Networked Robotics Systems, Cooperative Control and Swarming (ROB-GY 6333 Section A)

- **Today's lecture:**
  - Formation Control

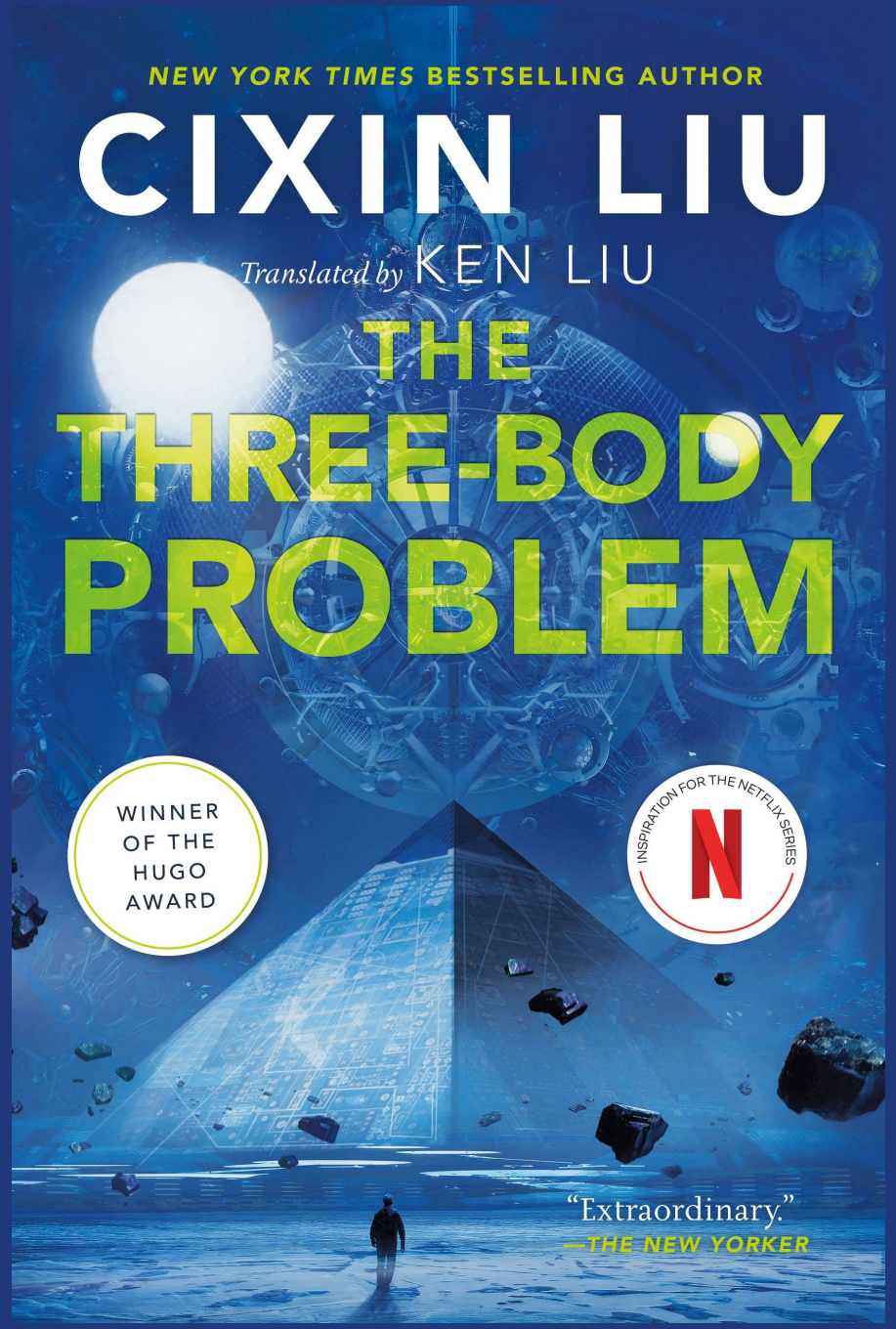
Special Lectures

- Next Tuesday: March 4, 10:00-11:00 (5MTC LC400)
- End of Semester: April 28, in-class

Turning Towards Lie Theory: Revisiting Rotations in Robotics

Since its development by Sophus Lie in the late nineteenth century, Lie theory has blossomed into a powerful mathematical framework used in a wide variety of disciplines from mathematical physics to nonlinear control theory. This talk explores how the specific problem of representing rigid-body rotations in three-dimensional space for robotics serves as both a natural motivation and intuitive example for understanding Lie groups and Lie algebras. As we explore the exponential map that links the two, we will see how Lie theory reveals the underlying unity of familiar rotation formalisms, such as Euler angles and quaternions, to produce much more than just another representation for SO(3). We will conclude with a practical note on the usefulness of this approach when applied to the statistical and learning methods that modern roboticists rely on for basic tasks such as navigation, localization, and perception.

There will be a small token of appreciation if you arrive early


NEW YORK TIMES BESTSELLING AUTHOR
CIXIN LIU
Translated by KEN LIU
THE THREE-BODY PROBLEM
WINNER OF THE HUGO AWARD
INSPIRATION FOR THE NETFLIX SERIES
"Extraordinary."
—THE NEW YORKER

Bad science fiction focuses on the *technology*

What is it?

How does it work?

Good science fiction examines the *choices* made

Who develops/uses it?

Why do they do that?

These choices are rooted in **space** and **time** — **Where?** and **When?**

Credit: U.S. Army photo by Aaron Duerk

The Biden administration in 2023 created the Replicator initiative, the first phase of which focuses on fielding thousands of autonomous drones within two years. The deadline is set for August of this year.

The second phase of Replicator, centered on countering those systems, was announced in October 2024.

Brad Dress - 01/26/25, The Hill: "Pentagon looks to leverage AI in fight against drones"

Credit: U.S. Army photo by Aaron Duerk

*DOD officials describe Replicator 1 as an all-domain initiative that could include **autonomous aerial, ground, surface, sub-surface, and/or space systems** representing a range of capabilities and mission sets.*

*... Replicator could include "distributed pods of self-propelled **ADA2*** [sensor] systems" to provide near-real time intelligence, "fleets of ground-based ADA2 systems delivering novel logistics support ... or securing DOD infrastructure," or space-based ADA2 systems to provide resilient communications.*

* all-domain, attritable, autonomous (ADA2)

Congressional Research Service, DOD Replicator Initiative: Background and Issues for Congress

Applications we will not pursue

In addition to the above objectives, we will not design or deploy AI in the following application areas:

Technologies that cause or are likely to cause overall harm. Where there is a material risk of harm, we will proceed only where we believe that the benefits substantially outweigh the risks, and will incorporate appropriate safety constraints.

Weapons or other technologies whose principal purpose or implementation is to cause or directly facilitate injury to people.

Technologies that gather or use information for surveillance violating internationally accepted norms.

Technologies whose purpose contravenes widely accepted principles of international law and human rights.

As our experience in this space deepens, this list may evolve.

