDimension
│ report! = alreadyComplete
└─────────────────────────────────────────
```

The shape is partially formed, if a number of localized robots is less than the shape's dimension value. The report output variable gives "incomplete" as value.

```
┌─ Incomplete ───────────────────────────
│ ΞShape
│ report! : REPORT
├────────────
│ sState = partial
│ #localize < shapeDimension
│ report! = incomplete
└─────────────────────────────────────────
```

3. *Schema calculus*

We combine success outcome of "JoinShape"schema for the shape formation process by using conjunction along with error outcome by using disjunction.

$$(\textit{JoinShape} \land \textit{success}) \lor \textit{alreadyComplete} \lor \textit{incomplete}$$

## 4. Results and Discussion

This study focused on self-organization of robots in a swarm at a large-scale by developing a simulation model using ABM under CABC for six different shapes. We performed a number of simulation experiments to analyze robots convergence. The proposed algorithm is capable to form complex symmetries and diverse shapes including hole-based shapes. The self-organization process is robust without depending on the size of swarm i.e. group of robots, as remaining extra moving robots separate themselves from the process. While the selection and localization of seeds is based on the robot id among a variable number of robots. A randomness in motion and initial locations is used in our approach. The obstacle-avoidance mechanism used in term of closed boundary walls as well as collision-avoidance among randomly moving robots also addressed. We analyzed average convergence time for the completely formed shapes by the fixed number of robots by performing (n=10) simulation experiments along with 95% confidence interval. The behavior analysis of our proposed approach shows that the average time convergence by the large group size robots is less than that for small group size robots.

### 4.1. Environment overview

We used NetLogo as the simulation environment for our self-organized shape formation model. NetLogo is a tool used for the ABM at an extensive level by the number of researchers across the multiple domains [33]. It is based on the programming language "Logo" used for the environment refers to "World" in the graphical interface of the tool. Agents exist in the world named as a "Turtles", which carry various characteristics and capabilities for performing a large number of actions. The state and behavior of agent changes against these actions performed in the world environment. Figure 6 shows the interface design of the NetLogo for our ABM based model.

### 4.2. Self-organized shape formation

We developed a simulation model for the formation of six shapes. These shapes include "star", "wrench", "k-letter", "rectangle", "tyre", and "spinner-shape". All final shapes formed by the self-organization of swarm robots in our simulation model by applying proposed shape formation algorithm developed in Netlogo are shown in figure 7.

### 4.3. Experiment results

In this section, we discuss the simulation-based experiment's results obtained by applying proposed self-organized shape formation algorithm. The results include number of robots a final desired shape comprises, the average time taken to complete the formation of desired shape after running n=10 experiment for the number of robots, and the dimensions of the desired shape formed as shown in the table 5. The convergence rate i.e. the total number of robots involved in

the formation process for the desired shape in a unit time along with the 95% confidence interval for convergence time are shown in the figure 8.

Next we describe the simulation results for all six shapes individually. We analyzed the behavior during simulation experiments by varying the number of robots and averaging their convergence time for the shape formation along with 95% confidence interval value. A completeness of the shapes in term of a total number of localized, unlocalized robots, and empty spaces or patches within the formed shape is also being evaluated. In addition, we observed convergence variations among five different robots group size values for the specific shape at the completion of the simulation. While at the end of this section, we also discuss the previously achieved results present in the study [7].

### 4.3.1. Star shape

By applying the proposed algorithm for a number of robots in our simulation model, robots converge to form a predefined "star" shape as imported into the model and shape information given to all agents at the start of the simulation process. We considered number of robots in five groups as (1025, 1030, 1035, 1040, and 1045) for the formation algorithm and run a simulation experiment n=10 times for each group. Average convergence times measured for these groups of robots along with 95% confidence interval. We analyzed that group of 1036 robots able to form a complete "star" shape by running n=10 simulation experiments. Robots showed average convergence time of 861.8 iterations and 95% confidence interval ranges from 599.1 to 1124.49 iterations as shown in figure 9.

### 4.3.2. Wrench shape

The shape information given to the algorithm after the imported image of the "wrench" to the world environment of simulation. Algorithm applied to five groups of robots (500, 525, 550, 575, and 600) by running simulation experiments for n=10 times for every group. We calculated average convergence time for these groups of robots along with 95% confidence interval as well as completeness of "wrench" by these groups of robots. We observed that 566 robots made complete "wrench" shape with average convergence time of 984.4 iterations and 95% confidence interval ranges from 804.48 to 1164.31 iterations as shown in figure 10.

### 4.3.3. K-letter shape

The proposed shape formation algorithm applied to the five groups (1300, 1325, 1350, 1375, and 1400) of robots for the formation of a "k-letter" shape. The simulation experiments executed n=10 times for these groups to calculate average convergence time with 95% confidence interval and shape completeness status. Figure 11 shows these results, where the complete shape achieved by 1352 robots group at average convergence time of 628.2 and 481.11 to 775.28 iterations for 95% confidence interval.

### 4.3.4. Rectangle shape

We performed n=10 simulation experiments upon five groups (400, 425, 450, 475, and 500) of robots for the formation of "rectangle" shape by using proposed shape formation algorithm.

We measured average convergence time with a confidence interval of 95% and completeness of the shape for all the groups. It can be observed from the figure 12, that the "rectangle" shape completely formed by the 438 robots with average convergence time of 1479.8 iterations along with the 95% confidence interval of 1253.58 to 1706.01 iterations.

### 4.3.5. Tyre shape

The five groups (1200, 1225, 1250, 1275, and 1300) of robots involved in n=10 simulation-based experiments for the "tyre" shape formation using proposed algorithm. At the completion of experiments, we observed the average convergence time with 95% confidence interval and completeness of the formed shape. The robot swarm of size 1282 formed the desired shape at average convergence time of 586.7 iterations with 95% confidence interval value as 431.93 to 741.46 as shown in figure 13.

### 4.3.6. Spinner shape

We applied our proposed shape formation algorithm upon desired "spinner" shape selected by the shape-type parameter for the five groups (1000, 1025, 1050, 1075, and 1100) of robots. The results achieved through n=10 number of simulation-based experiments demonstrated the average convergence times with the 95% confidence interval and completeness of the desired "spinner" shape. From figure 14, it is observed that the group of 1040 robots formed a complete shape with average convergence time 802.5 iterations with 457.41 to 1147.58 confidence interval of 95%.

Now we discuss the results previously achieved by the Rubenstein et al. in [7] by performing thirteen-experiments using thousand programmable robots swarm in order to make four predefined shapes by the self-assembly of robots. First three experiments were for "k-letter", "star", and "wrench" shapes while remaining ten experiments were for a "rectangle" shape. The results described in term of a number of robots involved in self-assembly process for the specific shape, time for shapes completion, and dimensions of the final shapes as shown in the table 4. The shape completion time measured for the "star" shape is 11.66 hours by 946 robots, the "k-letter" consumes 11.71 hours for 1018 robots, "wrench" formed in 5.95 hours by 533 robots, and the average time for "rectangle" shape formation is 1.7 hours by using average 103.8 robots. We illustrate these results explicitly in the figure 5.

### 4.4. Previous studies

Chen and Crilly suggested a cross-domain framework to relate applications of complex designs [34]. The author presented a comparative analysis based on the derived parameters from the three aspects which are characterization, purpose, and realization of complex systems design. The analysis depicts that the framework is capable of studying and designing complex systems across different domains. However, framework did not cover all the aspects of designing complex systems but can be used as a flexible and extendable framework by further partitioning the major aspects of complex system design.

