

MECH 6V29: Multiagent Robotic Systems- HW 3

Jonas Wagner

2022, March 22nd

Contents

a)	Definitions	2
Problem 1		4
a)	Big Picture Chart	4
Problem 2		5
a)	G_1	7
b)	G_2	8
c)	G_3	9
Problem 3		10
a)	10
b)	11
c)	12
Problem 4		13
Problem 5		15
a)	15
b)	15
Problem 6		16
a)	Problem	16
	i)	16
	ii)	16
b)	Network 1 Results	17
c)	Network 2 Results	18
d)	Network 3 Results	19
A MATLAB Code:		20

Preliminary Notes

a) Definitions

Definition 1. Graph $G(V, E)$ is constructed with vertex set

$$V = \{v_1, v_2, \dots, v_n\}$$

of n discrete vertices and edge set

$$E = \{e_1, \dots, e_m\} \subseteq V \times V$$

consisting of m edges $e_{k=(i,j)} = (v_i, v_j) \forall k=1, \dots, m$ connecting vertices v_i and v_j .

Definition 2. Let graph $G(V, E)$ with $V = \{v_1, \dots, v_n\}$ and $E \subseteq V \times V$.

a. $G(V, E)$ is considered undirected if

$$(v_i, v_j) \in E \iff (v_j, v_i) \in E$$

otherwise, $G(V, E)$ is considered directed.

b. An undirected graph $G(V, E)$ is considered connected if there exists a path between any two vertices.

c. A directed graph $G(V, E)$ is considered strongly connected if there exists a directed path between any two vertices.

d. A directed graph $G(V, E)$ is considered weakly connected if the corresponding undirected graph is connected.

Definition 3. Let graph $G(V, E)$ with $V = \{v_1, \dots, v_n\}$ and $E \subseteq V \times V$.

a. The degree matrix $\Delta \in \mathbb{R}^{n \times n}$ is a diagonal matrix defined as

$$\Delta := \begin{bmatrix} \deg(v_1) & & & \\ & \deg(v_2) & & \\ & & \ddots & \\ & & & \deg(v_n) \end{bmatrix}$$

b. The adjacency matrix $A \in \mathbb{R}^{n \times n}$ is a symmetric matrix ($A = A^T$) defined s.t.

$$A = [a_{ij}] : a_{ij} \begin{cases} 1 & (v_i, v_j) \in E \\ 0 & (v_i, v_j) \notin E \end{cases}$$

c. The incidence matrix $D \in \mathbb{R}^{n \times m}$ is defined as

$$D = [d_{ij}] : d_{ij} \begin{cases} 1 & (v_i, -) \in e_j \\ -1 & (-, v_i) \in e_j \\ 0 & \text{otherwise} \end{cases}$$

d. The Laplacian matrix $L \in \mathbb{R}^{n \times n}$ is a symmetric ($L = L^T$) and strictly semi-positive definite ($L \succeq 0$) is defined as

$$L := \Delta - A = DD^T$$

and

$$L = \begin{bmatrix} \deg(v_1) & -a_{12} & -a_{13} & \cdots & -a_{1n} \\ -a_{21} & \deg(v_2) & -a_{23} & \cdots & -a_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -a_{n1} & -a_{n2} & -a_{n3} & \cdots & \deg(v_n) \end{bmatrix}$$

e. For a weighted graph $G(V, E, W)$, the diagonal weighted matrix $W \in \mathbb{R}^{m \times m}$ is defined as

$$W = [w_{ij}] \forall_{ij \in E}$$

where w_{ij} are the corresponding weights for $e_{ij} = (v_i, v_j)$.

Definition 4. Let undirected and unweighted graph $G(V, E)$ with $V = \{v_1, \dots, v_n\}$ and $E \subseteq V \times V$. The consensus dynamics of network $G(V, E)$ is defined by

$$\forall_{i=1, \dots, n} \dot{x}_i = \sum_{j \in \mathcal{N}_i} (x_j - x_i) \iff \dot{x} = -Lx$$

For the case with weighted graph $G(V, E, W)$ with diagonal weight matrix $W = [w_{ij}]$, weighted consensus dynamics are given as

$$\dot{x}_i = \sum_{j \in \mathcal{N}_i} w_{ij} (x_j - x_i) \implies \dot{x} = -L_w x$$

where weighted Laplacian matrix L_w is defined as

$$L_w = DW D^T$$

Problem 1

State a summary of **Notes 11, 12 and 13**, preferably by creating a concept map diagram (flow diagram). The whole purpose is to make sure that we are clear about the bigger picture, and reiterate why are we doing and discussing the specific topics in the class. Do not merely write the topics, instead create connections between topics to clarify the flow of information.

