ABSTRACT

Title of thesis: PROOF OF CONCEPT FOR PAIR BASED APPROACH FOR SWARM ROBOTICS

Anshuman Singh, Master of Science, 2021

Thesis directed by: Professor Mumu Xu
Institute of Systems Research

Many cases exist where teams of agents in a multi-agent system perform better than individual agents acting alone. In swarm robots, a large number of low-cost robots with limited functionality interact with each other and the environment to result in a more complex emergent behavior capable of performing tasks collaboratively. There exist many robotic swarms composed of single agents however, the study of swarms composed of modular robots and/or smaller teams, each acting as an independent unit, is a relatively new area of study. This thesis provides a proof of concept for a pair-based approach for swarm robots where two individual robots act as a single unit in the swarm called "duos," and the emergent behavior of the swarm consisting of these duos is studied by making concentric circles pattern using duos. For small swarm sizes, the duo swarm converged 31.6 % faster than the single agent swarm.

PROOF-OF-CONCEPT FOR PAIR-BASED APPROACH FOR SWARM ROBOTS

by

Anshuman Singh

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2021

Advisory Committee:
Professor Mumu Xu: Chair
Professor Michael Otte: Advisor, Co-chair
Professor Pratap Tokekar
Professor Yancy Diaz-Mercado

Dedication

I dedicate my thesis work to my family and my supportive friends who have always been a source of inspiration and encouragement. A special feeling of gratitude to younger brother and Alex for believing in me at every moment.

Acknowledgments

I would like to express my gratitude towards professor Michael Otte, my thesis advisor, who has been more than a mentor and advisor to me. It is because of his support and the Kilobot Simulator, I am able to conclude my research. I would also like to thank committee chair, professor Mumu Xu for providing me thoughtful insights and inspirations through constructive meetings and discussions with them. I would also like to thank my defense committee not only for their time and extreme patience but also their constructive feedback. I would like to extend my appreciation towards my family for being there for me through thick and thin and always encouraging me to strive for the best. Finally, I thank my friends, far and near, for their support and motivation.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

GPS	Global Positioning System
BFO	Bacteria Foraging Optimization
UAV	Unmanned Aerial Vehicle
SEC	Smallest Enclosing Circle
DK	Defago and Konagaya
COG	Centre of Gravity
LRV	Least Recently Visited
OHC	Overhead Controller
RMSE	Root Mean Squared Error

Chapter 1: Overview

1.1 Introduction

In the last decade, robotics has found itself to be an emerging part of human lives. As robots become more conventional, the switch from a single robot system to a multi-robot system is inevitable [2]. Recent advancements in robotics have allowed researchers to develop relatively inexpensive robots, making it feasible to allocate a large number of robots to perform desired tasks. This approach to coordinate a large number of relatively simple robots to perform a task is termed Swarm robotics [3] Furthermore, the need for inexpensive and simpler robots arises from the trade-off between the size of the swarm and the complexity of the robots used. Swarm Robotics has found its inspiration from various natural phenomena such as ants forming trails to a food source, birds flocking, fishes swimming in schools etc. The resulting synergy from the interaction/cooperation of individuals in the swarm, the simultaneous complexity of the swarm and simplicity of individuals in swarms have always fascinated researchers and non-researchers alike.

One of the major properties that emerge from the complex behavior of the swarm is self-organization which is defined as "a process in which pattern at the global level of a system emerges solely from numerous interactions among the lower-

level components of the system" [4]. The lower-level components, i.e individual agents in the swarm, do not have a central controller but are still able to organize themselves in a pattern and hold it spatiotemporally. Such behavior can be seen in the process of nest building in bees and ants, where the individual insect determines its position based upon its neighboring insects and structure. Extending the same analogy to robots, a robotic system may be called self-organizing if given a random initial configuration, the robots are able to form a desired pattern autonomously. There has been extensive research on self-organizing multi-robot systems and swarm robots in the recent years and yet coordinating multiple robots to perform a task still presents a challenge. Hence, much of the research has been focused on simple geometric pattern formation, mostly circles which is detailed in Section 2.2. To the best of my knowledge, all the aforementioned studies have been performed on swarms consisting individual agents. A circle is one of the simplest and a perfectly symmetric geometric pattern which makes its formation easy. However, it is not such a simple task for anonymous and simple robots due to lack of sensory inputs and having a decentralized controller. One could argue that the fundamental idea behind forming circles is to enable a randomly initialized swarm to converge in a symmetric and regular shape where robots can localize each other, and special tasks can be allocated to some special robots to build more complex shapes. Hence, I believe that once the problem of circle formation is solved, the solution can be extended to solve more complex formation and organization problems. Concentric circles, on the other hand, form a more application-based topology. Let us assume the use of swarm robots to control wildfire [5], where robots over an area organize themselves

and suppress the fire omnidirectionally by forming a circular shape. The fire may damage one of the robots and could spread outwards from that point of failure. Hence, arranging robots in concentric circles would negate the failure of some agents in the system and hence providing us with a robust system. Furthermore, all the robots in the system will have a common point of census, i.e., the center and the shape can be scaled up or down radially for a large number of robots.

In this thesis, we expand on the idea of self-organization and study the emergent behavior, trade-offs and evaluate pair-based approach to swarm robotics. This work is inspired by modular robots where multiple individual modules attach together resulting in a more capable agent. Consider a non-homogeneous swarm with robots capable of performing specialized tasks. The hypothesis is that if two agents with different capabilities work together as a single unit, the resulting unit is will have more capabilities than the individual agent. The aim of the thesis is to provide a proof of concept for a pair-based approach for swarm robots where two individual robots known as "duo" act as a single unit in the swarm, and the emergent behavior of the swarm consisting of these duos is studied. The proof-of-concept is given by a proposed pair-based adaptive spatial formation algorithm to form concentric circles irrespective of the size of the swarm. Spatial formation is an emerging research area in swarm robotics and multi-agent systems, inspired by the biology and physics of bacterial colonies [6]. Much of the research on pattern formation has been performed in simulation [7] [8] [9] [10] [11] which assumes that the robot has been accurately modeled and also does not take sensing noise and disturbance from environments into consideration. Hence in order to validate pattern forming algorithms, one could

argue that the algorithms should be tested on large-scale actual robotic systems. Considering that the developed algorithm must be tested large-scale actual robotic systems, Kilobots were chosen due to their low lost and scalability. Kilobots have a diameter of 3.3 cm, a communication radius (r_{comm}) of 10 cm, no self-localization or any knowledge of their orientation. Considering these constraints of Kilobots existing pattern formation algorithms which use either a global coordinate system, have knowledge of position and/or orientation of other robots cannot be implemented using Kilobots. The small communication radius of Kilobots makes formation of large geometrical patterns difficult. Furthermore Kilobots also have a constraint of nine bytes on message payload which limits the amount of data transmitted or received. Hence, to form circles with radius greater than r_{comm} around a Kilobot, a new decentralized algorithm needs to be developed. In this thesis an algorithm to form concentric circles around a seed Kilobot has been presented. The algorithm converges a swarm of N Kilobots, arranged into $\frac{N}{2}$ duos which are composed of two Kilobot modules as shown in Fig 3.2, into n concentric circles.

1.2 Contribution of the thesis

There are 4 main contributions of this thesis:

1. Implementation of a swarm of duos. To best of my knowledge, this work is the first to implement a swarm of duos where two separate robot modules act as individual agents in the swarm.

2. Developed the first distributed algorithm for swarm consisting of 2 robot

modules acting as individual agents

3. Proof of concept in simulation and hardware of forming concentric circles using a swarm of duos

4. Extension of the capabilities of an already existing simulator to simulate the behavior of Kilobot duo swarm

1.3 Background on Swarm and Related Concepts

1.3.1 Swarm Robotics

Swarm robotics is the approach to coordinate large groups of relatively simple autonomous robots following a simple set of local rules to accomplish a task that would not be possible for an individual robot [12]. The robots interact with each other and the environment to result in a more complex emergent behavior capable of performing tasks collaboratively. The operation and performance of these robots, unlike traditional robots, do not depend on a sophisticated central controller or complex mechanical properties. Instead, swarm robotics finds its advantages by interacting with each other and the environment to work collaboratively. This approach results in some desirable behavior such as robustness, flexibility, and high reliability in task performance, e.g., if some robots in the system fail, other robots can take over finish the same task; different spatial formations in different environments. Another advantage of swarm robotics is scalability, i.e., the robot swarm can adapt to change in the swarm population without interrupting the whole swarm. Additionally,

due to robotics being such a vast field of research and nature being a source of inspiration for many research areas, a few research areas are often confused with swarm robotics [13] [6]

1. Multi-robot systems: Many multi-robot systems have a small number of agents whereas swarms are expected to scale up-to large numbers. Multi-robot systems generally use centralized control schemes but decentralized and distributed control schemes are also used. The agents can be heterogeneous for specialized tasks while trading off on flexibility, scalability, and re-usability [6].

2. Multi-agent systems: Multi-agent system is a super-category of swarm robotics.Multi-agent systems provides the implementation framework for the swarm-based methods to be implemented. The control scheme may be either centralized control, hierarchical or network control in a known environment. The agents may be homogeneous or heterogeneous depending upon the application of the system but are more flexible and scalable than multi-robot systems [6].

3. Sensor Networks: These systems usually have a fixed number of stationary agents (sensors) with centralized or remote control. The agents are usually homogeneous and monitor data in different locations and transmit them to a central location for further processing. They are usually implemented in *known* environments since they rely on human intervention to determine the data to be measured and determine their optimum placement in the environment [6].

1.3.2 Biological Inspiration for swarm robotics

Swarm Robotics research has been inspired by social insects, fishes, birds, and even humans [14] The collective behavior of social insects, such as foraging, cooperation, migration, has fascinated researchers. The hypothesis is that robots designed similar to their natural counterpart will have similar constraints on how the robots will perceive the environment, interact with each other and the environment. Furthermore, if the problem we are trying to address is similar to those solved by insects, then the algorithm developed by the insects may very well be used for designing algorithms for robots and even serve a baseline for performance comparisons [2]

1.3.2.1 Bacteria

Bacteria are one of the most fundamental life forms to exist. They function as a syntactic multicellular aggregate of microorganisms called biofilms while exchanging inter-cell communication signals for tasks like labor division, defense against dangers, and foraging [15]. Studies have shown that biofilms have about 500 times more resistance to antibacterial agents than individual bacteria of the same kind. Researchers have also developed a foraging algorithm inspired by E. Coli bacteria that uses chemical attractants and repellents to move towards amino acids like asparte and serine [16]. Since then, few modifications and iterations of Bacteria Foraging Optimization (BFO) have been made

1.3.2.2 Ants and Bee

Eusocial insects such as bees and ants provide a large number of algorithms for communication, foraging, task allocation that can be extended to swarm robots. The most common form of communication is to use pheromones in conjunction with other external cues e.g., ants use pheromones, touch, and sound to communicate with each other [17] to perform tasks like building mounds, organize foraging raids, defense. Ants are also a great example to showcase task allocation based on the individual's success during foraging. An ant with a successful foraging attempt leaves a pheromone trail on the shortest path on its way back to the nest. Successful paths are taken by more ants, hence identifying the best path [18]. Social insects also have the innate ability to get a global view of tasks, environment by only using the limited local information available, e.g., bees can estimate the global workload balancing by assessing simple and local indicators such as queuing delays [19]

1.3.2.3 Fish Schools

Many species of fish move in schools in a coordinated manner effortlessly. They are able to move upstream and downstream while keeping their formation with the help of "schooling marks" on their shoulders or tails [20]. These schooling marks serve as markers for other fishes, which helps them determine and adjust their school speed and hence maintain the formation. Swimming in disciplined phalanxes helps the fish schools evade predators [21] and have more success in foraging than individual fish [22]

1.3.2.4 Birds

Approximately 18% of the bird species are long-distance migrants [23] and flock in search of food and mating partners. Flocking may be defined as the collective motion of a group which arises from the emergent behavior due interaction of birds following simple rules. Many birds like pelicans migrate in flocks that are aerodynamically efficient and reduce energy consumption [24]. Migratory species of birds use various techniques including Magnetoreception, landmarks, olfactory cues, sun compass to help with navigation over long distances [25]

1.3.3 Advantages of Swarm Robotics

A vast number of operational robots are individual, autonomous with a central controller performing specific tasks. These robots designed with high mechanical complexity, sophisticated control, and communication architecture, which often results in high construction and life cycle costs. Furthermore, due to the complex design, they are prone to multiple failure points and make their repair and debugging a cumbersome task. Conversely, swarm robots make use of the emergent behavior to eliminate these issues and complete the same task through local interaction and cooperation. Following a simple set of rules and having simpler, cheaper hardware, swarm robots present various advantages over traditional single robots:

1. Scalability : As seen in natural swarms, the tasks are still performed successfully even if the population of the swarm is changed significantly. This can be attributed to the local interaction between agents, resulting in local decisions

which allows agents to enter and leave the swarm without interrupting the swarm. Whenever the population of a swarm is changed, a dynamic task reallocation takes place according to the new population without external intervention. Using the same analogy, in swarm robots, the swarm can adapt to the change in population without any change in hardware or software, which allows the use of a large number of robots in the swarm.

2. Robustness : In the context of swarm robotics, robustness of a system refers to the swarm's ability to still perform the desired task in the presence of some failed individuals in the swarm.One may observe the robustness of swarms in their own backyards by observing ant trails. If one disturbs the trail and removes some of the individuals, other ants take the place of the removed individuals and carry on the trail as before, albeit with lower performance. Swarms exhibit this behavior due to redundancy, i.e., in case of any malfunction of an individual, another individual could take its place and perform the same tasks. It eliminates the dependency of the swarm on any individual agent in the swarm and makes them expendable, unlike agents in traditional robot systems.

3. Parallelism and Flexibility : The large number of individual agents in the swarm allows for decomposing a task into sub-tasks and allocating them to a different group of robots to execute them concurrently to complete the task faster. This parallelism in task execution allows swarm robots to deal with multiple targets spread over a large area in search and rescue tasks. It also

allows for flexibility in swarm i.e., generate sub-solutions for sub-tasks hence generating modular solution for a problem. Flexibility allows us to perform different tasks in different environments with the same hardware and minimal changes in software.

4. Economical : One major factor that enables extensive research on swarm robots is the relatively low cost of design, manufacturing, and maintenance over its life cycle. This low cost allows researchers to have a large swarm and replace faulty units with a new one rather than replacing the faulty component which requires more effort and time. Furthermore, the use of inexpensive robots allow users to deploy robotic swarm for dangerous tasks such as disaster relief, search and rescue.

