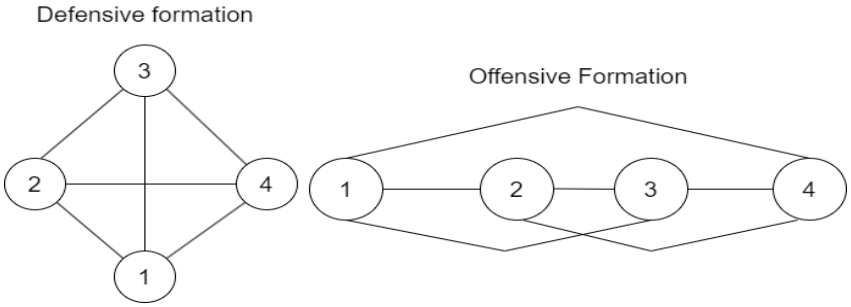
Shape Formation Using Decentralized Reference Based Graph Controller.

Multi -Robot Systems Final Project Report - Param Oza

Abstract: -

This project explores the dynamic behavior of a fully connected network of four military robots operating in formation. The primary objective is to investigate shape formation, and decentralized control mechanisms applicable to military tank formations. The proposed decentralized controller relies solely on lidar data for inter-robot communication, enhancing adaptability in diverse environments. The project utilizes a fully connected network graph, where each robot adheres to reference distances to converge into specific formations: a trapezium, a square, and a patrolling concentric circle. The robots' positions are updated through a velocity controller with a proportional error and a direction, adjusting their trajectories to minimize errors in the reference distances. The theoretical analysis involves proving convergence properties and stability characteristics of the formations. The decentralized controller is designed to achieve the desired collective behavior. The mathematical model is based on assumptions that reflect real-world scenarios, such as sensing and communication capabilities of military robots. Simulations, conducted over 4000-time steps, demonstrate the convergence of robots to the specified formations. The trajectories and final orientations of the robots are visualized, providing insights into the effectiveness of the decentralized controller. The simulations validate the proposed model and controller, emphasizing their potential applications in military scenarios.



Mathematical Model-

When applied to a scenario where 4 robots with turrets have to form an offensive and defensive formation, a decentralized approach is quite useful as a central approach is not robust in harsh scenarios as communication issues and the danger of a central controller being destroyed would incapacitate all the other robots that are dependent on the main central controller, to solve this, I use a fully connected graph based communication networks for a group of 4 robots , the robots use the distance between each other using the x and y coordinates of each other, such information would not be available in real world settings where there could be jammers or lack of GPS information, to solve this problem, lidar and local sensor information would be used to get information of the distances between each robot, as the sensor data is available from each robot to every other connected robot, the lidar data would be compared to finalize which robot is at a particular distance and the lidar distance for more than 2 robots to be same is extremely unlikely. The robots are spawning randomly between (0,30) and their task is to minimize the

error distance that everyone has in memory for the formation commanded to them, there are no obstacles that the robots have in the environment, they have collision avoidance that is based on the lidar data as well, when two robots come closer than a certain threshold, the repulsion velocity adds velocity in the direction away from the robot.

Description of Multi-Robot Behavior: The multi-robot behavior under investigation involves a team of robots exhibiting defensive, offensive, and patrolling formations, as well as a circular motion. The robots dynamically adjust their positions to maintain specified reference distances, depending on the selected command (defensive, offensive, offensive1, or patrolling). Collision avoidance is incorporated into the model to prevent robots from coming too close to each other.

Hypothetical Scenario: Military Tank Formations: In a military context, robots represent autonomous tanks forming defensive, offensive, or patrolling configurations. The robots' objective is to maintain specific formations to enhance strategic capabilities. The defensive formation guards a position, the offensive formation advances toward a target, and the patrolling formation involves circular motion around a central point.

Description of Multi-Robot Behavior: The multi-robot behavior being investigated is a combination of collective movement, distance maintenance, and orientation alignment within a swarm of robots. The hypothetical scenario involves a group of robots tasked with maintaining specified inter-robot distances while collectively moving within an environment. The robots aim to achieve and maintain a specific formation while avoiding collisions and dynamically adjusting their orientations. Additionally, circular trajectories are incorporated to enhance the overall motion pattern.