An algorithm called "potential field control" presented by Qin et al. in [35], based on an exponential functions in the form of bounded artificial forces for formation control in swarm robotic

systems. The simulation results showed the algorithm robust in term of swarm intelligence behavior in the real-world scenarios like local minima, avoiding obstacles and deformation of design. The proposed system more likely to generate local minima and less adaptive to the environment for the formation, based on the multi-robots.

Meng et al. in [36], used gene regulatory network (GRN) in morphogenetic approach for addressing complex shapes formation. The results demonstrated the robustness and effectiveness of the proposed approach in a decentralized environment for swarm robots in order to make the complex shapes. The proposed system is dependent on the size of the swarm and localization scheme is not much efficient for the global coordinate system.

The shape formation system using potential function based scheme, which controlled the formation of multiple shapes by addressing the issues of the collision, distance sensing and local minima presented by Jung and Kim in [37]. The proposed system is scalable for the size of the swarm and showed the effectiveness of the scheme for the formation of shapes like diamond, elliptical and heart by using agents in a swarm. The system showed stability, which analyzed by using "lyapunov" approach. While system lacks the dynamic behavior for the stochastic or obstacle-based environment, so system is unable to relate the real world scenario.

An agent-based self-organized algorithm proposed by Niazi in [38], for the modeling of emergent and dynamic snake-like structure. The achieved results showed the resilience behavior of the proposed approach, using agent-based modeling for the self-organization. The proposed algorithm lacks the decentralized environment where multiple factors may affect the agent.

MAS model based on the topology of mixed coupling proposed by Chen and Chu in [39], for the self-organized specific pattern or geometric shapes formation. The result showed the formation convergence for symmetric coupling as uniform behavior and complex for other. While using this, the control for coordination design can be achieved. However, the system is scalable only for a limited number of agents for the proposed approach to be effective in a multi-agent system model, which made it less adaptive.

In [40], Cheah et al. proposed control method by using "lyapunov" like function to analyze the convergence of robots with dynamic group and positions of robots in a region. The simulation experiment showed the analysis for the convergence of robots in the multi-robot system for shape formation controller. The proposed method lacks environment adaptability and considered only the obstacle-free environment.

An artificial GRN based framework presented by Guo et al. in [41] for the self-organized pattern generation and covering boundaries in the multi-robot system. For analyzing the effectiveness of the proposed framework, various case studies were taken and results showed the adaptive behavior in term of group size and decentralized nature of the self-organized algorithm. However, the current scheme is based on an assumption about the robots accurate localization in the coordinate system, which needs to be improve in term of efficient localization scheme.

In [42], Arvin et al. in their work implemented aggregation of honeybee based on the various parameter values as well as proposed two variations of honeybee aggregation called "waiting time" and "dynamic velocity". The experiments observed variation in collective behavior of swarm robotic systems against other algorithms. The results also showed the efficiency of the proposed variation of honey bee aggregation in term of aggregation time. The proposed system ignored local communication among homogeneous robots.

In [43], Ekanayake and Pathirana presented a scalable control algorithm for robots in a swarm by using artificial forces in order to form predefined 2D shapes. The simulation results illustrated the robustness of an algorithm. The performance analyzed by considering the issues in real applications, like communication distance and localization. However, the proposed architecture lacked the heterogeneity among swarm robots, ignored the obstacle-based environment and not scalable for making contours that are more complex.

Zhang and Pathirana in [44], proposed a solution to the formation control issue regarding a group of mobile robots for making random geometric patterns. A presented algorithm based on the optimization technique is distributed and focused on the desired geometric pattern for robot current arrangement. The strategy for movement assured continuous motion until the formation of desired geometric pattern. The system achieved the process of optimization which depends upon the perfect positioning for the desired pattern with respect to the current configuration of robots and communication that best suited for robots in the group. While system lacks the local interaction among robots and for a small set of robots, it optimizes independently.

The positioning algorithms are inspired by the systems having the model matching capability while traditional model matching systems are not much distributed. An algorithm presented by Seckin et al. in [45], designed to work in a distributed manner comprises of two stages, offline and online. Data collection and computation of the positions is completed in the offline stage, while transfer and communication of data in another stage. The results illustrated that the positions of the robots are based on their features given in the input image. The comparison made in the study is based only on the optimality for the various feature-matching algorithms.

The self-assembly based shape formation for swarm robotic system presented by Rubenstein et al. in [7]. An experiment carried out for the formation of shapes like "starfish", "wrench", "K-letter", and "rectangle" by applying collective behavior algorithm. However, at the start of the self-assembly process, robots are placed in aggregated form with fix locations and no random localization. Movement for robots is in sequential edge following manner without random motion. Self-assembly process depends on swarm size for formation, therefore no scalability for swarm size. The system limited only to solid shapes with no holes, this unable to form a complex symmetries. Furthermore, it also ignored the decentralized environment.

The self-organization in the swarm robotic system presented by Copenhagen et al. in [9] for sorting limits of the heterogeneous robots using multi-agents, made clusters based on the align-

ments of agents. There is an aggregation and segregation of agents in a swarm to make clusters. The proposed system has swarm size limitation, as well as clusters, comprise of both aligned and non-aligned agents instead of having separate clusters of each agent group in a two-dimensional space. The system also lacks the ABM approach for self-organized sorting of swarm robots.

We performed comparative analysis by reviewing literature discussed in this section, on the bases of the techniques used along with with the challenges faced or constraints in the swarm robotic systems for the self-organized formation task. The findings and comparisons of the previous work expressed in tabular form given in the table 6.

## 5. Conclusion

In this paper, we presented simulation model using ABM for the self-organized shape formation task in swarm robotic system. We also illustrated the formal descriptions for our model to validate all the informal requirements of the system. In this context, we proposed a self-organized shape formation algorithm, having no centralized controller instead rely on the individual actions and behaviors of agents for making flexible shapes. We performed number of simulation-based experiments for six different shapes and observed the convergence of agents for making these shapes. Simulation results showed the effectiveness of the proposed approach in term of agents adaptive behavior in the large-scale swarm for a decentralized environment, by successfully achieving the formation of six diverse shapes including hole-based shape. The behavior analysis for our proposed approach demonstrated that for a larger number of robots, the average time convergence is less than that for smaller swarm size in self-organization process of shape formation. In Addition to the future work, we intend to build the same scenario of self-organized flexible shapes formation in three-dimensional space with more adaptability feature in swarm robots.

**Table 4:** Results achieved by Rubenstein et al. in [7].

| Desire shape | Robots at start | Robots in final shape | Completion time (hrs) | Shape dimension (m) |
|---|---|---|---|---|
| K(E1) | 1024 | 1018 | 11.71 | 1.39 x 1.34 |
| Starfish(E2) | 1024 | 946 | 11.66 | 1.47 x 1.47 |
| Wrench(E3) | 543 | 533 | 5.95 | 1.60 x 0.63 |
| Rectangle(E4) | 116 | 104 | 1.18 | 0.35 x 0.61 |
| Rectangle(E5) | 116 | 104 | 1.10 | 0.35 x 0.61 |
| Rectangle(E6) | 116 | 103 | 1.11 | 0.35 x 0.61 |
| Rectangle(E7) | 116 | 104 | 1.15 | 0.35 x 0.61 |
| Rectangle(E8) | 116 | 105 | 1.25 | 0.35 x 0.61 |
| Rectangle(E9) | 116 | 104 | 1.15 | 0.35 x 0.61 |
| Rectangle(E10) | 116 | 100 | 1.06 | 0.35 x 0.61 |
| Rectangle(E11) | 116 | 106 | 1.11 | 0.35 x 0.61 |
| Rectangle(E12) | 116 | 104 | 1.10 | 0.35 x 0.61 |
| Rectangle(E13) | 116 | 104 | 1.16 | 0.35 x 0.61 |