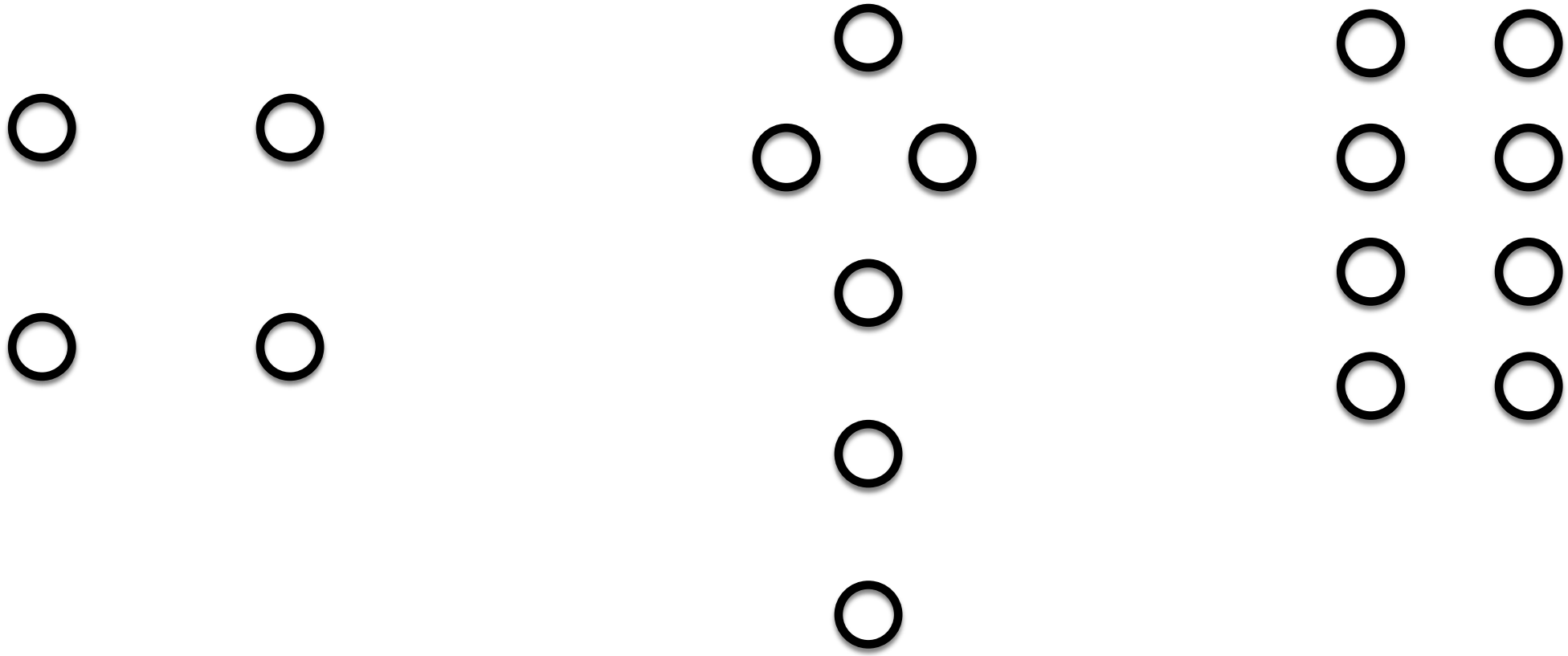
Now removed section of Google's AI principles

There's a global competition taking place for AI leadership within an increasingly complex geopolitical landscape. We believe democracies should lead in AI development, guided by core values like freedom, equality, and respect for human rights. And we believe that companies, governments, and organizations sharing these values should work together to create AI that protects people, promotes global growth, and ***supports national security****.*

Feb 4, 2025 Google Blog post: "Responsible AI: Our 2024 report and ongoing work"

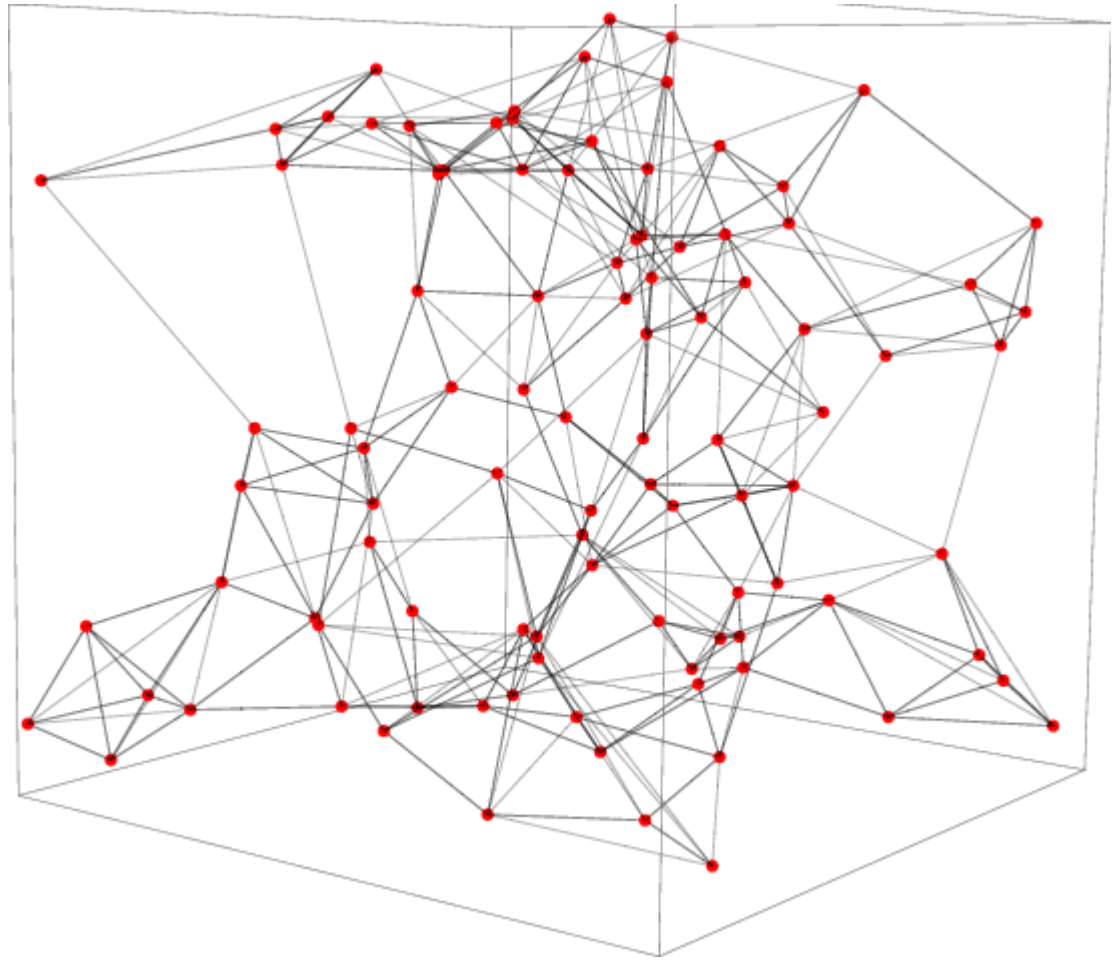
Formation Control: Beyond Rendezvous

- Formation control → control of geometrical patterns formed by multi-agent systems



