Collaborative Control

TSFS12: Autonomous Vehicles –planning, control, and learning systems

Lecture 8: Jan Åslund <jan.aslund@liu.se>

LINKÖPINGS UNIVERSITET

Swarm Robots

Most swarm approaches obtain inspiration from biological societies – particularly ants, bees, and birds – to develop similar behaviours in multirobot teams.

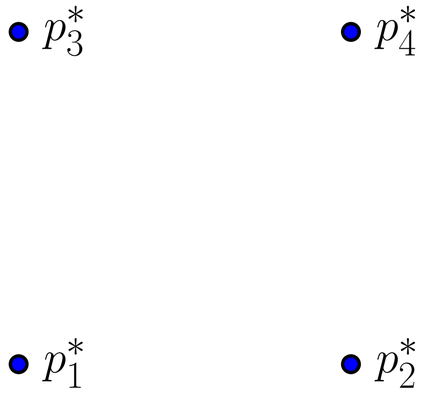
Swarm robotics systems are often called *collective* robotics, indicating that individual robots are often un-aware of the actions of other robots in the system, other than information on proximity.

Multi-agent Formation Control Problems

Multi-agent Formation Control Problems

The objective of formation control is for a set of N agents, reach a desired formation p^* given by a set of positions $p_1^*, \ldots, p_N^*$.

The formation $p = \begin{pmatrix} p_1^* \\ p_2^* \\ p_3^* \\ p_4^* \end{pmatrix}$:



What we mean by “reach” will be different in different cases and depends on limitations of the sensing capability of the agents. For example it can mean:

- Reach the formation p^*
- Reach the formation p^* modulo a translation
- Reach the formation p^* modulo a translation and a rotation

Multi-agent Formation Control Problems

The control strategy depends of the sensing capability of the agents. Today I will consider

- Position-based formation control
- Displacement-based formation control
- Distance-based formation control

There will be three models associated with each agent i:

- Dynamic model: $\dot{x}_i = f_i(t, x_i, u_i)$
  We shall consider single- and double-integrator models, introduced in Lecture 3.
- Measurement equations: $y_i = h_i(x)$
  The function $h_i(x)$ depends on the sensing capability of the agent.
- Control law: $u_i = g_i(y_i)$
  The control objective and design depends on available sensing capability.

Hand-in assignment four

The objective of the fourth hand-in assignment is to define and implement the functions f_i, h_i, and g_i in

$$\begin{cases} \dot{x}_i = f_i(t, x_i, u_i) \\ y_i = h_i(x) \qquad\qquad \text{where } i = 1, \ldots, N \\ u_i = g_i(y_i) \end{cases}$$

such that the complete formation fulfils different tasks specified in the assignment.

Position-Based Formation Control

Position-based formation control

Sensing capability: The agents are required to commonly have a global coordinate system. They need to sense their absolute positions with respect to the global coordinate system.

Interaction topology: The desired formation is specified by the desired absolute positions for the agents. In this case, interactions are not necessarily required because the desired formation can be achieved by position control of individual agents.

Position-based formation control

The objective is to achieve $p_i \rightarrow p_i^*$, where p_i is the position of agent i, and p_i^* is a known reference position.

Consider a single-integrator modelled agent

$$\dot{p}_i = u_i$$

where u_i the control signal. If the proportional control law

$$u_i = k_p(p_i^* - p_i),$$

is used, where $k_p > 0$ is the proportional gain.

The error $e_p = p - p^*$ satisfies the error dynamics

$$\dot{e}_p = -k_p e_p$$

which shows that p converges exponentially to p^*.

Position-based formation control

The desired formation p^* can be achieved by position control of individual agents, but interactions among the agents can be introduced to enhance control performance or addressing additional objectives such as formation shape keeping.

This can be achieved by introducing additional control inputs:

$$u_i = k_p(p_i^* - p_i) + \sum_{j \in \mathcal{N}_i} w_{ij}(p_j - p_i - p_j^* + p_i^*)$$

where $\mathcal{N}_i$ is a set of neighbours of node i, and w_{ij} are positive constants. It can be shown that this modification always improves the exponential convergence.

Double-integrator modelled agent

Now we shall consider the double integrator model

$$\ddot{p}_i = u_i$$

where $\ddot{p}_i$ is the acceleration and u_i is the control input of agent i. The model can be rewritten in state space form:

$$\dot{p}_i = v_i$$

$$\dot{v}_i = u_i$$

where v_i is the velocity vector.

In this case, the control law

$$u_i = k_v(v_i^* - v_i) + k_p(p_i^* - p_i)$$

can be used.

