Department of Mechanical Engineering

4DM70

Network Synchronization and Multi-Robot Coordination

Qinyao He 1945556
Lio Huntjens 1004092

September 6, 2024

1 Network Synchronization

1.1 HR Model Properties

a Network synchronization can be attained provided that the system exhibits exponentially convergent dynamic and demonstrates strictly semi-passive characteristics. The internal dynamics of known HR model neuron are given by Equation(1.1).

$$\begin{aligned}\dot{y}_i(t) &= 100\left(-y_i(t)^3 + 3y_i(t) - 4.7 + 5z_{i,1}(t) - z_{i,2}(t) + u_i(t)\right)\\ \dot{z}_{i,1}(t) &= 100(-y_i(t)^2 - 2y_i(t) - z_{i,1}(t))\\ \dot{z}_{i,2}(t) &= 0.5(4y_i(t) + 4.472 - z_{i,2}(t))\end{aligned} \tag{1.1}$$

Rewrite the model dynamics into the following form and use Demidovich condition to verify that the internal dynamics of the system are exponentially convergent. For a 2 by 2 matrix, the proof can be simplified to checking if the matrix's trace and determinant meet the specified criteria which is shown in Equation(1.3). Simultaneously, Equation(1.4) shows that when taking P as the identity matrix, a constant δ can be identified that ensures the system meets the test criteria. This also confirms the inherently exponential convergence of the HR model neuron.

$$\begin{pmatrix}\dot{z}_{i,1}\\ \dot{z}_{i,2}\end{pmatrix} = \begin{pmatrix}q_1(z_i,y_i)\\ q_2(z_i,y_i)\end{pmatrix} = \begin{pmatrix}-100 & 0\\ 0 & -0.5\end{pmatrix}\begin{pmatrix}z_{i,1}\\ z_{i,2}\end{pmatrix} + \begin{pmatrix}-100y_i^2 - 200y_i\\ 2y_i + 2.236\end{pmatrix} \tag{1.2}$$

$$\begin{aligned}tr(\frac{\partial q}{\partial z}(z,y)) &= \begin{pmatrix}-100 & 0\\ 0 & -0.5\end{pmatrix} = -100 - 0.5 = -100.5 < 0\\ det(\frac{\partial q}{\partial z}(z,y)) &= \begin{pmatrix}-100 & 0\\ 0 & -0.5\end{pmatrix} = -100 \times (-0.5) = 50 > 0\end{aligned} \tag{1.3}$$

$$\begin{gathered}P\left(\frac{\partial q}{\partial z}(z,y)\right) + \left(\frac{\partial q}{\partial z}(z,y)\right)^T P = \begin{pmatrix}-200 & 0\\ 0 & -1\end{pmatrix} \leq -\delta I\\ 0 < \delta \leqslant 1\end{gathered} \tag{1.4}$$

b S as a radially unbounded and positive definite storage function, if it satisfies the condition $\dot{S} = \frac{\partial S}{\partial x}(f(x,u) \leq y^T u - H(x))$ with $H(x) > 0$ for all $\|x\| > \rho \geq 0$, then it can be proved that the system is strictly semipassive. In the HR model, the storage function S is associated with $y_i, z_{i,1}, z_{i,2}$. The outcomes derived by applying the chain rule are as follows:

$$\begin{aligned}\dot{S} &= \frac{\partial S}{\partial y_i}y_i + \frac{\partial S}{\partial z_{i,1}}z_{i,1} + \frac{\partial S}{\partial z_{i,2}}z_{i,2}\\ &= \frac{1}{100}y_i \cdot \dot{y_i} + \frac{1}{40}z_{i,1} \cdot \dot{z_{i,1}} + \frac{1}{2}z_{i,2} \cdot \dot{z_{i,2}}\\ &= y_i u_i - [(y_i^4 + \frac{5}{2}z_{i,1}y_i^2 + \frac{5}{2}z_{i,1}^2) - 3y_i^2 + \frac{1}{4}z_{i,2}^2 + 4.7y_i - 1.118z_{i,2}]\end{aligned} \tag{1.5}$$

$H(x)$ is then defined as follows, further proof is given according to the hint, and similar terms are combined to simplify the equation which can be seen in Equation(1.6).

$$\begin{aligned}H &= (y_i^4 + \frac{5}{2}z_{i,1}y_i^2 + \frac{5}{2}z_{i,1}^2) - 3y_i^2 + \frac{1}{4}z_{i,2}^2 + 4.7y_i - 1.118z_{i,2}\\ &> \frac{1}{4}(y_i^4 + z_{i,1}^2) - 3y_i^2 + \frac{1}{4}z_{i,2}^2 + 4.7y_i - 1.118z_{i,2}\\ &= \frac{1}{4}y_i^4 - 3y_i^2 + 4.7y_i - 1.25 + \frac{1}{4}z_{i,1}^2 + \frac{1}{4}(z_{i,2} - 2.236)^2\end{aligned} \tag{1.6}$$

Both terms related to $z_{i,1}$ and $z_{i,2}$ are non-negative. Figure 1.1 displays a graph of a function related to y_i, which has zeros at -4.11 and 0.34, where the function remains positive in the interval $(-\infty, -4.11) \cup (0.34, +\infty)$. Consequently, the system exhibits strictly semipassive when $H(y_i, z_{i,1}, z_{i,2}) > 0$ with $\|y_i\| > 4.11 > 0$.

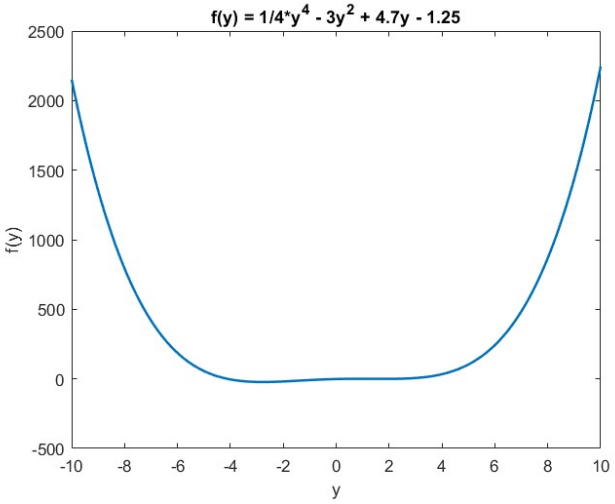


Figure 1.1: f(y)

1.2 Network Synchronization: Theory and Experiments

a Two electronically coupled HR neurons are connected with $w_{12} = w_{21} = 1$. The upper and lower bounds of the coupling strengths, which have been found through multiple experiments, achieve HR model neurons practical synchronization at different time delays. The measurement results are shown in Table 1.1.

Table 1.1: Lower-bounds and upper bounds for the coupling strength.

Time delay	Lower bound	Upper bound
0	0.51	10.00
1	0.57	8.62
2	0.62	5.25
3	0.68	4.10
4	0.73	3.51
5	0.78	3.30

b Figure 1.2 shows the network structure composed of five HR neurons, and based on the scaling rules of the system synchronization region, we can infer the synchronization conditions of more complex networks from the synchronization interval of the two HR neuron coupled systems.

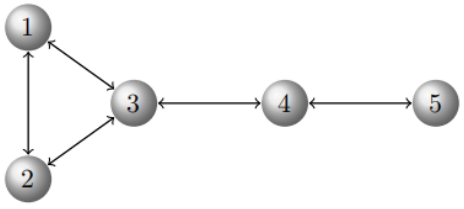


Figure 1.2: The network of five HR neurons.

For different time delays, lower-bounds and upper-bounds of the coupling strength can be theoretically calculated using Equation(1.7), where $(\sigma, \tau) \in S$ is the synchronization interval of the two neurons, and scaling factors λ_j are the non-zero eigenvalues of L under the new network structure. The final synchronization region of the network with $(\sigma, \tau) \in S_2^* \cap S_N^*$ is the intersection of the scaled set calculated from the maximum and minimum non-zero eigenvalues. After some calculations, with $\lambda_2 = 0.52, \lambda_N = 4.17$ the lower bound at different times will be magnified by a factor of

about 4, and the upper bound will be reduced by a factor of about 2. Table 1.2. shows the results of the theoretical calculation, if the derived lower bound is greater than the upper bound, it means that the system cannot be synchronized at the specific delay time, and the corresponding upper and lower bounds are denoted by the symbol "-". Table 1.3 presents the experimentally measured upper and lower bounds of the HR model network.

$$S_j^* := \left\{ (\sigma, \tau) \in \mathbb{R}_+ \times \mathbb{R}_+ \mid \left(\frac{\lambda_j}{2} \sigma, \tau \right) \in S \right\} \tag{1.7}$$

Table 1.2: Lower-bounds and upper bounds based on theoretical calculations.