Note: Since we have only covered Notes 11 and 12, this chart only includes those topics.

a) Big Picture Chart

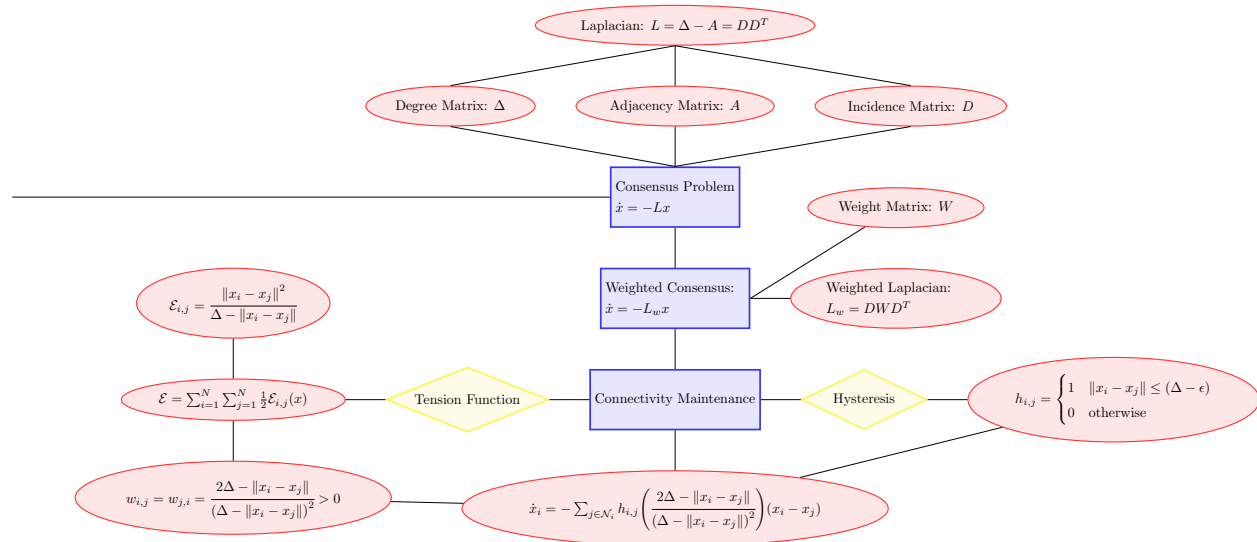


Fig. 1: Diagram of Course Topics (created w/ TikZ)

TO DO: Need to add all the Rigidity and maintaining formation stuff (refer to definitions from Problem 2)

Problem 2

Preliminaries

Definition 5. Consider a collection of N robots. Formation graph $G(V, E_f, \omega)$ consists of vertex set

$$V = \{v_1, v_2, \dots, v_N\}$$

of N vertices v_i associated with robot i , edge set E_f

$$E_f = \{e_1, \dots, e_m\} \subseteq V \times V$$

of m edges $e_{k=(i,j)} = (v_i, v_j)$ that indicate knowledge of the distance between robots v_i and v_j , and $\omega : E_f \rightarrow \mathbb{R}_+$ which associates a feasible desired inter-agent distance to each pair in E_f .

Definition 6. Consider a collection of N robots connected in formation graph $G(V, E_f, \omega)$. Let each robot be located at position $P_i \in \mathbb{R}^d$ within euclidean space $(\mathbb{R}^d, \|\cdot\|_2)$.

- a. The formation position set is defined as the collection of associated robot positions

$$P = \{P_1, \dots, P_N\} \subseteq \mathbb{R}^{d \times N}$$

- b. The set of pair-wise inter-robot distances is defined by

$$D = \{d_{ij} \geq 0 : d_{ij} = d_{ji}, \forall i, j \in \{1, \dots, N\}\}$$

where d_{ij} is the distance $\|P_i - P_j\|$.

- c. D is considered feasible if

$$\exists P_1, \dots, P_N \in \mathbb{R}^d : \|P_i - P_j\| = d_{ij} \forall i, j \in \{1, \dots, N\}$$

- d. A framework (G_f, P) is a combination of a formation graph G_f and a set of feasible points P .