Assumptions and Constraints: Sensing and Communication: The robots have accurate distance sensing capabilities, allowing them to measure the distances between themselves. Limited communication is assumed, allowing robots to share information about their positions and orientations with nearby robots. Environment: The environment is assumed to be unbounded and obstacle-free, simplifying the navigation task. Robots can move freely within the given space. The Graph is a globally rigid graph following [1], [2], which is considered in the book for this problem.

Dynamic Formation: The robots can dynamically adjust their formations based on the desired inter-robot distances and orientations. The model assumes that the robots can change their relative positions and orientations smoothly.

Control: The robots are equipped with velocity controller and orientation controller to adjust their positions and orientations over time based on the error distances following [2] . The controllers consider both attraction towards desired distances and orientations, repulsion to avoid collisions, and circular trajectories for enhanced motion patterns. The Orientation is handled in a separate module of the Notebook.

Coordinate System: the units of measurement are not explicitly defined. The code operates in an abstract space where distances, positions, and velocities are calculated and manipulated without a specific reference to real-world units. This is common in simulation scenarios where the focus is on the relative behavior of entities rather than precise real-world measurements. robots are not described relative to an inertial frame, they are only defined in their body frames.

Variables and Parameters:

- **Robot Positions:** Denoted as $\mathbf{P} = [x, y]$, representing the 2D positions of robots.
- **Robot Orientations:** Represented as Θ, indicating the heading direction of each robot.
- **Inter-Robot Distances:** D_{ij}, representing the distances between robots i and j.
- **Velocity Gain:** k_v, controlling the rate at which robots adjust their positions based on distance errors.
- **Orientation Gain:** k_Θ, controlling the rate at which robots adjust their orientations based on orientation errors.
- **Collision Threshold:** δ, specifying the minimum distance to trigger collision avoidance.
- **Rotational Velocities:** ω_i, representing the rotational velocity of each robot.
- **Circular Trajectory Parameters:** Circular radius (r) and angular speed (ω_c) for generating circular trajectories.

The mathematical model combines distance-based attraction, orientation alignment, collision avoidance, and circular trajectories. The control law for robot i is given by:

$$\dot{\mathbf{P}}_i = k_v \sum_{j \neq i} \left((D_{ij} - D_{ij}^*) \frac{\mathbf{P}_j - \mathbf{P}_i}{D_{ij}} + \text{repulsion force} \right) + \text{circular motion}$$

$$\dot{\Theta}_i = k_\Theta (\Theta_j - \Theta_i) + \omega_i$$

The repulsion force ensures collision avoidance by exerting a force when robots are within a certain threshold. The circular motion component generates circular trajectories to enhance the overall swarm motion. The simulation loop iterates through time steps, updating positions and orientations based on the control law. The final orientations are adjusted to reach a consensus.

Theoretical Analysis-

Equilibrium State-

Equilibrium is achieved when the error is minimized to zero, this also requires careful tuning of the controller so that the constants don't overshoot the robots from their desired position in every step.

The system is represented by the positions of multiple robots, and their motion is influenced by circular trajectories, reference distances, and velocity controllers with collision avoidance. Let's denote the state of the system as $\boldsymbol{x}$, where $\boldsymbol{x}$ includes the positions of all robots.

The equilibrium state is found by setting the derivative to zero:

$$f(x^*) = 0$$

$$x = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \vdots \\ x_n \\ y_n \end{bmatrix}$$

The dynamics of the system can be described by a set of differential equations:

$$\dot{x} = f(x)$$

Let's denote x_i and y_i as the x- and y-components of the position of robot i. The equilibrium equations for each robot i are given by:

For the x-component:

$$\text{velocity\_gain} \times \text{error} \times \text{direction}_{xi} + \text{circular\_radius} \times \text{angular\_speed} \times \cos(\text{angular\_speed} \times \text{time}) + 0.5 \times \text{velocity\_gain} \times (\text{collision\_threshold} - \text{current\_distance}) \times \text{repulsion\_direction}_{xi} = 0$$

For the y-component:

$$\text{velocity\_gain} \times \text{error} \times \text{direction}_{yi} + \text{circular\_radius} \times \text{angular\_speed} \times \sin(\text{angular\_speed} \times \text{time}) + 0.5 \times \text{velocity\_gain} \times (\text{collision\_threshold} - \text{current\_distance}) \times \text{repulsion\_direction}_{yi} = 0$$

Now, let's substitute the expressions for velocity, circular_position, and $\text{repulsion\_velocity}$ into these equations:

These equations form a system of equations for each pair of robots. The positions that satisfy these equations represent the equilibrium state of the system. Since the equilibrium equations are nonlinear, we need to use numerical methods to solve them.