Time delay	Lower bound	Upper bound
0	1.97	4.76
1	2.20	4.10
2	2.39	2.50
3	-	-
4	-	-
5	-	-

Table 1.3: Lower-bounds and upper bounds based on experiments.

Time delay	Lower bound	Upper bound
0	2.10	10
1	2.50	3.65
2	-	-
3	-	-
4	-	-
5	-	-

According to previous work, it is observed that the upper bound of the coupling strength is 10 at $\tau = 0$, which significantly differs from the theoretical value of 4.76. This discrepancy indicates that, in the absence of delay, the actual upper bound to achieve synchronization of the two neurons is much greater than 10. However, due to limitations in the measurement equipment, the maximum adjustable coupling strength is restricted to 10. Due to measurement errors, the actual synchronization interval is narrower than the theoretically calculated value. Specifically, the measured lower bound is higher and the upper bound is lower than expected. For $\tau = 2$, the system is theoretically capable of achieving synchronization within a coupling strength range of 2.39 to 2.50. However, this synchronization was not observed experimentally. This discrepancy may be due to the model not fully capturing the system's dynamics, and the measurement results may be affected by noise and environmental factors, leading to inaccuracies. These uncertainties result in more stringent synchronization conditions for HR circuits in practice.

c In order to add an edge to the above structure and increase the coupling strength interval that can realize system synchronization, the scaling effect is applied to calculate the eigenvalues of different networks and find the network with the largest intersection of scaled copies. The network in the Figure 1.2 exhibits a certain symmetry in its structure. Specifically, the connections between nodes 1 and 4, and nodes 1 and 5, are identical to those between nodes 2 and 4, and nodes 2 and 5. Therefore, we can simplify the edge-adding scheme to three cases: connecting nodes 1 and 4, nodes 1 and 5, or nodes 3 and 5. Table 1.4 shows the smallest and largest non-zero eigenvalues for the Laplace matrices L of different networks formed by adding different edges. From the scaling effect, a larger λ_2 will cause a smaller lower bound of the coupling strength, and a larger λ_N will also cause a smaller upper bound of the coupling strength. Therefore, a larger λ_2 and a smaller λ_N can improve the synchronization of the system. It is preliminarily observed that the connection of 1 and 5 produces the largest λ_2 and the second smallest λ_N, in which case the system can be calculated to produce the maximum coupling strength range [0.82, 3.73]. In addition, another reason for

choosing the above method is that node 5 is the farthest node in the current configuration and has fewer connections with other nodes, connecting nodes 1 and 5 should help in distributing the synchronization influence more evenly and reducing the longest path in the graph.

Table 1.4: The smallest and largest non-zero eigenvalues for different network L.

Cases for adding an edge	λ_2	λ_N
Add an edge between 1 & 5	1.382	4.618
Add an edge between 1 & 4	0.830	4.481
Add an edge between 3 & 5	1	5

Further experiments are carried out to verify that the coupling strength range of three different connection modes with $\tau = 1$ can synchronize the system is shown in the following table. Indeed, connecting node 1 to node 5 improves the system's synchronization to its maximum potential.

Table 1.5: Coupling strength range for three different cases.

Cases for adding an edge	Lower bound	Upper bound	Range of the coupling strength
Add an edge between 1 & 5	1.00	3.30	2.30
Add an edge between 1 & 4	1.59	3.38	1.79
Add an edge between 3 & 5	1.34	2.95	1.61

d It is known that in the network of eight coupled neurons, when the coupling strength $\sigma > 0.55$, the entire network will practically synchronize. The network presents a symmetrical structure, as illustrated in Figure 1.3 where the edge lengths represent the weights of the edges. Moreover, due to the system's inherent strict semi-passive property and exponential convergence characteristics, partial synchronization can be achieved through permutation matrices at specific coupling strength intervals and time delays.

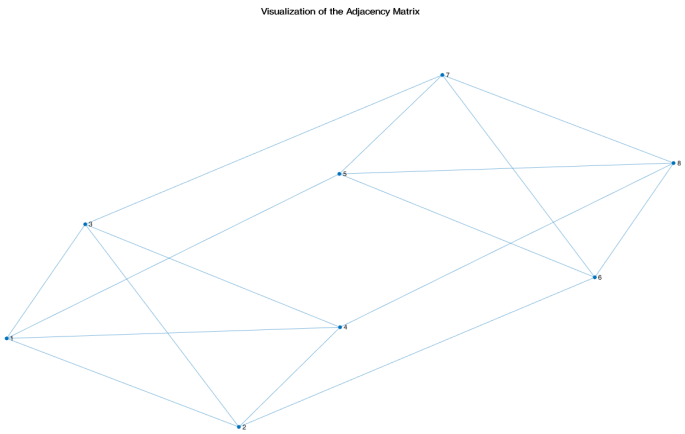


Figure 1.3: The network of eight HR neurons.

In the experiment, we observed that two clusters—nodes 1, 2, 3, 4 and nodes 5, 6, 7, 8—synchronized with $\sigma \in [0.22, 0.55]$ before the entire system achieved full synchronization. The responses of these clusters are depicted in Figure 1.4.

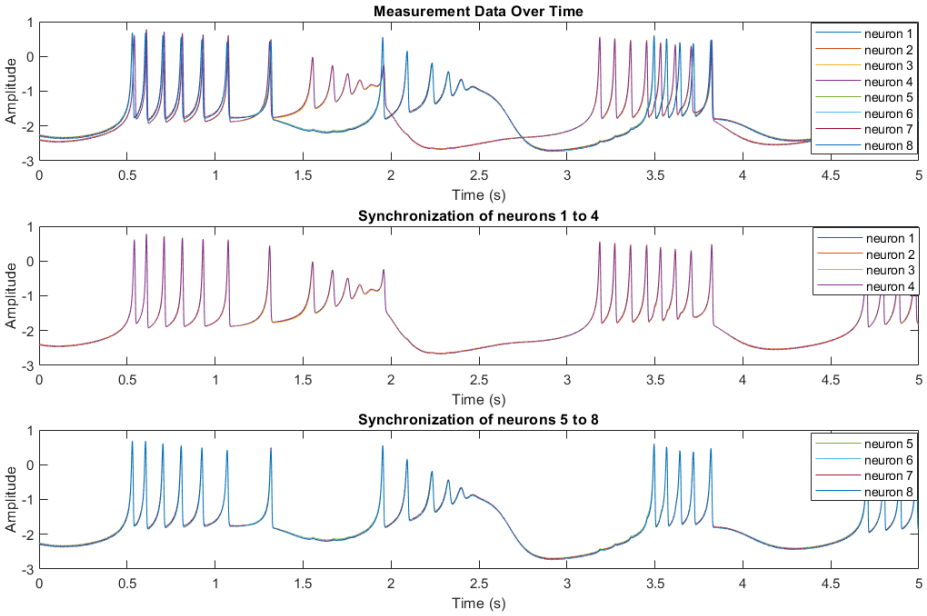


Figure 1.4: Partial synchronization of cluster1,2,3,4 and cluster 5,6,7,8.

e When the coupling strength is set to 2, the system transitions from practical synchronization to partial synchronization at $\tau = 0.6$. Notably, the two clusters that exhibit partial synchronization change to nodes 1, 3, 6, 8 and nodes 2, 4, 5, 7.

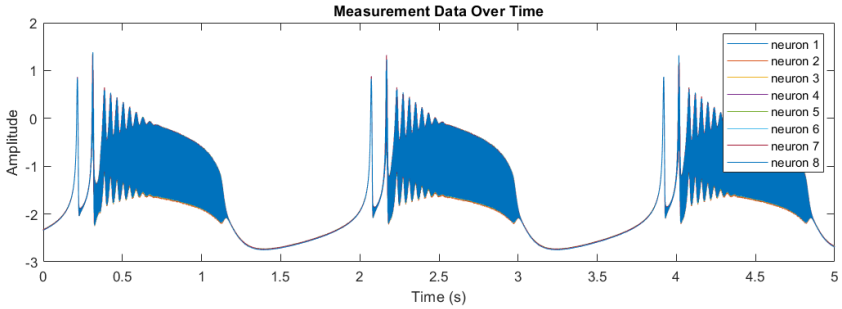


Figure 1.5: A critical state of practical synchronization to partial synchronization with $\tau = 0.6$.