Chapter 2: Related Works

2.1 Overview

Coordination, decentralized control, and self-organization are common themes of swarm robotics [12] [26] Most applications, algorithms in swarm robotics make use of the aforementioned characteristics for their design and development. This chapter focuses on self-organization in multi-robot systems and swarm robots, alongside a survey of most representative experiments and algorithms will be presented. Most of the algorithms presented in this chapter are for robots with localization information such as GPS, location of other robots, orientation. Furthermore, majority of the algorithms were implemented in simulation. The algorithm presented in this thesis takes inspiration from algorithms presented in Section 2.2.2 and Section 2.3 and can be implemented on robots which do not have neither of above information explicitly such as Kilobots.

2.2 Pattern Formation

Pattern formation is the most fundamental characteristic of swarms found in nature and has been a subject of great interest for researchers. Pattern formation

may be defined as organizing a group of robots in a global configuration from a given or an arbitrary initial configuration by manipulating the positions of individual robots [27] Pattern formation can be divided into two categories based on the control strategies: centralized and decentralized. Centralized control is the strategy of controlling agents in a networked environment where a single agent or a controller plans for other agents in the system. Centralized control can produce optimal plans for the group; however,fails when the leader fails. In Decentralized control, each agent autonomously decides for itself based on simple rules. This allows decentralized control to be parallel, faster, scalable, and robust since no single point of failure exists.

2.2.1 Pattern formation using centralized control

Sarno et al. [28] presented a path planning strategy to reconfigure a cluster of satellites autonomously. Based on convex-optimization [29],the algorithm plans the path by generating maneuvering trajectories based on the collision constraints. The initial state of formation is computed on a "chief unit" while exchanging absolute positions and then determining the spatial arrangement for the cluster of satellites. Next, to reconfigure into the desired configuration, the task-assignment is performed by a Genetic Algorithm by minimizing propellant consumption of the whole formation. Another centralized path-planning methodology is proposed by Koo and Shahruz [30] for Unmanned Aerial Vehicles to fly them in a desired formation. The study focuses on trajectory computation and consider two cases: Unmanned Aerial Vehicles (UAVs)

taking off successively and simultaneously. The path-planning is delegated to a leader UAV having special equipment such as cameras and sensors. The leader UAV determines the flight trajectories for the follower UAVs and send them to follower UAVs. A model-independent coordination strategy proposed by [31] allowed multiple mobile robots to arrange themselves in the desired configuration on a path. The strategy uses a virtual reference controlled by a remote supervisor on the desired trajectory to maintain the predefined positions of individual robots. The authors proved that if the tracking errors are bounded, the formation error is stabilized with the assumption that the information on the path to follow is perfect. Belta and Kumar [32] proposed a method of generating trajectories for multiple non-holonomic robots through kinetic energy shaping. The general idea behind the methodology is to change the kinetic energy metric to an artificial kinetic energy metric and tuning this metric leads to a desired rigid-body motion. Keshmiri and Payandeh [33] suggested a centralized workload distribution approach for multi-agent systems. The central controller collects data such as sensor data and positional data from other robots and delegates tasks to the most suitable robot.

2.2.2 Pattern formation using decentralized control

Coordination of a large number of robots presents a complex control problem due to the large workspace and voluminous communication between the robots. Hence, centralized control strategies are not feasible for swarm robots. Suzuki et al. [8] first explored the problem of approximate formation of a circle of diameter

"D" using multiple robots by proposing a heuristic method to control a group of robots. The robots are anonymous, i.e., have no identifiable label, identical in the sense that they all run the same algorithm, do not have a sense of orientation, and cannot determine their position in a global coordinate system.They assumed that each robot has the positional information about all other robots in the system in the unlimited visibility case and only the position information of its neighbors in the limited visibility case. Each robot calculates the distance "d" between its closest neighbor and the farthest neighbor. If $d > D - \alpha$, then it moves away from the farthest neighbors; if $d > D$, then the robot moves towards the farthest neighbor. The third condition distributes the circle more evenly.The algorithm was simulated on 50 robots, and sometimes the algorithm would converge to a reuleaux triangle instead of a circle. The algorithm was improved by Tanaka [7] by using the midpoint of the closest neighbor and the farthest neighbor. If the distance of the robot, while moving away from its closest neighbor to the midpoint, is approximately equal to the given target radius, then the robot adjusts its position treating the midpoint as the center of the circle to which all robots are converging. Extending the limited visibility case of Suzuki's work, Flocchini et al [34] focused on "gathering" multiple robots at a single location where the robots have a sense of orientation. They proved that only having a sense of orientation is sufficient to gather robots at an arbitrary point. In their future works, they assume robots with a sense of direction of the two axes, i.e., x and y [35]. They present the formation of arbitrary patterns and "gathering" under both limited visibility and unlimited visibility. In their subsequent work [36], they study the pattern formation problem under following cases: (i) robots know the

direction and orientation of both axes (ii) robots know one axis and orientation (iii) robots know one axis direction (iv) robots know neither the direction nor orientation. They show that pattern formation problem can be solved in cases i, ii and iii but not in case iv. Deshpande et al [9] and present a rule-based circle formation algorithm where all robots know their positions in a global coordinate system and can estimate other robot's distance from each other. The algorithm was simulated in MATLAB on 20,50 and 100 robots. However, one shortcoming of this algorithm is that it does not decide on the center of the circle formed hence cannot be used in applications where the location of the task is of importance e.g. containing an oil spill, fire etc. Yun et al. [37] presented an algorithm for line and circle formation. They presented a modified version of Sugihara and Suzuki's circle formation algorithm, which we discussed earlier. The modified algorithm calculates the robot's distance d (as mentioned in the original Sugihara's algorithm) from the centroid of its farthest neighbor, the closest neighbor, and the second closest neighbor. The robot then moves according to the rules in Sugihara's algorithm. They also present a novel algorithm called the Merge-Then-Circle algorithm, where all robots move towards the midpoint of their closest and farthest neighbor. Iteratively doing the same brings all robots to a cluster. Once all the robots are in the cluster, they start moving outwards towards empty space until they have traversed a distance equal to the radius of the circle. Finally, when all robots are on the circumference, each robot tries to move towards the midpoint of its immediate left and right neighbors to create a uniformly spaced circle.

Defago and Konagaya [10] present a circle formation algorithm using Suzuki's

robot model. The algorithm is separated into two parts, each of which solves a sub-problem. The first part of the algorithm uses Smallest Enclosing Circle (SEC)[1] to arrange all robots on the circumference of a circle. The second part of the algorithm distributes the robots on the circumference uniformly. A few movement constraints,based on Voronoi tessellation space, are also imposed on the robots to ensure that no two robots occupy the same position simultaneously and the SEC remains invariant, i.e., despite having different views of the environment, the SEC calculated by all the robots should be the same. All robots move to occupy the circumference of the SEC and then uniformly space themselves on the circumference by moving towards the midpoint of its immediate left and right neighbors. The authors called this algorithm the DK algorithm. Chatzigiannakis et al. [38] stated that Defago and Konagaya (DK) algorithm was computationally expensive due to Voronoi diagrams. The authors argued that when the number of robots is limited and/or the operating area is large, the aforementioned possibility is not probable. In their algorithm, after calculating the SEC, a particular robot decides a position on the circumference which is closest to itself and moves to occupy that position. After all the robots have occupied their positions on the circumference, they uniformly distribute themselves like in the DK algorithm. Ando et al. [39] presented a circumcenter algorithm for robots with limited visibility towards an undetermined point. They calculate a target point for every robot, dependent upon the SEC of the relative positions of its neighbors. The target point is chosen such that it is in the direction of center of the SEC at some distance, MOVE, from the robot. The distance MOVE

[1]SEC is the smallest circle that contains all of a given set of points in the Euclidean plane

is calculated such that if the robots and the neighbors move simultaneously, they are guaranteed to be within a specific distance to each other. However, the algorithm does not solve in finite time. Lee et al [40] presented a geometric approach to enable a large swarm of robots to form concentric circles in a two-dimensional plane. A robot locally communicates with its neighbors and forms an isosceles triangle which is followed by the robots generating circles circumscribed by regular n-polygons. Concurrently, the robots also reach consensus on the number of robots in the same circle which is followed by the agreement upon the centroid of individual circles. The algorithm was able to generate concentric circles in simulations. Hasan et al [41] presented an algorithm for circle formation for robots with limited visibility with a few assumptions : at initial placement of robots, one leader robot will have all other robots in its visibility range; all robots have an agreement on the orientation of x-axis. The algorithm is fairly straightforward; the leader robot stays stationary at the center of the circle and tells other robots the position to occupy on the circumference. The leader finds the farthest robot, uses it as the reference point, and determines the radius of the circle. Once all the robots are aware of the radius, target points are decided on the circumference by dividing the circumference into equal parts. Guzel et al. [42] presented a similar leader-based algorithm where robots move according to artificial attractive and repulsive potential fields. The leader gets the location of all robots in the swarm, calculates the Centre of Gravity (COG) point of the swarm and the relative distances of each robot with respect to the COG. The leader sorts the robots according to the angle they make with the COG while using an artificial vector from the leader to COG as the starting point. Then the coordinates

of possible positions that robots can take in the pattern are calculated.

2.3 Aggregation and Dispersion

Aggregation is one of the most frequently occurring phenomena in nature and is essential for creating function groups among social animals. It may be considered a prerequisite for accomplishing tasks such as collective movement, pattern formation, and self-assembly [43]. Trianni et al [44] studied the aggregation problem where the robots attract and repulse other robots based on the sound and infrared sensors. A robot generates its idea of the neighborhood and stochastically executes a predefined behavioral pattern which includes moving to and away from other robots. The probability of robot selecting a behavioral pattern depends on the robot's idea of the neighborhood. In their future work [43], an evolution-based algorithm was presented for static and dynamic aggregation. Static aggregation case results in compact and stable stationary aggregate, whereas the dynamic aggregate forms loose but a mobile aggregate. Jeanson et al. [45] presented an agent-based model to explain aggregation at a dynamic level that revealed that the emergence of clusters relies on the positive feedback from the individuals, i.e., more agents aggregate towards a cluster that already has a considerable size. Correll et al [46] adopted Jeanson's probabilistic model and used 20 Alice robots to show that when using probabilistic approaches for aggregation, one requires minimum combination metrics such as communication range and speed. They also showed that the approach is sensitive to agent's capabilities such as communication range and speed. Soysal and Sahin [47]

presented a probabilistic finite-state machine-based algorithm using a macroscopic model for estimating the final state of the aggregate which results from iteratively executing three steps: random walk, wait and approach. Each step is executed for a finite amount of time before moving on to the next step, and the robot's motion is defined by the current executing step.

Dispersion in multi-robot systems can be defined as maximizing the senor coverage area while maintaining swarm connectivity [48]. Howard et al. [49] presented an algorithm for dispersion based on potential fields where robots were assumed to virtual particles, and the motion was controlled by virtual forces which allowed robots to repel each other and get repelled by obstacles. Batin et al. [50] presented a Least Recently Visited (LRV) algorithm for dispersion based on a beacon-based network. A mobile agent deploys a set of beacons in the explored workspace in a triangular,hexagonal, or square pattern. The placed beacons instruct the mobile robot for future beacons placement. The beacons are responsible for driving the robots in the workspace. The results were good for dispersion; however not feasible for path planning and the algorithm had no termination point. Ugur et al [48] presented an dispersion algorithm based on wireless signal intensities. The intensity readings obtained are used as range estimates. However, the intensity readings do not give an estimate of the orientation of the robot and the intensity readings are affected by the two communicating robots' orientation. McLurkin and Smith [51] presented algorithms for bounded environment using only sensing other robot's position and communication between robots. The algorithm was tested on fifty-six iRobots which use an infrared communication system for obstacle avoidance and obtain relative

positions.

2.4 Modular Robots

Modular self-reconfiguring robots are anonymous mobile robots with variable morphology [52], which autonomously self-organize and change their morphology to adapt to different tasks or terrains. Self-reconfiguring modular robots connect and disconnect with each other physically without human intervention and are also capable of self-repair [53] [54]. Modular self-configuring robots can be classified based on geometric architecture [55].

2.4.1 Lattice-Based Architecture

In lattice-based architecture, modules are attached and arranged in a three-dimensional cubic or hexagonal pattern. Mytilinaios et al. [56] demonstrated homogenous modular robots called molecubes which are 10 cm cubes with two rotating halves. Each cube can rotate independently and parallelly. Jørgensen et al. [57] developed a self-reconfigurable robot called the ALTRON whose module is an assembly of two semi-spheres joined together, with each half being able to rotate independently in either direction. The modules are arranged in a subset of a surface-centered cubic lattice and arranged such that two connected modules' rotation axes are orthogonal to each other. In their future works, they presented a distributed cluster walk [2] algorithm for ALTRON which is advantageous in rough and unpredictable terrains

[2]Custer walk is a type of locomotion requiring self-reconfiguration where modules move from one part of the cluster to another, allowing the whole configuration to move

where modules can configure accordingly and adapt.

2.4.2 Chain-Based Architecture

In chain-based architecture, the modules are connected in a single chain or a tree topology. Shen et al. [58] developed hormone inspired control framework for adaptive communication and distributed control. Virtual hormones are created inside each module as a response to sensory inputs. These hormones can interact with each other and diffuse, representing information that propagates through the configuration of modules and acts on output hormones that drive the actuators. Polybot is another chain-inspired robotic module capable of forming 3-D structures whose assembly strategy was similar to CONRO. They can dock a six-module arm to a single module at a fixed and predefined location [59]. Using appropriate control for modules, chain architectures can reach any point within their workspace and hence more practically feasible than lattice architecture robots. However, chain architecture robots is harder to control and computationally more challenging to represent [54]

2.4.3 Mobile Architecture

In mobile architecture, the individual modules can move around in the workspace and are able to form complex chains and separate groups to carry out different tasks. Fukuda et al. [60] presented CEBOT which comprises three cells for movement, a wheel mobile cell, rotation joint cell, and a bending joint cell. For docking and

latching, the cells are equipped with SMA couplers. Castano et al. [61] presented CONRO a snake-like modular robot whose assembly consists of two segments consisting of two modules each. Each module consists of three different parts, the passive connector, the body, and the active connector. A docking methodlogy for CONRO was presented by Rubenstein et al [62] aligning male-female mechanisms with the help of an IR interface alignment. Christensen et al. [63] presented a distributed control scheme for S-BOTS that forms desired connected morphology. The algorithm uses a seed robot to initiate the pattern formation process. It opens a communication channel and waits for another robot to physically attach itself to the root robot with a physical gripper. When the pattern starts to form, the robots are already a part of the shape dictate where the other robots should assemble them.

Chapter 3: Kilobots and Kilobot Simulator

In this chapter a brief overview of Kilobots and a natively developed Kilobot simulator shall be given. The design and the dynamics of a Kilobot duo shall be presented as well.