- e. Framework (G_f, P) is considered generic if P is algebraically independent over \mathbb{Q} . (i.e. that the points are not collinear in 2-D or co-planer in 3-D)

Definition 7. Consider frameworks (G, P_0) and (G, P_1) .

- a. (G, P_0) and (G, P_1) are equivalent if

$$\|P_0(i) - P_0(j)\| = \|P_1(i) - P_1(j)\| \forall (i,j) \in E_f$$

meaning $d_{ij}^{(0)} = d_{ij}^{(1)}$ for the vertices that are neighbors.

- b. (G, P_0) and (G, P_1) are congruent if

$$\|P_0(i) - P_0(j)\| = \|P_1(i) - P_1(j)\| \forall (i,j) \in V \times V$$

meaning $d_{ij}^{(0)} = d_{ij}^{(1)}$ for every vertex.

- c. (G, P_0) is Globally Rigid if

$$(G, P_1) \text{ equivalent } (G, P_0) \implies (G, P_1) \text{ congruent } (G, P_0)$$

d. (G, P_0) is rigid if

$$\exists_{\epsilon > 0} \forall_{P_1} : (G, P_1) \text{ equivalent } (G, P_0) \wedge \forall_{i \in V} \|P_0(i) - P_1(i)\| < \epsilon \implies (G, P_1) \text{ congruent } (G, P_0)$$

Remark: A framework being rigidity is equivalent to saying that every continuous motion maintaining distances where edges exist also maintains the distances between all other vertex pairs.

Theorem 1. Let (G, P_0) be a d -dimensional generic framework.

Definition 8. The Rigidity Matrix is defined by the equations

$$(x_i - x_j)^T (\dot{x}_i - \dot{x}_j) = 0 \quad \forall_{i,j \in E}$$

resulting in a matrix $R(P_0)$ so that $R(P_0)\dot{P} = 0$.

Rigidity Test: (G, P_0) is rigid if and only if

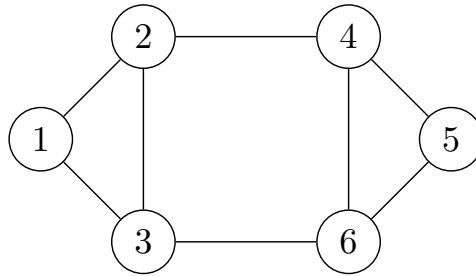
$$\text{rank}(R(P_0)) = \begin{cases} 2N - 3 & d = 2 \\ 3N - 6 & d = 3 \end{cases}$$

Proof. (see lecture notes) Essentially just a constructive proof to get $\frac{d}{dt}d_{ij} = 0 \forall_{i,j \in V}$ □

Additionally, the rank of the rigidity matrix remains the same for all generic realizations, thus G can be called Generically Rigid if any feasible generic realization is rigid.

Consider a two-dimensional plane ($d = 2$). Which of the following graphs are (generically) ridge? Use the rigidity test (based on the rank of the rigidity matrix) to support your answer.

a) G_1


$$R(P_0) = - \begin{bmatrix} -1 & 1 & & & & & & & & & \\ -1 & 1 & & & & & & & & & \\ -1 & & 1 & & & & & & & & \\ 1 & & -1 & & & & & & & & \\ & 0 & 0 & & & & & & & & \\ & 2 & -2 & & & & & & & & \\ -2 & & 2 & & & & & & & & \\ 0 & & 0 & & & & & & & & \\ & & 1 & -1 & & & & & & & \\ & & -1 & 1 & & & & & & & \\ & & & -1 & 1 & & & & & & \\ & & & -1 & 1 & & & & & & \\ & 2 & & & -2 & & & & & & \\ 0 & & & & 0 & & & & & & \\ & & 0 & & 0 & & & & & & \\ & & -2 & & 2 & & & & & & \end{bmatrix} \implies \text{rank}(R(P_0)) = 8 < 9 = 2N - 3 \implies G_1 \text{ is not rigid}$$

