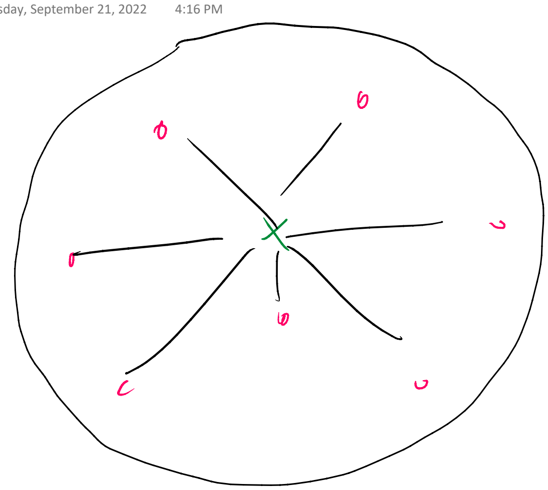
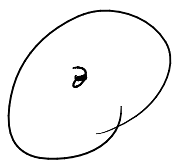
Flocking code

Wednesday, September 21, 2022 4:16 PM



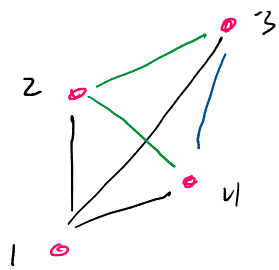
1. ESTABLISH GRAPH
2. DRIVE TOWARDS MIDPOINT OF NEAREST NEIGHBOR



RANGE

<u>HOW TO ESTABLISH GRAPH?</u>

- NEED TO KNOW SENSOR RANGE
- IF OTHER ROBOT IN SENSOR RANGE ESTABLISH EDGE BETWEEN
- EACH ROBOT NEEDS <u>UNIQUE ID</u>

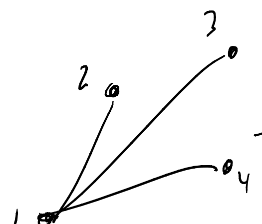


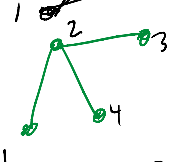
GRAPH 1

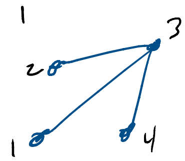
GRAPH 2

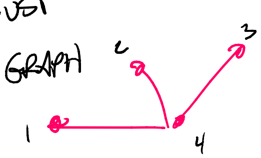
GRAPH 3

GRAPH 4









EVERY ROBOT MUST HAVE ITS OWN GRAPH BECAUSE THERE IS NO OVERSEER.

- EVERY ROBOT HAS ITS OWN GRAPH
- WHERE TO DRIVE TO?

↳ CONSENSUS EQUATION

$$\dot{x} = -Lx \qquad L \in \mathbb{R}^{N \times N}$$

PASS DEGREE MATRIX AND ADJACENCY MATRIX EVERY COMMS TICK...

DO COMMUNICATIONS NEED TO TICK? IS THERE SUCH THING AS CONTINUOUS CASE TALKING?

Laplacian matrix [edit]

Given a simple graph G with n vertices $v_1, \ldots, v_n$, its Laplacian matrix $L_{n \times n}$ is defined element-wise as[1]

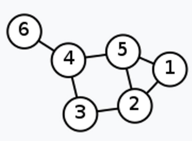
$$L_{i,j} := \begin{cases} \deg(v_i) & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and } v_i \text{ is adjacent to } v_j \\ 0 & \text{otherwise,} \end{cases}$$

or equivalently by the matrix

$$L = D - A,$$

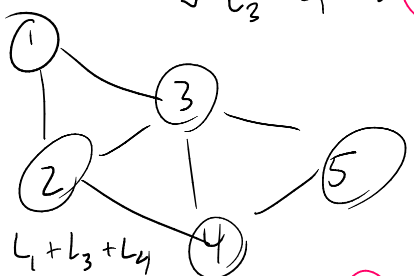
where D is the degree matrix and A is the adjacency matrix of the graph. Since G is a simple graph, A only contains 1s or 0s and its diagonal elements are all 0s.