3.1 Kilobots

Figure 3.1: Kilobot [1]

Kilobots, as shown in 3.1, are small, inexpensive robots designed by the Self-Organizing Systems Research Lab at Harvard University. Each robot has a diameter of 3.3 cm and is powered by a rechargeable 3.7 V Li-On battery. Kilobots uses a non conventional method of locomotion, they use two coin shaped vibration motors working on slip-stick principle. Each motor can be independently controlled by

adjusting the power level value from 0-255. This design significantly reduces the hardware complexity, cost of the robot and the robot's size while allowing the robots to move clockwise, anti-clockwise and straight with help of 3 legs. Kilobots have ATmega 328p 8 bit micro-controller which has 32KB flash memory (for user programs and bootloader) and 1KB EEPROM(storing calibration values) running at 8Hz. Kilobots communicate with each other within a range of approximately 10 cm at rates upto 30kb/s using infrared (IR) LED transmitter and infrared photo-diode receiver [64]. Kilobots estimate their distance from its neighbors based on the intensity of received IR light. Since the distance is estimated as a function of intensity, the effective r_{comm} of a Kilobot may be less than 10 cm depending on the surface on which kilobots are being used. Hence, for effective functioning of Kilobots, it is important that they are operated on reflective surfaces such as glass or a whiteboard. The Kilobots also house a ambient light sensor which can sense the level of ambient light shining on the robot. The small size and low cost of Kilobots make them an ideal platform to deploy large number of agents in a swarm hence to make the system scalable an Overhead Controller (OHC) is used. The OHC is an infrared transmitter capable of switching the Kilobots ON/OFF and upload programs to large number of kilobots simultaneously in a relatively short time. The kilobots are charged by grounding the top charging tab and providing 6V to any of the legs of the Kilobot. The user program running on the robot is compiled from C or C++ with the standard AVR tool chain, based on avr-gcc.

3.2 Kilobot Duo

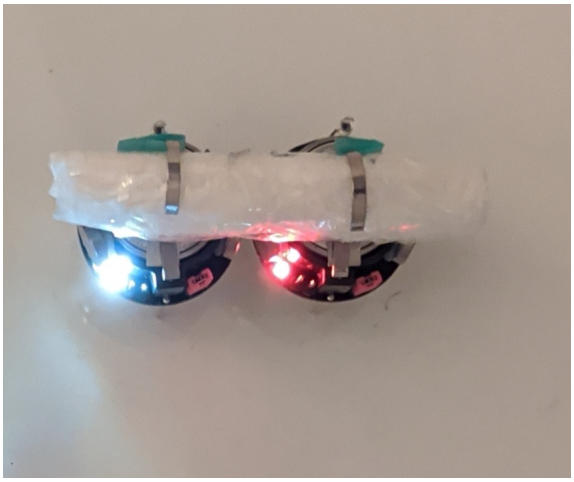


Figure 3.2: A Kilobot duo

A Kilobot duo as shown in Fig 3.2 is a pair of Kilobots physically constrained together along their abscissas. The Kilobots are physically constrained with the help of hollow paper tubes on which the Kilobots are clung with the help of the charging tab. This design of Kilobot provides us with a drive system akin to a differential drive and gives us better control on the motion of a single agent in a Kilobot duo swarm over a normal Kilobot duo swarm. Assuming N number of Kilobots in the swarm,the number of duos in the swarm will equal to half of the number of robots in the swarm i.e., $\frac{N}{2}$. The Kilobots are arranged such that the i^{th} duo has robots with IDs $i-1$ and $\frac{N}{2} \pm (i-1)$, such that IDs $\ngeq$ N. The dynamics of the Kilobot duo can be approximated as a combining two differential drive

3.3 Kilobot Simulator

Before performing hardware experiments it is beneficial to perform them in a virtual environment since in case we encounter unexpected behavior or issues on hardware, a detailed inspection can be carried out. Setting up Kilobots for experiments take a considerable amount of time and effort in addition to the time spent running the algorithm on hardware. Hence it advisable to perform experiments on simulators such as VREP [65], KBSim [66] , Kilombo [67], which are faster than real life experiments. KBSim and VREP require to maintain two seperate versions of codes- one for the simulator model and one that would be complied to HEX code to run on Kilobots. Maintaining two separate versions of code increases the probability of introducing new sources of errors and requires extra time and effort

I use a novel C based Kilobot simulator developed by Dr. Otte at University of Maryland. The simulator implements a physics based model of Kilobots to capture the fundamental and important features of real Kilobots. The simulator was developed with the following aims :

1. Accurate in depicting real-life behavior of Kilobots : The simulator is able to simulate the inter-robot interaction and interaction of the robots with the environment with good accuracy.

2. Code portability : Our simulator enables us to reuse the simulator code on Kilobots and vice versa with minimal changes hence producing portable code. Code portability offers improved speed and easier maintenance of code.

3. Faster : For simulations to be useful they must take less time than the corresponding real-life experiment. A shorter simulation execution cycle results in a shorter debugging and development cycles.

Extensive modifications to the existing Kilobot simulator were made to initialize the swarm as duos such as initialization of robots in pairs. A robot with ID i is initialized at a random location with co-ordinates (x, y) and another robot with ID $\frac{N}{2} + (i - 1)$ is initialized at coordinates (x+2ρ, y) where ρ is the radius of a Kilobot. Further modifications were made to the simulator to model the kinematics and dynamics of Kilobot duos.

Chapter 4: Preliminaries and Problem Statement

4.1 Definitions and Notations

1. The Swarm system : The Swarm system consists of N autonomous, homogeneous robots $R = R_1, R_2, R_3 \ldots . R_N$ each having a radius ρ. Each robot in the swarm has a unique ID $i \in (0, 1, 2, 3, 4, \ldots . N - 1)$

2. The environment : The environment in which the swarm will operate is defined as a set of points in a continuous 2D space $\mathbb{R}^2$

3. Duo : A pair of Kilobots ,(R_i, R_j), physically constrained together as shown in 3.2

4. Duo Swarm : A Duo swarm, denoted by D, is a swarm of $\frac{N}{2}$ duos such that $D = D_1, D_2, D_3 \ldots . D_{\frac{N}{2}}$

5. Sleep State : A Kilobot is in a sleep state if it does not move after a Circle 1 has been has been formed

6. Root Kilobot : A root Kilobot is the center Kilobot around which circles are formed.

7. Circle Kilobot/Duo : Any Kilobot which is not a root Kilobot/Duo is a Circle Kilobot/Duo.

8. Seeking State : Seeking state is defined in which a robot is moving either randomly or based on its distance from other stopped robots.

9. Non-Seeking State : A robot is in a non-seeking state if it is stopped on C_1

10. Ring Number(N) State : A robot in the Ring Number State determines the circle number of which the robot is a member. Whenever the formation of new circle c_n is initialized, all the robots in the c_{n-1} change their state from non-seeking to Ring_{n-1}

11. Counting State : A robot is in counting state if it is stopped on any C_i (except C_1) and keeping track of the number of its neighbors.

12. Random Motion : If a robot has not heard from a stopped robot (either Seeking or Counting or the root duo) then it moves forward for 1 second and takes a right turn every 30 seconds.

13. N_r : Number of Kilobots

14. N_d : Number of Kilobot Duos

15. r_i : Radius of i^{th} circle to be formed

16. r_{comm} : Communication radius of a Kilobot

4.2 Problem Statement

In this section we formally define the problems that are investigated in this thesis.

Problem 1 : Form a circle such that $\mathrm{r}_1 < \mathrm{r}_{comm}$ with the following swarm configurations

- A swarm consisting of N autonomous, homogeneous robots $R = R_1, R_2 \ldots . R_N$ in $\mathbb{R}^2$
- For N autonomous, homogeneous robots $R = R_1, R_2, R_3 \ldots . R_N$ in $\mathbb{R}^2$, a Duo swarm having $\frac{N}{2}$ duos such that $\forall D_i \in D$ exists $[R]^2$ where $[R]^2$ is set of two-element subsets of R;

Problem 2 : Form n concentric circles such that $\mathrm{r}_1 < \mathrm{r}_{comm}$ and $\mathrm{r}_n - \mathrm{r}_{n-1} = r_1$ for the same two swarm configurations as stated above.

Chapter 5: Methodology

5.1 Algorithm Description

We present an algorithm that converges the swarm of duos into n concentric circles $C_i = c_1, c_2, c_3 \ldots\ldots\ldots c_n$ having radius $r_i = r_1, r_2, r_3 \ldots . r_n$ where i represents the i^{th} circle. The circles are formed around a root duo D_0 having a communication radius r_{comm} such that $r_n > r_{comm}$ and $r_1 < r_{comm}$ i.e., the radius of the first circle will be smaller than the communication radius of the root duo, and the radius of the last concentric circle will be greater than the communication radius of the root duo. We assume, given a swarm of N homogeneous robots with limited sensing and communication capabilities, paired into $\frac{N}{2}$ duos based on their uniquely assigned IDs are placed arbitrarily in a two-dimensional plane R^2. Limited sensing means that the robots can only exchange their IDs and distance from each other based on infrared light intensity. The robots do not have orientation information or their positions in a global coordinate system.

Since the Kilobots do not have a large communication radius or position or orientation information, methods such as trilateration and other beacon-positioning techniques cannot be used to implement a global coordinate system in a large workspace. Hence, we take advantage of the resulting overlapping circles depicting

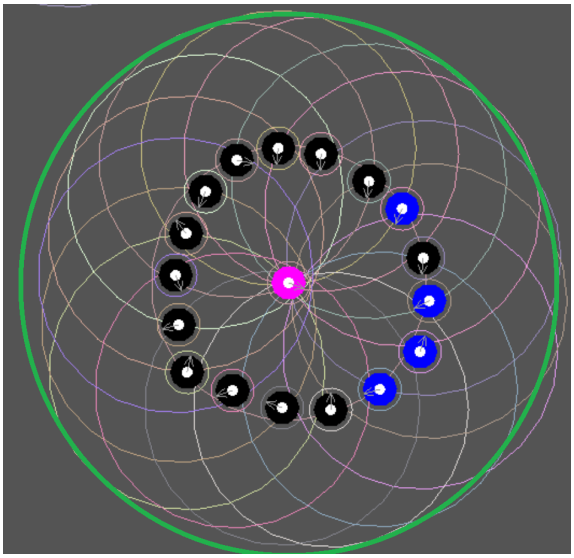


Figure 5.1: Circles depicting the range of communication of the 1^{st} circle

the communication range of a robot, as shown in Fig 5.1 , to form concentric circles. The central idea is that all neighbors of a robot in the previous ring should agree on the position of the robot in the next ring such that the distance of a robot from the previous ring neighbors is within the desired values i.e the 2^{nd} circle will be formed on the green circle which outlines its circumference in Fig 5.1. All the robots except the root duo run the same code. It should be noted that the algorithm does not have a termination point. One can argue that when the user observes that the Kilobots have converged to acceptable pattern, the user may stop Kilobots by broadcasting a message from the Overhead Controller (OHC) to pause the execution of the program.

The algorithm uses a "root duo" to seed the pattern's formation at a desired location. The duos start at a random location in the workspace in the Seeking state and initially do not communicate with each other.

5.1.1 Circle Formation

1. A duo moves randomly as shown in Fig 5.2 till it is within the r_{comm} of the root duo and estimates its distance from the robot duo

2. The duo then positions itself at a desirable distance from the root duo on the circumference of c_1. The root duo dynamically tracks the number of robots that are stopped on the circumference of c_1

3. The duo on c_1 goes to the Non-Seeking state and starts to communicate with other duos within r_{comm}

4. Any duo within r_{comm} of the stopped duo and in the Seeking state will move towards the stopped Non-Seeking duo till the Seeking duo stops on c_1 and changes its state from Seeking to Non-Seeking and starts communicating with other Seeking robots. This results in a virtual attraction force acting on Seeking duos within r_{comm} of Non-Seeking duos and pulls the Seeking robots towards c_1's circumference. This behaviour can be seen in Fig 5.3

5. Once the root duo determines, there are α number of duos on c_1, it sends a message to the Non-Seeking duos that circle 1 has been formed, and the system needs to start forming the next circle.

6. All Non-Seeking robots on c_1 change their state from Non-Seeking to ring 1 and broadcasts this information to all other Seeking robots

5.1.2 Concentric Circles Formation

1. Seeking robots within r_{comm} of c_1 robots. position themselves at a distance of d_{r1} from c_1 robots such that $r_2 = r_1 + d_{r1}$ where r_2 is the radius of the c_2 circle.

2. After positioning themselves on c_2 the duo changes their state to the Counting state where it keeps track of the number of its neighbors, β, on c_2 and broadcasts the information to other robots within r_{comm}

3. Once β reaches a desired value, the duo changes its state from Counting to Ring 2, stating that it is a part of the second circle and tells the other Seeking duos within its r_{comm} that the next circle, i.e., c_3 can start forming around it and its neighbors on c_2

4. Similarly, the duos on c_3 after having enough neighbors can initiate the process of formation of the next circle and so on.

The variables α and β represent the number of neighbors a robot has and affect the pattern formation process. For large value of a b the circle formed in the current layer and next layer will have a better degree of "perfect-ness" while trading off the time required to form the current layer. Smaller values of a and b lead to a "less-perfect" circle but take less time to form the current layer. The values of a and b are found through performing experiments.

The red robot is the root Kilobot and the green and black robots while they are moving, are in the seeking state. Once they stop, they go to the non-seeking

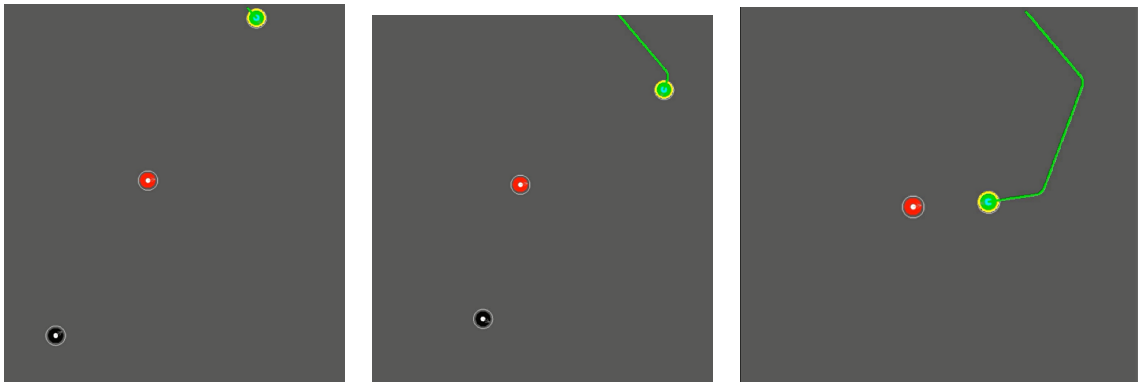


Figure 5.2: Random Walk

state. In Fig 5.2,the black and green robot perform random walk and the green robot stops when it is at a desired distance from the root Kilobot.