**Table 5:** Experiments summary for proposed shape formation algorithm for all desired shapes.

| Shape | Dimensions | No. of robots | Avg. time (sec) | Confidence interval (95%) |
|---|---|---|---|---|
| Star | 464 x 438 | 1036 | 861.8 | 599.1 — 1124.49 |
| K-letter | 446 x 462 | 1352 | 628.2 | 481.11 — 775.28 |
| Wrench | 466 x 430 | 566 | 984.4 | 804.48 — 1164.31 |
| Rectangle | 409 x 352 | 438 | 1479.8 | 1253.58 — 1706.01 |
| Tyre | 472 x 413 | 1282 | 586.7 | 431.93 — 741.46 |
| Spinner | 458 x 427 | 1040 | 802.5 | 457.41 — 1147.58 |

**Table 6:** Comparative analysis of previous studies.

| Ref. | Technique | Real exp. | ABM | Swarm limit | Dynamic localization | Random motion | Complex shapes | Obstacle-avoidance | Heterogeneous swarm |
|---|---|---|---|---|---|---|---|---|---|
| [35] | Potential-field control algorithm | ✗ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ |
| [36] | GRN based morphogenetic approach | ✓ | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ |
| [37] | Potential function with lyapunov approach | ✗ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| [38] | Self-organization algorithm | ✗ | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ |
| [39] | Mix coupling topology | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| [40] | Lyapunov function | ✗ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ |
| [41] | GRN based morphogenetic approach | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ |
| [42] | Honey-bee aggregation | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ | ✗ |
| [43] | Control algorithm | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ |
| [44] | Optimization based formation algorithm | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ |
| [45] | Positioning algorithm | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ |
| [7] | Self-assembly algorithm | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✓ |
| [9] | Self-organized sorting | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ |

**(a)**



**(b)**
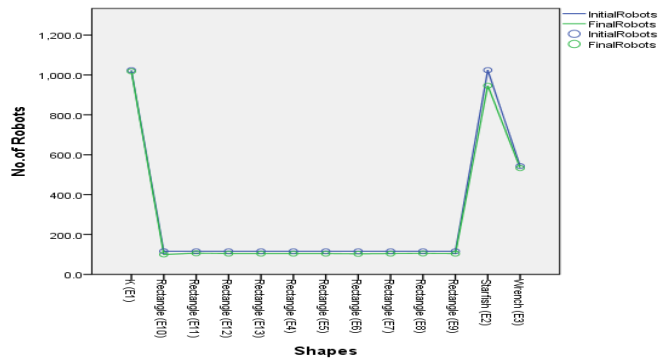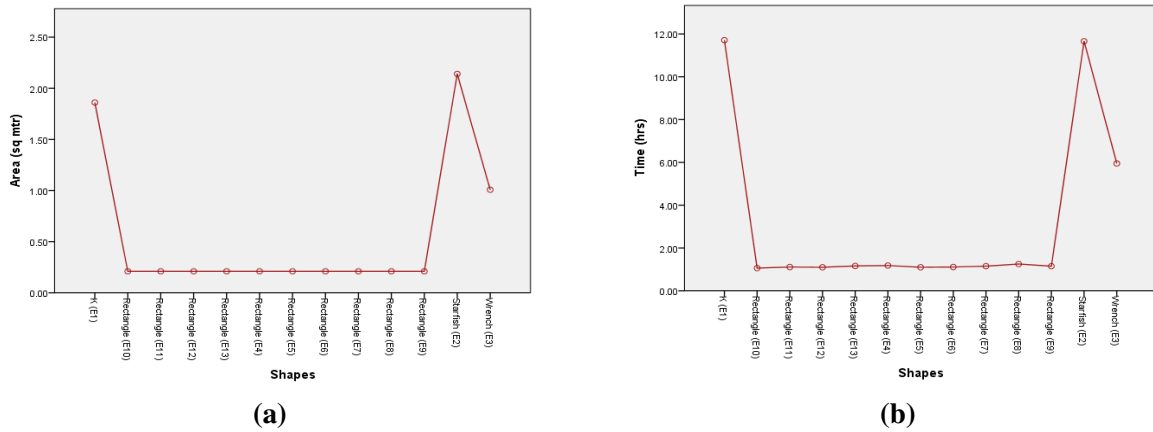


**Figure 5:** Replicated results for the data given in table 4 obtained from study by Rubenstein et al.
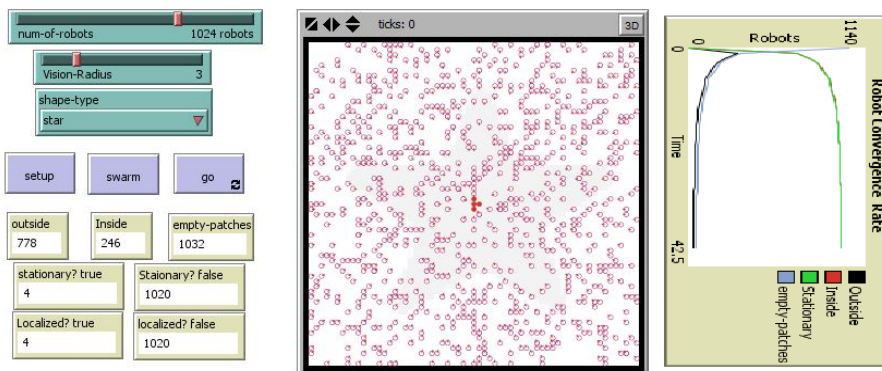