f According to the partial synchronization theorem, the undirected network of diffusively coupled system contains an asympototically stable subset if it satisfies $\sigma\lambda > \bar{\lambda}(\lambda \geqslant 0)$ with λ the smallest eigenvalue of L whose eigenvector is in range $I - \Pi$. In addition, the rearranged network through permutation matrix needs to meet $L\Pi = \Pi L$, that is, the network remains symmetrical and unchanged after swapping with some nodes. In the previous questions, we found that at $\tau = 0$, the network was partially synchronized with the sets of node 1, 2, 3, 4 and node 5, 6, 7, 8 before the practical synchronization, but with $\sigma = 2$ and $\tau = 0.6$ the synchronized groups became node 1, 3, 6, 8, and node 2, 4, 5, 7. To explain this phenomenon, it is necessary to observe the regions where the network is partially synchronized after being rearranged by different permutation matrices. Note that permutation matrice rearranges the position between the two nodes, and since we are observing synchronization in groups of four nodes, meaning that any two nodes in the same group can be synchronized, we need to discuss the areas where different pairs in the same group can be synchronized. The eigenvalues of L of the eight coupled neurons' network are $\{0, 2, 6, 6, 8, 8, 8, 10\}$, with $\lambda_2 = 2$ and $\lambda_N = 10$. Note that λ_2 and λ_N are the minimum and maximum non-zero eigenvalues of L, respectively.

$$\Pi_1 = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \tag{1.8}$$

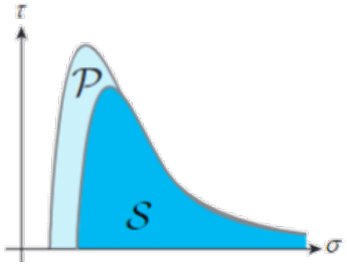
Using permutation matrix Π_1, positions of nodes 1 and 2, 3 and 4, 5 and 6, 7 and 8 will be swapped. After calculation, the rearranged network has eigenvectors in range$(I - \Pi)$, with $\lambda' = 8$, $\lambda^* = 10$, which are the smallest and largest eigenvalues corresponding to the eigenvectors in this dimension, respectively.

$$\Pi_2 = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \tag{1.9}$$

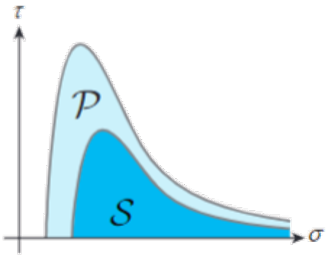
Similarly, with Π_2, the positions between nodes 1 and 3, 2 and 4, 5 and 7, 6 and 8 are swapped, and we get $\lambda' = 6$, $\lambda^* = 8$.

$$\Pi_3 = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix} \tag{1.10}$$

With Π_3, the positions between nodes 1 and 4, 2 and 3, 5 and 8, 6 and 7 are swapped, and $\lambda' = 8$ and $\lambda^* = 10$ can be obtained. Based on the above work, when Π_1 and Π_3 are used to rearrange the network, their eigenvalues meet the conditions $\lambda' > \lambda_2$, $\lambda^* = \lambda_N$. Besides, when Π_2 is applied, the eigenvalues satisfy $\lambda' > \lambda_2$, $\lambda^* < \lambda_N$. Figure 1.6a shows the results of partial synchronization of the network after applying Π_1 and Π_3, while Figure 1.6a shows the results of using Π_2. It can be seen that in the two groups of 1, 2, 3, 4 and 5, 6, 7, 8, using the permutation matrix arbitrarily swaps the positions of two nodes, and a partially synchronized region will be generated before the system reaches the practical synchronized region. Due to the overlap of partial synchronization regions between different pairs, a partial synchronization phenomenon with a group of four nodes was finally observed in the experiments.



(a) $\lambda' > \lambda_2$, $\lambda^* = \lambda_N$



(b) $\lambda' > \lambda_2$, $\lambda^* < \lambda_N$

Next, following the same procedure, each of the possible rearrangements were discussed that exist in the two groups of coupling synchronization groups 1, 3, 6, 8 and 2, 4, 5, 7 observed under $\sigma = 2$ and $\tau = 0.6$. Π_4 replaces the position between nodes 1 and 6, 2 and 5, 3 and 8, 4 and 7, with $\lambda' = 2$ and $\lambda^* = 8$. Consequently, partial synchronization of the region shifts to the area covered under higher coupling strength and greater latency as shown in Figure 1.7 with $\lambda' = \lambda_2$, $\lambda^* < \lambda_N$.

$$\Pi_4 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{1.11}$$

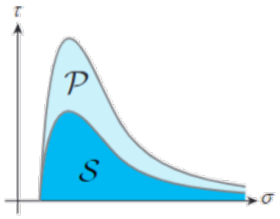


Figure 1.7: $\lambda' = \lambda_2$, $\lambda^* < \lambda_N$.

Moreover, with Π_5, the positions between nodes 1 and 8, 2 and 7, 3 and 6, 4 and 5 are swapped, and we get $\lambda' = 2$, $\lambda^* = 8$. It still meets the conditions $\lambda' = \lambda_2$, $\lambda^* < \lambda_N$. In summary, using permutation matrix Π_2, Π_4 and Π_5, rearrangement between groups 1,3,6,8 and 2,4,5,7 will result in a partially synchronized region with a large coupling strength and a large time delay. At the same time, due to the overlap between the synchronization regions of different node pairs, a partial synchronization phenomenon with a group of 4 nodes was observed which is different from the previous one.

$$\Pi_5 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{1.12}$$

g In question d, it was observed that at $\tau = 0$, there would be two groups of 1, 2, 3, 4, and 5, 6, 7, 8 synchronized with four neurons as a group. In order to achieve more granular part synchronization in which

- neurons 1 and 2 practically synchronize, and
- neurons 3 and 4 practically synchronize, and
- neurons 5 and 6 practically synchronize, and
- neurons 7 and 8 practically synchronize,

for a coupling strength lower than required for full practical synchronization. As can be seen in Figure 1.3, the network is highly symmetrical, and there are multiple configurations that can be rearranged through the permutation matrix. To achieve practical partial synchronization where only the desired neurons can be synchronized, we need to add two edges to break the symmetrical structure of the network to some extent.

In the original network structure, with the corresponding permutation matrix, the positions of the four nodes can be swapped arbitrarily, and a partially synchronized region will be generated between the complete synchronization of the system, which makes the synchronization of the four nodes occur. Then, add the edge between neurons 3 to 6 and neurons 4 to 5. In the new network, the non-zero minimum and maximum eigenvalues of L become $\lambda_2 = 2.76$ and $\lambda_N = 10$. In this case, when applying Π_1 to the coupled system, $\lambda' = 6$ and $\lambda^* = 10$ are obtained. Since it satisfies the condition $\lambda' > \lambda_2$, $\lambda^* = \lambda_N$, a partially synchronized area is still created before the full practical synchronization. However, it is not possible to apply Π_2 and Π_3 because the prerequisites $L\Pi = \Pi L$ for rearranging the network are not met. As a result, some of the symmetrical structure of the network is broken, and only the stable subsets 1 and 2, 3 and 4, 5 and 6, 7 and 8 obtained using permutation martrix Π_1 remain. Figures 5 and 6 illustrate the partial synchronization between the four different groups of neurons that we finally observed at $\tau = 0$ and $\sigma = 0.2$.

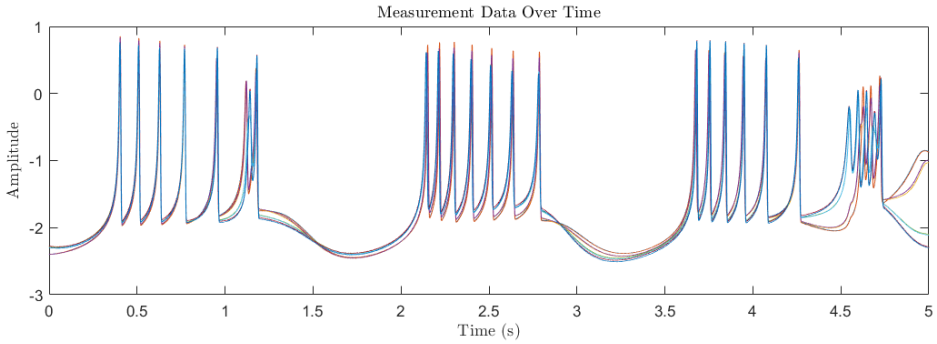


Figure 1.8: Partial synchronization at $\tau = 0$ and $\sigma = 0.2$.