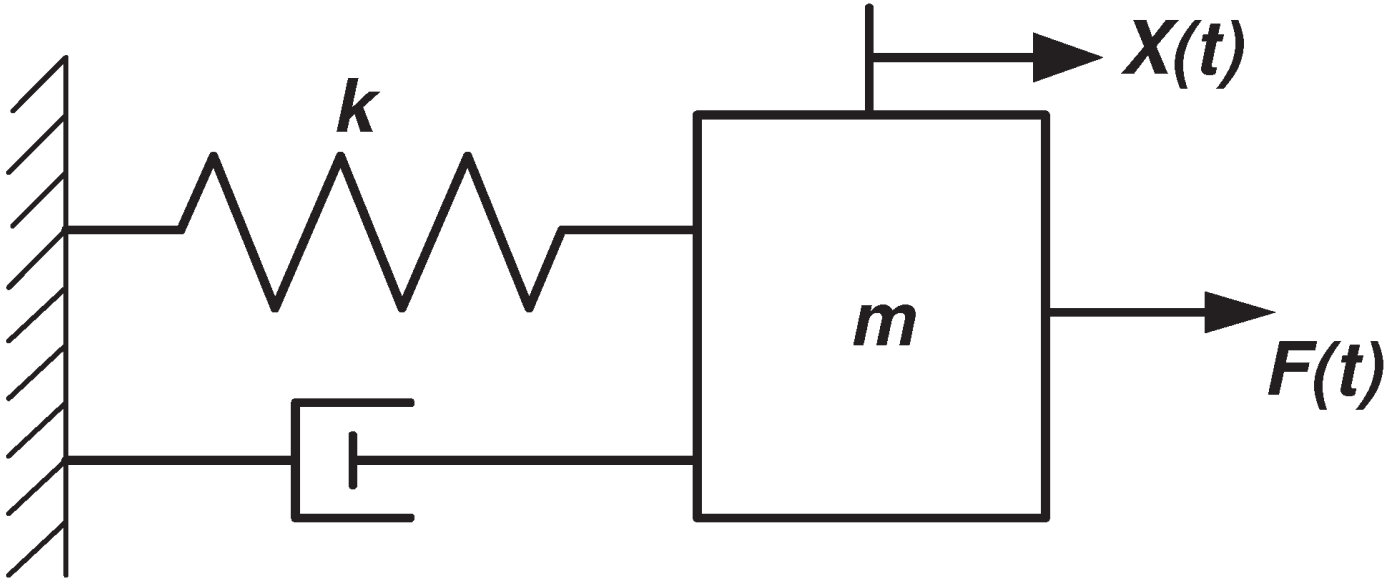
Swarms are often not flat

- But our PowerPoint slides are
- Reminder that not all our robots have to be in the same plane like in our examples



Other Approaches Not Covered Here