**Figure 6:** A screen shot of the interface design for self-organized shape formation model in Netlogo.

**(a)**

**(b)**

**(c)**

**(d)**

**(e)**

**(f)**
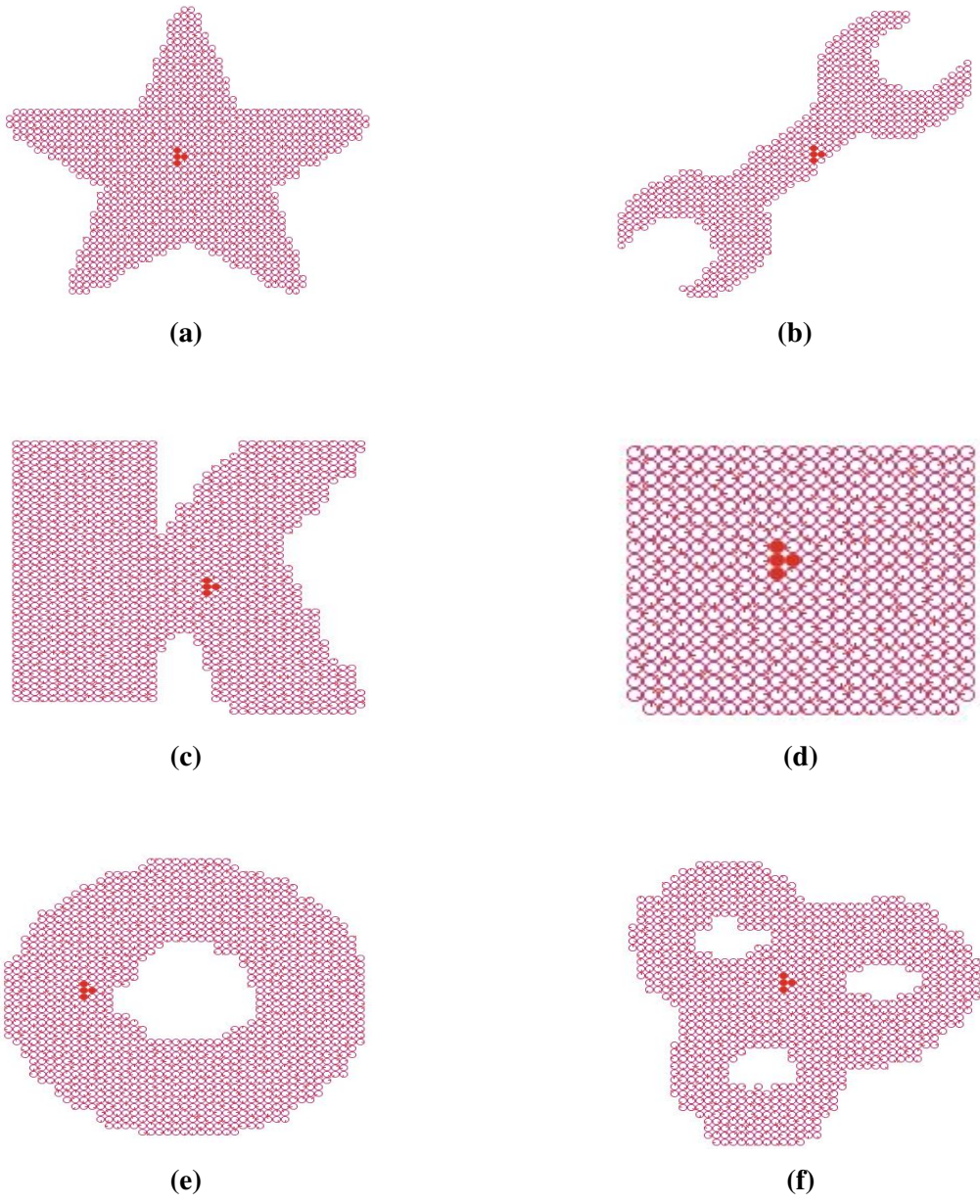
**Figure 7:** Desired six shapes formed by applying proposed shape formation algorithm in our simulation model.
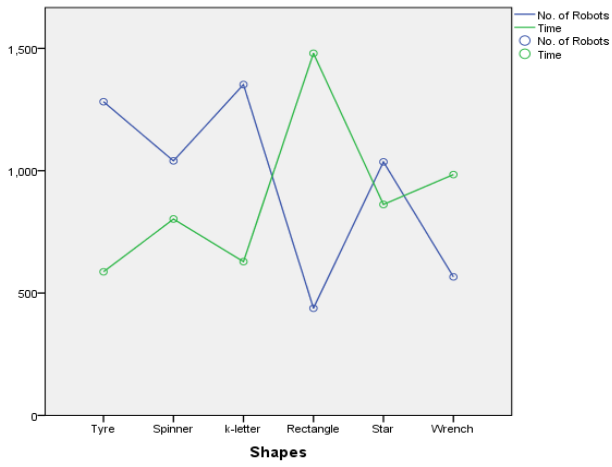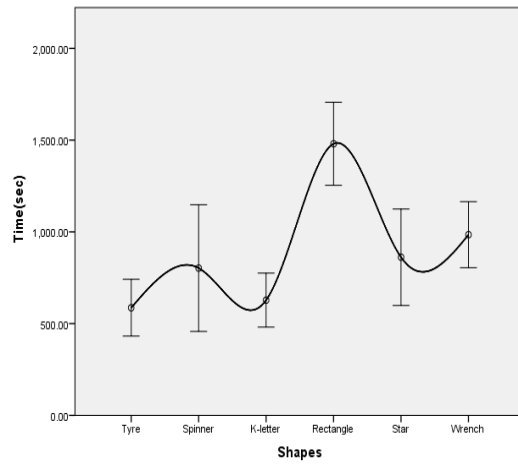
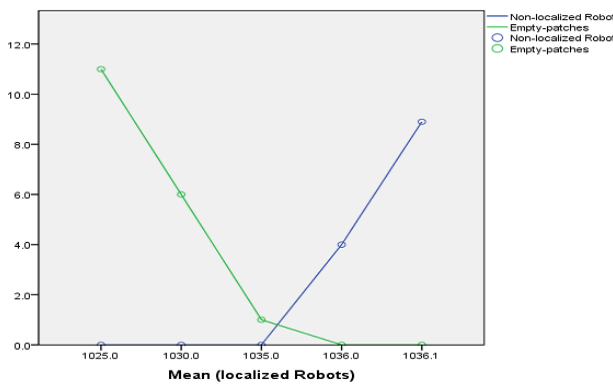(a)                                                         (b)

**Figure 8:** Experiment results for all six shape showing convergence time for number of robots along with 95% confidence interval.
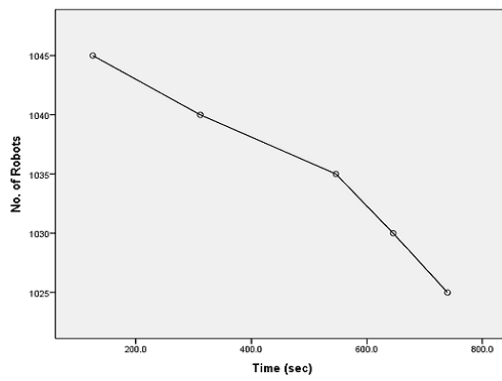
**(a)**
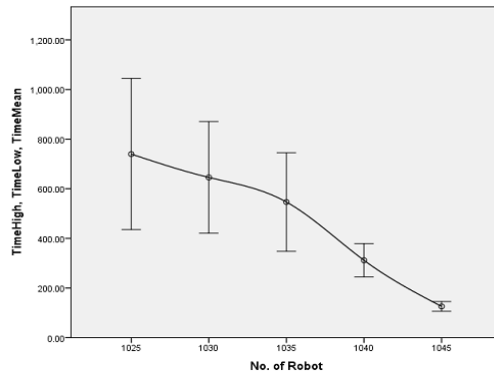


**(b)**



**(c)**
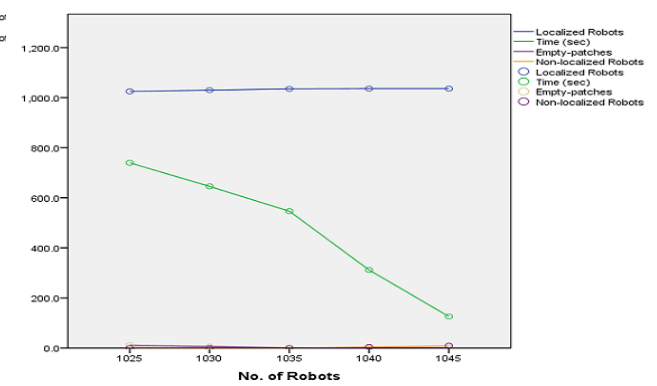


**(d)**

**Figure 9:** Experiment results for the "star" shape, show the time convergence and shape completeness for five different robot groups.