Double-integrator modelled agent

In the two-dimensional case, the state space form

$$\dot{p}_i = v_i$$
$$\dot{v}_i = u_i$$

has four states, $p_{x,i}$ and $p_{y,i}$ for position, and $v_{x,i}$ and $v_{y,i}$ for velocity. It can be written in the form $\dot{x} = Ax + Bu$ which you are familiar with from the basic course in automatic control:

$$\begin{pmatrix} \dot{p}_{x,i} \\ \dot{p}_{y,i} \\ \dot{v}_{x,i} \\ \dot{v}_{y,i} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} p_{x,i} \\ p_{y,i} \\ v_{x,i} \\ v_{y,i} \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} u_{x,i} \\ u_{y,i} \end{pmatrix}$$

Displacement-Based Formation Control

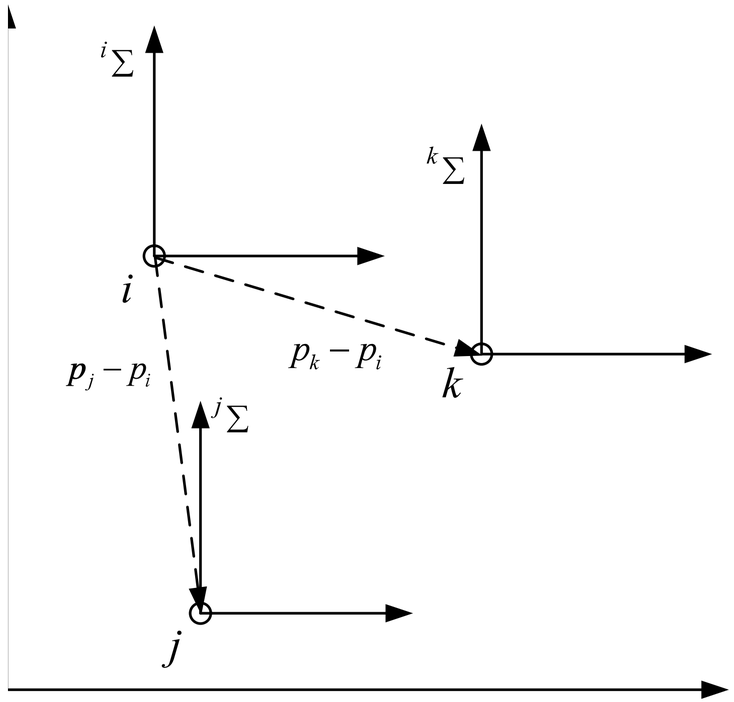
Displacement-based formation control

Sensing capability: The agents are required to have their own local coordinate systems, orientations of which are aligned to that of a global coordinate system. With respect to the local coordinate systems, the agents are required to sense relative positions of their neighbours with respect to the global coordinate system.

Interaction topology: The desired formation for the agents is specified by the desired displacements from any agents to the others.

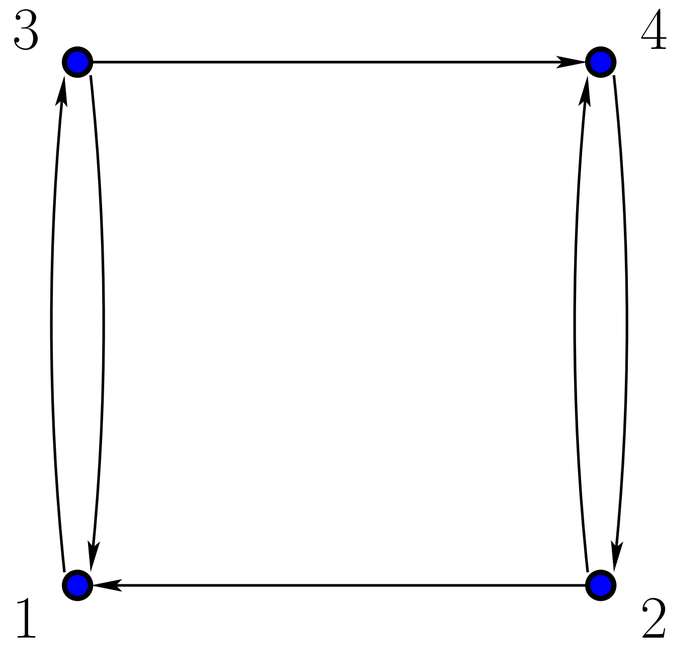
Displacement-based formation control

Local coordinate systems Σ^i with orientations that are aligned with the orientation of global coordinate system:



In this case it is assumed that the relative positions $p_j - p_i$ are available to agent i for some subset of the other agents, and a directed graph can be used to represent which these agent are.

Directed graphs: An example



Nodes $\mathscr{V} = \{1,2,3,4\}$

Edges $\mathscr{E} = \{(1,2),(1,3),(2,4),(3,1),(4,2),(4,3)\} \subset \mathscr{V} \times \mathscr{V}$

Neighbours $\mathscr{N}_1 = \{2,3\}$, $\mathscr{N}_2 = \{4\}$, $\mathscr{N}_3 = \{1\}$, and $\mathscr{N}_4 = \{2,3\}$

Displacement-based formation control

It is assumed that the relative positions

$$p_j - p_i, \quad j \in \mathcal{N}_i$$

are available to agent i.

The objective of the formation control is to achieve

$$p_i - p_j \to p_i^* - p_j^*, \quad i,j \in \mathcal{V},$$

i.e., to drive p to p^* up to a translation.