b) G_2

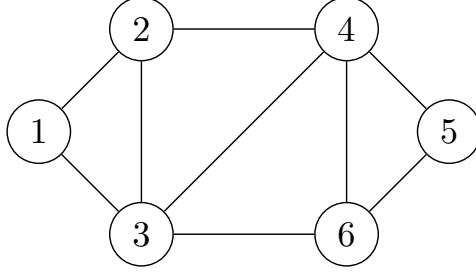


Fig. 3: Graph G_2

$$E = \{(1, 2), (1, 3), (2, 3), (2, 4), (4, 5), (5, 6), (3, 6), (4, 6), (3, 4)\}$$

$$P_0 = \{(0, 0), (1, 1), (1, -1), (3, 1), (4, 0), (3, -1)\}$$

$$R(P_0) = \begin{bmatrix} x_1 - x_2 & -(x_1 - x_2) & 0 & 0 & 0 & 0 \\ (x_1 - x_3) & 0 & -(x_1 - x_3) & 0 & 0 & 0 \\ 0 & (x_2 - x_3) & -(x_2 - x_3) & 0 & 0 & 0 \\ 0 & (x_2 - x_4) & 0 & -(x_2 - x_4) & 0 & 0 \\ 0 & 0 & 0 & (x_4 - x_5) & -(x_4 - x_5) & 0 \\ 0 & 0 & (x_3 - x_6) & 0 & 0 & -(x_3 - x_6) \\ 0 & 0 & 0 & (x_4 - x_6) & 0 & -(x_4 - x_6) \\ 0 & 0 & (x_3 - x_4) & -(x_3 - x_4) & 0 & 0 \end{bmatrix}$$

$$R(P_0) = - \begin{bmatrix} -1 & 1 & & & & & & & & & \\ -1 & 1 & & & & & & & & & \\ -1 & & 1 & & & & & & & & \\ 1 & & -1 & & & & & & & & \\ & 0 & 0 & & & & & & & & \\ & 2 & -2 & & & & & & & & \\ -2 & & 2 & & & & & & & & \\ 0 & & 0 & & & & & & & & \\ & & 1 & -1 & & & & & & & \\ & & -1 & 1 & & & & & & & \\ & & & -1 & 1 & & & & & & \\ & & & -1 & 1 & & & & & & \\ & & 2 & & -2 & & & & & & \\ & & 0 & & 0 & & & & & & \\ & & & 0 & 0 & & & & & & \\ & & & -2 & 2 & & & & & & \\ & & 2 & -2 & & & & & & & \\ & & 2 & -2 & & & & & & & \end{bmatrix} \implies \text{rank}(R(P_0)) = 9 = 2N - 3 \implies G_2 \text{ is rigid}$$

c) G_3

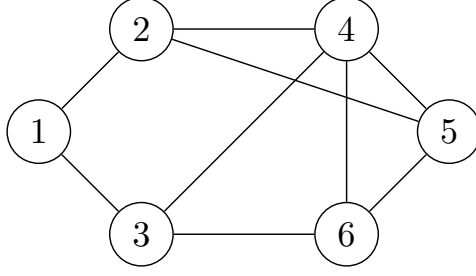


Fig. 4: Graph G_3

$$E = \{(1, 2), (1, 3), (2, 4), (4, 5), (5, 6), (3, 6), (4, 6), (3, 4), (2, 5)\}$$

$$P_0 = \{(0, 0), (1, 1), (1, -1), (3, 1), (4, 0), (3, -1)\}$$

$$R(P_0) = \begin{bmatrix} (x_1 - x_2) & -(x_1 - x_2) & 0 & 0 & 0 & 0 \\ (x_1 - x_3) & 0 & -(x_1 - x_3) & 0 & 0 & 0 \\ 0 & (x_2 - x_4) & 0 & -(x_2 - x_4) & 0 & 0 \\ 0 & 0 & 0 & (x_4 - x_5) & -(x_4 - x_5) & 0 \\ 0 & 0 & (x_3 - x_6) & 0 & 0 & -(x_3 - x_6) \\ 0 & 0 & 0 & (x_4 - x_6) & 0 & -(x_4 - x_6) \\ 0 & 0 & (x_3 - x_4) & -(x_3 - x_4) & 0 & 0 \\ 0 & (x_2 - x_5) & 0 & 0 & -(x_2 - x_5) & 0 \end{bmatrix}$$