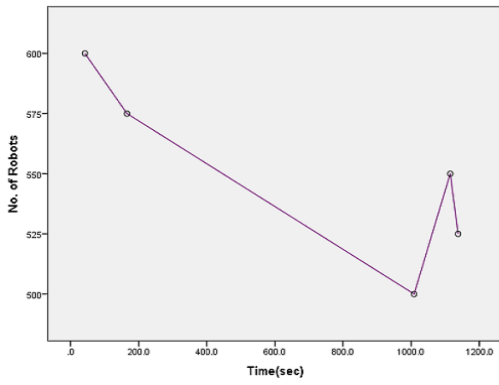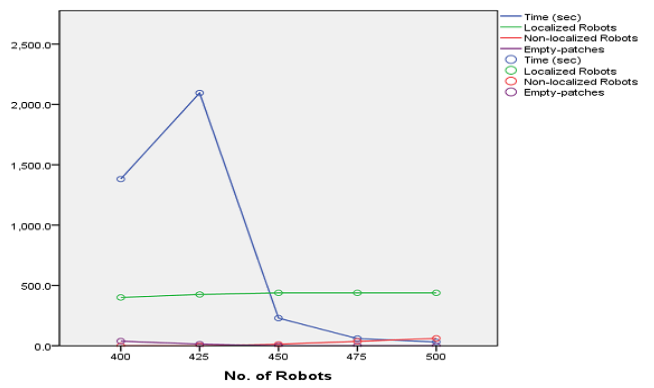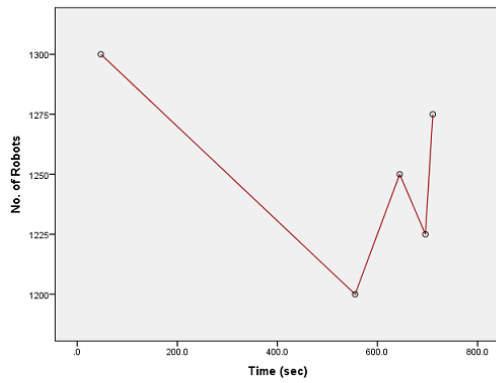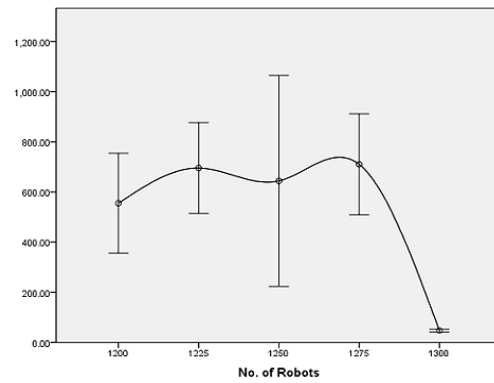
**(a)**
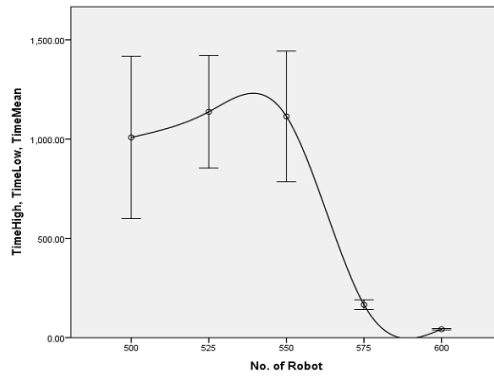


**(b)**
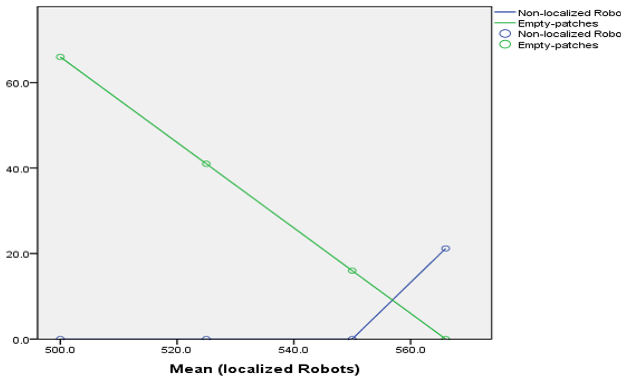


**(c)**
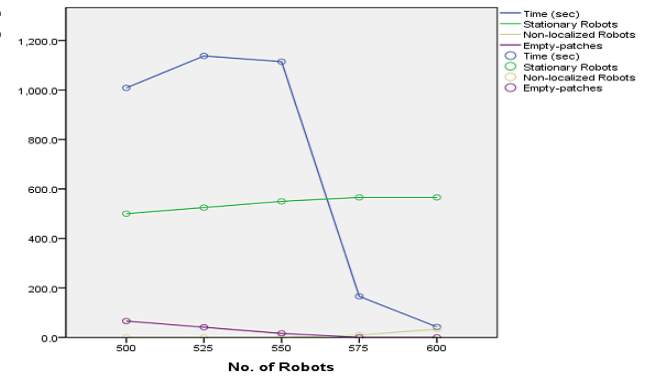


**(d)**

**Figure 10:** Experiment results for the "wrench" shape, show the time convergence and shape completeness for five different robot groups.
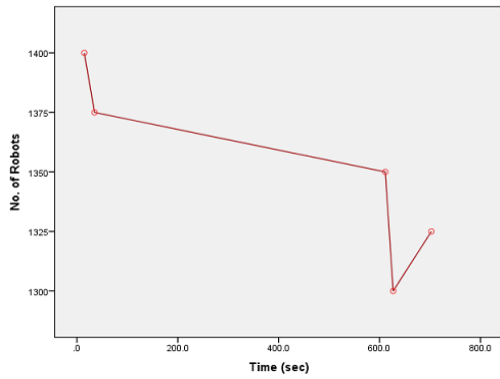
**(a)**
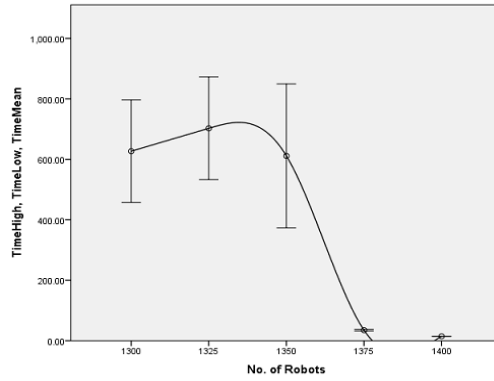


**(b)**
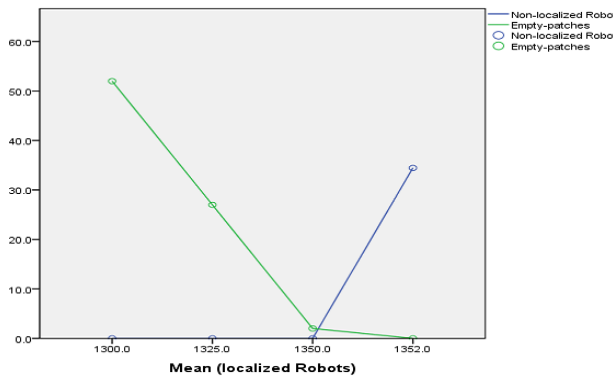


**(c)**
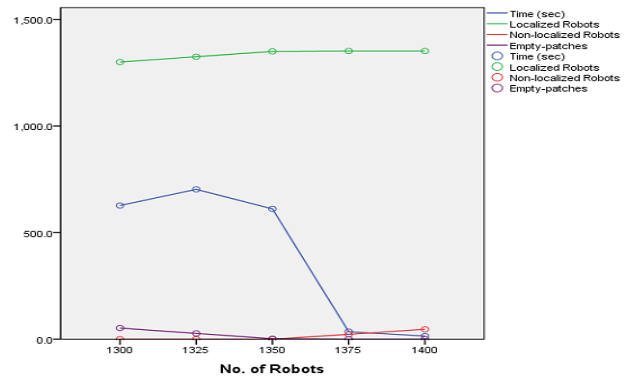


**(d)**

**Figure 11:** Experiment results for the "k-letter" shape, show the time convergence and shape completeness for five different robot groups.