- Energy-based methods
  - Passivity
  - Port-Hamiltonian Systems



Covalent Network Solids

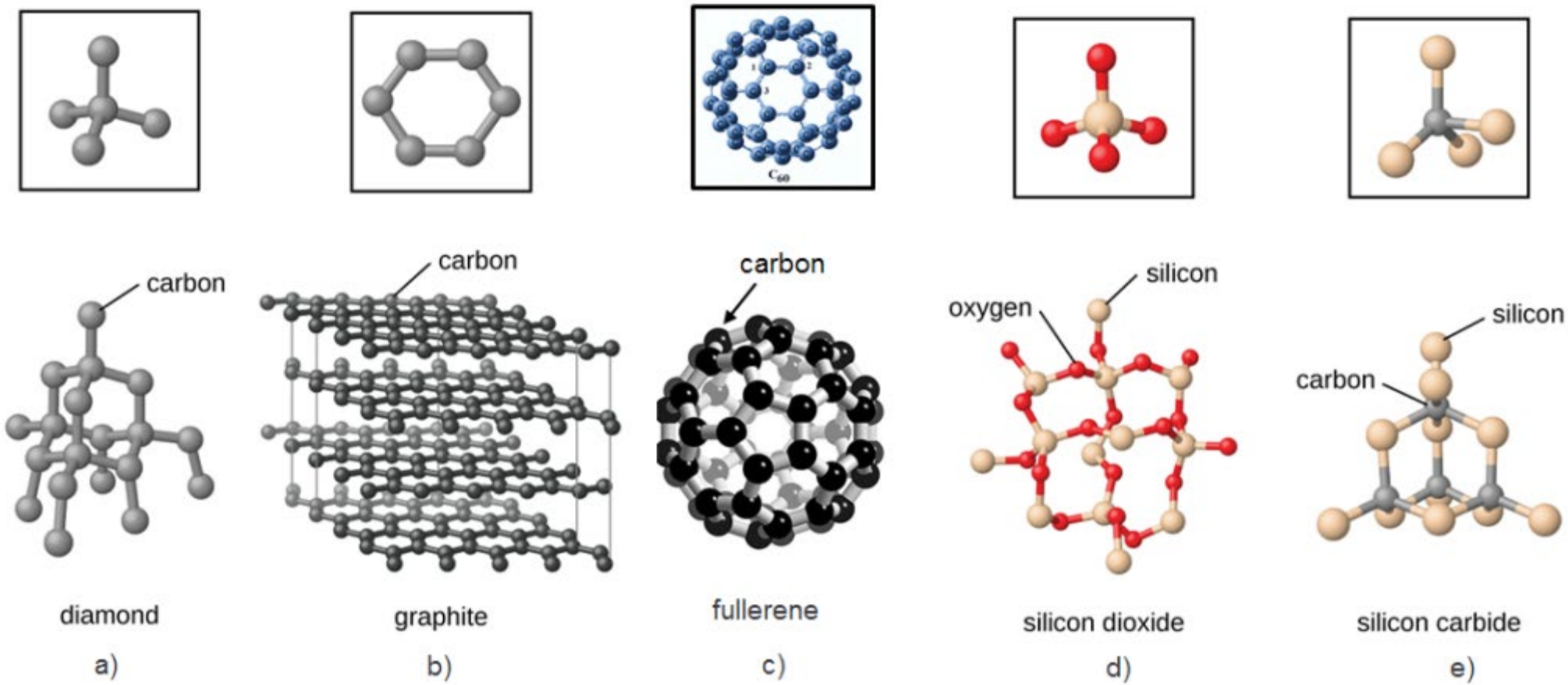
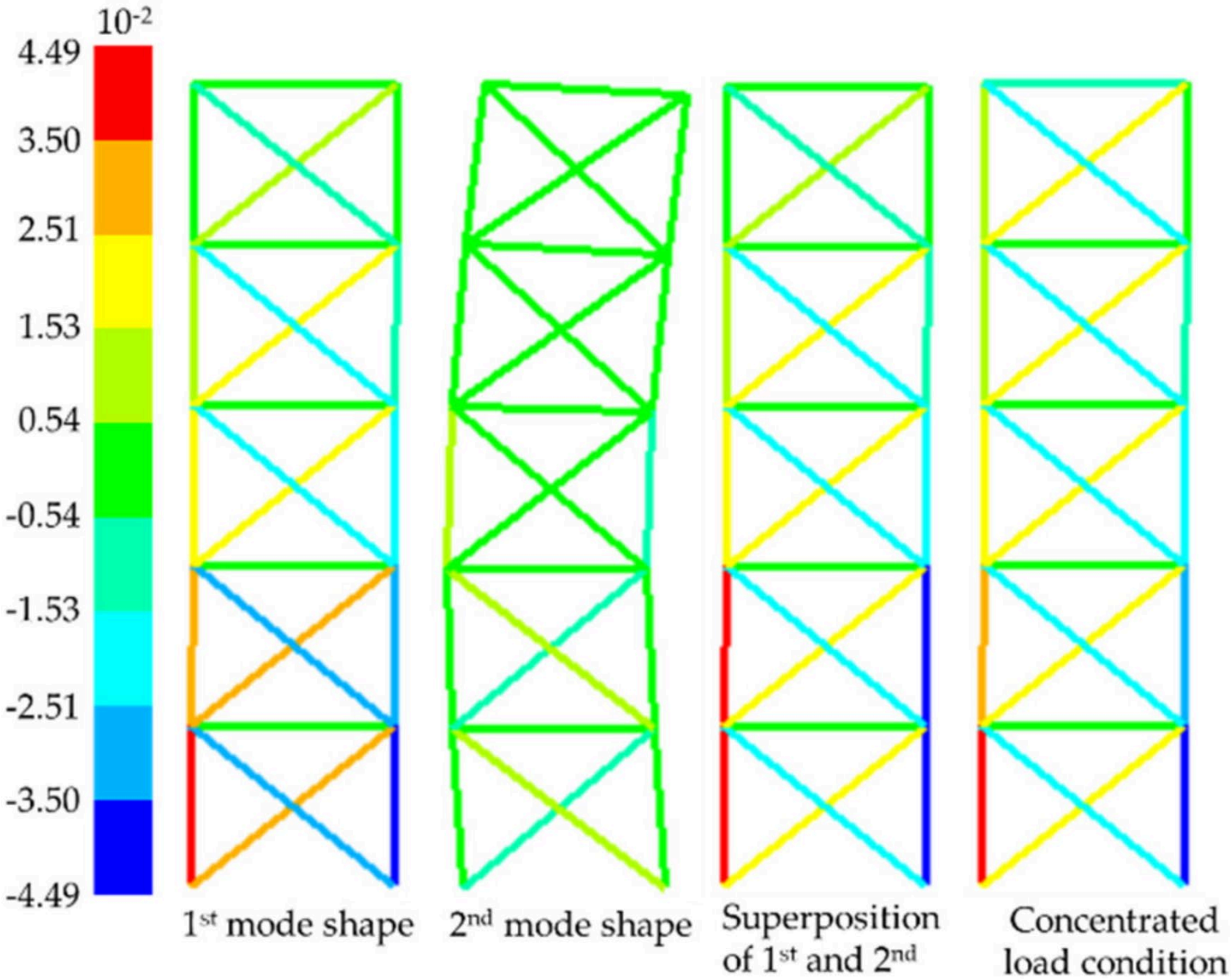
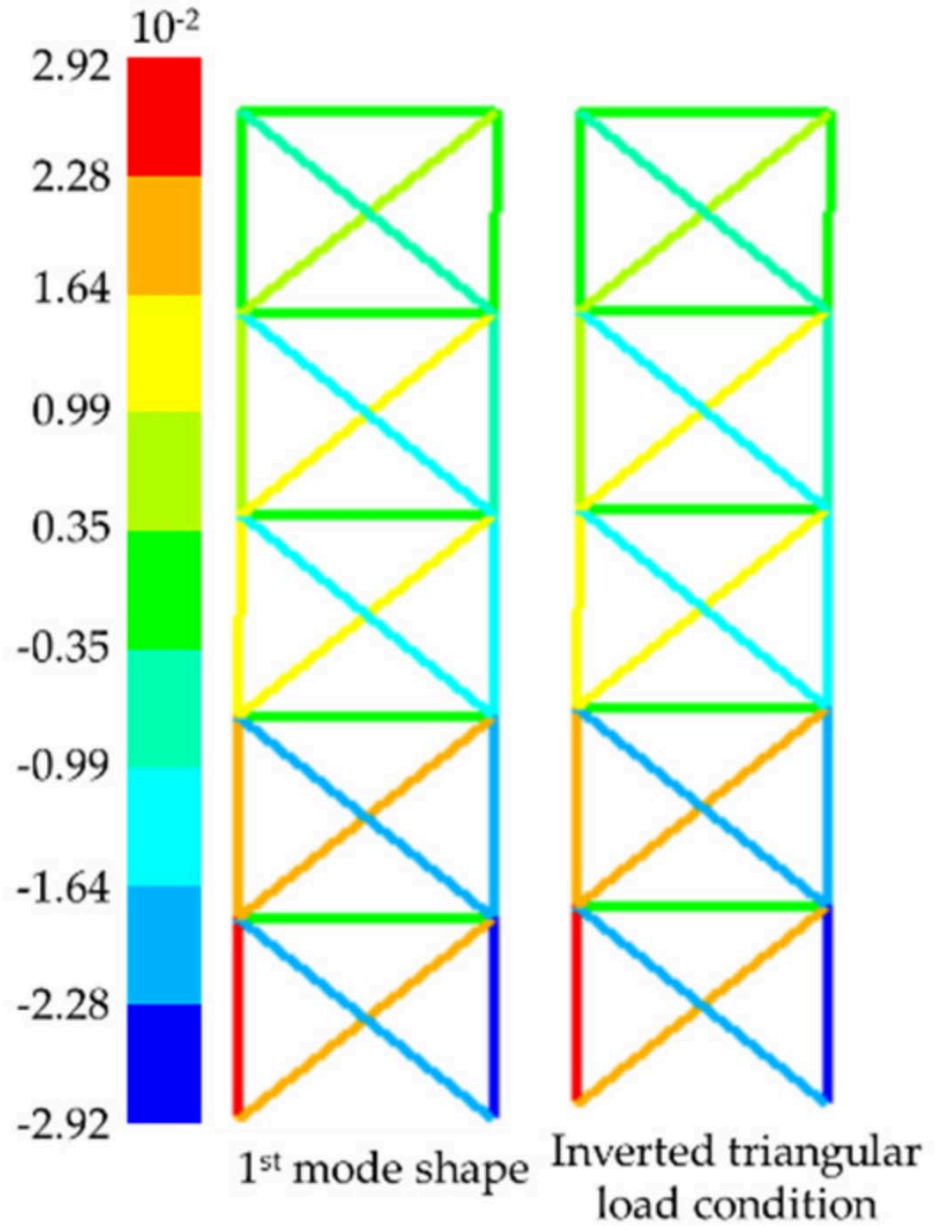