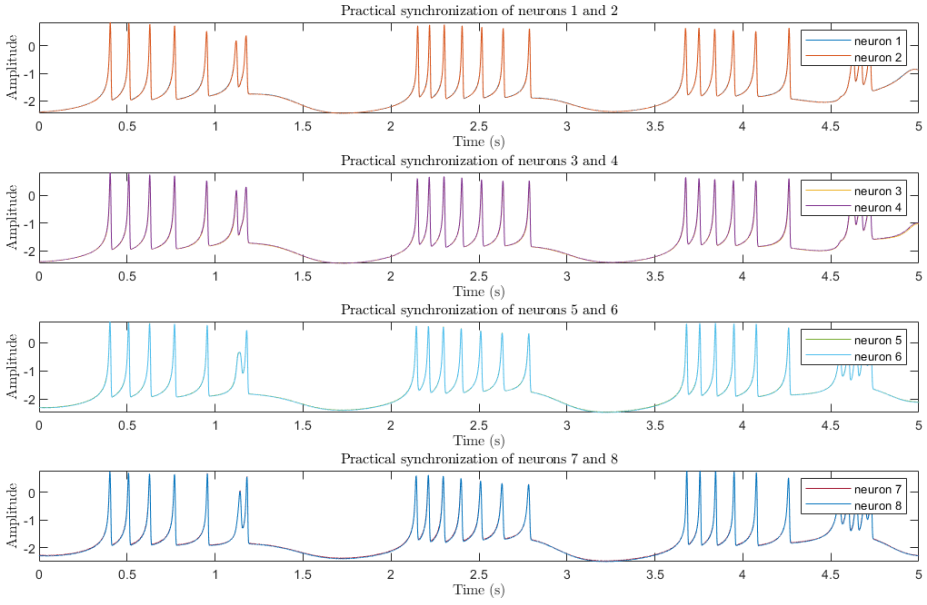


Figure 1.9: Partial synchronization of neurons 1 and 2, 3 and 4, 5 and 6, 7 and 8 after adding edges 3 to 6 and 4 to 5.

2 Multi-Robot Coordination

For this part of the assignment, the task "Option 1: Safe Equilateral Triangle Formation" was chosen and executed.

2.1 Controller Design

As starting point for creating the controller design, the attractive and repulsive potentials where based on the given lecture and problem set functions. The individual attractive potentials are based on the current position to desired goal formulation, where x_i is the current position and $x_i^\star$ is the desired goal position (with $i = A, B, C$). The desired goal positions, with 1 m distance between each of the robots, were defined as $x_A^\star = [0, 0]$, $x_B^\star = [1, 0]$ and $x_C^\star = [\frac{1}{2}, \frac{\sqrt{3}}{2}]$. These positions determine that the robots always go towards the same location and orientation of the triangle configuration independent of their initial position (see Figure 2.1). Meaning robot A is always on the bottom left corner, robot B always on the bottom right corner and robot C always on the top corner.

Attractive Potentials	Repulsive Potentials	
$V_A^{att}(x_A) = \frac{1}{2}\kappa_A \lvert\lvert x_A - x_A^\star\rvert\rvert^2$	$V_{AB}^{rep}(x_A, x_B) = \lvert\lvert x_A - x_B\rvert\rvert^2$	$V_{BA}^{rep}(x_A, x_B) = \lvert\lvert x_B - x_A\rvert\rvert^2$
$V_B^{att}(x_B) = \frac{1}{2}\kappa_B \lvert\lvert x_B - x_B^\star\rvert\rvert^2$	$V_{AC}^{rep}(x_A, x_C) = \lvert\lvert x_A - x_C\rvert\rvert^2$	$V_{CA}^{rep}(x_A, x_C) = \lvert\lvert x_C - x_A\rvert\rvert^2$
$V_C^{att}(x_C) = \frac{1}{2}\kappa_C \lvert\lvert x_C - x_C^\star\rvert\rvert^2$	$V_{BC}^{rep}(x_B, x_C) = \lvert\lvert x_B - x_C\rvert\rvert^2$	$V_{CB}^{rep}(x_B, x_C) = \lvert\lvert x_C - x_B\rvert\rvert^2$

The attractive and repulsive potentials are formulated in the coordination potentials 2.1a and can be used to calculate the coordination controls for each robot 2.1. It was chosen to not have a single coordination potential for all robots from which to determine the coordination controls. This is due to keeping the equations and computations as concise as possible.

$$V_i(x_i, x_j, x_k) = \frac{V_i^{att}}{s(V_{ij}^{rep}) * s(V_{ik}^{rep})} = \frac{\frac{1}{2}\kappa_i ||x_i - x_i^\star||^2}{s(||x_i - x_j||^2) s(||x_i - x_k||^2)} \tag{2.1a}$$

$$\dot{x}_i = -\frac{\partial V_i}{\partial x_i} \tag{2.1b}$$

$$= -\frac{s(V_{ij}^{rep}) s(V_{ik}^{rep}) \cdot \kappa_i (x_i - x_i^\star) - V_i^{att}[s'(V_{ik}^{rep}) s(V_{ij}^{rep}) \cdot 2(x_i - x_k) + s(V_{ik}^{rep}) s'(V_{ij}^{rep}) \cdot 2(x_i - x_j)]}{[s(V_{ij}^{rep}) s(V_{ik}^{rep})]^2}$$

$$= -\frac{s(V_{ij}^{rep}) s(V_{ik}^{rep}) V_i'^{att} - V_i^{att}[s'(V_{ik}^{rep}) s(V_{ij}^{rep}) V_{ik}'^{rep} + s(V_{ik}^{rep}) s'(V_{ij}^{rep}) V_{ij}'^{rep}]}{[s(V_{ij}^{rep}) s(V_{ik}^{rep})]^2}$$

with $i = \{A, B, C\}$, $j = \{B, C, A\}$, $k = \{C, A, B\}$ and $\kappa_i > 0$.

The saturation function used here is 2.2.

$$s(x) = tanh(kx) \tag{2.2a}$$

$$s'(x) = k \cdot (1 - tanh(kx)^2) \tag{2.2b}$$

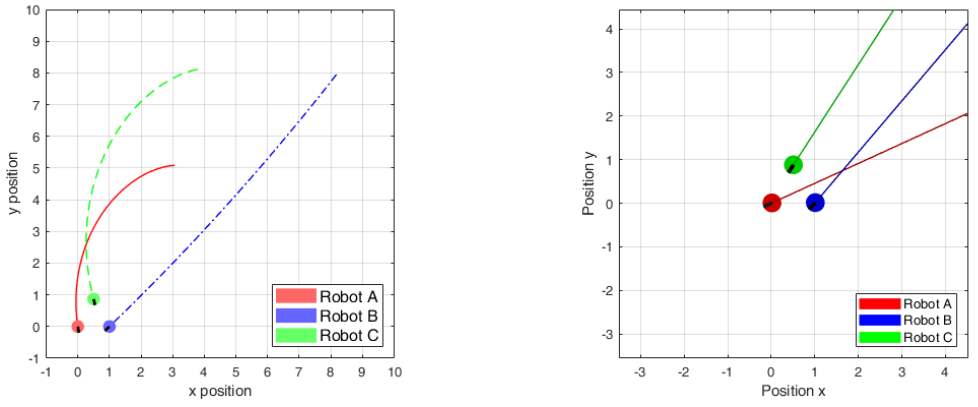
with $k = 3$.

Furthermore, the unicycle forward gradient control functions of 2.3 for linear and angular velocity are implemented for each of the robots.

$$v = max\left(-\kappa_v \begin{bmatrix} cos\theta \\ sin\theta \end{bmatrix}^T \frac{\partial V_i}{\partial x_i}(x_i, x_j, x_k), 0\right) \tag{2.3a}$$

$$\omega = \kappa_\omega atan2\left(-\begin{bmatrix}-sin\theta\\ cos\theta\end{bmatrix}^T \frac{\partial V_i}{\partial x_i}(x_i, x_j, x_k), -\begin{bmatrix}cos\theta\\ sin\theta\end{bmatrix}^T \frac{\partial V_i}{\partial x_i}(x_i, x_j, x_k)\right) \tag{2.3b}$$

These formula's have been implemented and tested by computing and running the *"demo_unicycle_test2.m"* and the *"triangle_formulation.m"* with the *"demo_turtlebot_simulator.m"* Matlab scripts (see Figure 2.1) (see Appendix A). This showed desirable behavior of the robot triangle formulation, therefore the next step could be taken of implementing consensus in relative positions.



(a) Test *"demo_unicycle_test2.m"*

(b) Test *"demo_turtlebot_simulator.m"*

Figure 2.1: Move-to-goal triangle configuration test.

The distance-to-goal based attractive potential from before is replaced by the consensus in relative position formation function 2.4a. This consensus function allows the robots to have a triangle configuration that is more adaptable to the environment and potential disturbances that occur within. The orientation of the triangle configuration stays the same, however the location is now relative (depending on initial starting conditions or disturbances) rather than being pre-determined.

$$V_i^{att}(x) = \frac{1}{2}\kappa_i \Sigma_{j\in N_i} ||x_i - x_j - (x_i^\star - x_j^\star)||^2 \tag{2.4a}$$

$$V_i'^{att} = \kappa_i \Sigma_{j\in N_i}(x_i - x_j - (x_i^\star - x_j^\star)) \tag{2.4b}$$

with $i = \{A, B, C\}$ and N_i being all neighbouring nodes/robots of i.