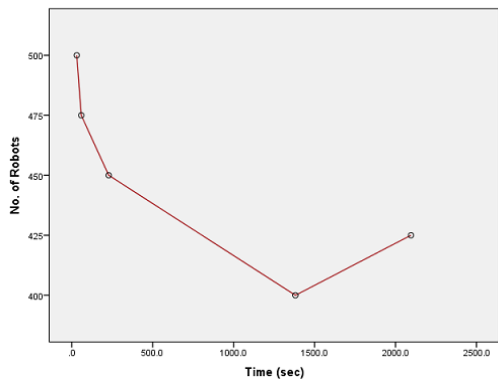
**(a)**
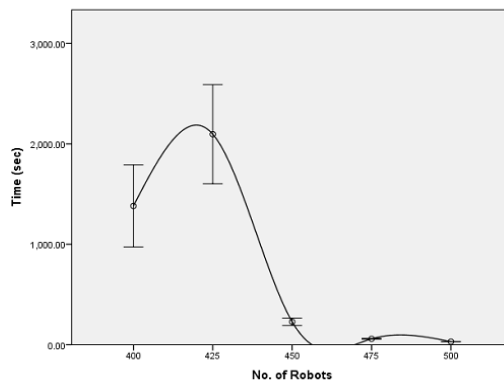


**(b)**
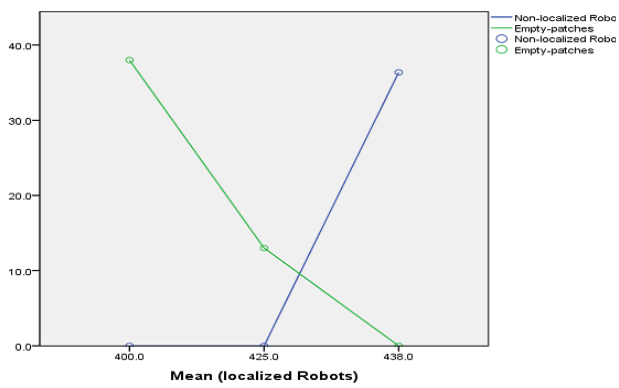


**(c)**



**(d)**

**Figure 12:** Experiment results for the "rectangle" shape, show the time convergence and shape completeness for five different robot groups.
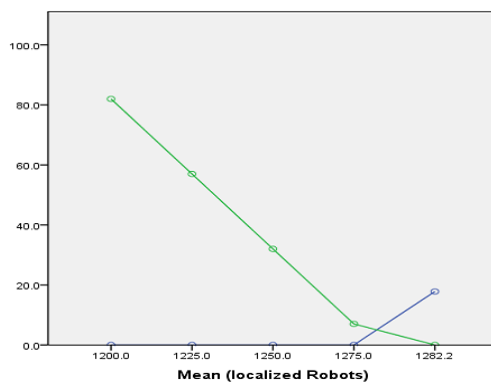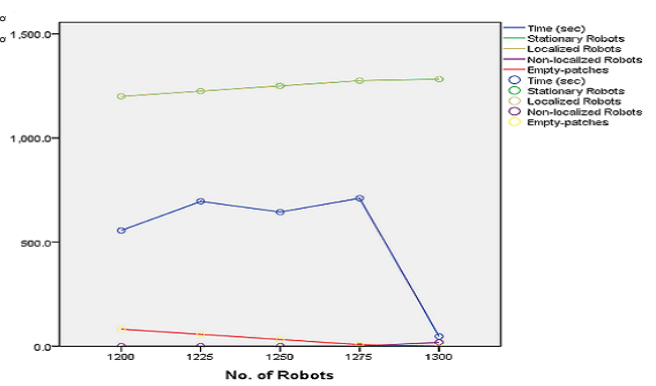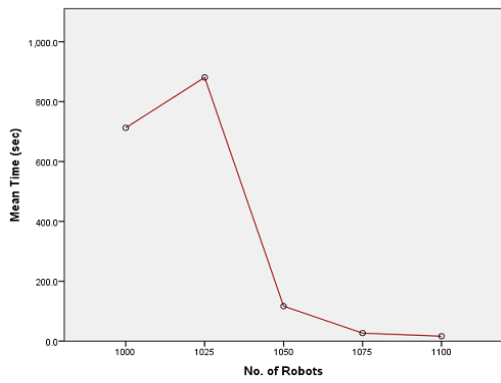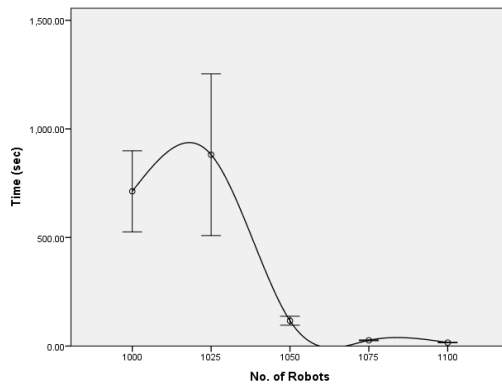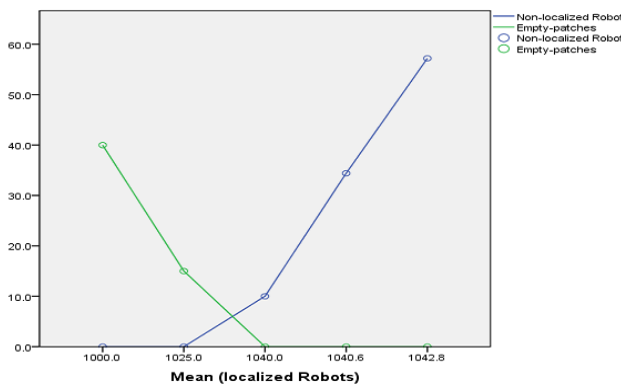
(a)



(b)



(c)



(d)

**Figure 13:** Experiment results for the "tyre" shape, show the time convergence and shape completeness for five different robot groups.
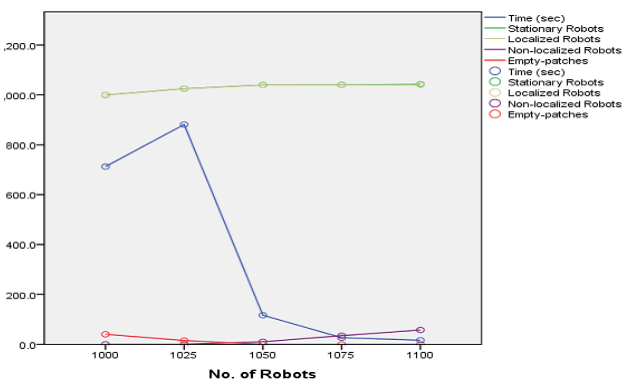
34

**(a)**



**(b)**



**(c)**



**(d)**

**Figure 14:** Experiment results for the "spinner" shape, show the time convergence and shape completeness for five different robot groups.

| Ref. : | Caption |
|---|---|

Fig 5: **a)** It represents the dimension or area in meters for the final formed shapes, **b)** The shape completion time measured in hours, shown for each of the thirteen experiments, **c)** A number of robots initially involved and a final number of robots after shape completion.

Fig 6: It shows the design interface for our agent-based model for the self-organized shape formation. The image shows the GUI of an environment world, where agents (purple colored) as robots placed at random locations while seed robots (red colored) placed near the origin at center of the world. There are sliders for controlling a number of robots and vision-radius for communication among neighbors. A chooser for selecting the desired shape, buttons for initializing and running simulation. The monitors are for monitoring the states of robots and plot for the robot convergence rate, which changes during simulation.