Displacement-based formation control

Example:

Single-integrator modelled agents

$$\dot{p}_i = u_i, \quad i = 1,\ldots,N$$

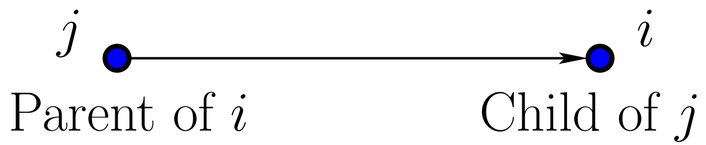
with the control law

$$u_i = k_p \sum_{j \in \mathcal{N}_i} w_{ij}(p_j - p_i - p_j^* + p_i^*)$$

where $w_{ij} > 0$.

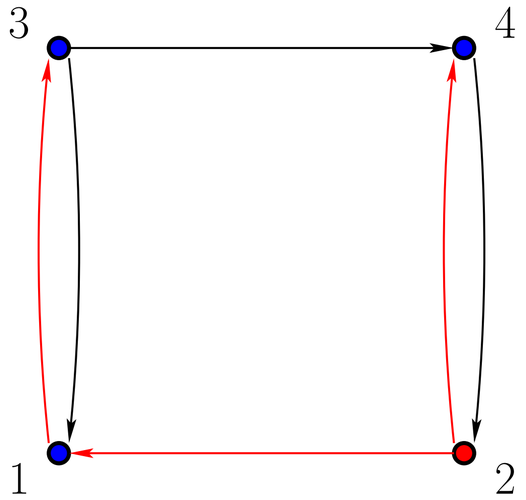
Directed graphs, spanning tree

A tree is a directed graph where a node, called the root, has no parent and the other nodes have exactly one parent.



A spanning tree of a directed graph is a directed tree containing every node of the graph.

A spanning tree consists the red edges, and node 2 is the root.



Displacement-based formation control

If the single-integrator model

$$\dot{p}_i = u_i, \quad i = 1, \ldots, N$$

is used with the control law

$$u_i = k_p \sum_{j \in \mathcal{N}_i} w_{ij}(p_j - p_i - p_j^* + p_i^*)$$

then the desired formation

$$\{p : p_j - p_i = p_j^* - p_i^*, i, j \mathcal{V}\}$$

is exponentially stable if and only if the directed graph $\mathcal{G}$ has a spanning tree.

Displacement-based formation control

With double-integrator modelled agents

$$\dot{p}_i = v_i$$

$$\dot{v}_i = u_i$$

the objective is to achieve the desired formation

$$\{[p^T v^T]^T : p_j - p_i = p_j^* - p_i^*, v_j - v_i = v_j^* - v_i^*\}$$

The control law

$$u_i = -k_p \sum_{j \in \mathcal{N}_i} w_{ij}(p_i - p_j - p_i^* + p_j^*) - k_v \sum_{j \in \mathcal{N}_i} w_{ij}(v_i - v_j - v_i^* + v_j^*)$$

can be used to achieve this goal.

Displacement-based formation control

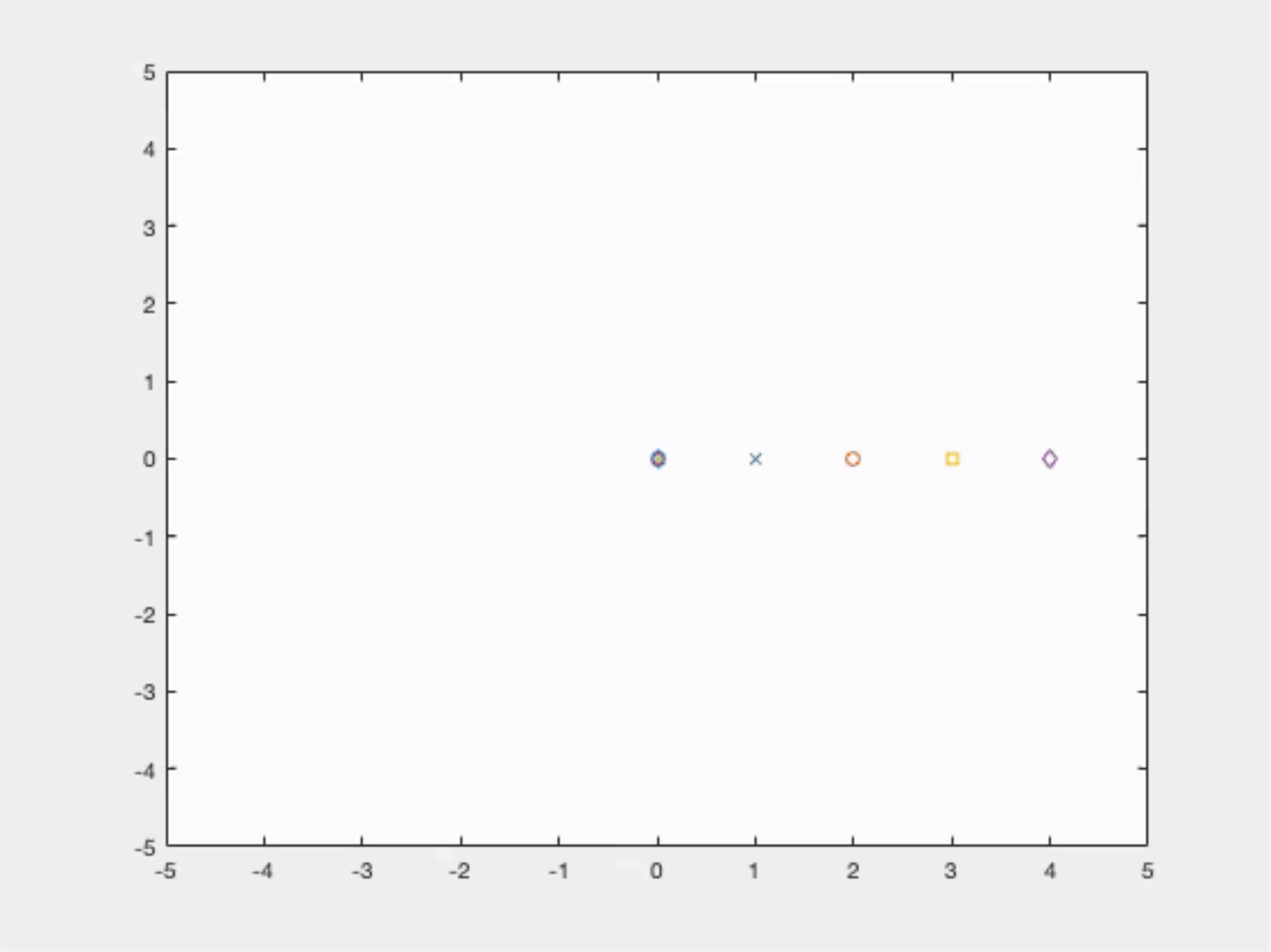
It is often the case that the formation needs to move to a prescribed absolute position. Assume that only a small number of agents are able to sense their absolute positions