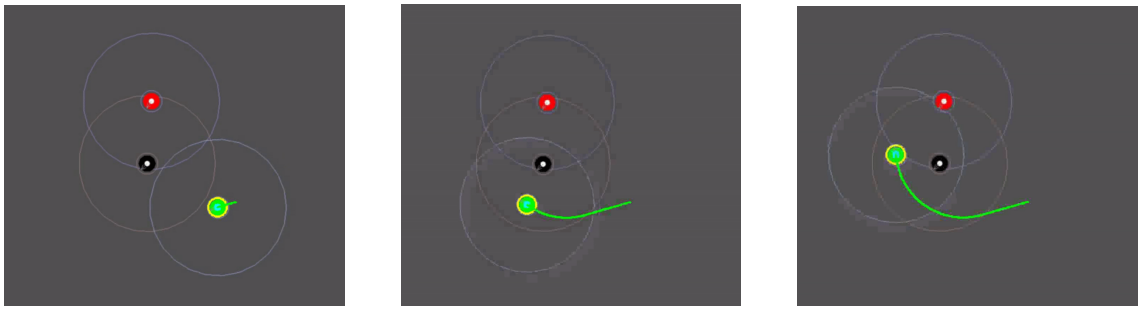


Figure 5.3: Virtual Attraction

Fig 5.3 shows virtual attraction, where a seeking robot (shown in green) within the communication range of a non-seeking robot (shown in black) with proper orientation moves towards the non-seeking robot in an attempt to position itself at a desired distance from the root Kilobot (shown in red)

The algorithm also implements collision avoidance so the if messages are lost or Kilobots are pushed by other Kilobots further towards the root Kilobot, the pushed Kilobot attempts to avoid colliding with the root Kilobot as shown in Fig 5.4

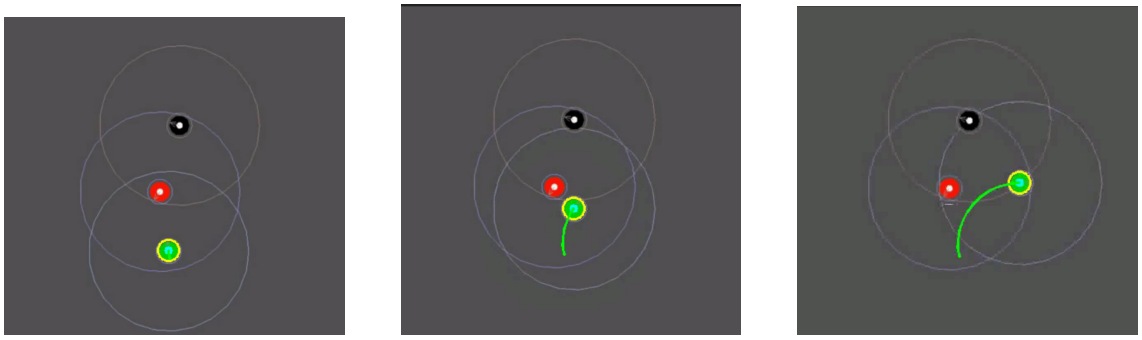


Figure 5.4: Collision Avoidance

Algorithm 1: First circle

Data: Distance, α
Result: Circle 1

```
1  Robot_state ← SeekingRing 1 while true do
2      if α < 12 then
3          if message from D0 and seeking and dist=r1 then
4              stop;
5              Robot_state←Non-Seeking        /* previous state was Seeking */
6          else if message from D0 and seeking and dist< r1 then
7              Move away;
8          else if message from Non-Seeking and seeking then
                   /* robot attempts to move towards communicating robot */
9              Move forward fpr 1 second ;
10             Turn right by x degrees ;
11         else
12             Random Motion ;
13     else if α >= 12 then
14         Robot_state ← Seeking              /* previous state was Non-Seeking */
15         Broadcast message that circle 1 has been formed;
16         if message from D0 and seeking and dist=< r1 then
17             stop
18     else
19         Random Motion
```

Algorithm 2: Concentric Circles

Data: Distance,recieved_β

/* Distance from communicating robot and the no of neighbors as seen by the communicating robot */

Result: Circle 2 and further

```
1  while true do
2      if recieved_β > β then
3          β =recieved_β
4      if β <= 20 then
5          if message from c1 and seeking and dist=r2 − r1 then
6              stop;
7              seeking goes to counting state;
8              broadcast β  /* send the number of neighhbors counted (β) to its
                neighbors */
9          else if message from c1 and seeking and dist< r2 − r1 then
10             Move away ;
11         else
12             Random Motion ;
13     else if β > 20 then
14         from counting state to c2;
15         broadcast β      /* send the number of neighhbors counted (β) to its
            neighbors */
16         if message from c2 and seeking and dist=r3 − r2 then
17             stop;
18             seeking goes to counting state;
19             Broadcast γ                /* for formation of the next circle */
20         else if message from c2 and seeking and dist< r3 − r2 then
21             Move away ;
22         else
23             Random Motion ;
24     else
25         Random Motion ;
```

Algorithm 3: Random Motion

Data: this text
Result: Random motion

```
1 initialization;
2 while true do
3     if time since last message>30 and seeking then
4         move straight;
5         if current time-last turned>15 then
6             last turned=current time;
7             turn right
```

Chapter 6: Simulation and Hardware Tests

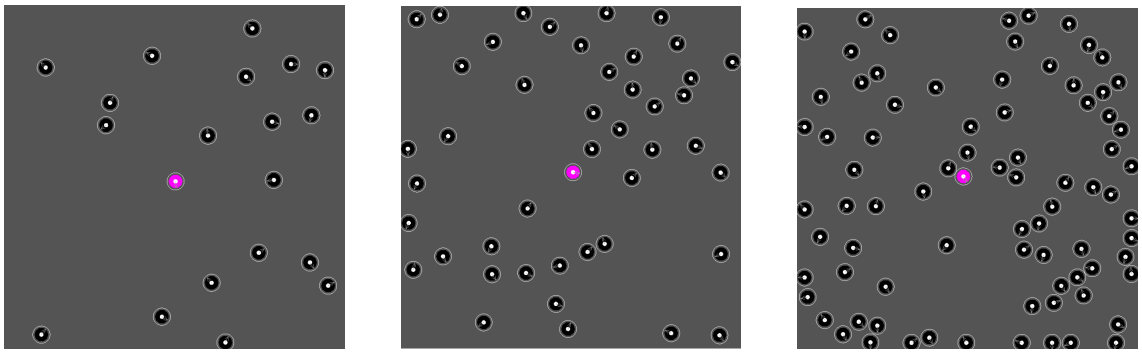
6.1 Design of Experiments and Experimental Setup

To evaluate the proposed algorithm on individual Kilobots and Kilobot Duos, experiments were performed first on the simulator followed by Kilobot hardware. The simulation was performed on maximum of 80 individual Kilobots robots and 23 Kilobot duos, initialized at random positions as shown in Fig 6.1. The root robot is manually dragged and placed at a desired location for the circle formation. The robots/duos have no information about the environment or the position of the root robot/duo. To obtain a uniform pattern two assumptions were made; since the distance between Kilobots is a function of the intensity of infrared light received, the distance measurement depends upon various factors such as the surface on which Kilobots are being used, number of neighbors and lighting in the environment. Hence, to obtain a uniform and stable pattern, instead of choosing a singular value for the desired radius, a range of radius is chosen to be a desired radius. Second, the distance between any 2 circles is the same as the distance between the root Kilobot and the Circle 1 Kilobots.

For hardware experiments, we used a whiteboard as our table top with the overhead controller at an appropriate position. Fig 6.2 shows the test-bed for the

Table 6.1: Number of Robots for different number of circles in pattern

No of Circles	Number of Individual Kilobots	Number of Kilobot Duos
1	20 in simulation and 50 in real-word	10 in Simulation and 15 in real world
2	45 in simulation and 60 in real world	23 in simulation
3	80 in simulation	–



(a) N_r=20 (b) N_r=45 (c) N_r=80

Figure 6.1: Simulator initialized with different number of Kilobots

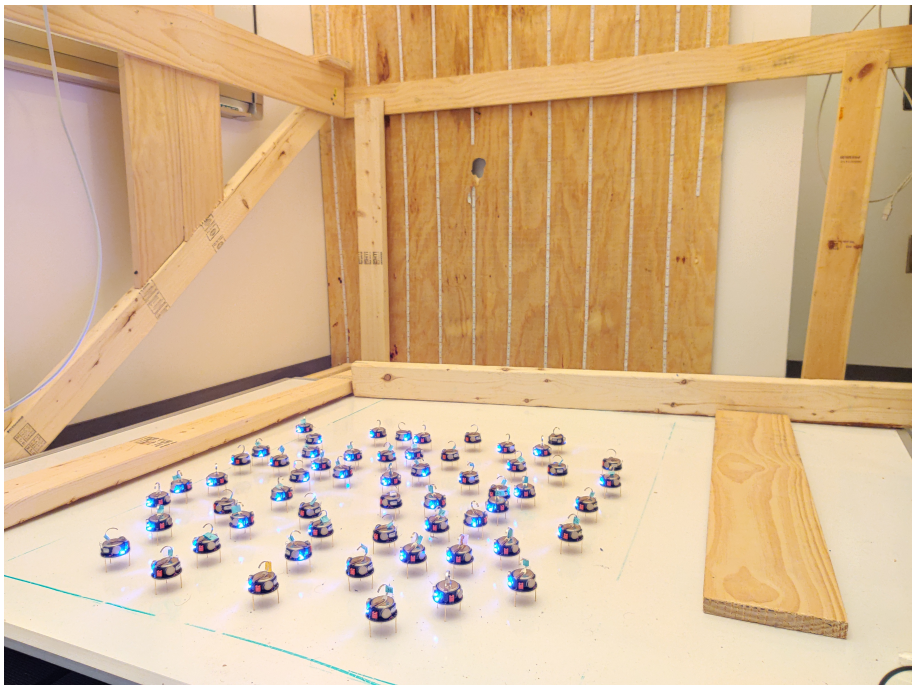


Figure 6.2: Experimental Test-bed

hardware experiments. While performing experiments it was found that using a Halogen lamp light on the test-bed allowed for proper communication between Kilobots and the OHC. The Kilobots cannot sense the metallic edges of the whiteboard where their legs get stuck and Kilobots are unable to move. In that scenario, Kilobots are rotated by hand allowing them to continue moving randomly.

Table 6.2: Message Structure

Byte	Root Message	Circle Message
1	Type of robot	Type of robot
2	α	Ring Number
3	-	ID
4	-	Circle 1 formation flag
5	-	Bit-array
6	-	Bit-array
7	-	Bit-array
8	-	β

Table 6.2 shows the message structure used by robots to communicate with each other. The robots use the 1st byte to indicate the kind of the robot i.e. if it is the root robot or a circle robot. The 2nd byte is used by the root robot to inform the other robots about the number of its neighbors whereas the circle robots used it to indicate their Ring Number. The 3rd byte is used by circle robots to send its ID to other robots so the receiver can count the number of its neighbors. The 4th byte is used as a flag to indicate whether the first circle has been formed or not. Bytes 5,6 and 7 are used to send its bit-array to other robots so the receiving robots may perform an OR operation with its bit-array to count the total number of neighbors the sender and receiver have. The robots use byte 8 to send the number

of its neighbors to other robots.

6.2 Simulation

A large number of simulations were performed to tune various parameters to produce good concentric circles. Simulations were performed on both individual Kilobots and Kilobot Duos while varying different metrics such as the radius of the circle to be formed and the number of circles in the concentric pattern.

Table 6.3: Design of Experiments

Test Case	System Type	Number of Circles	Radius(mm)	No. of trials
1	Individual	1	90-100	5
2	Individual	1	70-80	5
3	Individual	1	50-60	5
4	Individual	2	90-100	5
5	Individual	2	70-80	5
6	Individual	2	50-60	5
7	Individual	3	90-100	5
8	Individual	3	70-80	5
9	Individual	3	60-60	5
10	Duo	1	90-100	5
11	Duo	1	70-80	5
12	Duo	1	50-60	5
13	Duo	2	90-100	5
14	Duo	2	70-80	5
15	Duo	2	50-60	5

Table 6.3 lists the experiments performed on the simulator to find the optimal values of the parameters α and β. Table 6.4 lists the values of the various parameters which produced optimal results with respect to the time required to converge to circles and how accurately the pattern could be formed. Five simulations were run on every test case to collect data and compare the performance of the simulation.

Table 6.4: Parameter Values used in simulation

Radius (mm)	α	β
90-100	12	20
70-80	11	20
50-60	10	18

The Root Mean Squared Error (RMSE) of the distance between the center of a robot and the center of the radius range is calculated for r $\in$ [90,100] as shown in Fig 6.3. The red circle denotes the upper limit of the range i.e. 100 mm and green circle dentes the lower limit of the circle range i.e. 90 mm. The distance "x" is measured from the locus of the mid-point of the range i.e. a circle of 95 mm.

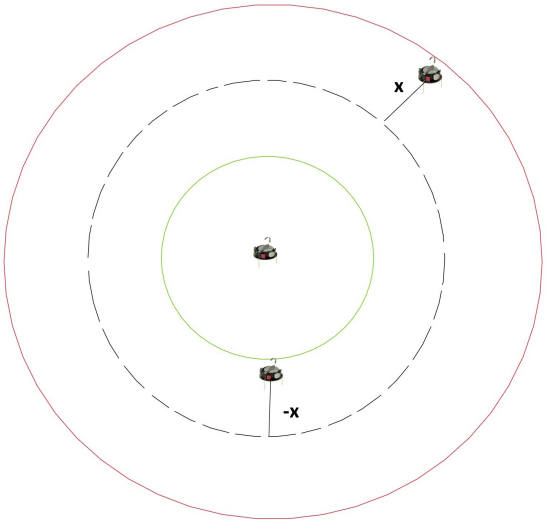


Figure 6.3: Error measurement

From 6.4 we can see that as we increase the value of α the error initially increases significantly due to inter-robot collision. However for alpha, at value 12, there does seem to be a local minima of the RMSE which shows that at 12 neighbors the robots form the most accurate circle.

