ENAE488O Multi-Robot Swarms HW1 | Romeo Perlstein

Tuesday, March 25, 2025 11:09 AM

Q1

1) Given That:

$$\Delta(G) = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$A(G) = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$$L = \Delta(G) - A(G)$$

$$L = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ -1 & 0 & 0 & -1 & 2 \end{bmatrix}$$

Thus:

$$\dot{\vec{x}}(t) = -L\vec{x}$$

$$\begin{bmatrix} \dot{x}_1(t) \\ \dot{x}_2(t) \\ \dot{x}_3(t) \\ \dot{x}_4(t) \\ \dot{x}_5(t) \end{bmatrix} = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ -1 & 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_5(t) \end{bmatrix}$$

2) Agreement value

using

$$c = \sum_{i=1}^{N} p_i x_i(0) \quad , \quad \lambda_{1_L} = [p_1, p_2, \dots p_N]$$

Left eigenvector:

$$\boxed{C = 5.8138}$$

From propogation

$$\boxed{2.600}$$

time to agreement:

$$\boxed{t_{agree} \approxeq 6.21}$$

(rounding state to 3 decimals)

3) see matlab

4) <u>given that</u>:

$x_i \in R^2$

$$\dot{\vec{x}} = \begin{bmatrix} \dot{x}_1 & y_1 \\ \dot{x}_2 & \dot{y}_2 \\ x_3 & \dot{y}_3 \\ \dot{x}_4 & \dot{y}_4 \\ x_5 & \dot{y}_5 \end{bmatrix}$$

For $x_i \in R^N$, $N > 1$, the agreement dynamics become:

$$\dot{\vec{x}} = -(L \otimes I)\vec{x}$$

where $\otimes$ is the kronecker product.

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

<u>then</u>:

$$L \otimes I =$$

$$
\begin{bmatrix}
2 & 0 & -1 & 0 & 0 & 0 & 0 & & -1 & 0 \\
0 & 2 & 0 & -1 & & & & & & -1 \\
-1 & 0 & 2 & 0 & -1 & & & & & 0 \\
0 & -1 & 0 & 2 & 0 & -1 & & & & 0 \\
0 & & -1 & 0 & 2 & 0 & -1 & & & 0 \\
0 & & & -1 & 0 & 2 & 0 & -1 & & 0 \\
0 & & & & -1 & 0 & 2 & 0 & -1 & \\
-1 & & & & & -1 & 0 & 2 & 0 & \\
0 & -1 & 0 & 0 & 0 & 0 & -1 & 0 & 2 &
\end{bmatrix}
$$

Then finally

$$
\dot{\vec{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{y}_1 \\ \dot{y}_2 \\ \dot{y}_3 \\ \dot{y}_4 \\ y_5 \end{bmatrix}
=
\begin{bmatrix}
2 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 2 & 0 & -1 & & & & & & -1 \\
-1 & 0 & 2 & 0 & -1 & & & & & 0 \\
0 & -1 & 0 & 2 & 0 & -1 & & & & 0 \\
0 & & -1 & 0 & 2 & 0 & -1 & & & 0 \\
0 & & & -1 & 0 & 2 & 0 & -1 & & 0 \\
0 & & & & -1 & 0 & 2 & 0 & -1 & 0 \\
0 & & & & & -1 & 0 & 2 & 0 & -1 \\
-1 & & & & & & -1 & 0 & 2 & 0 \\
0 & -1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 2
\end{bmatrix}
\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix}
$$

$\vec{x}$

5) from MATLAB integration.

for node 2, node 4

Agreement value $[0, 0.8]$

Agreement value [0, 0.8]

Agreement time: 7.1750

For node 1, node 3, node 5

Agreement value: [0.8, 0]

Agreement time: 7.35

6) see MATLAB for plots

Interestingly, two of the kilobots rendezvous at one location, while the other three kilobots rendezvous at a different location, but these locations have their x and y values flipped! I'm not sure why this is the case but it's interesting that they rendezvous to these locations separately yet the locations they go to are still connected.

Seems like the kilobots that are the farthest north and south converge together, while the kilobots that are closet to the x and y axis converge together.

Q2)

1) In the sync lab, the robots attempt to synchronize their LED flashing period with the other kilobots in the network, so that each kilobot is flashing in sync (hence the name). The kilobots do this by converting their clock value to be a number bounded between 0-31 and storing it in a separate variable called modulo_clock and sending it out to the network.

When a kilobot receives a neighbors modulo_clock, it checks if its 0, and if it's not it then checks if the offset between the local modulo_clock and the neighbors modulo_clock is less than 16 (half the period). If the offset is less than 16, the offset is stored locally , however if it's greater than 0, the offset is adjusted by taking the max modulo_clock value (32) and subtracting the neighbors modulo clock, so that a kilobot will sync with its neighbor while ensuring its neighbor doesn't try to sync with the kilobot. Finally, the LED only blinks when the local modulo_clock is 0, and at the same time find the average offset from its neighbors and adjusts its own local offset!

2) Blue!

3) We know that the kilobot has received all of the code becaues it goes from slowly flashing blue, to quickly flashing blue and green, and when all of the code has been transferred the kilobot quickly blinks green