Putting this into the combined potential functions gives 2.5. The control $\dot{x}_i$ with this new potential function is calculated the same as 2.1.

$$V_i = \frac{V_i^{att}}{s(V_{ij}^{rep}) * s(V_{ik}^{rep})} = \frac{\frac{1}{2}\kappa_i(||x_i - x_j - (x_i^\star - x_j^\star)||^2 + ||x_i - x_k - (x_i^\star - x_k^\star)||^2)}{s(||x_i - x_j||^2) * s(||x_i - x_k||^2)} \tag{2.5a}$$

with $i = \{A, B, C\}$, $j = \{B, C, A\}$, $k = \{C, A, B\}$ and $\kappa_i > 0$.

Lastly, during the numerical analysis, the robots showed to maintain some velocity even after reaching the desired triangle configuration. To solve this, a "Stop Function" is implemented. The "Stop Function" takes the absolute value of the attractive potential function per robot 2.4a and when this value is below a set tolerance = 0.02, the linear and angular velocity of that robot are set to 0.

2.2 Stability Analysis

The stability of the system is analyzed using the Lyapunov Theory and plotting the potential functions.

The first step is to determine if the functions V_i with $i = \{A, B, C\}$ are positive definite and radially unbounded. The function V_i^{att} in the numerator is a sum of squared norms, which makes it positive definite and radially unbounded. $V_i^{att} = 0$ only for $x_i - x_j = x_i^\star - x_j^\star$ and $x_i - x_k = x_i^\star - x_k^\star$. The functions $s(V_{ij}^{rep})$ and $s(V_{ik}^{rep})$ with $s(x) = tanh(kx)$ $(k > 0)$ in the denominator are always a positive value. The squared term of V_{ij}^{rep} and V_{ik}^{rep} makes their values non-negative. Furthermore, assuming that the robots cannot start at the same initial position (i.e. cannot be stacked on top of each other) and the robots have a physical body dimension (i.e. are not point objects) $x_i - x_j$ and $x_i - x_k \neq 0$. Knowing that V_{ij}^{rep} and V_{ik}^{rep} are positive values, $s(V_{ij}^{rep})$ and $s(V_{ik}^{rep})$ are also positive values with a range of $(0, 1]$. The conclusions from the numerator and denominator functions show that the functions V_i with $i = \{A, B, C\}$ are indeed positive definite and radially unbounded.

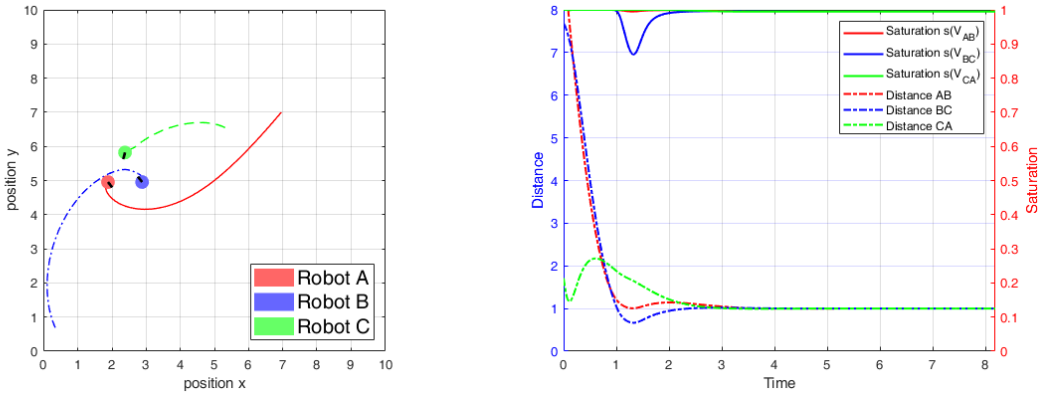
The next step is to determine if the potential functions $\dot{V}_i$ with $i = \{A, B, C\}$ are negative (semi-)definite (i.e. asymptotically converge to the desired relative positions).

$$\dot{V}_A = \frac{\partial V_A}{\partial x_A} * \frac{dx_A}{dt} + \frac{\partial V_A}{\partial x_B} * \frac{dx_B}{dt} + \frac{\partial V_A}{\partial x_C} * \frac{dx_C}{dt} = -||\frac{\partial V_A}{\partial x_A}||^2 - \frac{\partial V_A}{\partial x_B} * \frac{\partial V_B}{\partial x_B} - \frac{\partial V_A}{\partial x_C} * \frac{\partial V_C}{\partial x_C} \leq 0 \quad (2.6a)$$

$$\dot{V}_B = \frac{\partial V_B}{\partial x_A} * \frac{dx_A}{dt} + \frac{\partial V_B}{\partial x_B} * \frac{dx_B}{dt} + \frac{\partial V_B}{\partial x_C} * \frac{dx_C}{dt} = -\frac{\partial V_B}{\partial x_A} * \frac{\partial V_A}{\partial x_A} - ||\frac{\partial V_B}{\partial x_B}||^2 - \frac{\partial V_B}{\partial x_C} * \frac{\partial V_C}{\partial x_C} \leq 0 \quad (2.6b)$$

$$\dot{V}_C = \frac{\partial V_C}{\partial x_A} * \frac{dx_A}{dt} + \frac{\partial V_C}{\partial x_B} * \frac{dx_B}{dt} + \frac{\partial V_C}{\partial x_C} * \frac{dx_C}{dt} = -\frac{\partial V_C}{\partial x_A} * \frac{\partial V_A}{\partial x_A} - \frac{\partial V_C}{\partial x_B} * \frac{\partial V_B}{\partial x_B} - ||\frac{\partial V_C}{\partial x_C}||^2 \leq 0 \quad (2.6c)$$

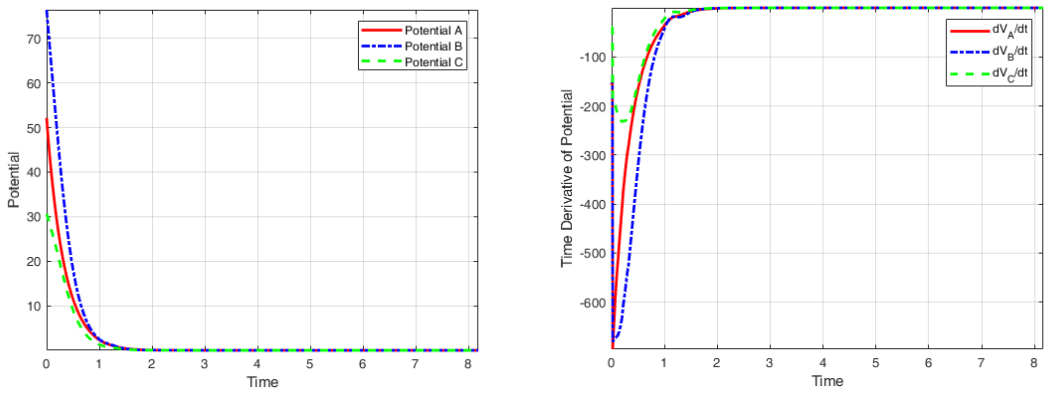
Unfortunately, due to how the potential functions are constructed, there are some cross-terms which make determining negative (semi-)definiteness very difficult. This is because of terms such as $V_i^{'att}$, $V_i^{'rep}j$ and $V_i^{'rep}k$ of which the value can be either positive or negative ($V_i^{'att}$ can also be 0). However, knowing that the range of the denominator is $(0, 1]$, the squared terms $-||\frac{\partial V_i}{\partial x_i}||^2$ will have a much smaller denominator compared to the other cross-terms. Therefore, the squared terms might be the dominant terms in the derivative equations. If this is indeed the case, The functions $\dot{V}_i$ are negative semi-definite. To confirm this, numerical simulations (see 2.3) where done and the potential functions are plotted.



(a) XY plot stability test

(b) Distance and saturation plot

Figure 2.2: First Matlab test Scenario 1



(a) Potential plot

(b) $\dot{V}_i$ plot

Figure 2.3: Stability Test $\dot{V}_i$ functions

Figure 2.3 shows an example of a simulation which plots both potential V_i and $\dot{V}_i$ functions. One could also just look at the potential plot if it is strictly decreasing, but just for clear visualization of the behavior of $\dot{V}_i$ it is also plotted in this stability test example. These plots show that the turtlebots all asymptotically stabilize towards the triangle configuration.

2.3 Numerical Simulations

Three different scenario's were tested with different Matlab scripts. The first simulation was done using the *"demo_unicycle_test6.m"* Matlab script, the second simulation uses *"triangle_formation_v1_june17th.m"* in combination with *"demo_turtlebot_simulator_v2.m"* and the third simulation uses *"triangle_formation_v1_june17th.m"* in combination with the Gazebo platform. The provided video A for the report shows the replays of the Gazebo simulations. From the three simulation environments, both the second and third simulation indeed showed a more realistic trajectory of the turtlebots compared to the first simulation. Comparing the potential graphs for simulation 1 and simulation 2, where those of simulation 1 have an almost horizontal part during the trajectory, those of simulation 2 show more clearly to be strictly decreasing. Hereby also showing that the turtlebots asymptotically converge to the triangle configuration.