$$R(P_0) = - \begin{bmatrix} -1 & 1 & & & & & & & & & \\ -1 & 1 & & & & & & & & & \\ -1 & & 1 & & & & & & & & \\ 1 & & -1 & & & & & & & & \\ & -2 & & 2 & & & & & & & \\ & 0 & & 0 & & & & & & & \\ & & 1 & -1 & & & & & & & \\ & & -1 & 1 & & & & & & & \\ & & & -1 & 1 & & & & & & \\ & & & -1 & 1 & & & & & & \\ & & 2 & & -2 & & & & & & \\ & & 0 & & 0 & & & & & & \\ & & & 0 & 0 & & & & & & \\ & & & -2 & 2 & & & & & & \\ & & 2 & -2 & & & & & & & \\ & & 2 & -2 & & & & & & & \\ & 3 & & -3 & & & & & & & \\ -1 & & & 1 & & & & & & & \end{bmatrix} \implies \text{rank}(R(P_0)) = 9 = 2N - 3 \implies G_3 \text{ is rigid}$$

Problem 3

We can “grow” a minimally rigid graph by adding nodes one by one using Henneberg construction.

Preliminaries

Definition 9. G_f is considered minimally rigid if it is rigid and the removal of any single edge renders it not rigid.

Theorem 2. G_f is minimally rigid if and only if it is rigid and contains

$$\begin{cases} 2N - 3 \text{ edges} & d = 2 \\ 3N - 6 \text{ edges} & d = 3 \end{cases}$$

a)

Problem:

Briefly state the two rules of Henneberg construction (vertex addition and vertex split) by consulting pages 16 and 17 of Notes 12.3 uploaded at the elearning page. You can also consult page 6 of the reference [1] (which is also uploaded at the elearning course page). [1] B. Anderson, et al. ”Rigid graph control architectures for autonomous formations.” IEEE Control Systems Magazine 28.6 (2008): 48-63.

Solution:

Given a seed graph that is already minimally rigid, you “grow” the graph in two steps:

a. **Vertex Addition:**

- Starts as $(i, j) \in E$
- Add new vertex v
- Connect v to two adjacent vertices i and j
- Ends with $(i, j), (v, i), (v, j) \in E$

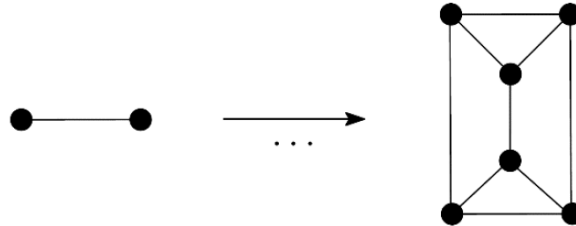
b. **Edge Splitting:**

- Starts as $(i, j), (j, k) \in E$
- Add new vertex v
- Connect v to vertices i, j , and k (i.e.) add $(v, i), (v, j), (v, k)$
- Remove one of the edges that previously existed (i.e.) remove either (i, j) or (j, k) .
- Ends with $(i, j), (v, i), (v, j), (v, k) \in E$ or $(j, k), (v, i), (v, j), (v, k) \in E$

b)

Problem:

Starting with the seed graph on the left, obtain the minimally rigid graph on the right by applying the Henneberg construction rules. Clearly state the rule used in each step.



Solution:

Note: Due to a lack of time this week, hand drawings were used. If too difficult to view or explanation is lacking then I can share the TikZ version in the future.

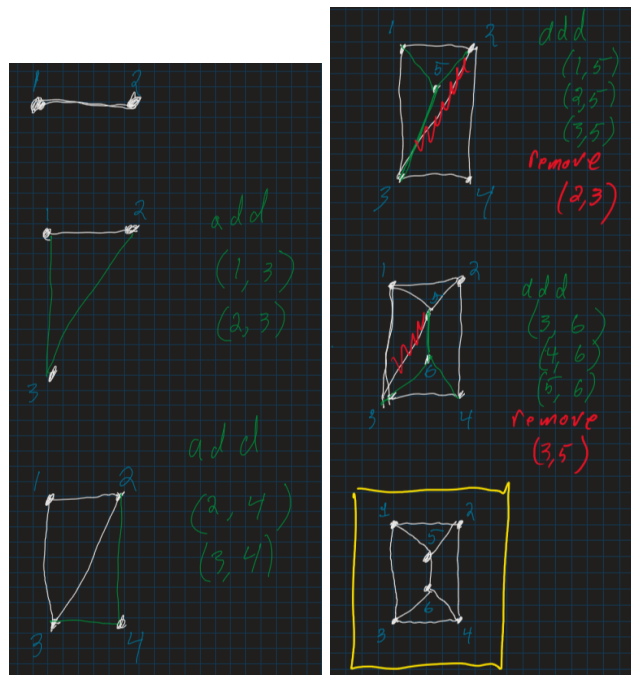
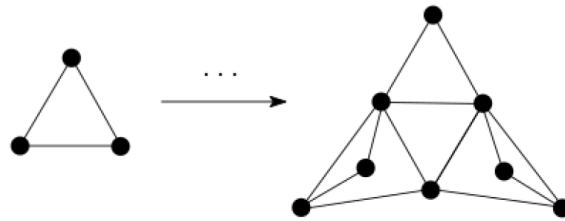