Fig 7: Final formed shapes by swarm robots: **(a)** A complete "star" shape formed by 1036 robots including four seed robots in our simulation model, **(b)** A final "wrench" shape formed through simulation in our model for self-organized shape formation by the convergence of 566 robots initially placed and move randomly in the world, **(c)** A complete "k-letter" shape formed by 1352 robots including four seed robots in our simulation model, **(d)** A "rectangle" shape formed through simulation in our model for self-organized shape formed by the convergence of 438 robots, **(e)** A "tyre" shape formed through simulation in our model for self-organized shape formation by the convergence of 1282 robots initially placed and move randomly in the world, **(f)** A final "spinner" shape formed through simulation in our model for self-organized shape formation by the convergence of 1040 robots.

Fig 8: The experiment results for all the desired shapes: **(a)** It shows the convergence rate for all six desired shapes, i.e. the total number of robots involved in forming the desired shapes with respect to the time taken for the formation, **(b)** It represents the convergence time with 95% confidence interval for all six shapes.

| Ref. : | Caption |
|---|---|
| Fig 9: | The simulation results for the "star" shape formation achieved by performing a number of experiments as n=10 for five (1025, 1030, 1035, 1040, and 1045) groups of robots and averaging their results: **(a)** The average convergence time for a different number of robots ranges from 125.7 to 739.7 iterations, shows less time for large robot group size as compare to small group size, **(b)** It shows the 95% of the confidence interval for convergence time for the above-mentioned groups as 105.96 to 1044.26 iterations, **(c)** The shape completeness shows the average number of localized and stationary robots as 1025 to 1036 robots with the remaining unlocalized robots 0 to 8.9 and empty spaces or patches within the shape range from 0 to 11 for all groups respectively. It showed that the group size of 1036 robots completely formed the desired "star" shape without leaving any empty patches within the shape and unlocalized extra robots. Below this group size, there will be empty spaces and shape will be incomplete while above this group size, there are unlocalize robots still remaining and waiting to become localize and stationary. Thus, they separate themselves from the self-organization process, **(d)** In this graph, convergence behavior variations are shown for the "star" shape. |
| Fig 10: | The "wrench" shape formation achieved by performing n=10 simulation experiments for five (500, 525, 550, 575, and 600) groups of robots and averaging their results: **(a)** The average convergence time for a different number of robots ranges from 42.3 to 1137.7 iterations, shows less time for the large robots group than small group size, **(b)** It shows the convergence time for the above-mentioned groups of robots with 95% of the confidence interval as 38.81 to 1443.49 iterations, **(c)** The "wrench" shape completeness measures as the average number of localized and stationary robots which are 500 to 566 with the remaining unlocalized are 0 to 33.4 and empty patches within the shape are 0 to 66 for each of the robot group size respectively. The group size of 566 robots completely formed the desired "wrench" shape without leaving any empty patches within the shape and un-localize extra robots. Below this group, the shape will be incomplete and above this size, there will more unlocalize robots outside the shape, **(d)** The variations in convergence in "wrench" shape comprises average time convergence, shape completeness for each of the robots group sizes. |
| Fig 11: | The simulation results for the "k-letter" formation by performing n=10 for five (1300, 1325, 1350, 1375, and 1400) groups of robots: **(a)** The average convergence time lies between 14.6 to 702.7 iterations for all groups, **(b)** 95% of the confidence interval 14.1 to 872.34 of convergence time for all groups, **(c)** The completeness shows the average number of localized and stationary robots ranges from 1300 to 1352 and 0 to 46.4 remaining unlocalize robots. The empty-patches within the shape are 0 to 52 for all groups. The group of 1352 robots completely formed a "k-letter" shape without leaving any empty-patches and unlocalize extra robots, **(d)** The convergence variations for the desired "k-letter" shape are shown for each of robots group. |

| Ref. : | Caption |
|---|---|

Fig 12: The results of (n=10) simulation experiments for "rectangle" shape formation by five (400, 425, 450, 475, and 500) groups of robots: **(a)** The value lie between 31.2 to 2095.6 iterations for the average convergence time of all five groups, it shows less time for the large robots group than small group size, **(a)** The 95% confidence interval value as 29.39 to 2589.03 for the convergence times of all groups, **(c)** The completeness of the "rectangle" shape depicts average number of localized robots as 400 to 438, unlocalize robots 0 to 60.8, and 0 to 38 empty-patches for all groups. It observed that the group of 438 robots completely formed "rectangle" shape without leaving any empty-patches and unlocalize robots. While below this group, the incomplete shape formed and unlocalize robots remained above this group size, so they separated themselves from the formation process, **(d)** The convergence variations are shown as average time convergence and shape completeness for each group of robots.

Fig 13: The five groups (1200, 1225, 1250, 1275, and 1300)of robots used in simulation experiment for the "tyre" shape and the obtained results are averaged : **(a)** The convergence time for a different number of robots lie between 47.2 to 710.5 iterations, **(b)** It shows the convergence time with 95% confidence interval as 41.82 to 1065.37 for all groups, **(c)** The shape completeness shows the average number of localized and stationary as 1200 to 1282 robots while remaining unlocalize are 0 to 17.8 robots with 0 to 82 value for the number of empty-spaces within the shape. A group of 1282 robots completely formed the desired "tyre" shape without leaving any empty patches within the shape, **(d)** The average time convergence and shape completeness for each group of robots is shown together as a variation in convergence .

Fig 14: The simulation results for the "spinner" shape formation by performing n=10 experiments for five groups (1000, 1025, 1050, 1075, and 1100) of robots: **(a).** The average convergence time is between 16.1 to 881.2 iterations, it showed less time for the large group than small number of robots, **(b)** The convergence time with 95% confidence interval of 15.57 to 1253.37 for number of robots, **(c)** The shape completeness depicts the average number of localized and stationary robots as 1000 to 1042.8 with remaining unlocalize robots as 0 to 57.2 along with 0 to 40 empty-patches within the shape for all groups. The group of 1040 robots completely formed "spinner" shape without leaving any empty patches and unlocalize robots. While less than this formed the incomplete shape and above this group size, there were unlocalize robots left, **(d)** Convergence variations of the "spinner" showing average time convergence and shape completeness for each of robots group.

# References

[1] Hyondong Oh, Ataollah Ramezan Shirazi, Chaoli Sun, and Yaochu Jin. Bio-inspired self-organising multi-robot pattern formation: A review. *Robotics and Autonomous Systems*, 91:83–100, 2017.

[2] Thomas Schmickl and Karl Crailsheim. Trophallaxis within a robotic swarm: bio-inspired communication among robots in a swarm. *Autonomous Robots*, 25(1):171–188, 2008.

[3] Reza Haghighi and Chien-Chern Cheah. Multi-group coordination control for robot swarms. *Automatica*, 48(10):2526–2534, 2012.

[4] Manish Kumar, Devendra P Garg, and Vijay Kumar. Segregation of heterogeneous units in a swarm of robotic agents. *IEEE Transactions on Automatic Control*, 55(3):743–748, 2010.

[5] Alan Oliveira de Sá, Nadia Nedjah, and Luiza de Macedo Mourelle. Distributed efficient localization in swarm robotic systems using swarm intelligence algorithms. *Neurocomputing*, 172:322–336, 2016.

[6] Jimming Cheng, Winston Cheng, and Radhika Nagpal. Robust and self-repairing formation control for swarms of mobile agents. In *AAAI*, volume 5, 2005.