For a single-integrator model the control law can be modified as

$$u_i = k_p \sum_{j \in \mathcal{N}_i} w_{ij}(p_j - p_i - p_j^* + p_i^*) + g_{ii}(p_i^* - p_i)$$

where $g_{ii} > 0$ if agent i senses p_i, and 0 otherwise.

It can be shown that the error tends to zero exponentially if the graph has a spanning tree where the agent corresponding to the root node senses its absolute position


5
4
3
2
1
0
-1
-2
-3
-4
-5
-5
-4
-3
-2
-1
0
1
2
3
4
5

Distance-Based Formation Control

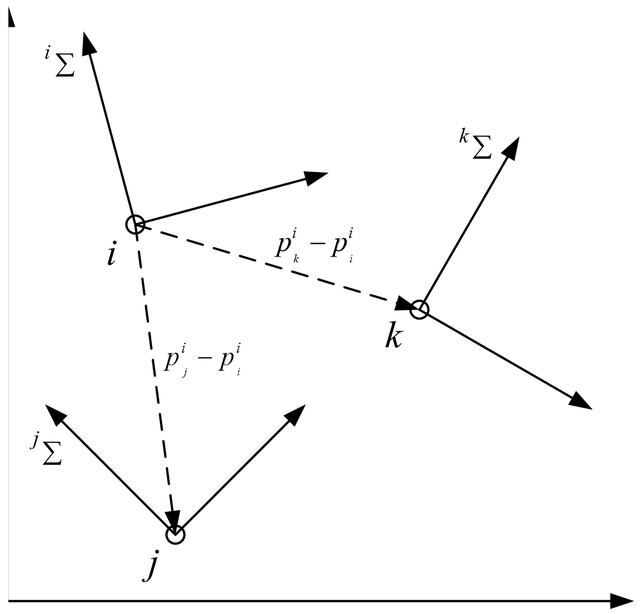
Distance-based formation control

Sensing capability: The agents are required to carry their own local coordinate systems. The orientations of the coordinate systems need not to be aligned to each other. The agents are required to sense relative positions of their neighbours.

Interaction topology: The desired formation is specified by the desired distances between any pair of agents.

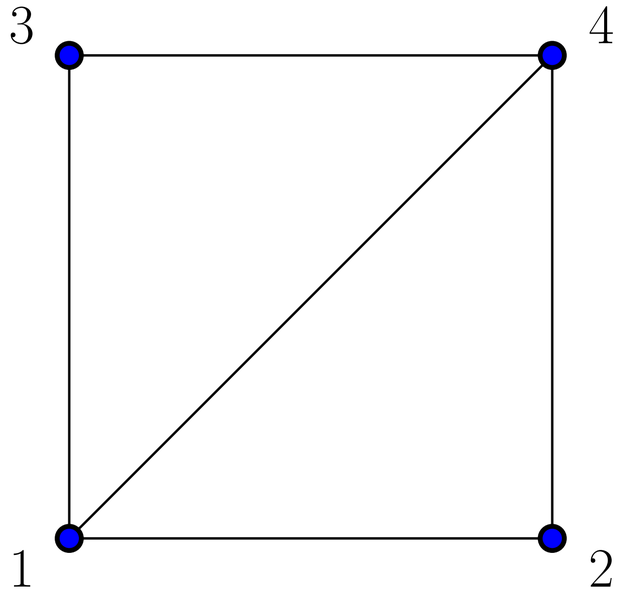
Distance-based formation control

Local coordinate systems that are not aligned to each other:



It is assumed that the relative position $p_j - p_i$ are available to agent i for its neighbours j.