FIGURE 4 : *A covalent crystal contains a 3-dimensional network of covalent bonds, as illustrated by the structures of diamond, graphite, fullerene, silicon dioxide and silicon carbide*


10-2
4.49
3.50
2.51
1.53
0.54
-0.54
-1.53
-2.51
-3.50
-4.49
1st mode shape
2nd mode shape
Superposition of 1st and 2nd
Concentrated load condition

(a)


10-2
2.92
2.28
1.64
0.99
0.35
-0.35
-0.99
-1.64
-2.28
-2.92
1st mode shape
Inverted triangular load condition

(b)

Our approach for this and next week

- We will use
  - Graph Theory
  - **Geometry (Geometric Rigidity)**

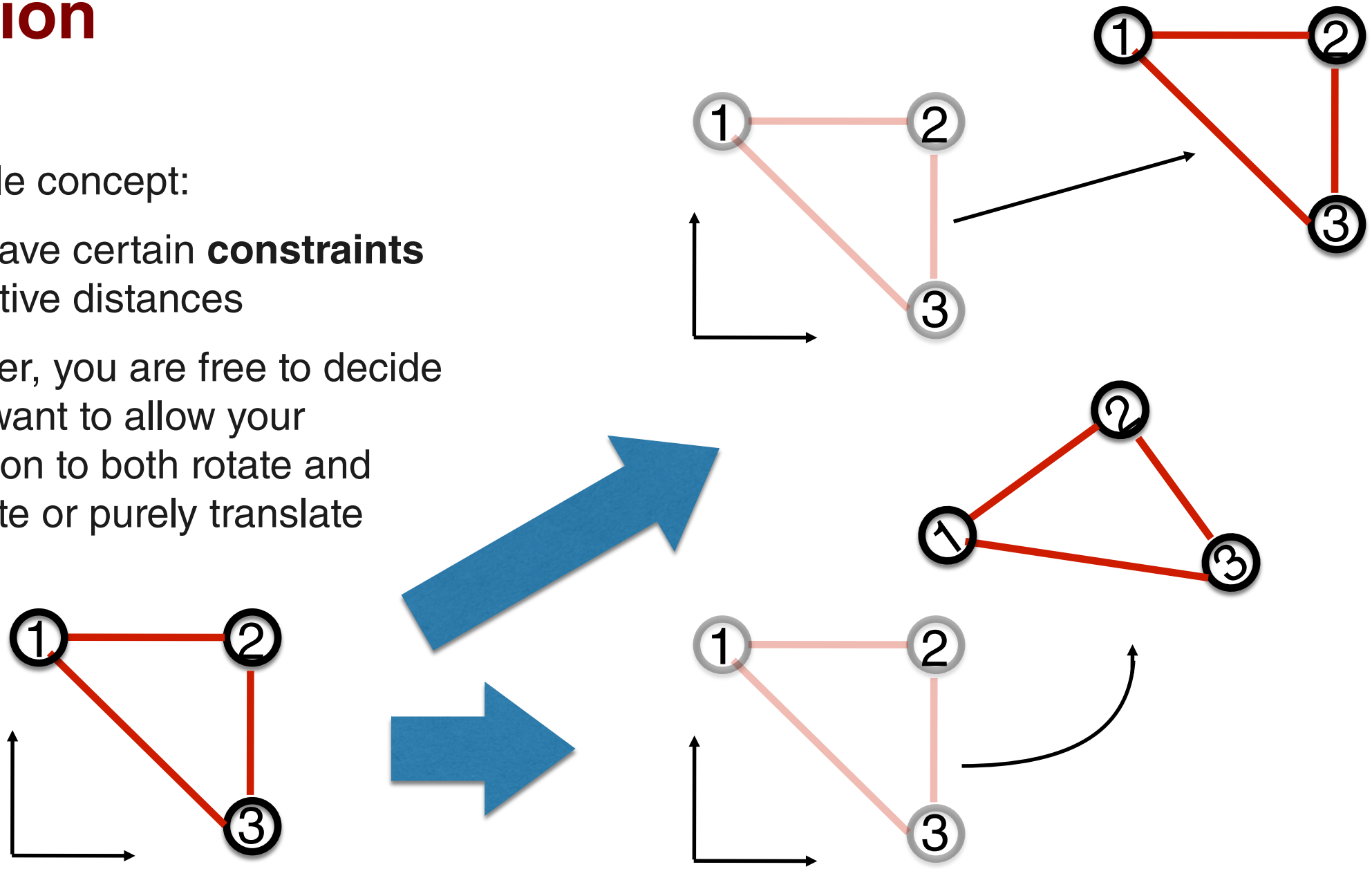
- https://crazyswarm.readthedocs.io/en/latest/

Note:

- Today's lecture is heavy on notation, but most of the notation are obvious, straightforward ideas tortured with clunky mathematical notation. We will touch on the following terms:
  - **Specifications**
  - **Framework**
  - **Rigidity mapping**
  - **Rigidity matrix**
  - **Rigidity**
  - **Infinitesimal Rigidity**

Formation

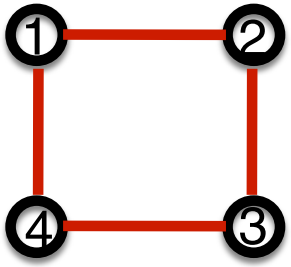
- Very flexible concept:
  - Must have certain **constraints** on relative distances
  - However, you are free to decide if you want to allow your formation to both rotate and translate or purely translate



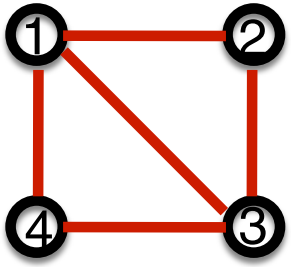
Formation Specification

- The constraints that define the formation can always be written as relative distance constraints between agents (pair-wise geometrical constraints).
- A **specification** $\mathcal{D}$ is the set of desired relative distances between agents

$$\mathcal{D} = \{d_{ij} \in \mathbb{R} \,|\, d_{ij} > 0, \quad i, j = 1, \ldots, n \quad i \neq j\}$$



$$\mathcal{D} = \{d_{12} = 1, d_{23} = 1, d_{34} = 1, d_{41} = 1\}$$



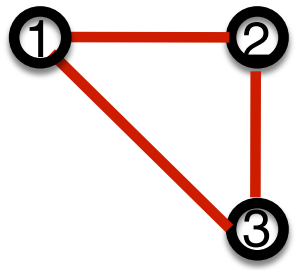
$$\mathcal{D} = \{d_{12} = 1, d_{23} = 1, d_{34} = 1, d_{41} = 1, d_{13} = \sqrt{2}\}$$

- We have chosen, quite practically, not to allow zero distance between agents in a specification to avoid considering collisions.
- More interesting when the specification is not constant but varies as a function of time.
- The term specification is a clunky notation for an obvious idea, how to maintain spacing in formation

Formation Specification

- A valid formation specification must be **feasible**
  - Physically realizable!

$$\mathcal{D} = \{d_{12} = 1, d_{23} = 1, d_{31} = \sqrt{2}\}$$



  - Not physically realizable in either 2-D or 3-D space because it violates the triangle inequality!

$$\mathcal{D} = \{d_{12} = 1, d_{23} = 1, d_{31} = 3\}$$

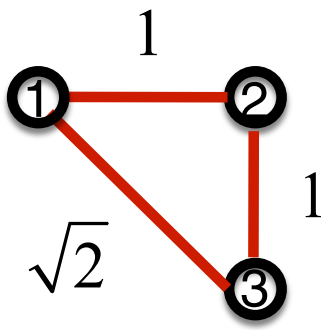
**Possibly realizable if we consider non-Euclidean spacetime but we are not touching that in this class.*

Connection to Graph Theory

- A formation can also be modeled as a **weighted graph**
  - Simply treat the pair-wise distance constraints as weighted edges of a graph

$$\mathcal{D} = \{d_{12} = 1, d_{23} = 1, d_{31} = \sqrt{2}\}$$

Specification

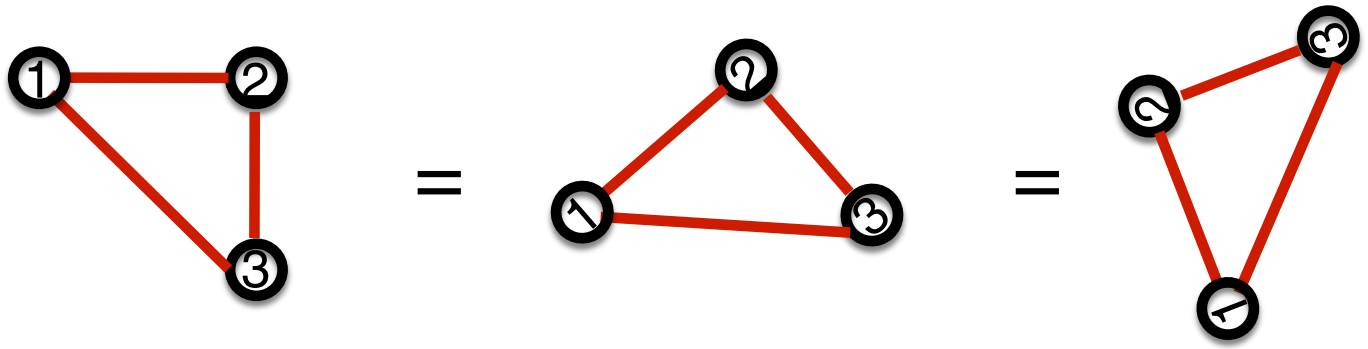


Weighted undirected graph

Formation Specification

- By itself, a specification can only determine shape, it is satisfied by different formations that are translated or rotated from each other