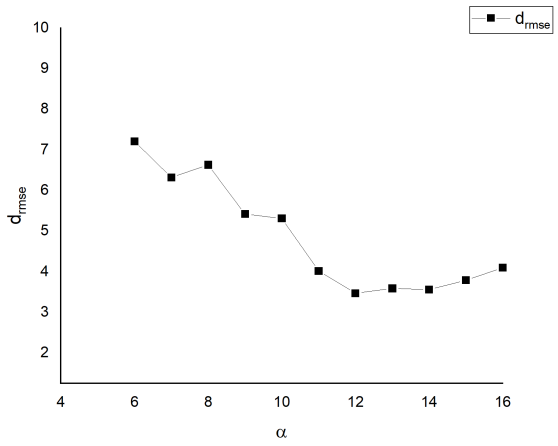


Figure 6.4: Variation of the root mean square error with α

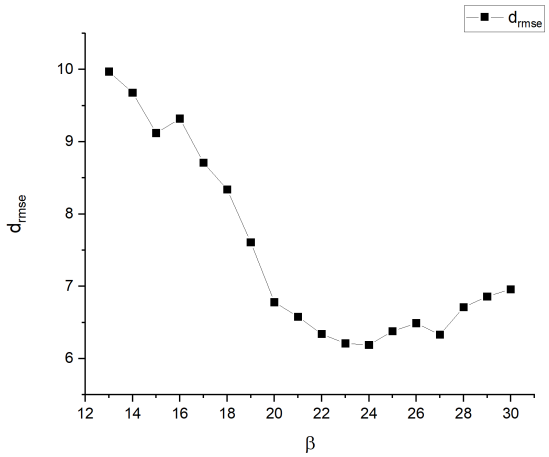


Figure 6.5: Variation of the root mean square error with β

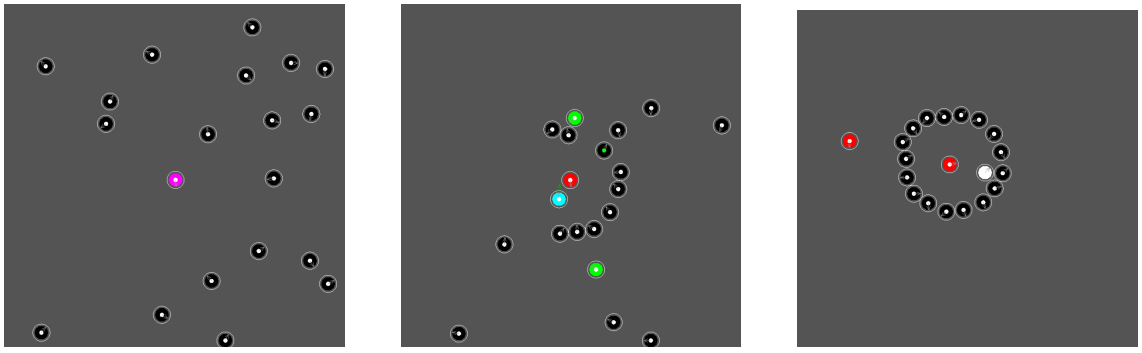
For β from fig 6.5 we can see that initially, the high value of error is due to when some robots are between 2nd circle and circle 1 and they stop before they can follow any rules since they are at a desired distance from any C1 robot; and since RMSE penalizes outliers heavily, we get a large value of error. As we increase the value of beta, the number of robots arrange themselves into a semblance of a circle that the RMSE reaches a minima. As we keep increasing the value of β we see an increase in the error again. This is due to the fact that robots which were supposed to be a part of circle 3 for a smaller value of beta, are now Seeking robots

which causes them to collide with already stopped robots and hence changing their distance from the theoretical estimated distance. We can also see that the RMSE starts to reach a significantly smaller value for β=20. However this is not the local minima as lower of RMSE exist for larger values of β. We still chose β=20 as our parameter since it required shorter amount of time and produced acceptable results.

6.2.1 Individual Kilobots

Simulations were performed for 20, 45 and 80 robots for one, two and three circles concentric respectively. Each robot start outs at a random position and run the same algorithm to converge on the circumference of a desired circle. Fig 6.6 shows the time-lapse of formation of the first circle around the root robot using 20 Kilobots. The root Kilobot is shown in magenta and the circle robots are depicted in black(at initialization) in Fig 6.6a. As the simulation progresses, the circle robots perform random walk or move according to one of the rules as detailed in Section 4.2. The root Kilobot keeps tracks the number of robots and repels away any Kilobot which gets too close to avoid collision and disturb the already formed pattern. The Kilobots being repelled are shown in electric-blue color in Fig 6.6b. When the root Kilobot has reached its desired αvalue, it tells the other robots that the 2_{nd} circle should start forming, shown as red outlier robot in Fig 6.6c. After the circle has been formed, if there exists any Kilobot such that dist $< r_1$ then that Kilobot goes into Sleep State, shown as the white Kilobot in Fig 6.6c, so that it may not collide with other Kilobots and disturb the pattern. From Fig 6.7 we can see that the mean

times required to form circle with radii ≈ 9 cm, 7 cm and 5 cm are 8.54 minutes,8.02 minutes and 6.4 minutes respectively.



(a) $t = 0$ (b) $t = 4min$ (c) $t = 9min$

Figure 6.6: Formation the 1^{st} circle with 20 Kilobots

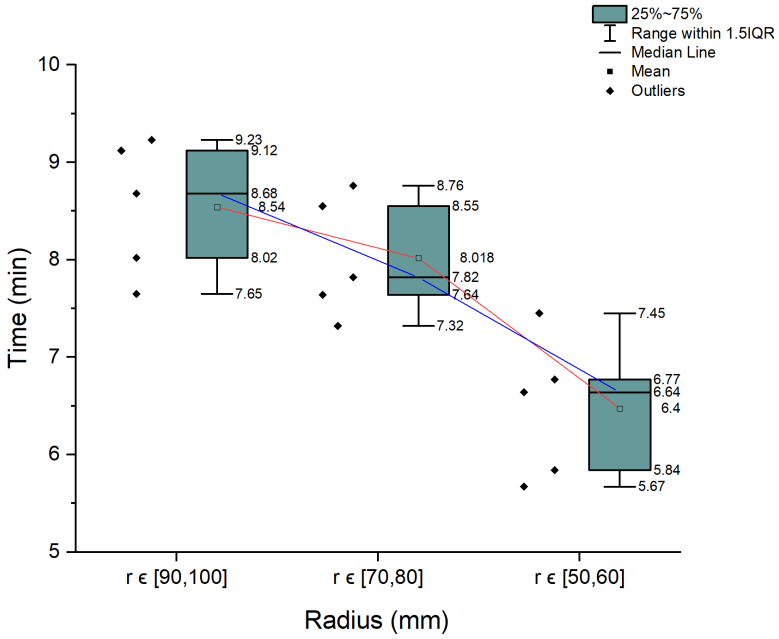
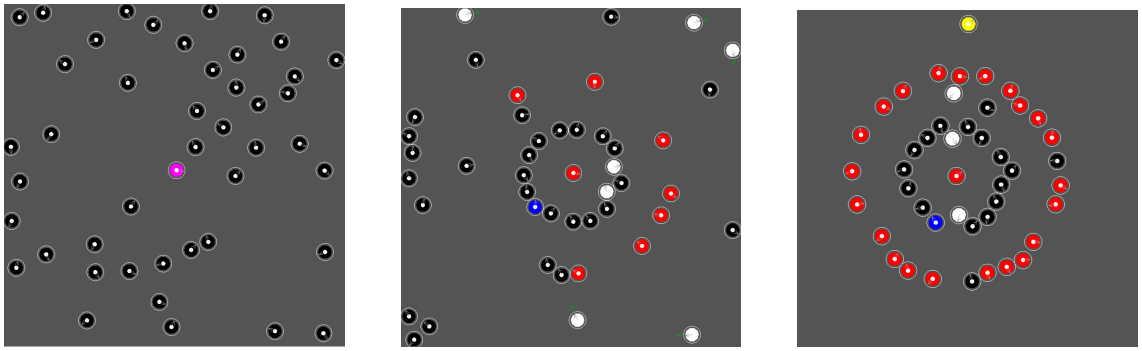


Figure 6.7: Time required to form Circle 1 for various radii

Simulations for the 2 concentric circles is run on 45 Kilobots, also initialized at random locations. After circle 1 has been formed, all the Kilobots in Ring 1 state communicates with other Seeking Kilobots which then move according to rules detailed in Section 4.2. Each robot keeps tracks of the number of its neighbors and

when the desired value if β is reached, it tells the remaining Seeking Kilobots that circle 2 has been formed and the 3_{rd} circle should start forming, shown by the yellow outlier Kilobot in Fig 6.8b.From Fig 6.9 we can see that the mean times required to form circle with radii $\approx$ 9 cm, 7 cm and 5 cm are 16.78 minutes,16.24 minutes and 14.85 minutes respectively.



(a) $t = 0$ (b) $t = 10min$ (c) $t = 17min$

Figure 6.8: Formation of 2 concentric circles with 45 Kilobots

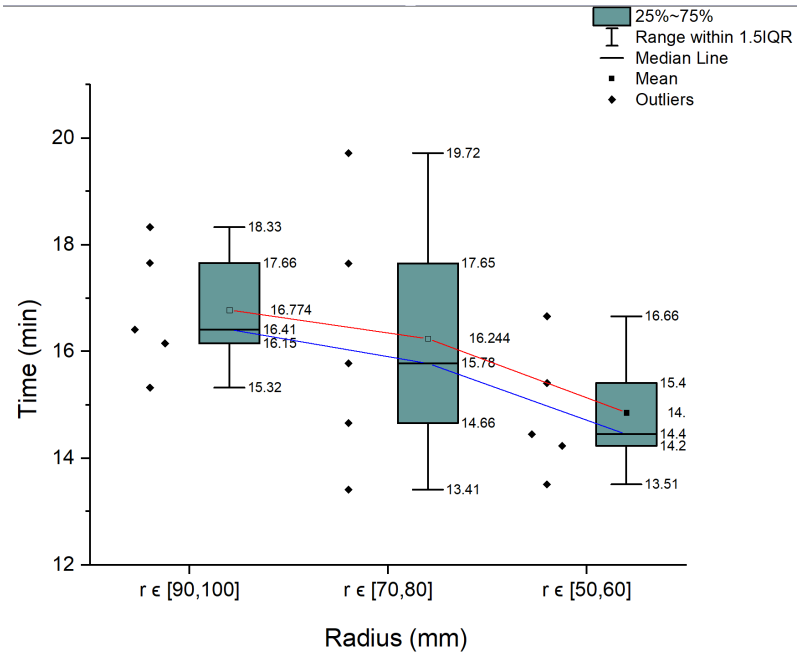


Figure 6.9: Time required to form Circle 2 for various radii

Simulation for 3 concentric circles is run on 80 Kilobots. After formation of

the circle 1 and circle 2 according to rules described in Section 4.2 and as seen above, the remaining Seeking Robots follow the same rules to form circle 3, shown in yellow in Fig 6.11. From Fig 6.9 we can see that the mean times required to form circle with radii ≈ 9 cm, 7 cm and 5 cm are 23.36 minutes,22.10 minutes and 19.8 minutes respectively.

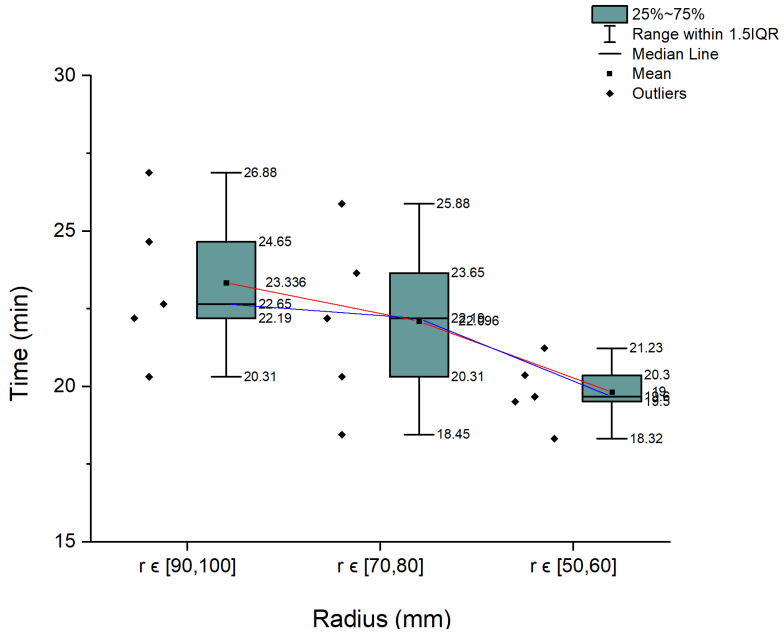
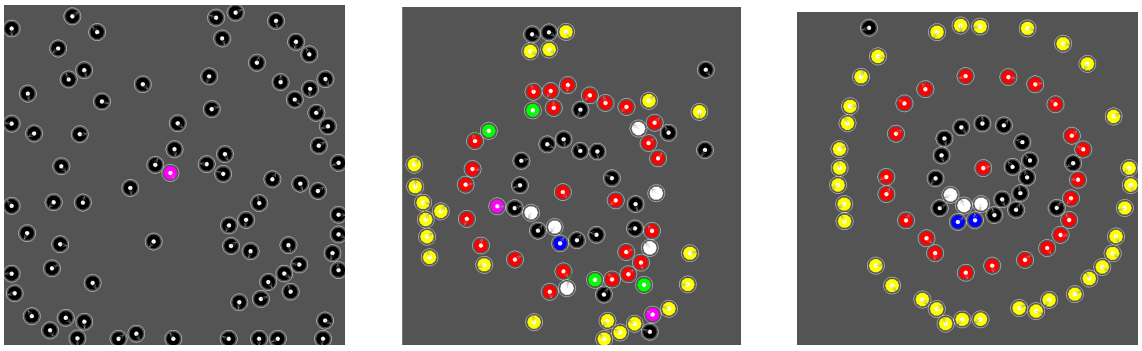


Figure 6.10: Time required to form Circle 3 for various radii

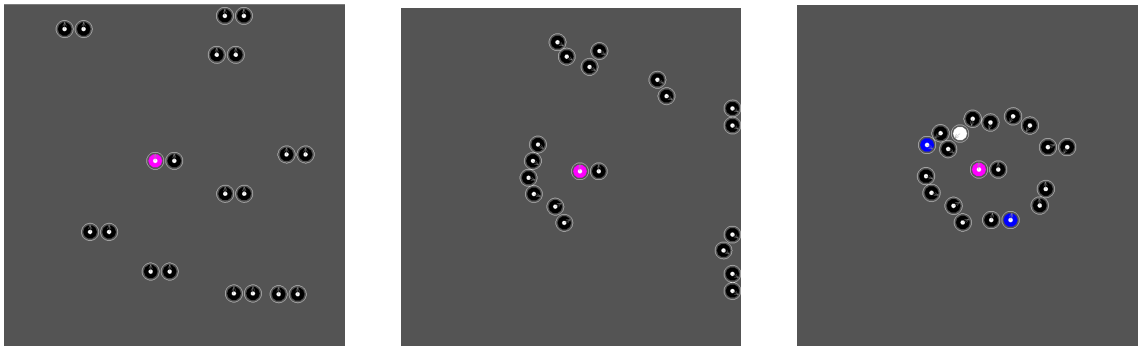


(a) $t = 0$ (b) $t = 15s$ (c) $t = 24min$

Figure 6.11: Formation of 3 concentric circles with 80 Kilobots

6.2.2 Kilobot Duos

Kilobot duos are simulated for forming circle 1 and circle 2 using 10 Kilobot duos and 23 Kilobot duos respectively. Fig 6.12 shows the formation of circle 1. The Kilobots in duos run the same code as individual Kilobot case except for a slight modification; Kilobot with ID i will not receive a message from Kilobot with ID $\frac{N}{2} \pm i$ i.e. from its duo pair. Fig 6.13 shows the mean, median, the maximum and minimum values of time elapsed to make circles of varying radii suing Kilobot Duos. It is interesting to note that the time elapsed to form a circle of radius r $\in$ [70,80] is less than time elapsed in forming a circle of radius $\in$ [90,100]. This may be attributed to the fewer number of Kilobots Duos that are required to be on the circumference of the desired circle.