Fig. 5: Hand Drawings of Henneberg Construction method for Problem 3b

c)

Problem:

Repeat the same problem (as in part (B)) for the following:



Solution:

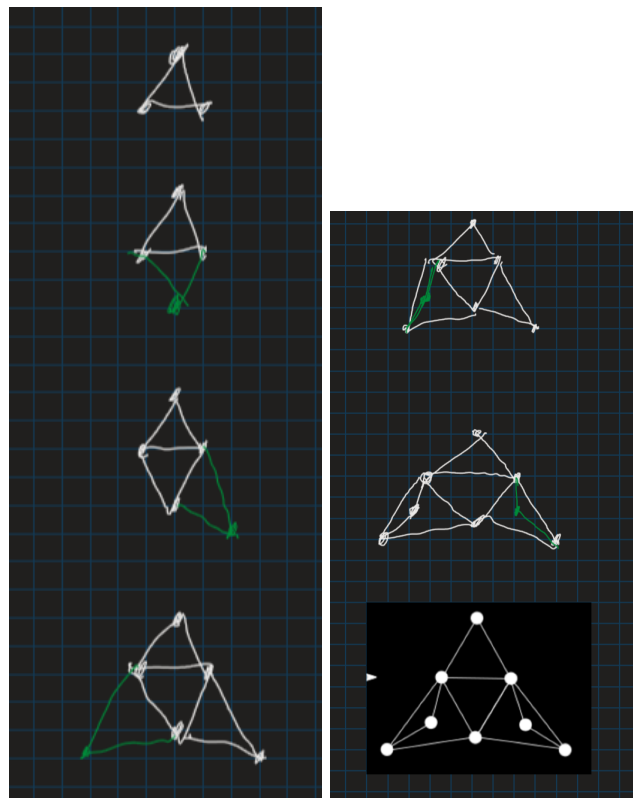


Fig. 6: Hand Drawings of Henneberg Construction method for Problem 3c

Problem 4

Preliminaries

Definition 10. A graph $G = (V, E)$ is considered a Tree if it satisfies any of the following equivalent conditions.

- G is connected and acyclic (contains no cycles).
- G is acyclic, and a cycle is formed if any edge is added.
- G is connected, but would be disconnected if any edge is removed.
- Any two vertices in G can be connected by a unique simple path.

If G is a finite graph with N vertices, then these additional conditions are also equivalent:

- G is connected and has $N - 1$ edges.
- G has no simple cycles and has $N - 1$ edges.

Problem

Let's consider two trees $T_1 = (V, E_1)$ and $T_2 = (V, E_2)$ that have the same vertex sets containing N vertices, but possibly different edge sets. Is it possible that the union of two such trees result in a minimally rigid graph? If yes, explain and construct an example. If not, explain why not.

Solution

By definition, a tree will have exactly $N - 1$ edges. One necessary condition for a minimally rigid graph, when $d = 2$, is to have $2N - 3$ edges, which is possible as $T_1 \cup T_2$ would contain $\leq 2N - 2$ edges. Additionally, within all connected graphs there exists induced subgraphs that are trees, so in many¹ minimally rigid graphs there will be two distinct trees that can be defined to be T_1 and T_2 .

Regardless, the answer is **YES**. See Fig.7, Fig.8, and Fig.9 for an example.