2.3.1 Scenario 1

The turtlebots start from one vertical straight line. For simulation 1 and 2 it starts from initial positions of robot A: $[5, 2.5]$, robot B: $[5, 5]$ and robot C: $[5, 7.5]$.

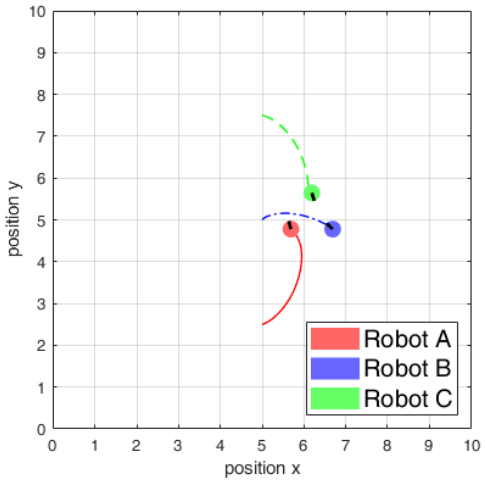
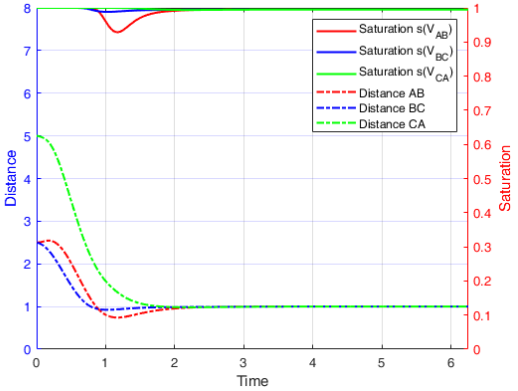
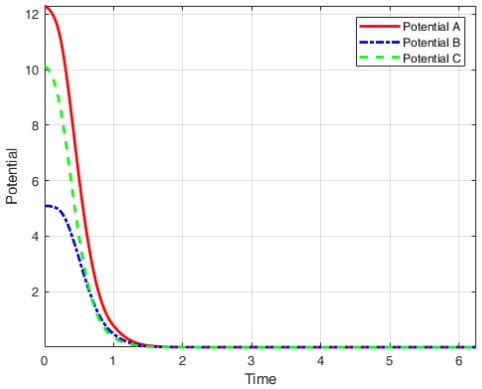


Figure 2.4: Simulation 1 XY Plot Scenario 1



(a) Saturation and Distance Plot Scenario 1



(b) Potential Plot Scenario 1

Figure 2.5: Simulation 1 of Scenario 1

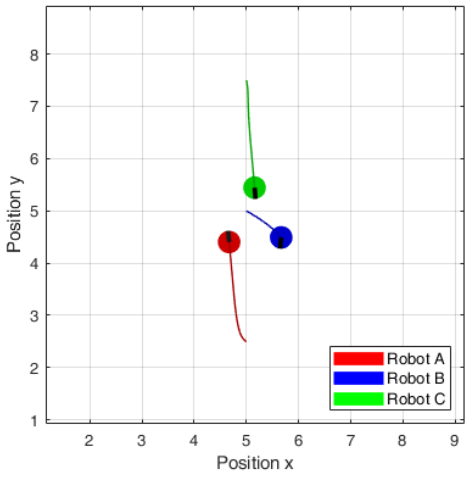
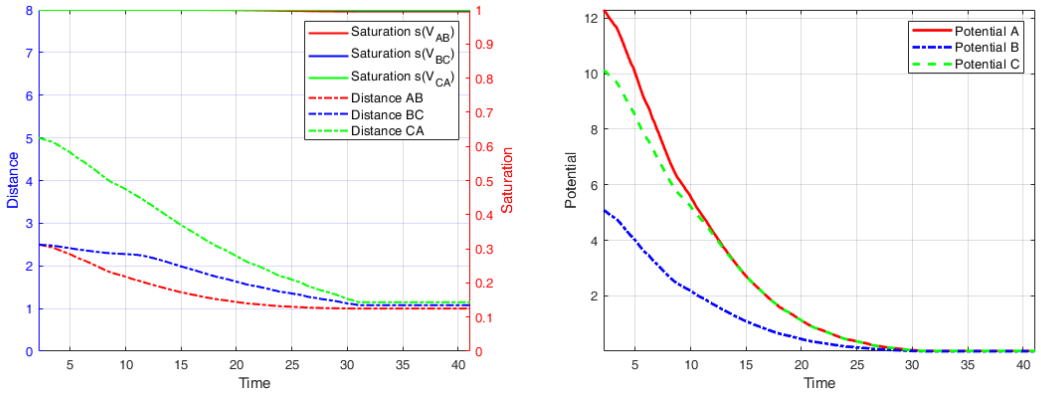


Figure 2.6: Simulation 2 (Turtlebot Simulator) XY Plot Scenario 1



(a) Saturation and Distance Plot Scenario 1

(b) Potential Plot Scenario 1

Figure 2.7: Simulation 2 (Turtlebot Simulator) of Scenario 1

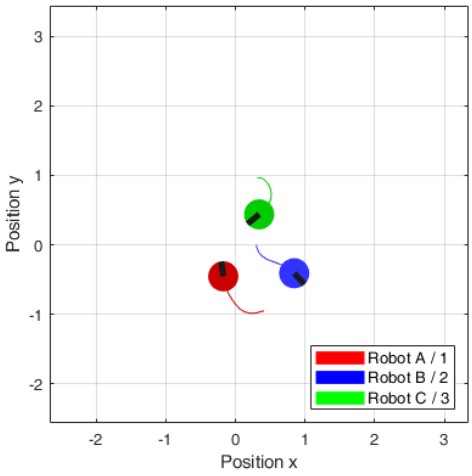


Figure 2.8: Simulation 3 (Gazebo) of Scenario 1

2.3.2 Scenario 2

The turtlebots start from triangle position, however the triangle is turned counterclockwise (compared to the final relative configuration). The initial positions for simulation 1 and 2 are robot A: $[7, 3]$, robot B: $[5, 3 + (2\sqrt{(3)})]$ and robot C: $[3, 3]$.

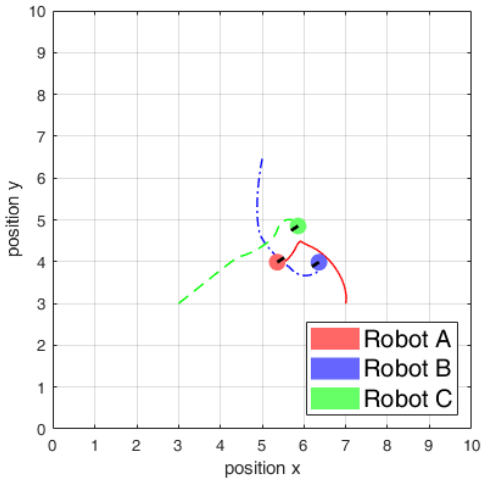
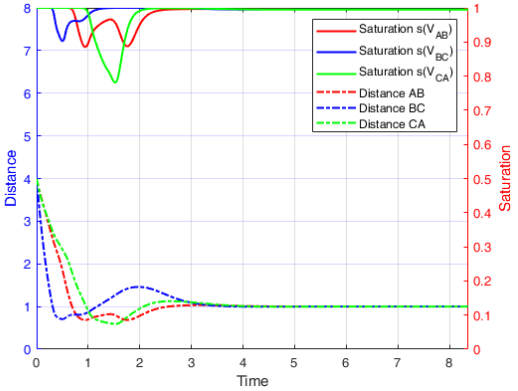
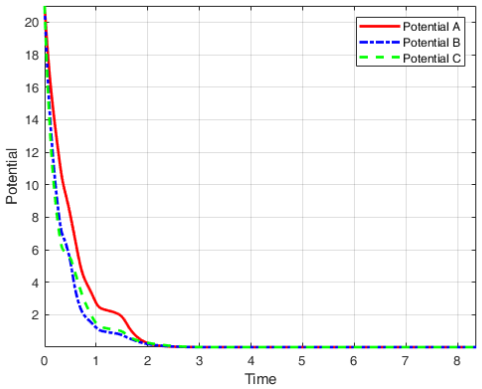