Undirected graphs: An example



Nodes $\mathscr{V} = \{1,2,3,4\}$

Edges $\mathscr{E} = \{(1,2),(1,3),(1,4),(3,4),(2,4)\} \subset \mathscr{V} \times \mathscr{V}$

Neighbours $\mathscr{N}_1 = \{2,3,4\}$, $\mathscr{N}_2 = \{1,4\}$, $\mathscr{N}_3 = \{1,4\}$, and $\mathscr{N}_4 = \{1,2,3\}$

Distance-based formation control

It is assumed that the relative positions

$$p_j - p_i, \quad j \in \mathcal{N}_i$$

in the local coordinate system are available to agent i.

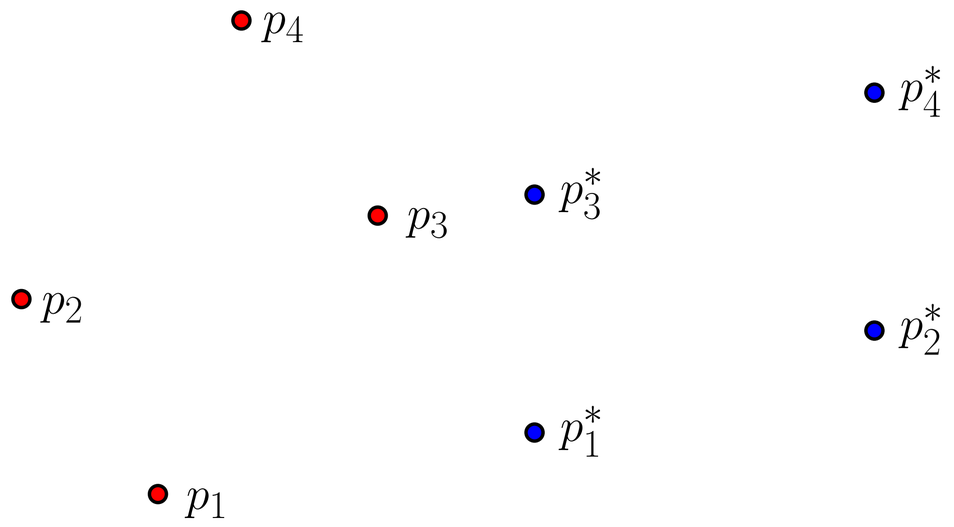
The objective of the formation control is now to achieve

$$\|p_i - p_j\| \to \|p_i^* - p_j^*\|, \quad i,j \in \mathcal{V}$$

But this condition alone does not imply that you get the formation of the agents that you were looking for.

Distance-based formation control

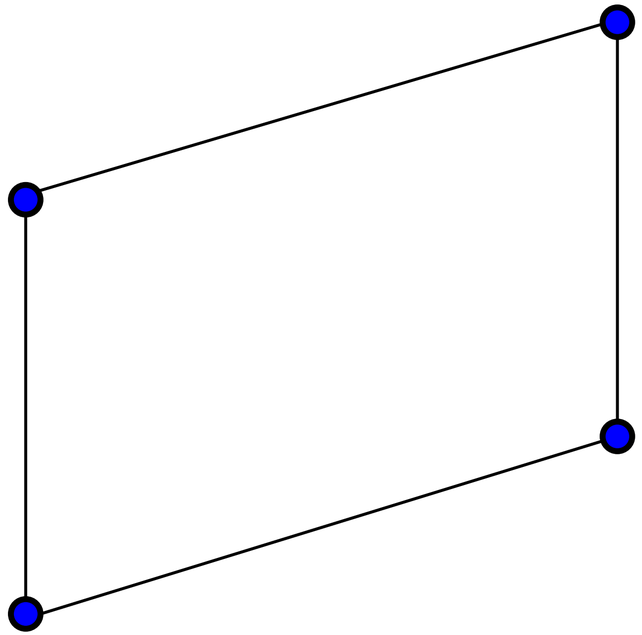
The red formation is obtained by translating and rotating the blue formation and it satisfies the condition $\|p_i - p_j\| = \|p_i^* - p_j^*\|$



Observation: It is not possible to distinguish between the red and blue formations with the limited sensing capability of the agents in this case.

Graph rigidity

Consider the following undirected graph:

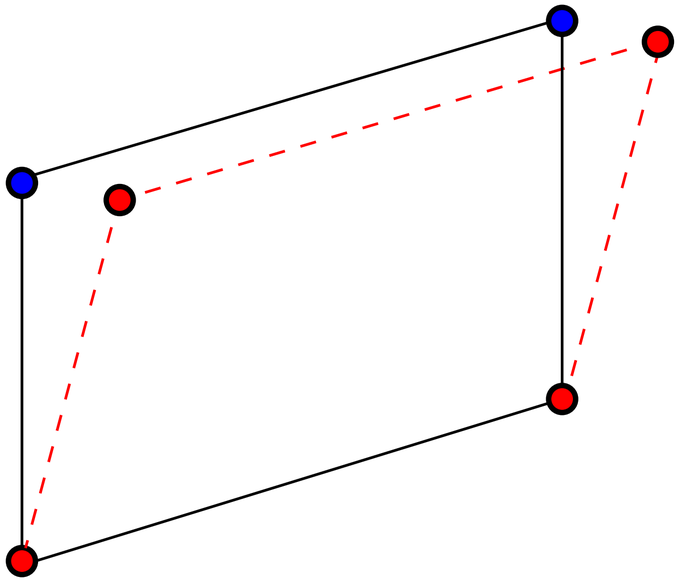


Question: Is this graph suitable for the task?