¹mabye all, but I don't have a proof for that

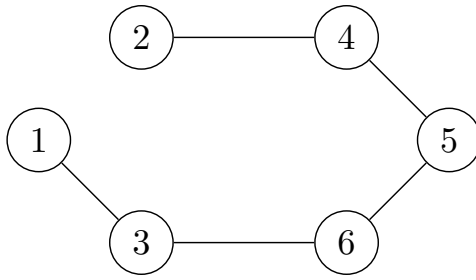


Fig. 7: Graph T_1

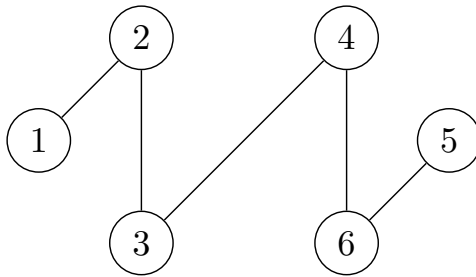


Fig. 8: Graph T_2

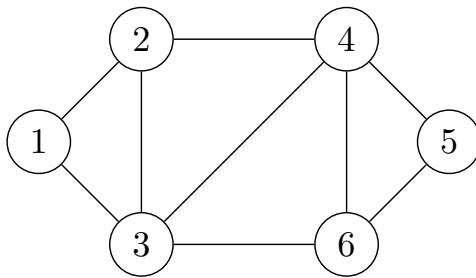


Fig. 9: Graph $T_1 \cup T_2$

Problem 5

In class, we saw that if we place a total of $(P + 2)$ agents on a line, with P agents at location 0, one agent at location Δ , and one agent at location 2Δ , then if the network is a Δ -disk proximity graph it will get disconnected under the linear consensus equation if $P > 2$.

a)

Problem

Show that this particular configuration is indeed the worst-case scenario from a connectivity point-of-view.

Solution

As far as the definition of the Δ -disk proximity graph goes, the positioning of v_1 at 2Δ and v_2 at Δ places them at the exact point of disconnection. Additionally, having the points $v_{i=3,\dots,2+P}$ at 0 places them at the extreme to maximize the consensus ‘pull’ on v_2 . This results in the worst-case scenario where v_2 is ‘pulled’ as hard as allowed by the Δ -disk proximity graph definition, while the v_1 and v_2 edge is already at it’s ‘breaking point’.

b)

Problem

Is the following theorem valid? A Δ -disk proximity graph network of 4 agents will always stay connected under the consensus equation if it starts connected.

Solution

Technically yes. Since in this case $P = 2$, $\dot{x}_2 = (1 - 2)\Delta = -\Delta$ (which is equal to $\dot{x}_1 = -\Delta$) and therefore they will maintain a distance smaller than Δ and remain connected.

Problem 6

See attached MATLAB File for code.

a) Problem

i)

Make sure that the network stays connected for all three initial networks. (I am not looking for a proof here - just try and be clever and produce some appropriate control law that works in simulation.)

Proposed Solution: No need to be clever since the edge-tension algorithm is already implemented which guarantees that once a connection exists it will not disappear. The Hysteresis that is implemented does make it so that it isn't great near the outskirts but it works regardless.

An option (as tested) to improve the far away ones without making Hysteresis infinite, is to add a standard baseline weight if an edge exists but is still out of range. (i.e)

```
w = weight(X(:,j), X(:,i)) * H(i, j);  
w = w + 1 * not(H(i,j)); % w = 1 if not below epsilon
```

ii)

Make sure that the agents do not run into each other, i.e., that collisions are avoided, at the same time as the network remains connected. Again, I do not want a proof I want the simulations to work

Proposed Solution: The edge-tension algorithm also already takes care of this for the most part. An additional modification of the Hysteresis function component can allow for more of a 'motivator' to force the agents to stay away from each other.

```
if dist(X(:,j), X(:,i)) < dist_goal  
    H(i, j) = 1 / dist(X(:,j), X(:,i));  
else  
    H(i,j) = 1;  
end
```