For the x-component:

$$\text{velocity\_gain} \times (\text{current\_distance} - S \times G[i][j]["reference\_distance\_defensive"]) \times \frac{(\text{robot\_positions}_{yi} - \text{robot\_positions}_{yj})}{\text{current\_distance}} + \text{circular\_radius} \times \text{angular\_speed} \times \cos(\text{angular\_speed} \times \text{time}) + 0.5 \times \text{velocity\_gain} \times (\text{collision\_threshold} - \text{current\_distance}) \times \frac{(\text{robot\_positions}_{xi} - \text{robot\_positions}_{xj})}{\text{current\_distance}} = 0$$

For the y-component:

$$\text{velocity\_gain} \times (\text{current\_distance} - S \times G[i][j]["reference\_distance\_defensive"]) \times \frac{(\text{robot\_positions}_{xi} - \text{robot\_positions}_{xj})}{\text{current\_distance}} + \text{circular\_radius} \times \text{angular\_speed} \times \sin(\text{angular\_speed} \times \text{time}) + 0.5 \times \text{velocity\_gain} \times (\text{collision\_threshold} - \text{current\_distance}) \times \frac{(\text{robot\_positions}_{yi} - \text{robot\_positions}_{yj})}{\text{current\_distance}} = 0$$

In the case where we are not patrolling, the angular speed becomes 0, thus only leaving terms.

For the x-component:

$$\text{velocity\_gain} \times (\text{current\_distance} - S \times G[i][j]["reference\_distance\_defensive"]) \times \frac{(\text{robot\_positions}_{yi} - \text{robot\_positions}_{yj})}{\text{current\_distance}} + 0.5 \times \text{velocity\_gain} \times (\text{collision\_threshold} - \text{current\_distance}) \times \frac{(\text{robot\_positions}_{xi} - \text{robot\_positions}_{xj})}{\text{current\_distance}} = 0$$

For the y-component:

$$\text{velocity\_gain} \times (\text{current\_distance} - S \times G[i][j]["reference\_distance\_defensive"]) \times \frac{(\text{robot\_positions}_{xi} - \text{robot\_positions}_{xj})}{\text{current\_distance}} + 0.5 \times \text{velocity\_gain} \times (\text{collision\_threshold} - \text{current\_distance}) \times \frac{(\text{robot\_positions}_{yi} - \text{robot\_positions}_{yj})}{\text{current\_distance}} = 0$$

The term current_distance-S* G[i][j]["reference_distance_offensive1"]

Becomes 0 as the distance is minimized, thus the robots are in equilibrium state when the robots converge to the reference distances.

Lyapunov stability Analysis and Invariant Set: -

We can Lyapunov Stability Analysis to verify whether our system is stable or not and to find the invariant set where the Lyapunov Function V'(x)=0, Lyapunov stability analysis is a method used to assess the stability of an equilibrium point or trajectory of a dynamical system. The analysis involves the use of a

Lyapunov function, which is a scalar function that measures the system's "energy" and helps determine whether trajectories starting from certain initial conditions will converge to, diverge from, or remain within a given region in the state space. The time derivative of the Lyapunov function, V˙(x), represents how this scalar value changes over time, providing insights into the stability of the system. Let the Lyapunov function be a function of the distances since distances is the main state variable that I originally wanted to use with purely Lidar Data, the Lyapunov Function must be Positive Definite

$$x_{ij} = C_{ij} - R_{ij}$$

Here's a breakdown of the terms:

- x_{ij}: Displacement or relative position between the robots i and j.
- C_{ij}: Current distance between the robots i and j.
- R_{ij}: Reference or desired distance between the robots i and j.