Here is a simple example of a labelled, undirected graph and its Laplacian matrix.

Labelled graph	Degree matrix	Adjacency matrix	Laplacian matrix
	$\begin{pmatrix} 2&0&0&0&0&0 \\ 0&3&0&0&0&0 \\ 0&0&2&0&0&0 \\ 0&0&0&3&0&0 \\ 0&0&0&0&3&0 \\ 0&0&0&0&0&1 \end{pmatrix}$	$\begin{pmatrix} 0&1&0&0&1&0 \\ 1&0&1&0&1&0 \\ 0&1&0&1&0&0 \\ 0&0&1&0&1&1 \\ 1&1&0&1&0&0 \\ 0&0&0&1&0&0 \end{pmatrix}$	$\begin{pmatrix} 2&-1&0&0&-1&0 \\ -1&3&-1&0&-1&0 \\ 0&-1&2&-1&0&0 \\ 0&0&-1&3&-1&-1 \\ -1&-1&0&-1&3&0 \\ 0&0&0&-1&0&1 \end{pmatrix}$

We observe for the undirected graph that both the adjacency matrix and the Laplacian matrix are symmetric, and that row- and column-sums of the Laplacian matrix are all zeros.

$L_1 \rightarrow L_3 = L_1 + L_2 + L_4 + L_5$



$L_5 = L_4 + L_3$

$L_2 = L_1 + L_3 + L_4$

$L_4 = L_2 + L_3 + L_5$

SO THE PROBLEM HERE IS THAT WE GET DUPLICATE INFORMATION.

WE CAN'T CONSTRUCT AN ENTIRE MAP... RIGHT?

YES BECAUSE EACH UNIT HAS A UNIQUE ID.

→ THIS "DUPLICATE" DATA ACTUALLY RESULTS IN LARGER MATRICES WITH MORE UNIQUE DATA RATHER THAN CONVOLUTED INFORMATION

$p = 2$ IS $p > 1$

$\rightarrow \dot{x} = -(L \otimes I_p)x$

$\vec{x}$ = POSITIONS | $x_{i,j} \in \mathbb{R}^2$

(1) ESTABLISH UNIQUE ID'S → (2) ESTABLISH ADJACENCY DEGREE MATRICES → (3) BEHAVIOR

ON INITIALIZATION

(1) CHECK SENSORS (SENSOR RANGE)
(2) FIND NEAREST NEIGHBOR
(3) ESTABLISH LINK (WRITE MATRICES)
(4) BROADCAST COMPILED L MATRIX

FLOCKING:
(1) $\dot{\phi} = -L\phi$
(2) DRIVE FORWARD

OFFSET:

(3) BIHEUR…

(4) BROADCAST COMPILED L MATRIX

OFFSET:

$$\dot{x}_i = R(-\phi)(x_{i+1} - x_i)$$

THIS

WHAT THE HECK DOES THIS MEAN IF A ROBOT HAS 2+ EDGE CONNECTIONS?

https://mathworld.wolfram.com/KroneckerProduct.html

Kronecker Product

Download Wolfram Notebook

Given an $m \times n$ matrix A and a $p \times q$ matrix B, their Kronecker product $C = A \otimes B$, also called their matrix direct product, is an $(m\,p) \times (n\,q)$ matrix with elements defined by

$$c_{\alpha\beta} = a_{ij} b_{kl}, \tag{1}$$

where

$$\alpha \equiv p(i-1)+k \tag{2}$$

$$\beta \equiv q(j-1)+l. \tag{3}$$

For example, the matrix direct product of the 2×2 matrix A and the 3×2 matrix B is given by the following 6×4 matrix,

$$A \otimes B = \begin{bmatrix} a_{11} B & a_{12} B \\ a_{21} B & a_{22} B \end{bmatrix} \tag{4}$$