Figure 2.9: Simulation 1 XY Plot Scenario 2



(a) Saturation and Distance Plot Scenario 2



(b) Potential Plot Scenario 2

Figure 2.10: Simulation 1 of Scenario 2

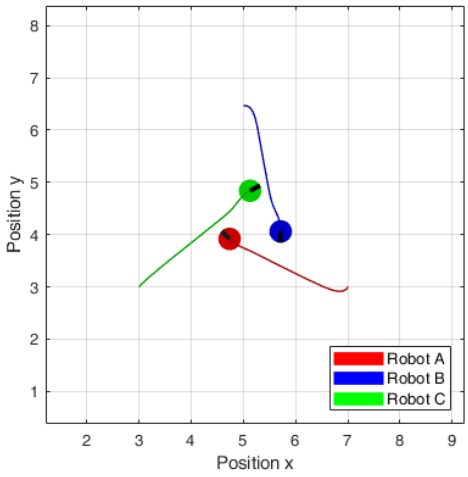
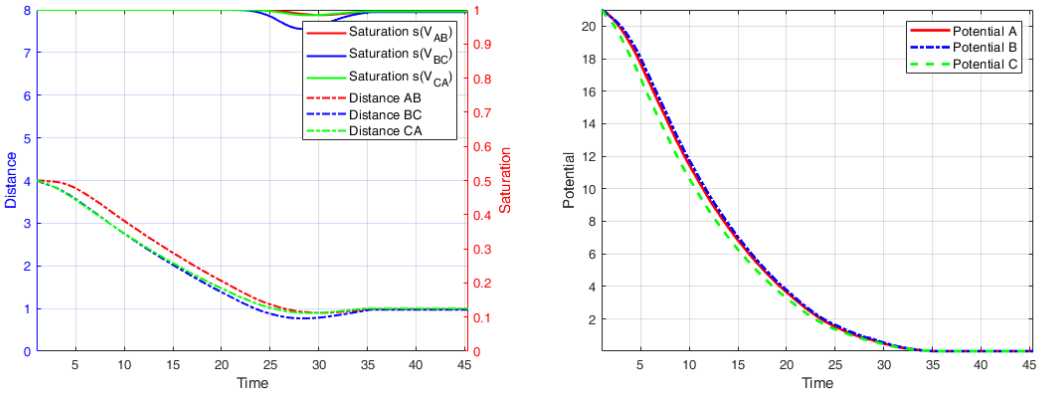


Figure 2.11: Simulation 2 (Turtlebots Simulator) XY Plot Scenario 2



(a) Saturation and Distance Plot Scenario 2

(b) Potential Plot Scenario 2

Figure 2.12: Simulation 2 (Turtlebots Simulator) of Scenario 2

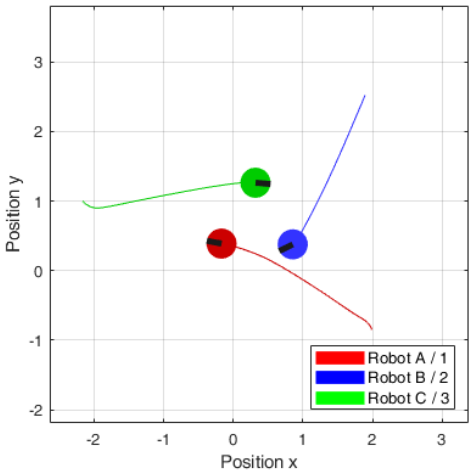


Figure 2.13: Simulation 3 (Gazebo) of Scenario 2

2.3.3 Scenario 3

The turtlebots start from a random initial position. For the Gazebo platform, it was possible to change the position of the robots randomly during the experiment itself.

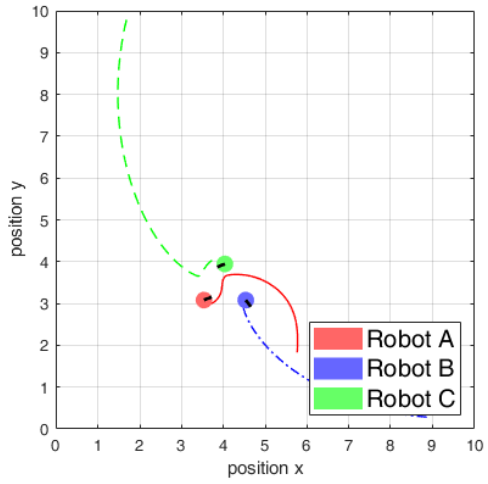
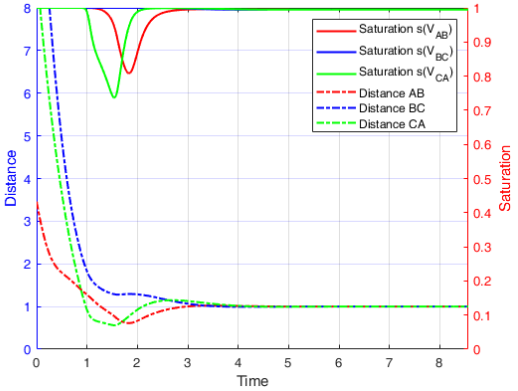
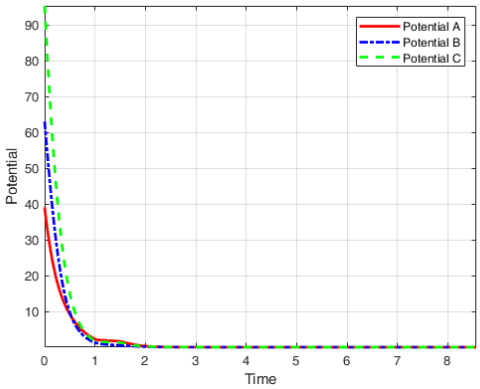


Figure 2.14: Simulation 1 XY Plot Scenario 3



(a) Saturation and Distance Plot Scenario 3



(b) Potential Plot Scenario 3

Figure 2.15: Simulation 1 of Scenario 3

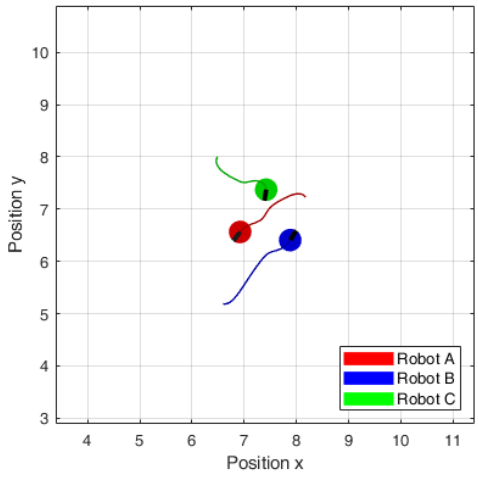
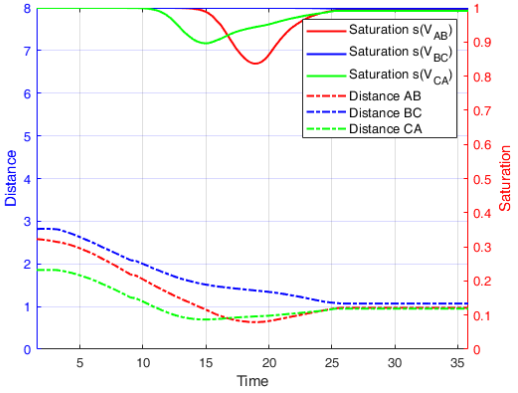
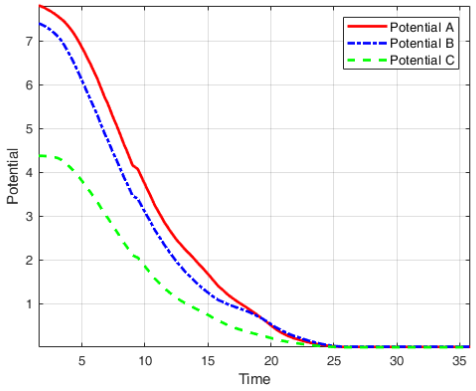


Figure 2.16: Simulation 2 (Turtlebot Simulator) XY Plot Scenario 3



(a) Saturation and Distance Plot Scenario 3



(b) Potential Plot Scenario 3

Figure 2.17: Simulation 2 (Turtlebot Simulator) of Scenario 3

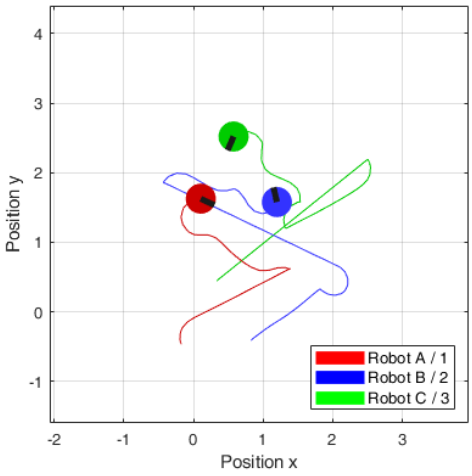


Figure 2.18: Simulation 3 (Gazebo) of Scenario 3

2.4 Experimental Demonstration

As the numerical simulations were successful, the experimental demonstrations could be conducted. For the experimental demonstration, scenario 1 and 2 were demonstrated and a 3rd scenario where the turtlebots were randomly re-allocated during the experiment. The graphs show the recorded data of each of the experiments and their past trajectory. The data can be replayed by running both *"demo_turtlebot_print_pose_2.m"* and *"demo_ros2bag_replay.m"*. The instructions on replaying the different scenario's are given in A. Furthermore, a video is provided A that shows the physical turtlebots in action. Comparing the graphs with those of the numerical analysis (specifically the Gazebo simulations) it can be seen that the past trajectory of the experimental demonstration show less 'direct' paths towards the final configuration (i.e. more curves in the path). This can indeed be expected due to potential disturbances in the physical system set-up, such as e.g. time-delays, which make the turtlebots' trajectories vary from the simulations. Nonetheless, the experimental demonstrations where successfully executed as the turtlebots converged towards the triangle configuration.