b) Network 1 Results


Proximity Disk Graph

Delta 
network_no

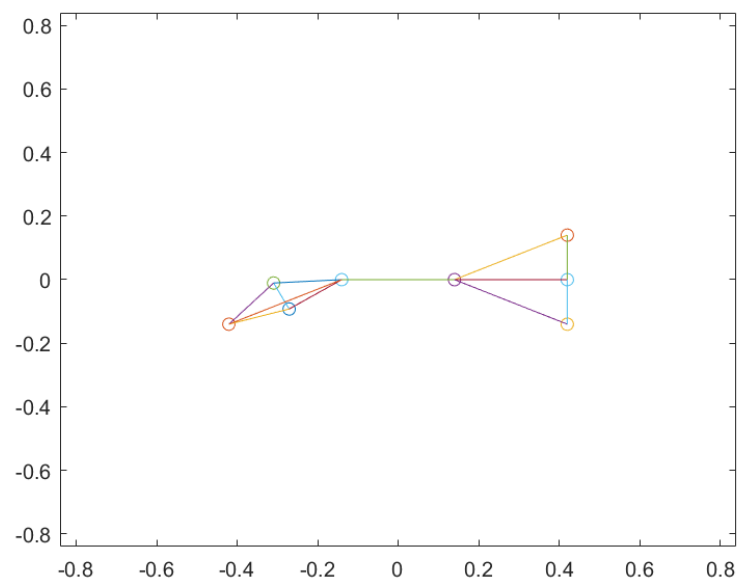
Some numerical integration parameters

dt
Tf

Weight Function Controls

dist_goal
w_min 
epsilon 


Run Simulation and Plot



dmin = 0.0720

c) Network 2 Results

Proximity Disk Graph

Delta 
network_no

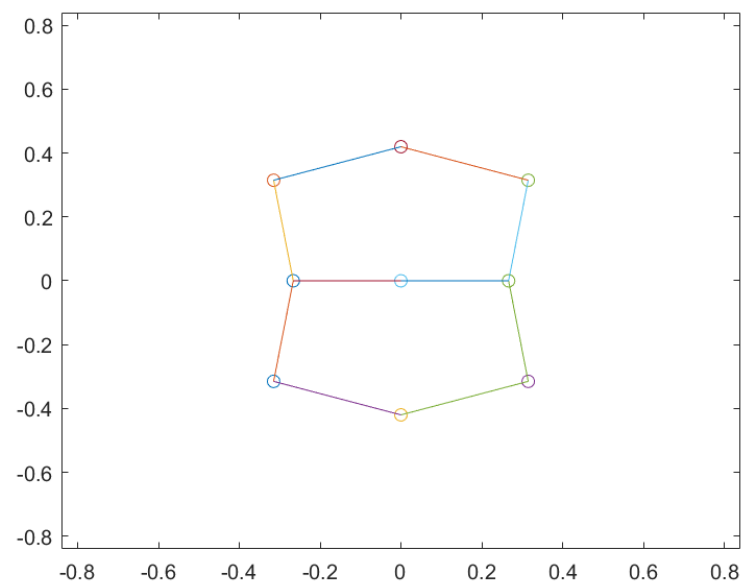
Some numerical integration parameters

dt
Tf

Weight Function Controls

dist_goal
w_min 
epsilon 

Run Simulation and Plot



dmin = 0.2665

d) Network 3 Results

Proximity Disk Graph

Delta 

network_no 


Some numerical integration parameters


dt

Tf

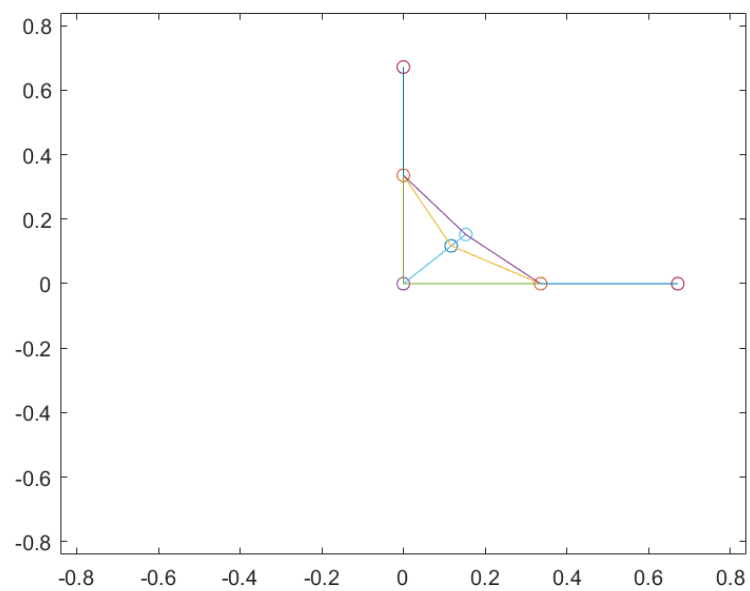
Weight Function Controls

dist_goal

w_min 

epsilon 

Run Simulation and Plot



dmin = 0.0493

A MATLAB Code:

All code I write in this course can be found on my GitHub repository:

https://github.com/jonaswagner2826/MECH6V29_MultiagentRoboticSystems