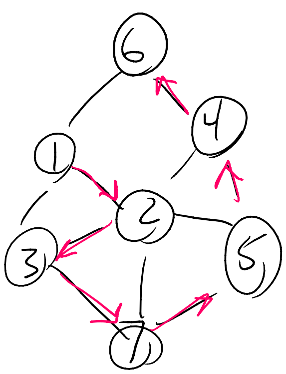
$$= \begin{bmatrix} a_{11} b_{11} & a_{11} b_{12} & a_{12} b_{11} & a_{12} b_{12} \\ a_{11} b_{21} & a_{11} b_{22} & a_{12} b_{21} & a_{12} b_{22} \\ a_{11} b_{31} & a_{11} b_{32} & a_{12} b_{31} & a_{12} b_{32} \\ a_{21} b_{11} & a_{21} b_{12} & a_{22} b_{11} & a_{22} b_{12} \\ a_{21} b_{21} & a_{21} b_{22} & a_{22} b_{21} & a_{22} b_{22} \\ a_{21} b_{31} & a_{21} b_{32} & a_{22} b_{31} & a_{22} b_{32} \end{bmatrix}. \tag{5}$$

DO I NEED TO ESTABLISH AN ORDER?

IF YES → HOW TO DYNAMICALLY CHANGE FORMATION NODES?

IF NO → HOW TO CHOOSE WHAT IS "NEXT" AND "PREV"?

→ MAYBE UNIQUE ID IMPLIES ORDER.



IF YOU CHOOSE ONE POINT THEN YOU CAN CHOOSE AN ORDER SEQUENTIALLY.

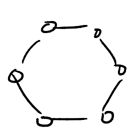
BUT WE CANT CHOOSE. ALL ROBOTS ARE SELFISH.

REQUIREMENT: MUST HAVE CYCLE TOPOLOGY...

BORING! → HOW TO CHANGE TOPOLOGY? (LOOK INTO PROTEIN FOLDING)

GRAPHS:

Cycle

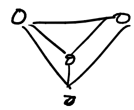


STATIC

EDGES DO NOT CHANGE AFTER INITIALIZATION

EDGES DO NOT CHANGE
AFTER INITIALIZATION
→ CAN GET TANGLED
WITH NO WAY TO UNTANGLE

PLANAR



→ DYNAMIC

EDGES OCCASIONALLY CHANGE
WHEN NEW CONTACT IS MADE
→ OSCILLATIONS . . .

ASSUMPTIONS

1. SENSORS CAN DETECT RELATIVE DISTANCE BETWEEN ROBOTS

   ULTRASONIC SENSOR ?
   CV (PI CAMERA) ?
   RADIO (WIFI) ?

2. ROBOTS HAVE ACCESS TO BIDIRECTIONAL COMMUNICATION

   CLOUD COMMUNICATION
   EVEN THOUGH IN THE CLOUD, AGENTS STILL NEED SENSOR CONFIRMATION TO FIND RELATIVE DISTANCE

3. DETECTION RANGE IS SOFTWARE LIMITED BECAUSE OTHERWISE DETECTION MIGHT BE TOO EASY

→ SO . . . (IN THE SIMULATION)

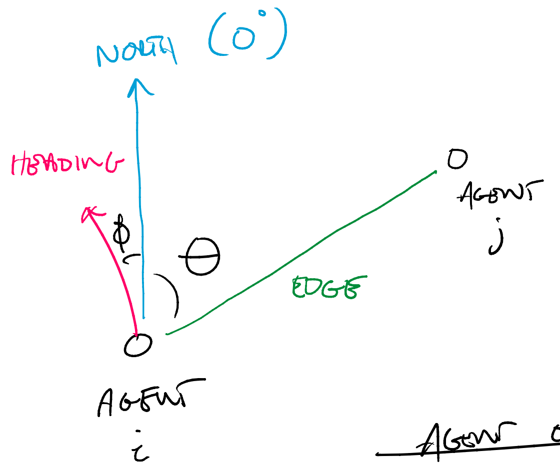
A WE WILL USE ABSOLUTE POSITIONING TO FIND ROBOT POSITIONS

B AN OVERARCHING PROGRAM WILL RUN THE SIMULATION, BUT AGENTS WILL NOT HAVE DIRECT ACCESS TO EACH OTHER. (ONLY LAPLACIAN, RELATIVE DISTANCE, AND ANGLE FROM BOW (FRONT))