Graph rigidity: Not rigid frameworks

Answer: No!

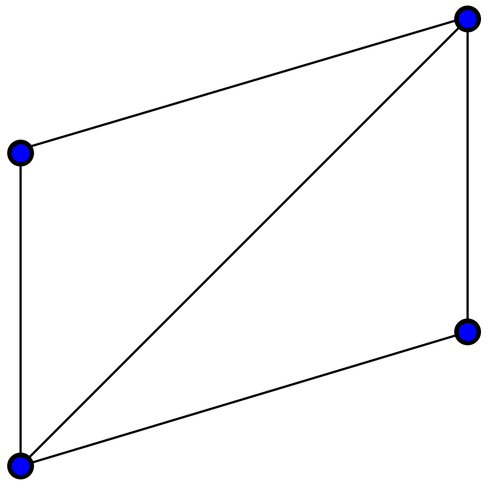
The red nodes satisfies the condition $\|p_i - p_j\| = \|p_i^* - p_j^*\|$

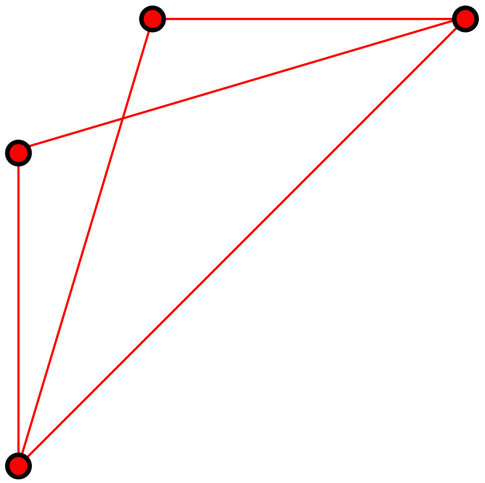


The red formation is usually not acceptable in the formation control problem. The framework above is called ***not rigid***.