(a) $t = 0$ (b) $t = 4min$ (c) $t = 8mins$

Figure 6.12: Formation of circle 1 using 10 Kilobot Duos

Fig 6.14 shows the formation of circle 2 using 23 Kilobot Duos. During simulation it was observed that for approximately $N_{duo} > 25$ the algorithm is non-admissible. Hence, to successfully form two concentric circles, the values of α and β were changed to a smaller value. The Kilobot duos were still able to converge into

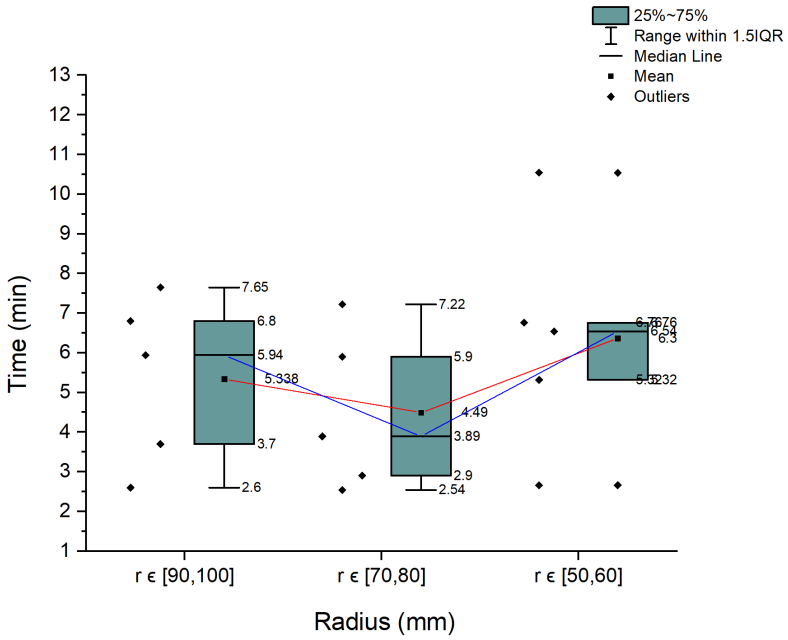
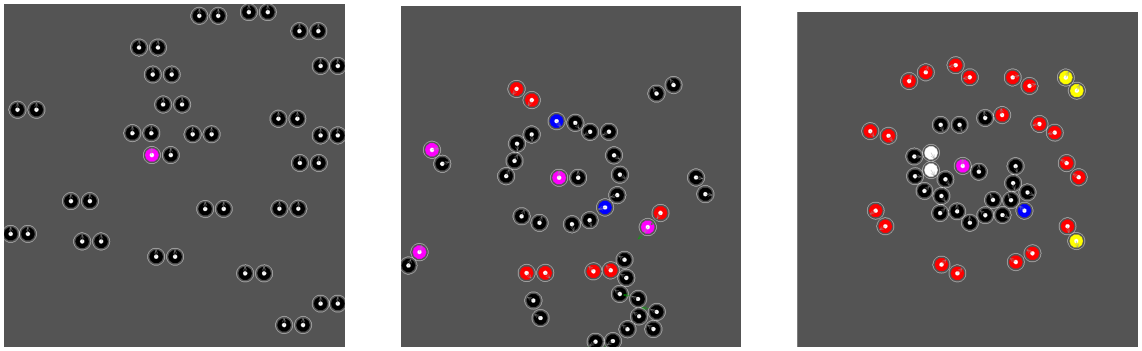


Figure 6.13: Time required to form Circle 1 using Kilobot Duos for various radii

two concentric circle although less accurately.



(a) $t = 0$ (b) $t = 9min$ (c) $t = 13min$

Figure 6.14: Formation of circle 2 using 23 Kilobot Duos

Fig 6.16 shows the time required by individual Kilobots and Kilobot Duos to converge into approximate circles and concentric circles. Interestingly, when $N_d = 10$, the Kilobot duo swarm converges faster into a circle as compared to individual Kilobot case in spite having the same number of Kilobots in the system.

Fig 6.17 6.18 6.19 show the time required for formation of three concentric

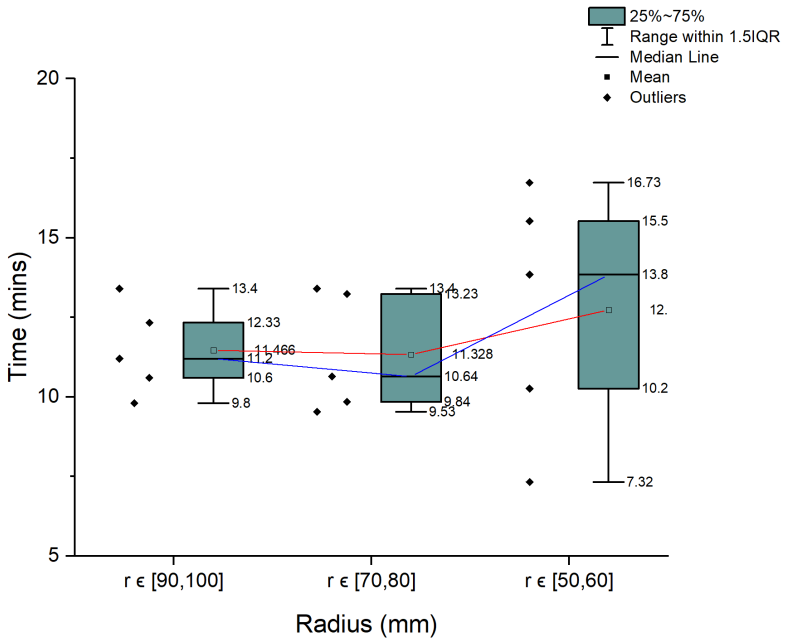


Figure 6.15: Time required to form Circle 2 using Kilobot Duos for various radii

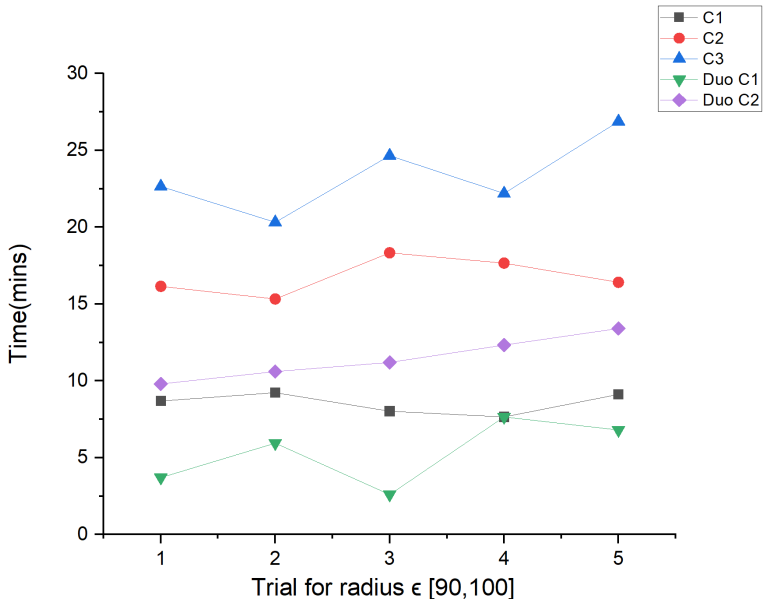


Figure 6.16: Time required by individual Kilobots and Kilobot Duos to form three circles (C1, C2, C3) and two circles respectively in 5 trials

circles having the same radius. As expected, the time required to form circles increases with the increase in number of Kilobots in the swarm. Hence Fig 6.17 6.18 6.19 investigate the scalability of the proposed algorithm for individual Kilobots. For

small number of Kilobots in the swarm, the Kilobots converge within approximately 10 minutes of initialization of the swarm. However as the number of Kilobots in the system increases the time required for the Kilobots to converge into different circles also increases.

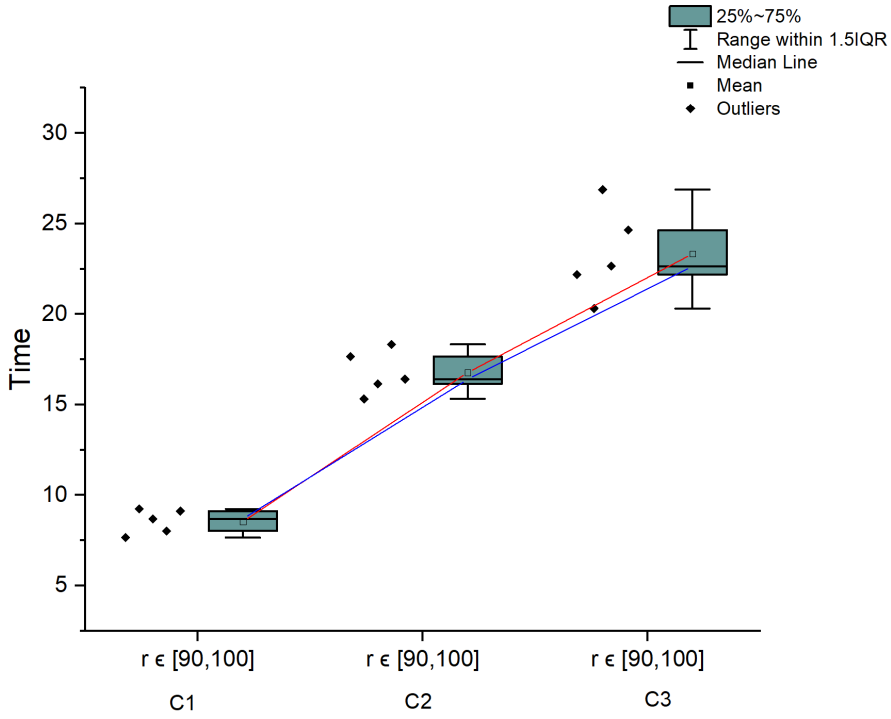


Figure 6.17: Circle 1,Circle 2,Circle 3 formation with radius ∈[90,100]

Similarly, for Kilobot Duo,s as the number of Kilobot Duos in the system increases, the time required for the Kilobots Duos to converge into different circles also increases.

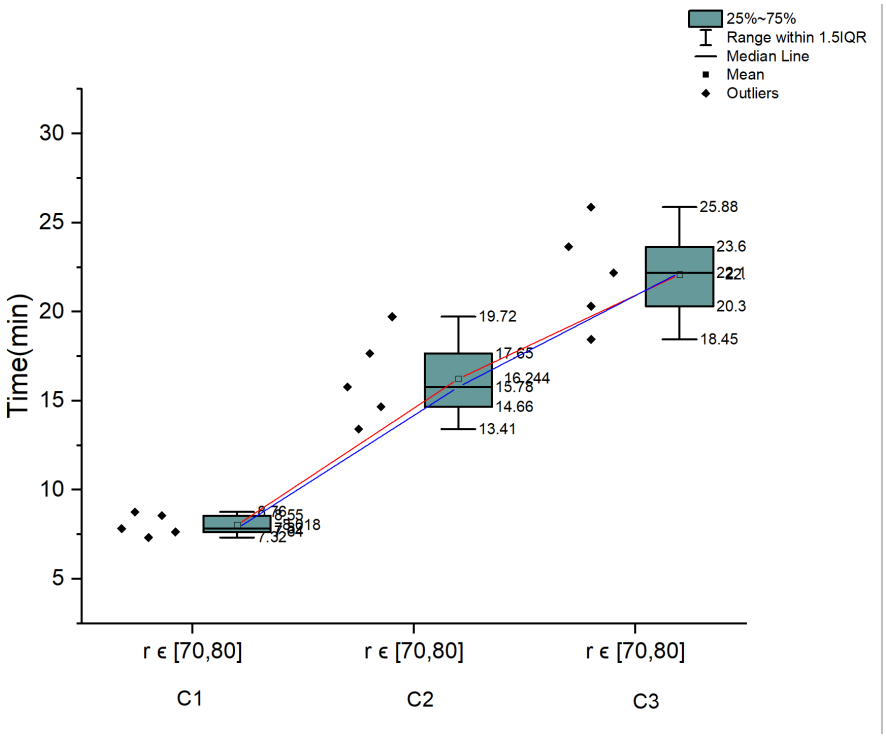


Figure 6.18: Circle 1,Circle 2,Circle 3 formation with radius ∈[70,80]

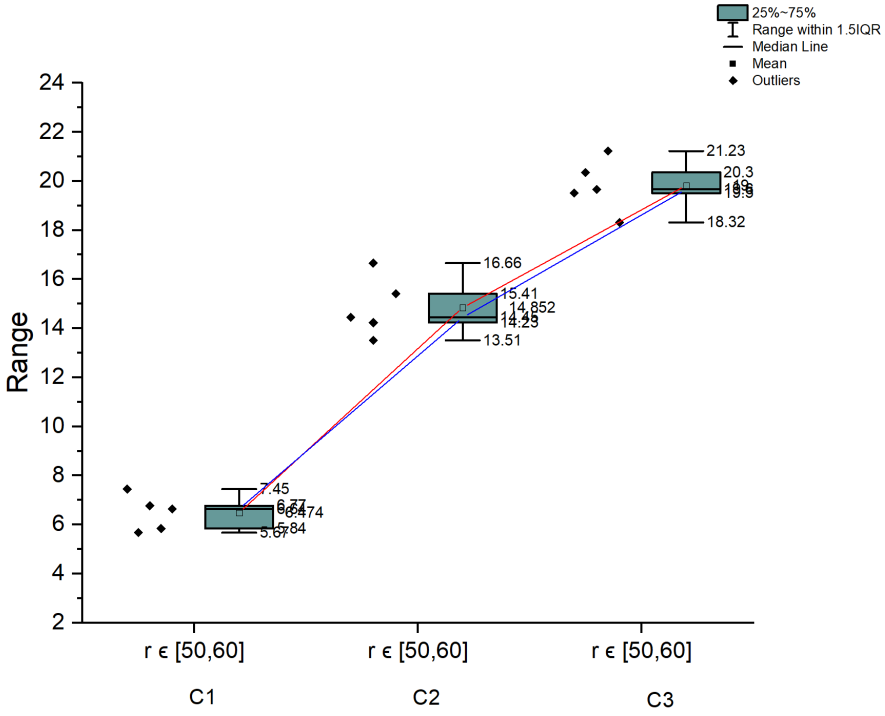


Figure 6.19: Circle 1,Circle 2,Circle 3 formation with radius ∈[50,60]