RELATIVE DISTANCE,
AND ANGLE FROM BOW (FRONT)

(4) ROBOTS MOVE AS PARTICLES — PERHAPS WITH OMNIWHEELS

↳ CHANGE VELOCITY INSTANTANEOUSLY — BECAUSE INTRODUCING PHYSICS INCREASES DIFFICULTY UNNECESSARILY.



AGENT OBJECT

- CURRENT POSITION x
- CURRENT HEADING ϕ
- CURRENT VELOCITY $\dot{x}$
- MAX VELOCITY $\dot{x}_{MAX}$
- VISIBLE EDGES WITH DISTANCES {KEY, DISTANCE} UNIQUE TO
- MAX SENSOR RANGE $RANGE_{MAX}$
- ALL EDGES WITH DISTANCES {KEY, DISTANCE}
- MY LAPLACIAN GRAPH L
  - ↳ EDGES
  - ↳ ADJACENCIES

THERE ARE TWO APPROACHES TO FINDING NEARBY AGENTS →

✓ (1) HOLD ALL EDGES IN FULL EDGE LIST WITH UP-TO-DATE VALUES.

→ PLACE INTO "VISIBLE" EDGE LIST

(2) QUADTREE APPROACH ?

FUNCTIONS

CALCULATE MY L

② QUADTREE APPROACH →

CALCULATE MY L

RECONSTRUCT L FROM NEIGHBORS

GET NEAREST NEIGHBORS

MOVE (DISTANCE, HEADING)

INITIALIZATION SEQUENCE

AGENTS

RANDOM ORIENTATION/POSITION.

ROTATION MATRIX

FOR CYCLIC PURSUIT

$$\vec{x} = \begin{bmatrix} \overrightarrow{AGENT}_1 \\ \vdots \\ \overrightarrow{AGENT}_N \end{bmatrix}, \quad \overrightarrow{AGENT}_i = \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\vec{x} \in \mathbb{R}^{N \times 2}$$

$$R(-\psi) = \begin{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi \\ -\sin\psi & \cos\psi \end{bmatrix}_1 & \cdots & \begin{bmatrix} \cdot \end{bmatrix}_N \end{bmatrix}^T$$

I THINK I NEED TO USE A TENSORDOT

TEST TENSORDOT!

$$\begin{bmatrix} c & s \end{bmatrix} \begin{bmatrix} x \end{bmatrix} = \begin{bmatrix} xc + ys \end{bmatrix}$$

TEST TRANSPOSED:

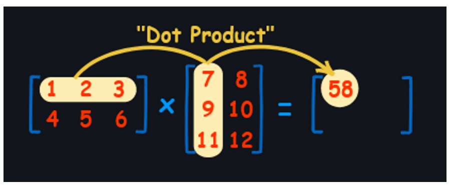
$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} xc + ys \\ -xs + yc \end{bmatrix}$$

$$\longrightarrow \begin{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}_1' \\ \vdots \\ \begin{bmatrix} x \\ y \end{bmatrix}_N' \end{bmatrix}$$

SEEMS RIGHT

... LET'S CHECK MATRIX MULTIPLICATION

MATRIX MULTIPLICATION



$$\begin{bmatrix} x & x & x & x & x & x \\ x & x & x & x & x & x \end{bmatrix} \begin{bmatrix} x & x \\ x & x \\ x & x \end{bmatrix} \longrightarrow \text{NOPE}$$