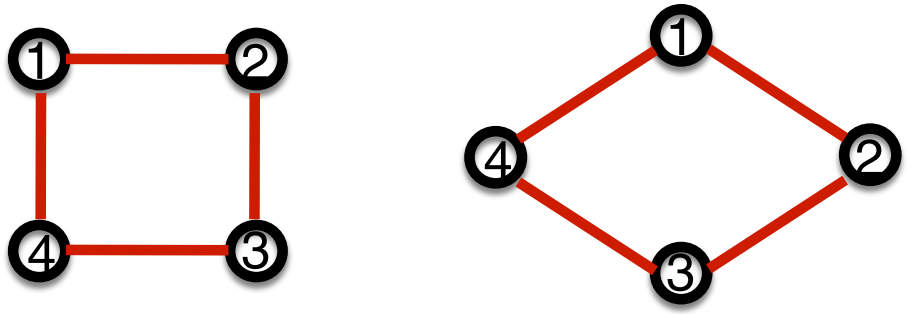
$$\mathcal{D} = \{d_{12} = 1, d_{23} = 1, d_{31} = \sqrt{2}\}$$



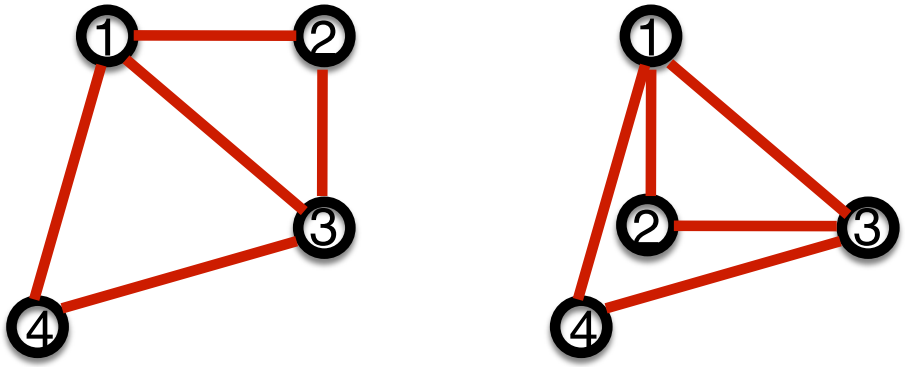
Formation Specification

- In fact, sometimes a specification can allow for multiple shapes
  - This is not necessarily bad or good depending on your goals

$$\mathcal{D} = \{d_{12} = 1, d_{23} = 1, d_{34} = 1, d_{41} = 1\}$$

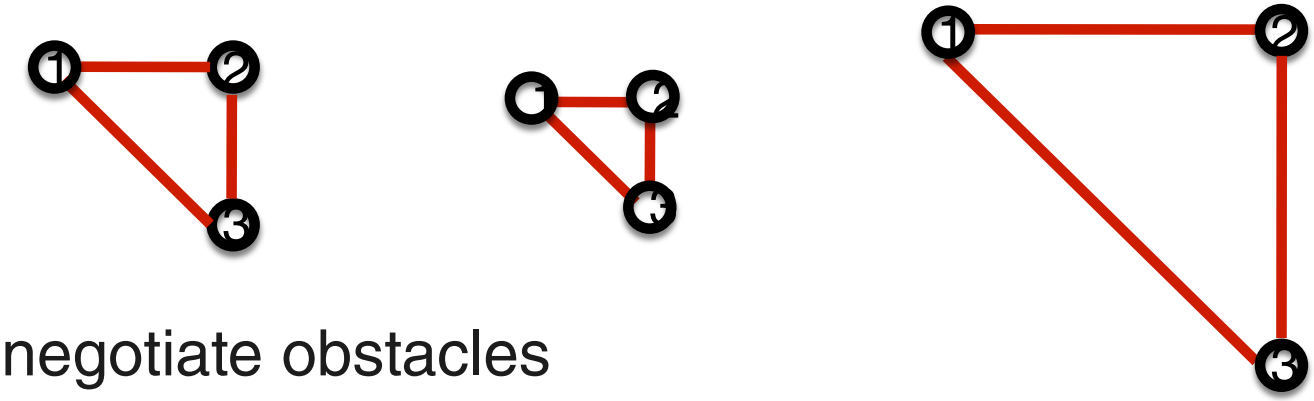


$$\mathcal{D} = \{d_{12} = 1, d_{23} = 1, d_{34} = 1.5, d_{41} = 1.5, d_{13} = \sqrt{2}\}$$

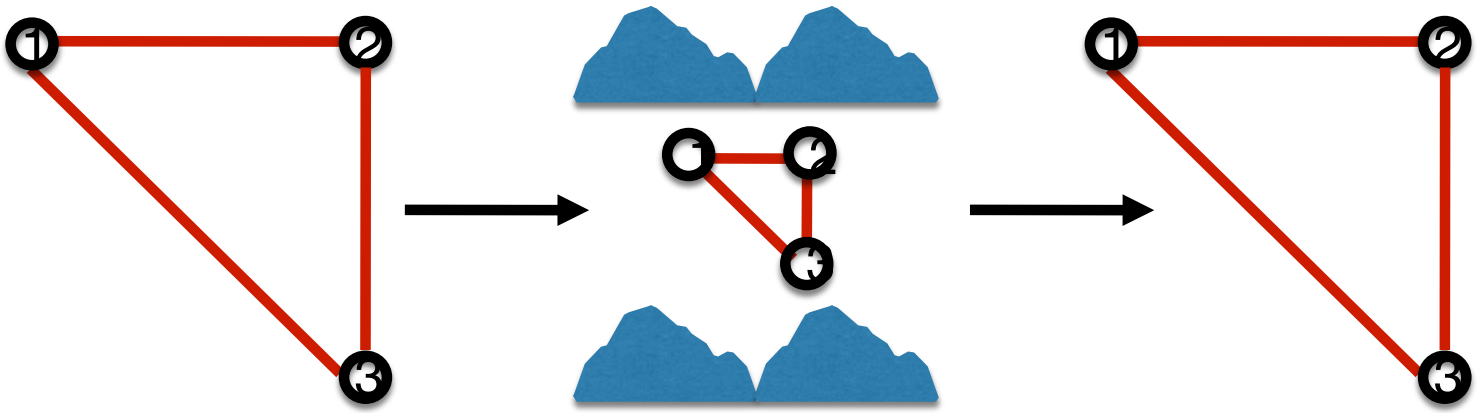


Formation Specification

- Even constant angle constraints (bearing constraints) can be written as pair-wise distance constraints (all distances are according to a common ratio)

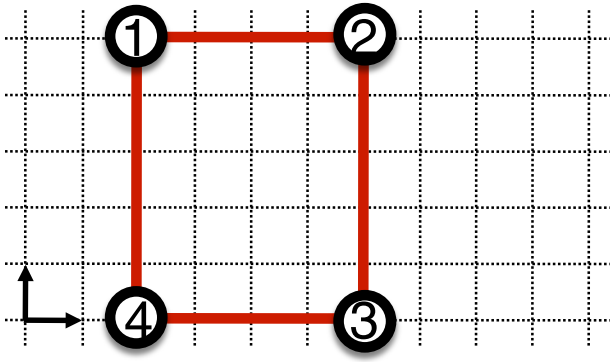


- Useful to negotiate obstacles

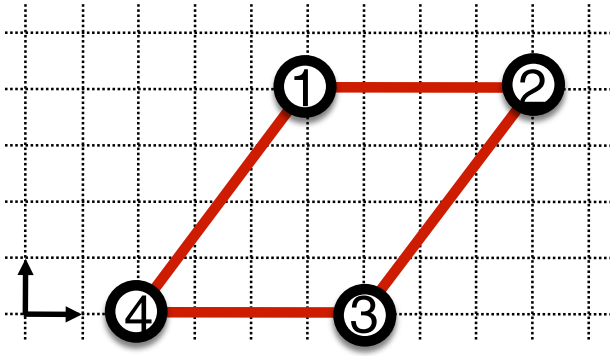


Framework

- A **framework** is composed of a **weighted undirected graph** $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a function $p : \mathcal{V} \rightarrow \mathbb{R}^d$ that maps every vertex to a point in space with dimension d



$$\mathcal{V} = \{v_1, v_2, v_3, v_4\}$$
$$\mathcal{E} = \{d_{12} = 4, d_{23} = 5, d_{34} = 4, d_{41} = 5\}$$
$$p = \{(v_1, (2,5)), (v_2, (6,5)), (v_3, (6,0)), (v_4, (0,2))\}$$



$$\mathcal{V} = \{v_1, v_2, v_3, v_4\}$$
$$\mathcal{E} = \{d_{12} = 4, d_{23} = 5, d_{34} = 4, d_{41} = 5\}$$
$$p = \{(v_1, (5,4)), (v_2, (9,4)), (v_3, (6,0)), (v_4, (0,2))\}$$

These two frameworks have the same $\mathcal{E}$ but not p. Can you imagine two frameworks with the same p but not $\mathcal{E}$?