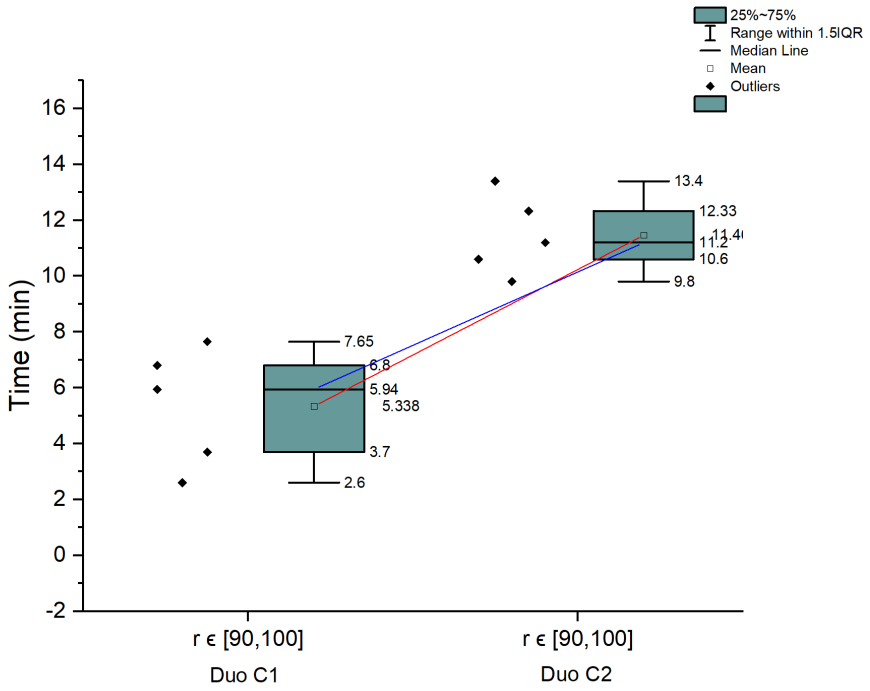


Figure 6.20: Circle 1 and Circle 2 formation using Kilobot Duo with radius ∈[90,100]

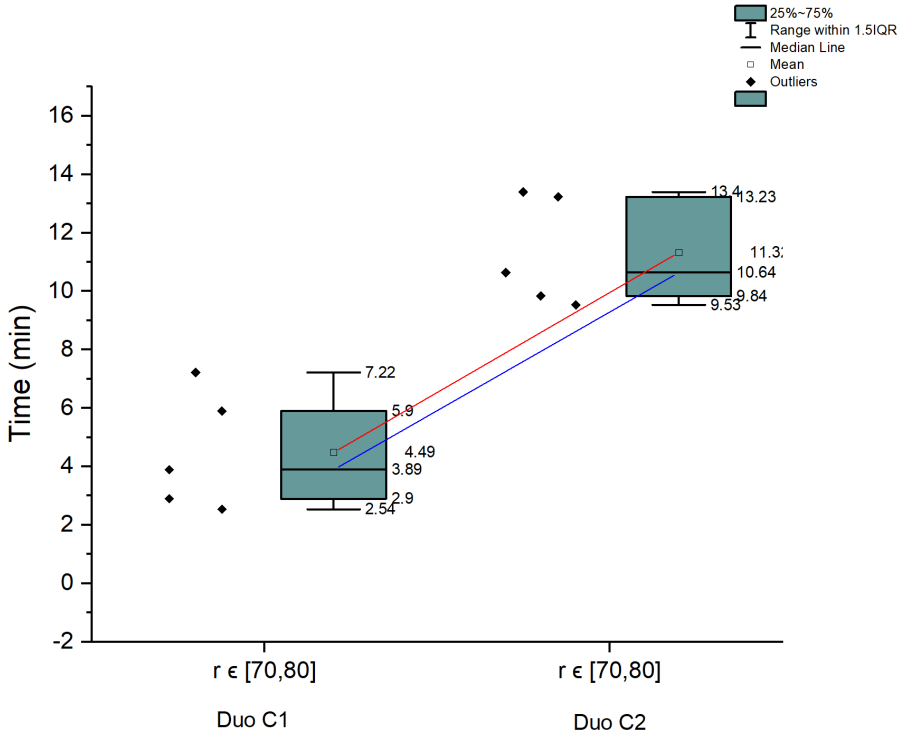


Figure 6.21: Circle 1 and Circle 2 formation using Kilobot Duo with with radius ∈[70,80]

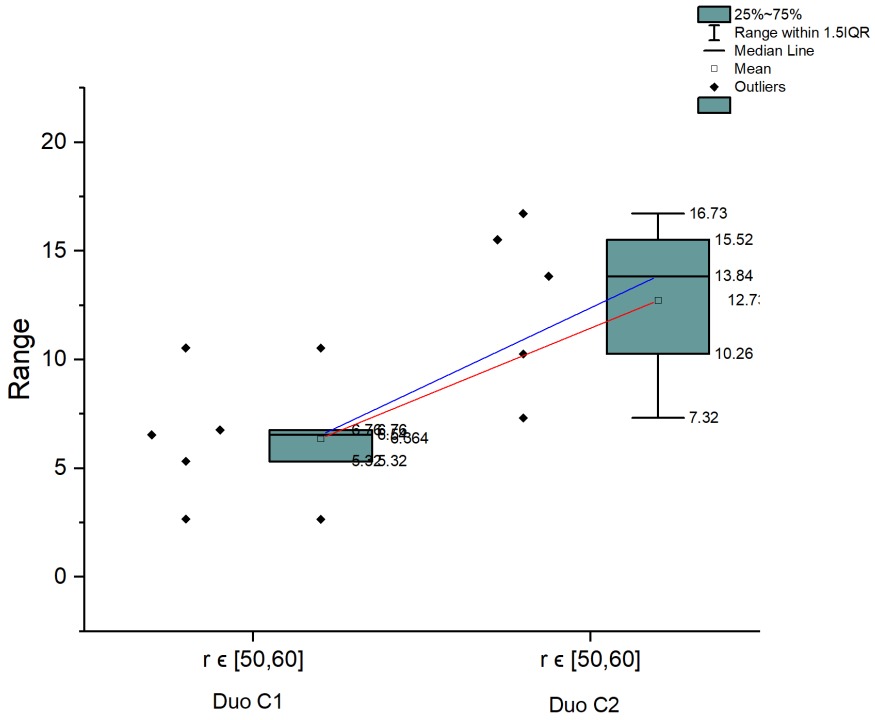


Figure 6.22: Circle 1 and Circle 2 formation using Kilobot Duo with radius ∈[50,60]

6.3 Hardware Results

In this section, results of experiments performed on Kilobot hardware are presented. Although Kilobots may not be useful in real-life situations, they provide a low-cost experimental test bed for proof-of-concept. Fig 6.2 shows the experimental setup. All the Kilobots are placed randomly on the white-board and the root Kilobot/Duo is placed at approximately at the center of the test-bed so appropriate space can be provided for the robots to converge.

Experiments for formation of Circle 1 was performed on 50 Kilobots. Five trials were performed to collect data for further analysis. Fig 6.23 shows the final pattern generated for the first circle.

Fig 6.24 shows the final formation for Circle 2 performed on 60 Kilobots.

Figure 6.23: Circle 1 formation with 50 Kilobots

Figure 6.24: Circle 2 formation with 60 Kilobots

From the results we can see that in-spite the presence of noise, a good approximation of circles is achieved. Kilobots are colliding as expected due to the small communication range of the Kilobots however the proposed algorithms reduced the frequency of collisions in the single Kilobot case. Furthermore, we can also see that due to the virtual attractive and repulsive forces, the swarm is able to re-converge if the pattern is disturbed.

Figure 6.25: Circle 1 formation with 15 Kilobot Duos

In the experiment with real Kilobot Duos, 5 trials were conducted with 15 Kilobot Duos. From the results in Fig 6.25 we can see that the Kilobot Duos were successfully able to form a good approximation of a circle considering that Kilobots have no position data, no bearing information, no agree-upon common axis, no camera and no GPS.

During experimentation it was also found that for approximately $N_d > 15$,

Figure 6.26: Clustering of Kilobot Duos for $N_d = 20$

the Kilobot duos began forming clusters as seen in Fig. 6.26 This may be due to the small communication range of Kilobots ($r_{comm} \approx 3$ Kilobot diameters) and the fact that the Kilobot Duos are physically attached together using a paper cylinder whose length is greater than 66 mm which jam into each other if robots are too close. It was also observed during the experiments that the motion of Kilobot Duos is much more harder to control since to achieve perfect straight motion, both the Kilobots in the duo should be calibrated such that both robots have the same speed while moving forward. If either of the Kilobot has a different speed, the duo instead of moving forward, begins to turn which increases the probability of colliding into another robots rather than avoiding it. Furthermore, calibrating a pair of Kilobots to achieve perfect motion is time and effort consuming.

Chapter 7: Discussion of Results

7.1 Performance with different number of circles

The algorithm produces good results in simulation for all three circles as seen in Figs 6.6, 6.8 and 6.11. As seen from figure 6.17, 6.18, 6.19 we can see that the mean and median time required for for the individual Kilobot swarm to converge into circles increases almost linearly as the number of robots in the swarm increases. It is also worth noting that the time between forming circle 1 and circle 2 was almost always greater than time between forming circle 2 and circle 3. This may be explained by the fact while forming circle 1, the Seeking robots only have single point source of communication i.e. the root Kilobot/Duo however the Kilobots/Duos forming circle 2 and circle 3 have multiple Kilobots to communicate with i.e. the Kilobots on the circumference of circle 1. From Fig 6.20 we can see that the formation of circle 2 using Kilobot Duos that variation of time required to form circle 2 is less than to form circle 1, which again can be explained by the availability of single and multiple points of communication. However, for smaller radii (Fig 6.21 and Fig 6.22)the variation of time is more due to the collision of Kilobot duos with the root duo.

7.2 Performance of Individual Kilobots and Kilobot Duos

From 7.1 we can see that for small swarm size, the Kilobot Duos perform temporally better than its individual counterpart while trading off on the accuracy of the pattern formed(see fig 6.23 vs 6.25, 6.6c vs 6.12c, 6.8c vs 6.14c). However, for larger swarm sizes, the Kilobot Duos do not converge and and individual Kilobots perform better than Kilobot Duos. We can see that in both cases that the mean and median time required to form the second circle is almost twice as of the time required to form the first circle.

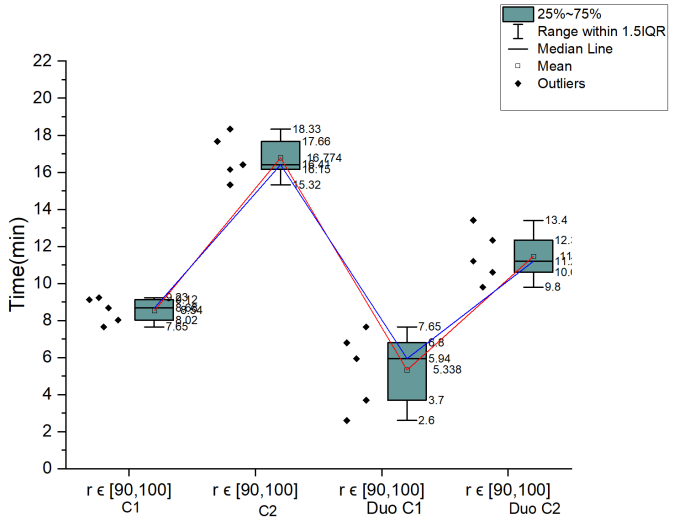


Figure 7.1: Individual Kilobot and Kilobot Duo Performance

7.3 Performance of Individual Kilobots and Kilobot Duos with varying radii

Fig 6.7, 6.9, 6.10, 6.13, 6.15 show the variation of time required to form concentric circles for different radii while keeping the number of robots in the swarm

constant for each circle according to table 6.1. For individual Kilobot system the time required to converge decreases as the the swarm size reduces as seen in Fig 6.7, 6.9 and 6.10 which is the expected behavior since less Kilobots are required to converge to the circumference. For Kilobot Duos, the time required decreases to form when the radius is decreased from [90,100] to [70,80], as seen in 6.13 and 6.15. However, when the radius of the desired circle was made smaller, the time required to converge increased significantly. This behavior may be explained due to the more frequent inter-robot collision. It was observed for r $\in$ [50,60], the Kilobot Duos, being closer to the Root duo, displaced the root Kilobot more frequently while performing a left or right turn since r<4*ρ. Since the root Kilobot Duo initializes the circle formation, its displacement disturbs the already formed pattern, in which case the previously Non-Seeking Kilobots go to the Seeking State, hence increasing the time required to re-converge.

7.4 Clustering

The simulator was developed assuming ideal and friction-less behavior however while performing hardware experiments, it was found that the friction between Kilobot Duos did not scale with the number of robots and caused clustering of Kilobot Duos as seen in 6.26. It would be recommended that any future work involving duos should account for friction.

Chapter 8: Conclusion

In this thesis, pair-based approach for spatial formation for swarm robots have been explored. Most of the existing research assume information about global position, orientation information and direction sensing is available to agents in the swarm which makes them infeasible to be implemented on simple Kilobot robots. I developed a rule based, cross-platform algorithm for Kilobot and Kilobot Duos was developed to form multiple concentric circles around a desired location. A detailed comparison of performance of the algorithm on individual Kilobots and Kilobot Duos was also performed by varying metrics such as number of robots in the swarm and the radius of the circle to be formed.

The algorithm was tested on a novel Kilobot Simulator and real Kilobot robots. The algorithm was evaluated for scalability and completeness by running simulation and experiments on individual Kilobots and Kilobot Duos. The scalability of the algorithm was evaluated by performing simulations on 20, 45 and 80 individual Kilobots in simulation, 15 and 50 individual real Kilobots. For Kilobot Duos, the algorithm was tested on 10 and 23 Kilobot Duos in simulation and 15 real Kilobot Duos. The algorithm produced good results for individual Kilobots for all three concentric circles in simulation and one circle on real Kilobots. It was also observed

that for small swarm sizes with same number of Kilobots, the Kilobot Duo swarm took less time to converge into pattern than its individual counterpart.