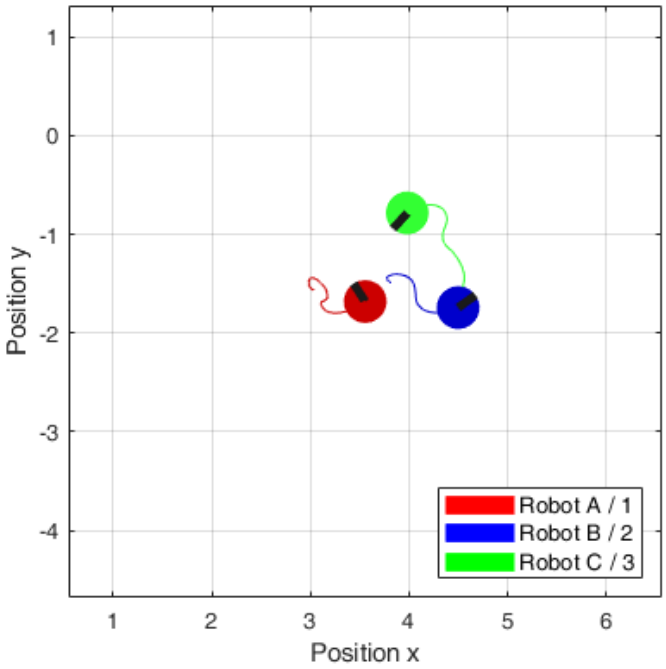


Figure 2.19: Experimental Demonstration Plot Scenario 1

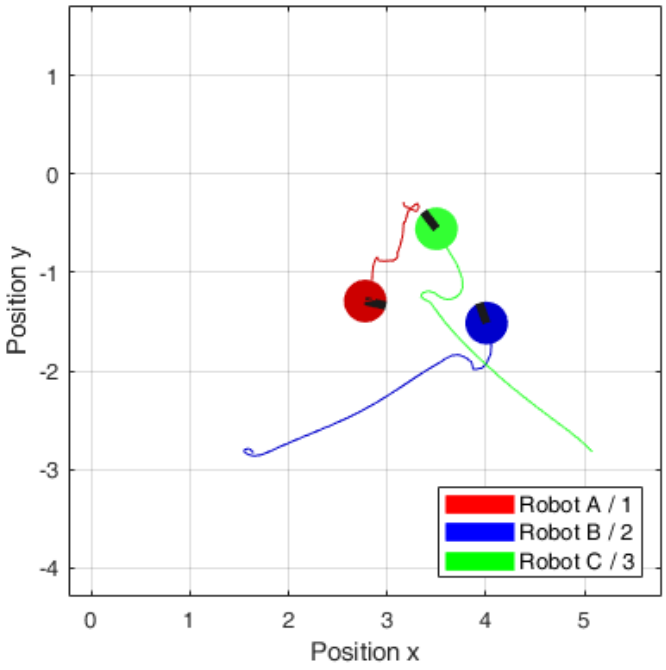


Figure 2.20: Experimental Demonstration Plot Scenario 2

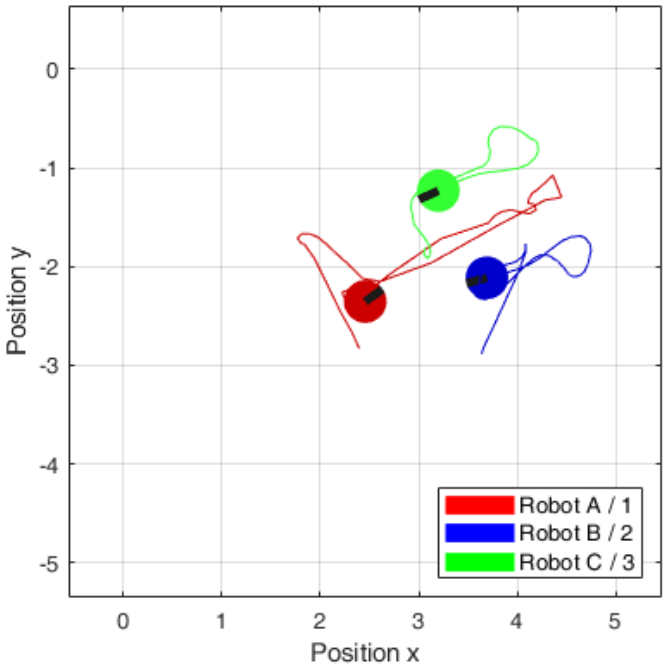


Figure 2.21: Experimental Demonstration Plot Scenario 3

A Appendix

A.1 Matlab Files

The following files (which can be found in the folder *"demo"*) are provided to demonstrate the multi-robot coordination controller design. First run `startup.m` in the main folder before running the other files.

1. `demo_unicycle_test2.m` Triangle configuration controller with specified goal position.

2. `demo_unicycle_test6.m` Triangle configuration controller with relative goal position.

3. `triangle_formation.m` Triangle configuration controller with specified goal position for ROS.

4. `triangle_formation_v1_june17th.m` Triangle configuration controller with relative goal position for ROS.

5. `demo_turtlebot_simulator.m` Given code for turtlebot simulation.

6. `demo_turtlebot_simulator_v2.m` Adjusted code from *demo_turtlebot_simulator.m* including plots and video saving functions.

7. `demo_turtlebot_print_pose.m` Given code for printing turtlebot position from replay.

8. `demo_turtlebot_print_pose_2.m` Adjusted code from *demo_turtlebot_print_pose.m* for 3 turtlebots and including visualization based on *demo_turtlebot_simulator.m* and video saving functions.

For Numerical Analysis using Gazebo, change the turtlebot names to `turtlebotName1 = 'turtlebot_red'`, `turtlebotName2 = 'turtlebot_green'` and `turtlebotName3 = 'turtlebot_blue'`.

Note that for getting the correct colors of the robots they have to be changed to `turtlebotColors = [0 1 0]*0.8, [1 0 0]*0.8, [0 0 1]*0.8 + 0.2;`. Some error occured here even though the same code was used for the other simulations.

For the Experimental Demonstrations, change the turtlebot names to `turtlebotName1 = 'turtlebot1'`, `turtlebotName1 = 'turtlebot2'` and `turtlebotName1 = 'turtlebot3'`.

9. `demo_ros2bag_replay.m` Replay recorded turtlebot(s) data.

For the Numerical Analysis using Gazebo, change the bagPath and bagTopics as follows:

bagTopics = {`"/mocap/turtlebot_red/pose"`, `"/mocap/turtlebot_green/pose"`, `"/mocap/turtlebot_blue/pose"`, `"/turtlebot_red/cmd_vel"`, `"/turtlebot_green/cmd_vel"`, `"/turtlebot_blue/cmd_vel"`};

Scenario 1: *bagPath* = `fullfile(getenv("HOME"), "Documents", "tmp", "demo_gazebo2");`

Scenario 2: *bagPath* = `fullfile(getenv("HOME"), "Documents", "tmp", "demo_gazebo1");`

Scenario 3: *bagPath* = `fullfile(getenv("HOME"), "Documents", "tmp", "demo_gazebo3");`

For the Experimental Demonstrations, change the bagPath and bagTopics as follows. *bagTopics* = {`"/mocap/turtlebot1/pose"`, `"/mocap/turtlebot2/pose"`, `"/mocap/turtlebot3/pose"`, `"/turtlebot1/cmd_vel"`, `"/turtlebot2/cmd_vel"`, `"/turtlebot3/cmd_vel"`};

Scenario 1: *bagPath* = `fullfile(getenv("HOME"), "Documents", "tmp", "demo_test1");`

Scenario 2: *bagPath* = `fullfile(getenv("HOME"), "Documents", "tmp", "demo_test6");`

Scenario 3: *bagPath* = `fullfile(getenv("HOME"), "Documents", "tmp", "demo_test5").`

If not works from the *"demo"* folder, open the one in the main file.

10. `demo_ros2bag_record.m` Record turtlebot(s) data.

11. Link to video: https://youtu.be/nut2WBpspjk. The file of the video is also provided with the deliverables.