$$V(x) = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=i+1}^{N} x_{ij}^2$$

And the time derivative $\dot{V}(x)$ becomes:

$$\dot{V}(x) = \sum_{i=1}^{N} \sum_{j=i+1}^{N} x_{ij} \dot{x}_{ij}$$

In the quadratic form,

$$V(x) = \frac{1}{2} x^T Q x$$

Let Q be the identity matrix, which is positive definite, so V'(x) becomes.

$$V'(x) = \nabla V(x) \cdot x'$$

$$\frac{dx}{dt} = \frac{dc}{dt}$$

$$\frac{dc}{dt} = -\|v_i - v_j\|$$

$$\mathbf{V}'(x) = -\alpha(\nabla \mathbf{V}) \cdot \mathbf{v}$$

- α is a constant.
- ∇V is the gradient of the Lyapunov function.
- $\mathbf{v}$ is the velocity vector between two robots.

$$\nabla V(x) = \begin{bmatrix} \frac{\partial V}{\partial x_1} \\ \frac{\partial V}{\partial x_2} \\ \vdots \\ \frac{\partial V}{\partial x_n} \end{bmatrix}$$

To find $\nabla V(x)$, we differentiate $V(x)$ with respect to each component of x:

$$\frac{\partial V}{\partial x_i} = x_i$$

Therefore, the gradient $\nabla V(x)$ is simply the vector x:

$$\nabla V(x) = x$$

$$V'(x) = -\alpha \cdot x \cdot v$$

Since velocity is always positive or 0, alpha is a constant, and x=c-r is also always positive as error is positive, as it is limited by the fact that the robots are spawned at a distance greater than the reference distances.

By analyzing its values,

- $V(x)$ is positive definite: $V(x) > 0$ for all $x \neq 0$ and $V(0) = 0$.
- $\dot{V}(x)$ is negative definite (or non-positive definite): $\dot{V}(x) < 0$ (or ≤ 0) for all $x \neq 0$.

In our case V'(x) is Negative Semi Definite. With equilibrium values at x=0.

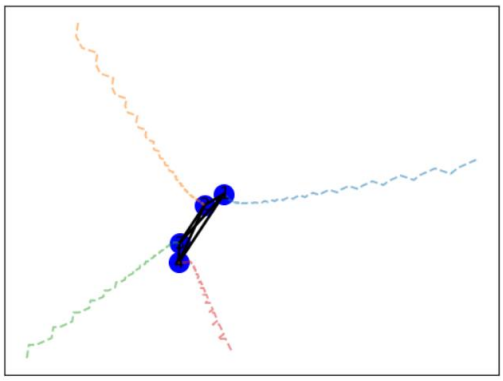
Thus, the system is Stable.

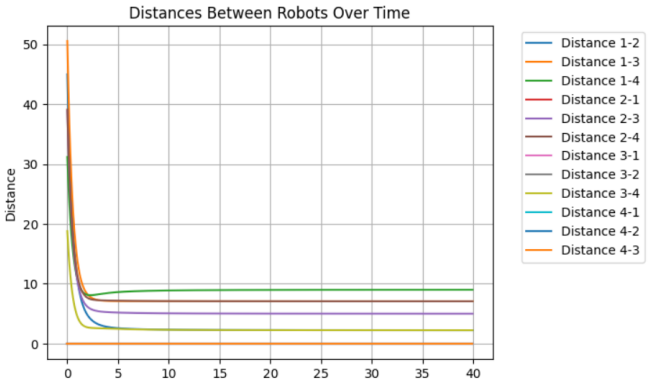
The LaSalle Invariance Principle is a stability criterion that provides conditions for the invariance of a set in the phase space of a dynamical system. The principle states that if a Lyapunov function $V(x)$ is such that $\dot{V}(x) \leq 0$ for all x in a region Ω (except possibly at the equilibrium points), and $\dot{V}(x) = 0$ only at the equilibrium points in Ω, then the system trajectories starting in Ω are attracted to the largest invariant set contained in $\{x \in \Omega \mid \dot{V}(x) = 0\}$.