Bibliography

[1] Kilobot – K-Team Corporation.

[2] James D. Mclurkin, Thsis Supergsor, Arthur C. Smith, and James D. Mclurkin. Stupid Robot Tricks: A Behavior-Based Distributed Algorithm Library for Programming Swarms of Robots. Technical report, 2004.

[3] Erol Sahin. *Swarm Robotics: From Sources of Inspiration to Domains of Application*, volume 3342. January 2005.

[4] Scott Camazine, Jean-Louis Deneubourg, Nigel R. Franks, James Sneyd, Guy Theraula, and Eric Bonabeau. *Self-Organization in Biological Systems*. Princeton University Press, May 2020.

[5] Mauro Innocente and Paolo Grasso. *Swarms of Autonomous Drones Self-Organised to Fight the Spread of Wildfires*. July 2018.

[6] Ying Tan and Zhong-yang Zheng. Research Advance in Swarm Robotics. *Defence Technology*, 9(1):18–39, March 2013.

[7] O Tanaka. Forming a Circle by Distributed Anonymous Mobile Robots, Bachelor thesis, 2002.

[8] K. Sugihara and I. Suzuki. Distributed motion coordination of multiple mobile robots. In *Proceedings. 5th IEEE International Symposium on Intelligent Control 1990*, pages 138–143 vol.1, September 1990.

[9] Aditya Milind Deshpande, Rumit Kumar, Mohammadreza Radmanesh, Nalini Veerabhadrappa, Manish Kumar, and Ali A. Minai. Self-Organized Circle Formation around an Unknown Target by a Multi-Robot Swarm using a Local Communication Strategy. In *2018 Annual American Control Conference (ACC)*, pages 4409–4413, Milwaukee, WI, June 2018. IEEE.

[10] Xavier Défago and Akihiko Konagaya. Circle formation for oblivious anonymous mobile robots with no common sense of orientation. In *Proceedings of the Second ACM International Workshop on Principles of Mobile Computing*, POMC '02, page 97–104, New York, NY, USA, 2002. Association for Computing Machinery.

[11] Xavier Défago and Samia Souissi. Non-uniform circle formation algorithm for oblivious mobile robots with convergence toward uniformity. *Theoretical Computer Science*, 396(1-3):97–112, May 2008.

[12] Iñaki Navarro and Fernando Matía. An Introduction to Swarm Robotics. https://www.hindawi.com/journals/isrn/2013/608164/, September 2012.

[13] Aufar Zakiev, Tatyana Tsoy, and Evgeni Magid. Swarm Robotics: Remarks on Terminology and Classification. In *Interactive Collaborative Robotics*, Lecture Notes in Computer Science, pages 291–300, Cham, 2018. Springer International Publishing.

[14] Caroline E. Harriott, Adriane E. Seiffert, Sean T. Hayes, and Julie A. Adams. Biologically-Inspired Human-Swarm Interaction Metrics. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 58(1):1471–1475, September 2014.

[15] Guenther Witzany. Biocommunication of Unicellular and Multicellular Organisms. *tripleC: Communication, Capitalism & Critique. Open Access Journal for a Global Sustainable Information Society*, 6:24–53, January 1970.

[16] Swagatam Das, Arijit Biswas, Sambarta Dasgupta, and Ajith Abraham. Bacterial Foraging Optimization Algorithm: Theoretical Foundations, Analysis, and Applications. In Ajith Abraham, Aboul-Ella Hassanien, Patrick Siarry, and Andries Engelbrecht, editors, *Foundations of Computational Intelligence Volume 3: Global Optimization*, Studies in Computational Intelligence, pages 23–55. Springer, Berlin, Heidelberg, 2009.

[17] Duncan E. Jackson and Francis L. W. Ratnieks. Communication in ants. *Current biology: CB*, 16(15):R570–574, August 2006.

[18] S. Goss, Serge Aron, Jean-Louis Deneubourg, and Jacques Pasteels. Self-organized shortcuts in the Argentine Ant. Naturwissenschaften 76: 579-581. *Naturwissenschaften*, 76:579–581, December 1989.

[19] F. L. W. Ratnieks and C. Anderson. Task partitioning in insect societies. *Insectes Sociaux*, 46(2):95–108, May 1999.

[20] Dmitrii Pavlov and A. Kasumyan. Patterns and mechanisms of schooling behavior in fish: A review. *Journal of Ichthyology*, 40:S163–S231, December 2000.

[21] Peter F. Major. Predator-prey interactions in two schooling fishes, Caranx ignobilis and Stolephorus purpureus. *Animal Behaviour*, 26:760–777, August 1978.

[22] Lee A. Dugatkin and Jean-Guy J. Godin. Predator inspection, shoaling and foraging under predation hazard in the Trinidadian guppy,Poecilia reticulata. *Environmental Biology of Fishes*, 34(3):265–276, July 1992.

[23] Cagan Sekercioglu. Conservation Ecology: Area Trumps Mobility in Fragment Bird Extinctions. *Current biology : CB*, 17:R283–6, January 2006.

[24] James R. Usherwood, Marinos Stavrou, John C. Lowe, Kyle Roskilly, and Alan M. Wilson. Flying in a flock comes at a cost in pigeons. *Nature*, 474(7352):494–497, June 2011.

[25] Henrik Mouritsen. Magnetoreception in Birds and Its Use for Long-Distance Migration. In *Sturkie's Avian Physiology: Sixth Edition*, pages 113–133. July 2015.

[26] Giandomenico Spezzano. Editorial: Special issue "swarm robotics". *Applied Sciences*, 9(7), 2019.

[27] Mario Coppola, Jian Guo, Eberhard Gill, and Guido C. H. E. de Croon. Provable self-organizing pattern formation by a swarm of robots with limited knowledge. *Swarm Intelligence*, 13(1):59–94, March 2019.

[28] S. Sarno, M. D'Errico, J. Guo, and E. Gill. Path planning and guidance algorithms for SAR formation reconfiguration: Comparison between centralized and decentralized approaches. *Acta Astronautica*, 167:404–417, February 2020.

[29] Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK ; New York, 2004.

[30] T.J. Koo and S.M. Shahruz. Formation of a group of unmanned aerial vehicles (UAVs). In *Proceedings of the 2001 American Control Conference. (Cat. No.01CH37148)*, pages 69–74 vol.1, Arlington, VA, USA, 2001. IEEE.

[31] M. Egerstedt and Xiaoming Hu. Formation constrained multi-agent control. *IEEE Transactions on Robotics and Automation*, 17(6):947–951, Dec./2001.

[32] C. Belta and V. Kumar. Trajectory design for formations of robots by kinetic energy shaping. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, volume 3, pages 2593–2598 vol.3, May 2002.

[33] Soheil Keshmiri and Shahram Payandeh. A Centralized Framework to Multi-robots Formation Control: Theory and Application. volume 6066, pages 85–98. January 2011.

[34] Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Peter Widmayer. Gathering of asynchronous robots with limited visibility. *Theoretical Computer Science*, 337:147–168, June 2005.

[35] Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Peter Widmayer. Pattern Formation by Autonomous Mobile Robots. In Ali A. Minai and Yaneer Bar-Yam, editors, *Unifying Themes in Complex Systems*, pages 241–248. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[36] Paola Flocchini, Giuseppe Prencipe, Nicola Santoro, and Peter Widmayer. *Hard Tasks for Weak Robots: The Role of Common Knowledge in Pattern Formation by Autonomous Mobile Robots*, volume 1741, pages 93–102. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999.

[37] Xiaoping Yun, Gokhan Alptekin, and Okay Albayrak. Line and circle formation of distributed physical mobile robots. *Journal of Robotic Systems*, 14(2):63–76, 1997.

[38] Ioannis Chatzigiannakis, Michael Markou, and Sotiris Nikoletseas. Distributed circle formation for anonymous oblivious robots. In Celso C. Ribeiro and Simone L. Martins, editors, *Experimental and Efficient Algorithms*, pages 159–174, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[39] H. Ando, Y. Oasa, I. Suzuki, and M. Yamashita. Distributed memoryless point convergence algorithm for mobile robots with limited visibility. *IEEE Transactions on Robotics and Automation*, 15(5):818–828, Oct./1999.

[40] Geunho Lee, Seokhoon Yoon, Nak Young Chong, Henrik Christensen, School of Information Science, Japan Advanced Institute of Science and Technology,, and College of Computing, Georgia Institute of Technology,. A Mobile Sensor Network Forming Concentric Circles Through Local Interaction and Consensus Building. *Journal of Robotics and Mechatronics*, 21(4):469–477, August 2009.

[41] Eman Hasan, Khaled Al-Wahedi, Belal Jumah, Diana W. Dawoud, and Jorge Dias. Circle Formation in Multi-robot Systems with Limited Visibility. In Anibal Ollero, Alberto Sanfeliu, Luis Montano, Nuno Lau, and Carlos Cardeira, editors, *ROBOT 2017: Third Iberian Robotics Conference*, volume 693, pages 323–336. Springer International Publishing, Cham, 2018.

[42] Mehmet Serdar Guzel, Emir Cem Gezer, Vahid Babaei Ajabshir, and Erkan Bostanci. An adaptive pattern formation approach for swarm robots. In *2017 4th International Conference on Electrical and Electronic Engineering (ICEEE)*, pages 194–198, Ankara, Turkey, April 2017. IEEE.

[43] Vito Trianni, Roderich Groß, Thomas H. Labella, Erol Şahin, and Marco Dorigo. Evolving Aggregation Behaviors in a Swarm of Robots. In Wolfgang Banzhaf, Jens Ziegler, Thomas Christaller, Peter Dittrich, and Jan T. Kim, editors, *Advances in Artificial Life*, Lecture Notes in Computer Science, pages 865–874, Berlin, Heidelberg, 2003. Springer.

[44] V Trianni, T H Labella, R Groß, E Şahin, M Dorigo, and J L Deneubourg. Modeling Pattern Formation in a Swarm of Self-assembling Robots, 2002.

[45] Raphael Jeanson, Colette Rivault, Jean-Louis Deneubourg, Stephane Blanco, Richard Fournier, Christian Jost, and Guy Theraulaz. Self-organized aggregation in cockroaches. *Animal Behaviour*, 69(1):169–180, January 2005.

[46] Nikolaus Correll and Alcherio Martinoli. Modeling and designing self-organized aggregation in a swarm of miniature robots. *The International Journal of Robotics Research*, 30(5):615–626, April 2011.

[47] Onur Soysal and Erol Sahin. *A Macroscopic Model for Self-Organized Aggregation in Swarm Robotic Systems*, volume 4433. September 2006.

[48] Emre Ugur, Ali Turgut, and Erol Sahin. *Dispersion of a Swarm of Robots Based on Realistic Wireless Intensity Signals*. December 2007.

[49] A. Howard, M. J. Mataric, and G. S. Sukhatme. An incremental deployment algorithm for mobile robot teams. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 2849–2854 vol.3, September 2002.

[50] Maxim A. Batalin and Gaurav S. Sukhatme. Coverage, Exploration and Deployment by a Mobile Robot and Communication Network. *Telecommunication Systems*, 26(2):181–196, June 2004.

[51] James Mclurkin and Jennifer Smith. Distributed Algorithms for Dispersion in Indoor Environments Using a Swarm of Autonomous Mobile Robots. volume 6, pages 399–408. January 2007.

[52] Yogeswaran Mohan and S. G. Ponnambalam. An extensive review of research in swarm robotics. In *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*, pages 140–145, Coimbatore, India, 2009. IEEE.

[53] K. Tomita, S. Murata, H. Kurokawa, E. Yoshida, and S. Kokaji. Self-assembly and self-repair method for a distributed mechanical system. *IEEE Transactions on Robotics and Automation*, 15(6):1035–1045, December 1999.

[54] Mark Yim, Kimon Roufas, David Duff, Ying Zhang, Craig Eldershaw, and Sam Homans. Modular Reconfigurable Robots in Space Applications. *Autonomous Robots*, 14(2):225–237, March 2003.

[55] M. Yim, Ying Zhang, and D. Duff. Modular robots. *IEEE Spectrum*, 39(2):30–34, February 2002.

[56] J. Pollack, M. A. Bedau, P. Husbands, R. A. Watson, and T. Ikegami. Designed and Evolved Blueprints For Physical Self-Replicating Machines. In *Artificial Life IX: Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems*, pages 15–20. MIT Press, 2004.

[57] M.W. Jorgensen, E.H. Ostergaard, and H.H. Lund. *Modular ATRON: Modules for a Self-Reconfigurable Robot*, volume 2. November 2004.

[58] Wei-Min Shen, B. Salemi, and P. Will. Hormone-inspired adaptive communication and distributed control for CONRO self-reconfigurable robots. *IEEE Transactions on Robotics and Automation*, 18(5):700–712, October 2002.

[59] Emanuela Del Dottore, Ali Sadeghi, Alessio Mondini, Virgilio Mattoli, and Barbara Mazzolai. Toward Growing Robots: A Historical Evolution from Cellular to Plant-Inspired Robotics. *Frontiers in Robotics and AI*, 5:16, March 2018.

[60] T. Fukuda and S. Nakagawa. Dynamically reconfigurable robotic system. In *1988 IEEE International Conference on Robotics and Automation Proceedings*, pages 1581–1586 vol.3, April 1988.

[61] Andres Castano, Wei-Min Shen, and Peter Will. CONRO: Towards Deployable Robots with Inter-Robot Metamorphic Capabilities. page 16.

[62] M. Rubenstein, K. Payne, P. Will, and Wei-Min Shen. Docking among independent and autonomous CONRO self-reconfigurable robots. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, volume 3, pages 2877–2882 Vol.3, April 2004.

[63] A. L. Christensen, R. O'Grady, and M. Dorigo. Morphology control in a multirobot system. *IEEE Robotics Automation Magazine*, 14(4):18–25, December 2007.

[64] Michael Rubenstein, Christian Ahler, and Radhika Nagpal. Kilobot: A low cost scalable robot system for collective behaviors. In *2012 IEEE International Conference on Robotics and Automation*, pages 3293–3298, St Paul, MN, USA, May 2012. IEEE.

[65] Robot simulator CoppeliaSim: Create, compose, simulate, any robot - Coppelia Robotics. https://www.coppeliarobotics.com/.

[66] GitHub - ajhalme/kbsim: Kilobot simulator. https://github.com/ajhalme/kbsim.

[67] Fredrik Jansson, Matthew Hartley, Martin Hinsch, Ivica Slavkov, Noemı Carranza, Tjelvar S G Olsson, Roland M Dries, Johanna H Gronqvist, Athanasius F M Maree, James Sharpe, Jaap A Kaandorp, and Veronica A Grieneisen. Kilombo: A Kilobot simulator to enable effective research in swarm robotics. page 13.