Framework

- A framework has much more information than a specification, since it has the actually locations of each vertex.
  - Again, more interesting when the function p varies with time rather than takes constant values
- In the context of frameworks, I defined the function $p : \mathcal{V} \to \mathbb{R}^d$, which for 2D formations ($d = 2$) is $p : \mathcal{V} \to \mathbb{R}^2$. In this usage, p will have a subscript that indicates which agent I am referring to. For example, p_3 refers to the coordinates of v_3.
- **WARNING:** From now on, when I use p without a subscript, p is the coordinates **for the entire swarm.** In other words, p is a point in Nd-dimensional-space.

For example, 2D formation ($d = 2$) of 3 robots ($N = 3$) has $p \in \mathbb{R}^{Nd} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \end{bmatrix}$ which I have chosen to represent as a column vector of length $N \times d = 2 \times 3 = 6$

Rigidity Mapping

- **Rigidity mapping** (sometimes referred to as a **cumulative constraint function**)

$$\rho_{\mathcal{G}} = \{\cdots \ \rho_{ij} \ \cdots\} : \mathbb{R}^{Nd} \rightarrow \mathbb{R}^{|\mathcal{E}|}$$

  - For each edge $(i, j) \in \mathcal{E}$, the rigidity mapping is typically defined as:

$$\rho_{ij}(p_i, p_j) = \| p_i - p_j \|^2$$

  - Note that it is the **squared** distance.
  - The rigidity mapping maps from the space of swarm points p (Nd-dimensional space) to the space of relative distances ($|\mathcal{E}|$-dimensional space, where $|\mathcal{E}|$ is the number of edges)

Rigidity Mapping

- Again, clunky notation, but the idea is straightforward. You can think of a rigidity mapping as vector function (vector input → vector output) although there are subtle distinction between coordinates defining a point in space vs. vector in vector space mapping from one set to another.

$$\rho_{\mathcal{G}} = \{\cdots \ \rho_{ij} \ \cdots\} : \mathbb{R}^{Nd} \rightarrow \mathbb{R}^{|\mathcal{E}|}$$

$$\rho_{ij}(p_i, p_j) = \| p_i - p_j \|^2$$

Rigidity Mapping: Example

$$\rho_{\mathcal{G}} = \{\cdots \ \rho_{ij} \ \cdots\} : \mathbb{R}^{Nd} \rightarrow \mathbb{R}^{|\mathcal{E}|}$$

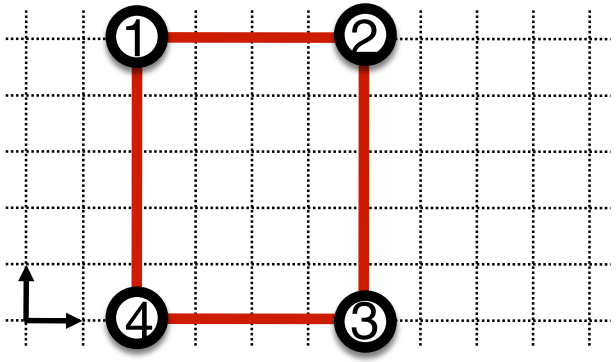
$$\rho_{ij}(p_i, p_j) = \| p_i - p_j \|^2$$

Easiest to think of the rigidity mapping entry by entry

$$\rho_{12}(p_1, p_2) = \|p_1 - p_2\|^2 = (x_1 - x_2)^2 + (y_1 - y_2)^2 = 4^2 + 0^2 = 16$$

General form

Specific value (16 in this case)



$$\mathcal{V} = \{v_1, v_2, v_3, v_4\}$$

$$\mathcal{E} = \{d_{12} = 4, d_{23} = 5, d_{34} = 4, d_{41} = 5\}$$

$$p = \{(v_1, (2,5)), (v_2, (6,5)), (v_3, (6,0)), (v_4, (0,2))\}$$

$$\rho_{\mathcal{G}}\left(\underbrace{\begin{bmatrix} 2 \\ 5 \\ 6 \\ 5 \\ 6 \\ 0 \\ 0 \\ 2 \end{bmatrix}}_{p \in \mathbb{R}^{Nd}}\right) = \begin{bmatrix} 16 \\ 25 \\ 16 \\ 25 \end{bmatrix} \in \mathbb{R}^{|\mathcal{E}|}$$

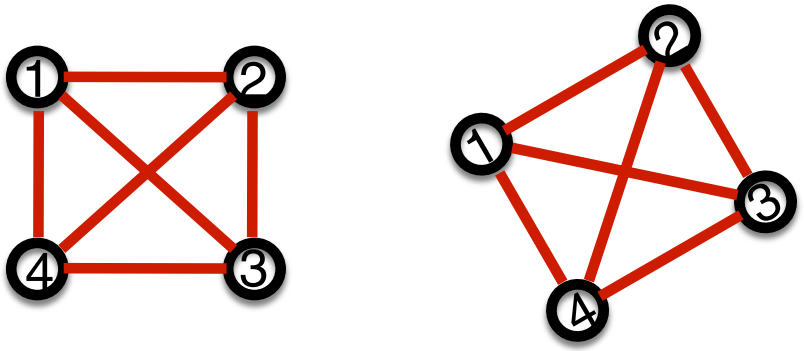
Basic Notation Done

- We have gone through the clunky notation of
  - Specifications
  - Frameworks
  - Rigidity mapping

- Next we will move on to rigidity and the different types of rigidity that don’t neatly relate to one another, so try not to be confused.

Geometric Rigidity: Idea

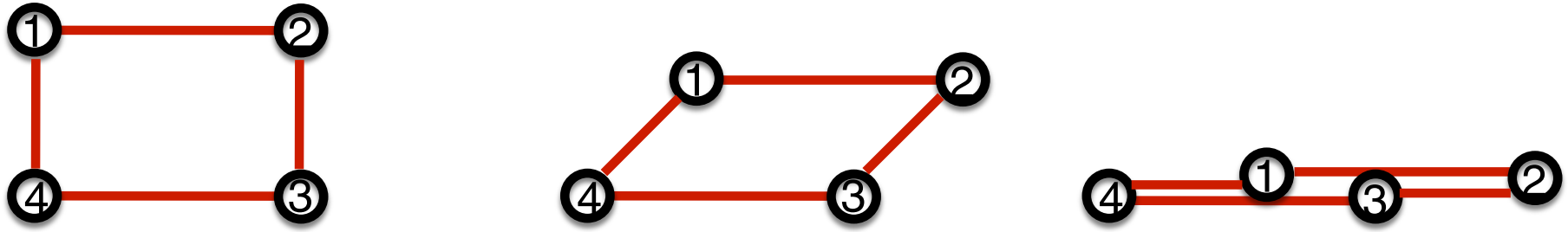
- Given N agents and M pair-wise geometrical constraints (edges), do the constraints unambiguously determine the shape of the agents?
  - If $M = N(N–1)/2$, we have a complete graph K_n whose shape of the formation is unambiguous up to a rotation and translation in space



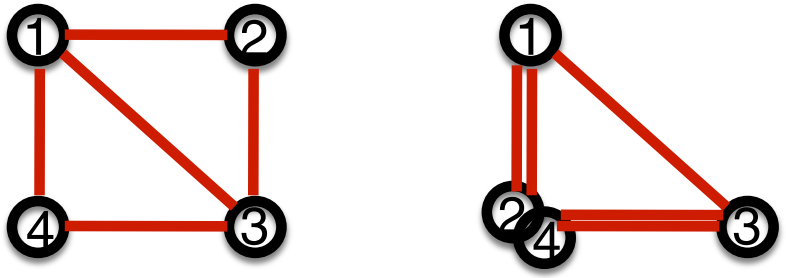
  - What if $M < N(N–1)/2$?