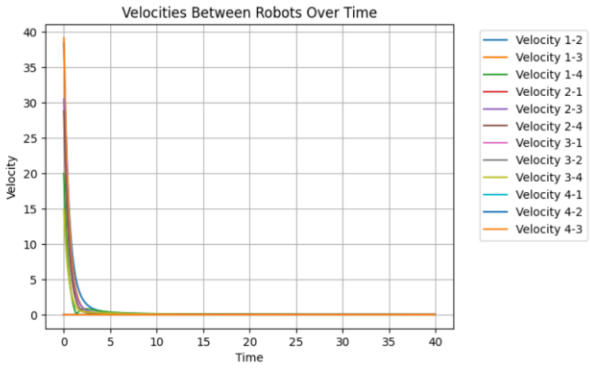
- Using LaSalle's Invariance Principle, we can find the invariant set, this principle states that the trajectories of the system converge to the largest invariant set in the region where V'(x)=0
- The Invariant set consists of state vector x where V'(x)=0, corresponds to the set M.
- we find that the only state x which is in invariant set M is the when x=0 and since x=c-r, it means the error is zero when the ***Current distance=Reference distance***, thus proven that the system is stable with equilibrium at error zero which is the when change in distance is zero and the equilibrium state is the same as invariant set, in the way that any initial state **x** in the state space Ω converges to invariant set M.
- As our Graph formation are based on rigid graph frameworks following the literature [2], the control strategy is stable and convergence is obtained by the second eigen value of L matrix.

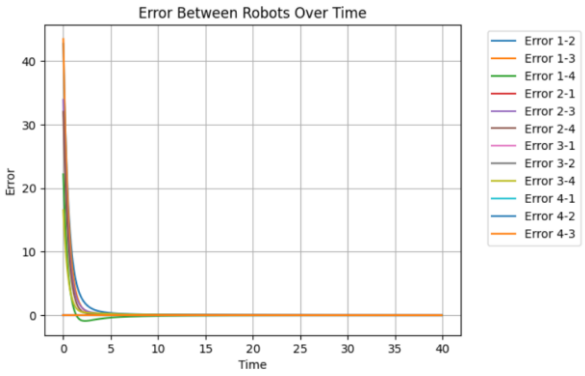
Results

Offensive Formation:

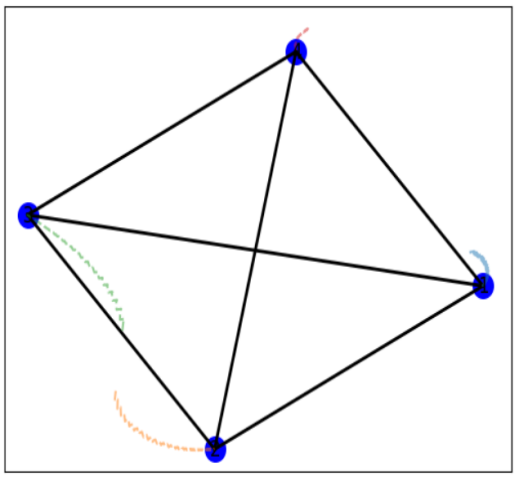


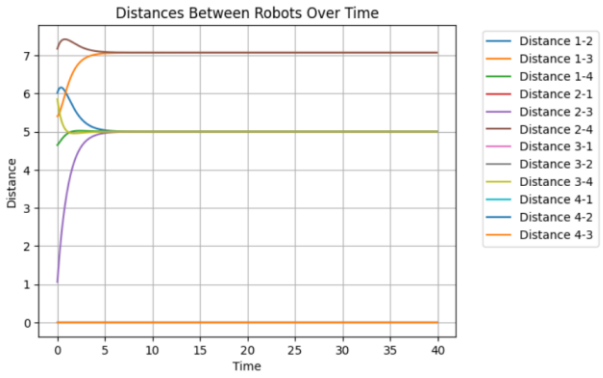


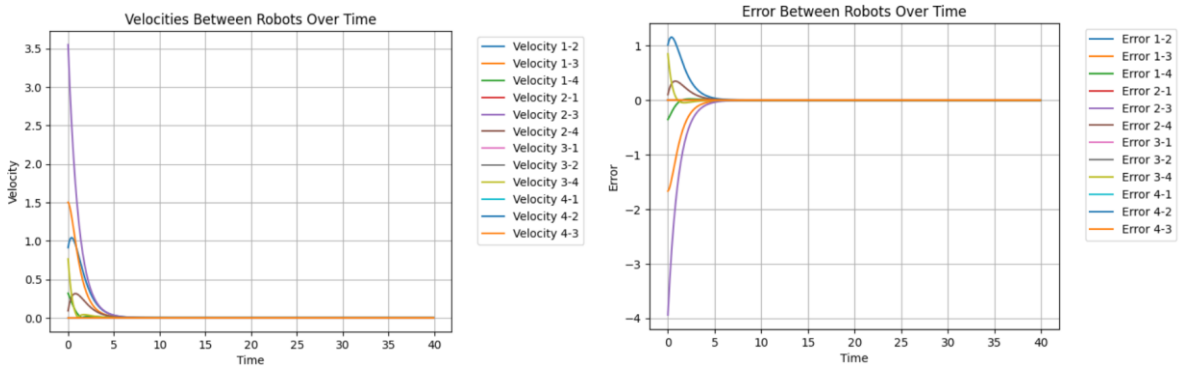




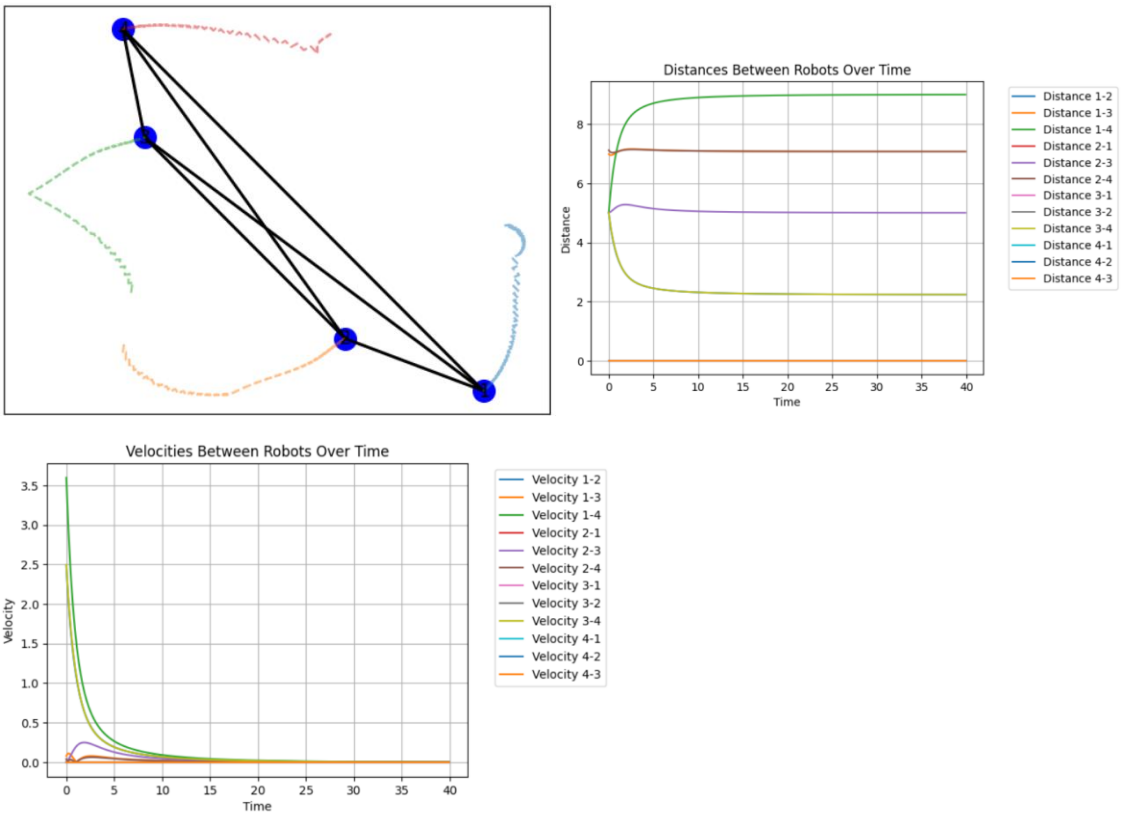
Defensive Formation:



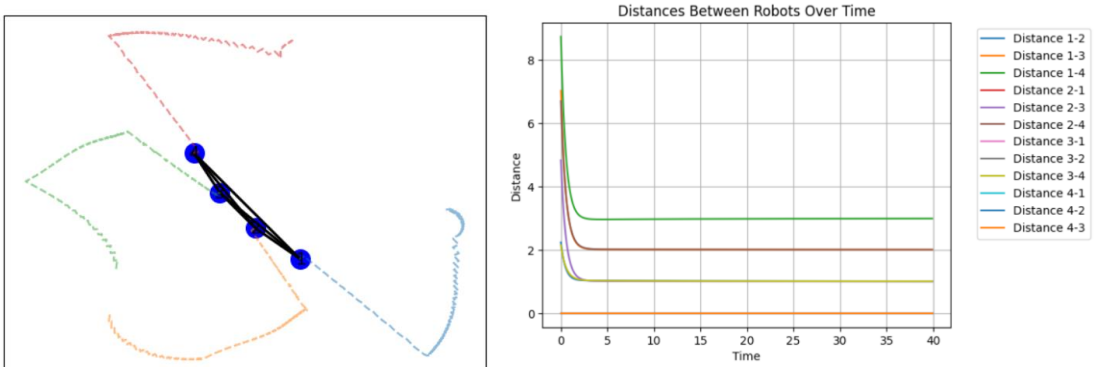




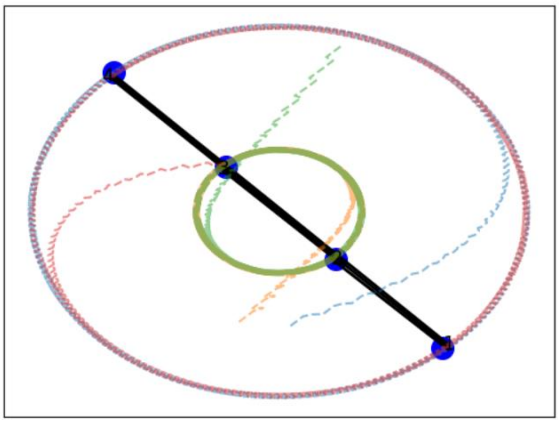
Offensive Formation:

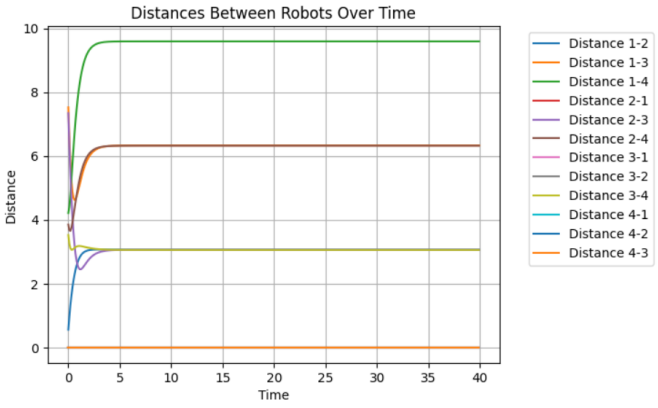


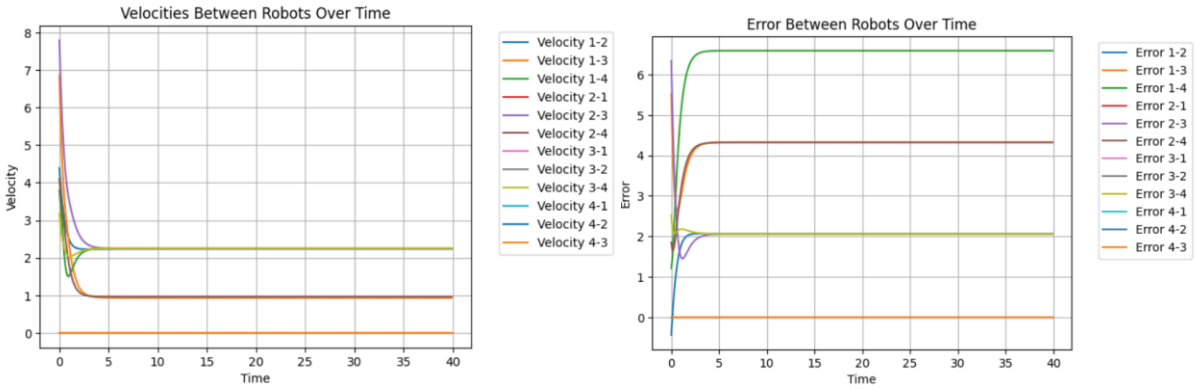
Offensive Formation1:



Patrolling:

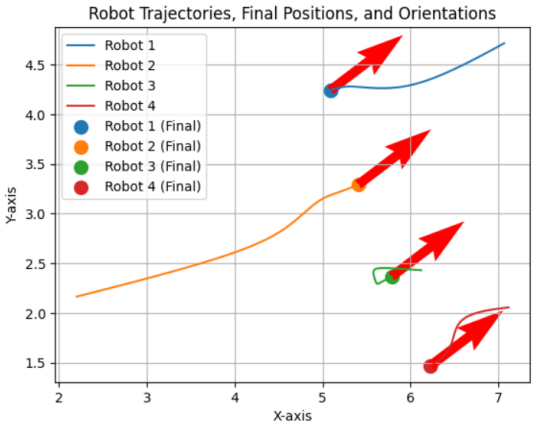




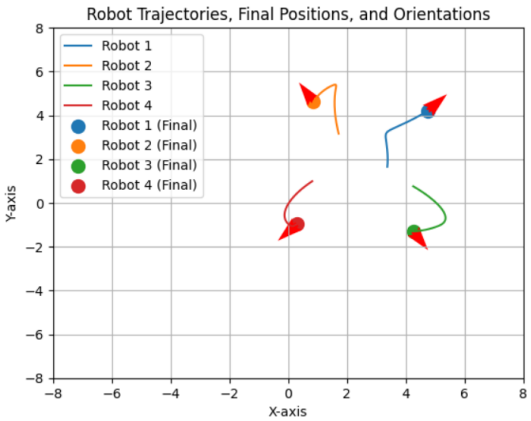


Orientation Control with Average Consensus:

Offensive Orientations



Defensive Orientations



Conclusion

From the results obtained from the simulations, robots can converge to the formation according to the memory of what that formation would be like, which coincides with idea of how people create formations based on the memory of what that formation looks like and then comparing their distances with other people and adjusting their velocity to form the shape. Collision Avoidance and Orientation consensus further adds to the stability of a real system, as the orientations are controlled decentralized based on the graph center that is calculated locally after convergence and the offensive formation orientations that are based on average orientation consensus plus a constant to make it a desired direction. Patrolling mode helps in the scenario where the robots quickly needs to dispatch from the formations into motion and back into formations as required, this sort of system would benefit quite from decentralization as such changes require quick response based on purely local data which minimizes the amount of overhead a single central control would create and adds robustness to the system without the risk of a central controller which could be damaged in active warzones.

Video Link- https://youtu.be/2_AUeeo0_xg

References

[1]Mesbahi, M., & Egerstedt, M. (2010). Graph-Theoretic Methods in Multiagent Networks. Princeton University Press.

[2]Jiang, C., Chen, Z., & Guo, Y. (2019). Learning Decentralized Control Policies for Multi-Robot Formation. IEEE

[4]Cai, X. (2013). Graph rigidity-based formation control of planar multi-agent systems. LSU Scholarly Repository.

[3]Lafferriere, G., Caughman, J., & Williams, A. (2004). Graph Theoretic Methods in the Stability of Vehicle Formations. IEEE.

[5]Turpin, M., Michael, N., & Kumar, V. (2012). Decentralized Formation Control with Variable Shapes for Aerial Robots. IEEE.

[6]Ren, W. (2006). Multi-vehicle consensus with a time-varying reference state. ScienceDirect.com