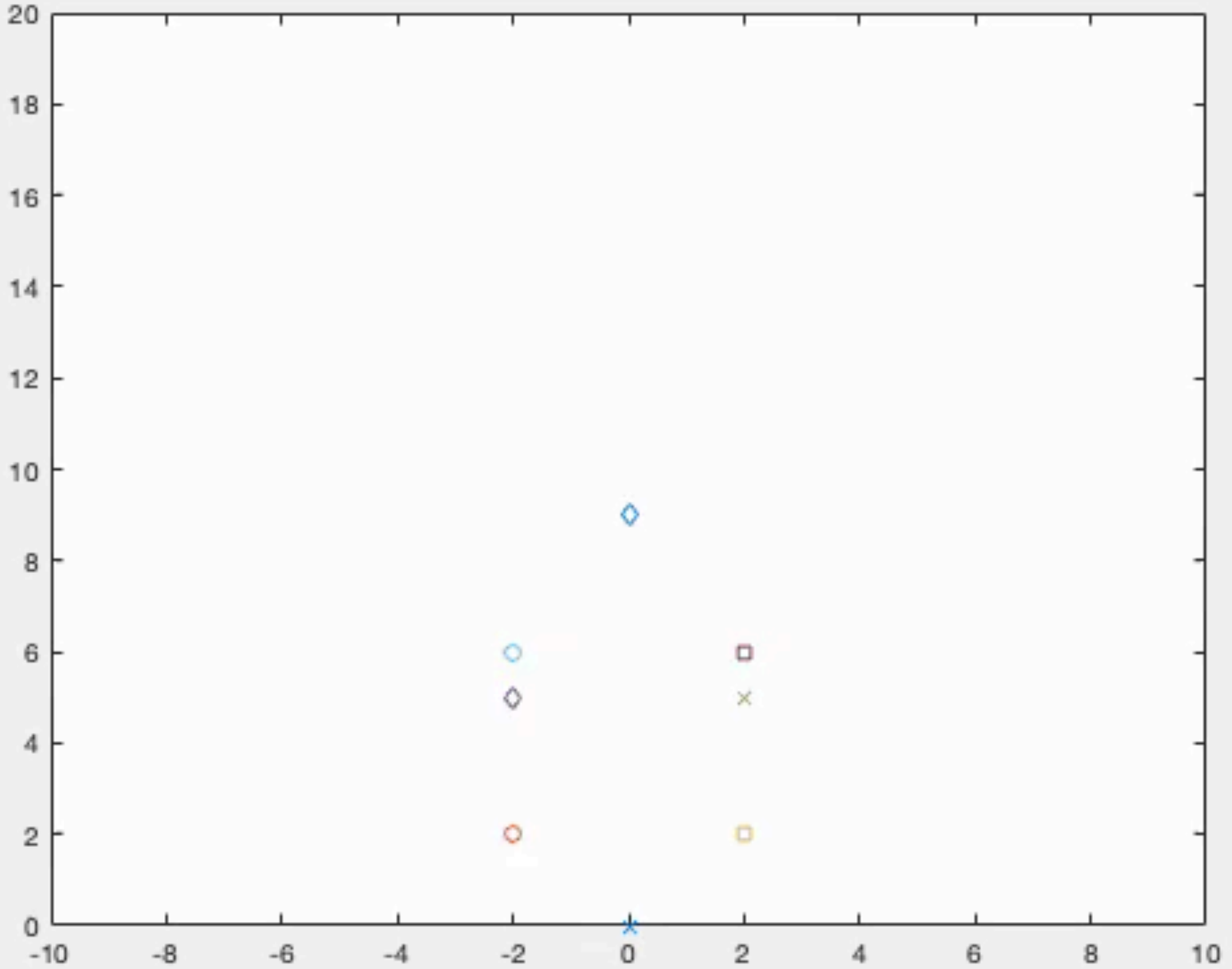
Graph rigidity: Rigid framework

The framework to the left is called rigid. The diagonal edge prevents the nodes to deviate from the desired formation



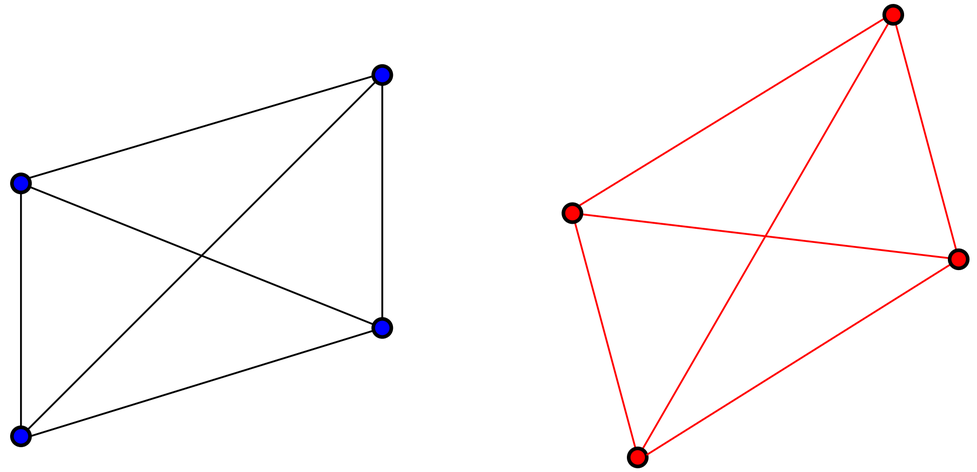


The drawback with this graph is shown to the right. The nodes satisfy $\|p_i - p_j\| = \|p_i^* - p_j^*\|$, but this is not the formation we wanted. This can happen if the initial conditions are unfavourable.


20
18
16
14
12
10
8
6
4
2
0
-10
-8
-6
-4
-2
0
2
4
6
8
10

Graph rigidity: Global rigidity

The framework to the left is called globally rigid. The second diagonal edge prevents a node to flip to the other side of the diagonal.

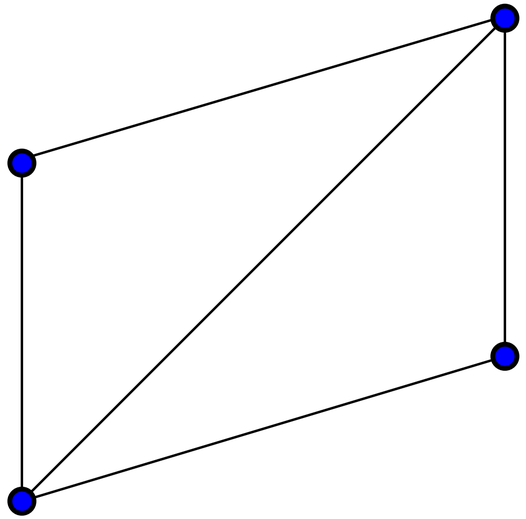


The red framework to the right is obtained by a translation and a rotation, and the nodes satisfy the condition $\|p_i - p_j\| = \|p_i^* - p_j^*\|$.

As pointed out before, this can not be avoided with the sensing capability in this case.

Graph rigidity

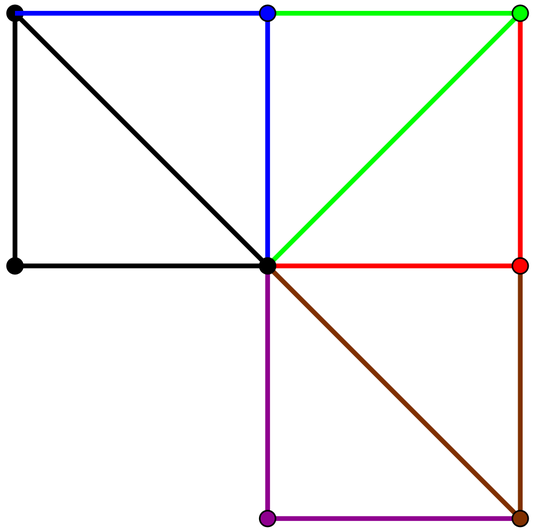
The rigid graph below is minimally rigid, which means that if any edge is removed, then the graph is no longer rigid.