[7] Michael Rubenstein, Alejandro Cornejo, and Radhika Nagpal. Programmable self-assembly in a thousand-robot swarm. *Science*, 345(6198):795–799, 2014.

[8] Muaz A Niazi. Technical problems with" programmable self-assembly in a thousand-robot swarm". *arXiv preprint arXiv:1708.03341*, 2017.

[9] Katherine Copenhagen, David A Quint, and Ajay Gopinathan. Self-organized sorting limits behavioral variability in swarms. *Scientific reports*, 6, 2016.

[10] Muaz A Niazi and Amir Hussain. *Cognitive agent-based computing-I: a unified framework for modeling complex adaptive systems using agent-based & complex network-based methods*. Springer Science & Business Media, 2012.

[11] Dervis Karaboga, Beyza Gorkemli, Celal Ozturk, and Nurhan Karaboga. A comprehensive survey: artificial bee colony (abc) algorithm and applications. *Artificial Intelligence Review*, 42(1):21–57, 2014.

[12] Ying Tan and Zhong-yang Zheng. Research advance in swarm robotics. *Defence Technology*, 9(1):18–39, 2013.

[13] Jan Carlo Barca and Y Ahmet Sekercioglu. Swarm robotics reviewed. *Robotica*, 31(3):345–359, 2013.

[14] Naoki Nishikawa, Reiji Suzuki, and Takaya Arita. Coordination control design of heterogeneous swarm robots by means of task-oriented optimization. *Artificial Life and Robotics*, 21(1):57–68, 2016.

[15] Blesson Varghese and Gerard McKee. A review and implementation of swarm pattern formation and transformation models. *International Journal of Intelligent Computing and Cybernetics*, 2(4):786–817, 2009.

[16] Zongsheng Wu, Weiping Fu, Ru Xue, and Wen Wang. A novel global path planning method for mobile robots based on teaching-learning-based optimization. *Information*, 7(3):39, 2016.

[17] Levent Bayındır. A review of swarm robotics tasks. *Neurocomputing*, 172:292–321, 2016.

[18] Muaz A Niazi. Self-organized customized content delivery architecture for ambient assisted environments. In *Proceedings of the third international workshop on Use of P2P, grid and agents for the development of content networks*, pages 45–54. ACM, 2008.

[19] Zhongshan Zhang, Keping Long, Jianping Wang, and Falko Dressler. On swarm intelligence inspired self-organized networking: its bionic mechanisms, designing principles and optimization approaches. *IEEE Communications Surveys & Tutorials*, 16(1):513–537, 2014.

[20] Rehan OGrady, Roderich Groß, Anders Lyhne Christensen, and Marco Dorigo. Self-assembly strategies in a group of autonomous mobile robots. *Autonomous Robots*, 28(4):439–455, 2010.

[21] Andrew Vardy. Aggregation in robot swarms using odometry. *Artificial Life and Robotics*, 21(4):443–450, 2016.

[22] Farshad Arvin, Ali Emre Turgut, Tomáš Krajník, and Shigang Yue. Investigation of cue-based aggregation in static and dynamic environments with a mobile robot swarm. *Adaptive Behavior*, 24(2):102–118, 2016.

[23] Valerio Sperati, Vito Trianni, and Stefano Nolfi. Self-organised path formation in a swarm of robots. *Swarm Intelligence*, 5(2):97–119, 2011.

[24] Shervin Nouyan, Alexandre Campo, and Marco Dorigo. Path formation in a robot swarm. *Swarm Intelligence*, 2(1):1–23, 2008.

[25] Erkin Bahceci, Onur Soysal, and Erol Sahin. A review: Pattern formation and adaptation in multi-robot systems. *Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-03-43*, 2003.

[26] Muaz A Niazi and Amir Hussain. A novel agent-based simulation framework for sensing in complex adaptive environments. *IEEE Sensors Journal*, 11(2):404–412, 2011.

[27] Muaz Niazi and Amir Hussain. Agent-based tools for modeling and simulation of self-organization in peer-to-peer, ad hoc, and other complex networks. *IEEE Communications Magazine*, 47(3), 2009.

[28] Muaz Ahmed Khan Niazi. *Towards A Novel Unified Framework for Developing Formal, Network and Validated Agent-Based Simulation Models of Complex Adaptive Systems*. PhD thesis, University of Stirling, 2011.

[29] Samreen Laghari and Muaz A Niazi. Modeling the internet of things, self-organizing and other complex adaptive communication networks: a cognitive agent-based computing approach. *PloS one*, 11(1):e0146760, 2016.

[30] Jim Woodcock and Jim Davies. *Using Z: specification, refinement, and proof*, volume 39. Prentice Hall Englewood Cliffs, 1996.

[31] Jonathan Peter Bowen. *Formal specification and documentation using Z: A case study approach*, volume 66. International Thomson Computer Press London, 1996.

[32] Mark d'Inverno, Michael Luck, and Michael M Luck. *Understanding agent systems*. Springer Science & Business Media, 2004.

[33] Uri Wilensky and I Evanston. Netlogo: Center for connected learning and computer-based modeling. *Northwestern University, Evanston, IL*, 4952, 1999.

[34] Chih-Chun Chen and Nathan Crilly. Describing complex design practices with a cross-domain framework: learning from synthetic biology and swarm robotics. *Research in Engineering Design*, 27(3):291–305, 2016.

[35] Long Qin, Yabing Zha, Quanjun Yin, and Yong Peng. Formation control of robotic swarm using bounded artificial forces. *The Scientific World Journal*, 2013, 2013.

[36] Yan Meng, Hongliang Guo, and Yaochu Jin. A morphogenetic approach to flexible and robust shape formation for swarm robotic systems. *Robotics and Autonomous Systems*, 61(1):25–38, 2013.

[37] Hahmin Jung and Dong Hun Kim. Potential-function-based shape formation in swarm simulation. *International Journal of Control, Automation and Systems*, 12(2):442–449, 2014.

[38] Muaz A Niazi. Emergence of a snake-like structure in mobile distributed agents: an exploratory agent-based modeling approach. *The Scientific World Journal*, 2014.

[39] Zhifu Chen and Tianguang Chu. Multi-agent system model with mixed coupling topologies for pattern formation and formation splitting. *Mathematical and Computer Modelling of Dynamical Systems*, 19(4):388–400, 2013.

[40] Chien Chern Cheah, Saing Paul Hou, and Jean Jacques E Slotine. Region-based shape control for a swarm of robots. *Automatica*, 45(10):2406–2411, 2009.

[41] Hongliang Guo, Yaochu Jin, and Yan Meng. A morphogenetic framework for self-organized multirobot pattern formation and boundary coverage. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 7(1):15, 2012.

[42] Farshad Arvin, Khairulmizam Samsudin, Abdul Rahman Ramli, and Masoud Bekravi. Imitation of honeybee aggregation with collective behavior of swarm robots. *International Journal of Computational Intelligence Systems*, 4(4):739–748, 2011.

[43] Samitha W Ekanayake and Pubudu N Pathirana. Formations of robotic swarm: an artificial force based approach. *International Journal of Advanced Robotic Systems*, 7(3):23, 2010.

[44] Huan Zhang and Pubudu N Pathirana. Optimization-based formation of autonomous mobile robots. *Robotica*, 29(4):515–525, 2011.

[45] Ahmet Çağdaş Seçkin, Ceyhun Karpuz, and Ahmet Özek. Feature matching based positioning algorithm for swarm robotics. *Computers & Electrical Engineering*, 2016.