It can be shown that any minimally rigid graph in the plane with N nodes has $2N - 3$ edges. I will not give a formal definition of minimally rigid. Instead I will illustrate how to construct a minimally rigid graph.

How to build a minimally rigid graph

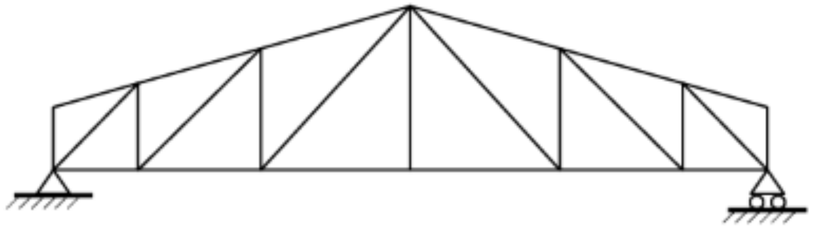
Start with a triangle (black):



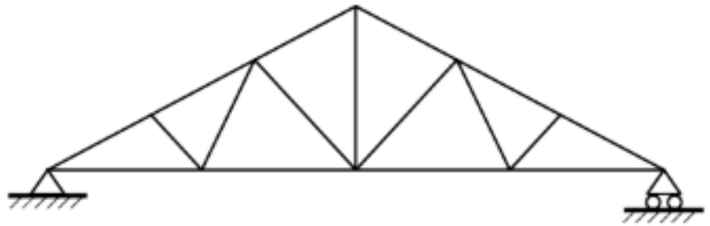
Attach one node and two edges (blue) to the structure as in the figure. Continue two add one node and two edges at a time (green, red, brown magenta). The resulting framework will alway be minimally rigid.

This is actually how you build rigid structures:

f)



g)



In three dimensions you start with 4 nodes and 6 edges, and then add one node and three edges at a time to construct a minimally rigid structure.

Distance-based formation control

Assume that we have a rigid graph. We shall study single-integrator modelled agents

$$\dot{p}_i^i = u_i^i, \quad i = 1,\ldots,N$$

where p_i^i and u_i^i denote the position and control input of agent i with respect to the local coordinate system.

Gradient control laws are popular to use in distance based formation control.

Gradient based control

For each agent i, a local potential function is defined as

$$\phi_i = k_p \sum_{j \in \mathcal{N}_i} \gamma_{ij}(\|p_j^i - p_i^i\|)$$

and one example of a function that can be used in the definition is

$$\gamma_{ij}(\|p_j - p_i\|) = (\|p_j - p_i\|^2 - \|p_j^* - p_i^*\|^2)^2$$

If the the condition $\|p_j - p_i\| = \|p_j^* - p_i^*\|$ is fulfilled for all neighbours $j \in \mathcal{N}_i$, i.e., all neighbours are at the desired distance, then $\phi_i = 0$.

If this is not the case, then $\phi_i > 0$.

Gradient based control

Hence, the objective to fulfil the desired formation can be reformulated as trying to find a formation that fulfils $\phi_i = 0$ for all agents, or equivalently minimise ϕ_i.

A control law that strive to fulfil the latter formulation is

$$u_i^i = -\nabla_{p_i^i}\phi_i = -k_p \sum_{j \in \mathcal{N}_i} \nabla_{p_i^i}\gamma_{ij}(\|p_j^i - p_i^i\|)$$

In order to calculate the gradient $\nabla_{p_i^i}\gamma_{ij}(\|p_j^i - p_i^i\|)$, note that $\gamma_{ij}(\|p_j^i - p_i^i\|)$ is a compound function, where $\gamma_{ij}(s) = (s^2 - \|p_j^* - p_i^*\|^2)^2$ is a scalar valued function that depends on the scalar s, and $s(p_i^i) = \|p_j^i - p_i^i\|$, for a fixed p_j^i, is a scalar valued function that depends on the vector p_i^i.

Gradient based control

By using the chain rule of differentiation we obtain

$$\nabla_{p_i^i}\gamma_{ij} = \gamma'_{ij}(\|p_j^i - p_i^i\|)\,\nabla_{p_i^i}\|p_j^i - p_i^i\| = -\gamma'_{ij}(\|p_j^i - p_i^i\|)\frac{p_j^i - p_i^i}{\|p_j^i - p_i^i\|}$$

where γ'_{ij} denotes the derivative of γ_{ij}. By substituting this into the sum above, the following control law is obtained:

$$u_i^i = -\nabla_{p_i^i}\phi_i = k_p \sum_{j\in\mathcal{N}_i} \gamma'_{ij}(\|p_j^i - p_i^i\|)\frac{p_j^i - p_i^i}{\|p_j^i - p_i^i\|}$$

By transforming this expression into global coordinates, by a rotation and a translation, we obtain the control law:

$$u_i = k_p \sum_{j\in\mathcal{N}_i} \gamma'_{ij}(\|p_j - p_i\|)\frac{p_j - p_i}{\|p_j - p_i\|}$$