Geometric Rigidity: Idea

- Examples in 2D: constraints are relative distances between agents
- If $M = 4$, there is an infinite number of shapes that agree with the constraints

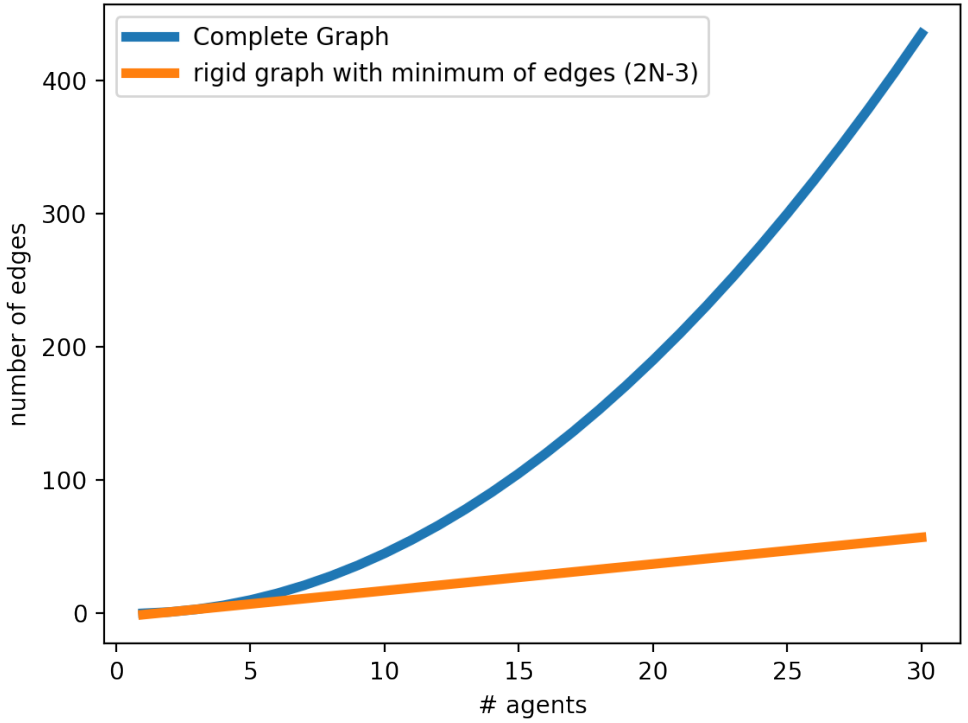


- If $M = 5$, the shape is preserved
  - (continuous deformations are not possible anymore)



Geometric Rigidity: Idea

- How can we characterize/find rigid structures that are not complete graphs?
- Complete graph: $N(N-1)/2$ edges
  - Number of connections/control/constraint enforcement is $O(N^2)$
- In 2D, $2N-3$ properly chosen edges are sufficient to enforce a certain shape ($O(N)$ number of edges) → distributed control, locality, etc.)
  - Can prove this inductively
  - How about 3D?



Why is real analysis back again?

- Rigidity is about resistance to deformations
  - Deformations relate to continuity
  - What happens if I wiggle the formation?

- Rigidity will be defined with respect to a **neighborhood** and a **rigidity mapping**

2.18 Definition Let X be a metric space. All points and sets mentioned below are understood to be elements and subsets of X.

(*a*) *A neighborhood* of p is a set $N_r(p)$ consisting of all q such that $d(p, q) < r$, for some $r > 0$. The number r is called the *radius* of $N_r(p)$.

(*b*) A point p is a *limit point* of the set E if *every* neighborhood of p contains a point $q \neq p$ such that $q \in E$.

(*c*) If $p \in E$ and p is not a limit point of E, then p is called an *isolated point* of E.

(*d*) E is *closed* if every limit point of E is a point of E.

(*e*) A point p is an *interior* point of E if there is a neighborhood N of p such that $N \subset E$.

(*f*) E is *open* if every point of E is an interior point of E.

(*g*) The *complement* of E (denoted by E^c) is the set of all points $p \in X$ such that $p \notin E$.

(*h*) E is *perfect* if E is closed and if every point of E is a limit point of E.

(*i*) E is *bounded* if there is a real number M and a point $q \in X$ such that $d(p, q) < M$ for all $p \in E$.

(*j*) E is *dense in* X if every point of X is a limit point of E, or a point of E (or both).

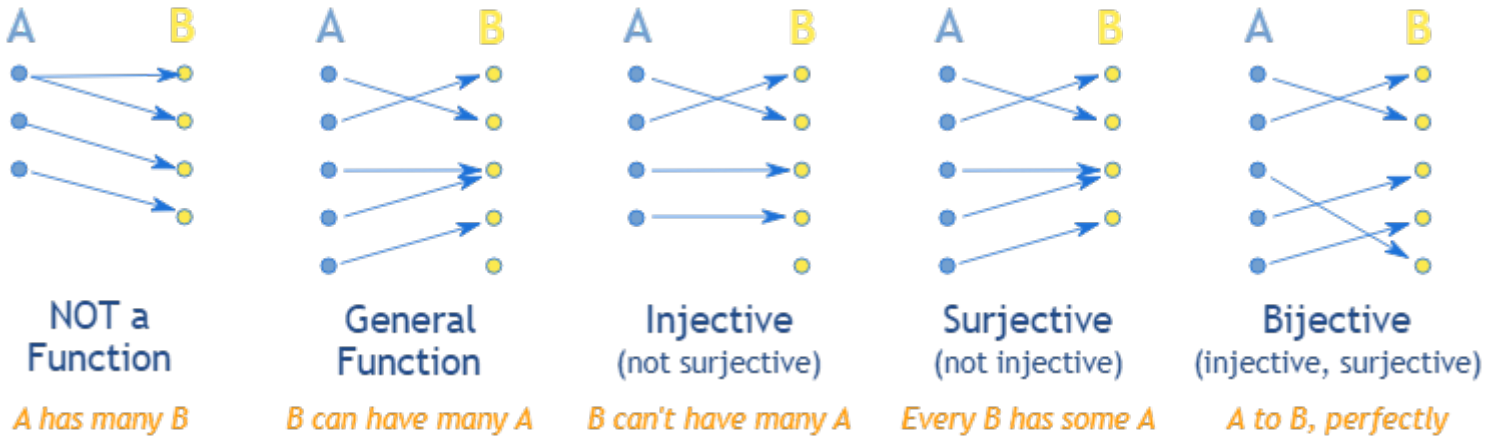
Let us note that in R^1 neighborhoods are segments, whereas in R^2 neighborhoods are interiors of circles.

- Principles of Mathematical Analysis, Walter Rudin

- Note that limit points include both what we think of as "boundary" points as well as interior points.

Rigidity Mapping

- Inverse mappings are not always functions. The inverse of the rigidity mapping is often not a function!



1st Definition of Rigidity

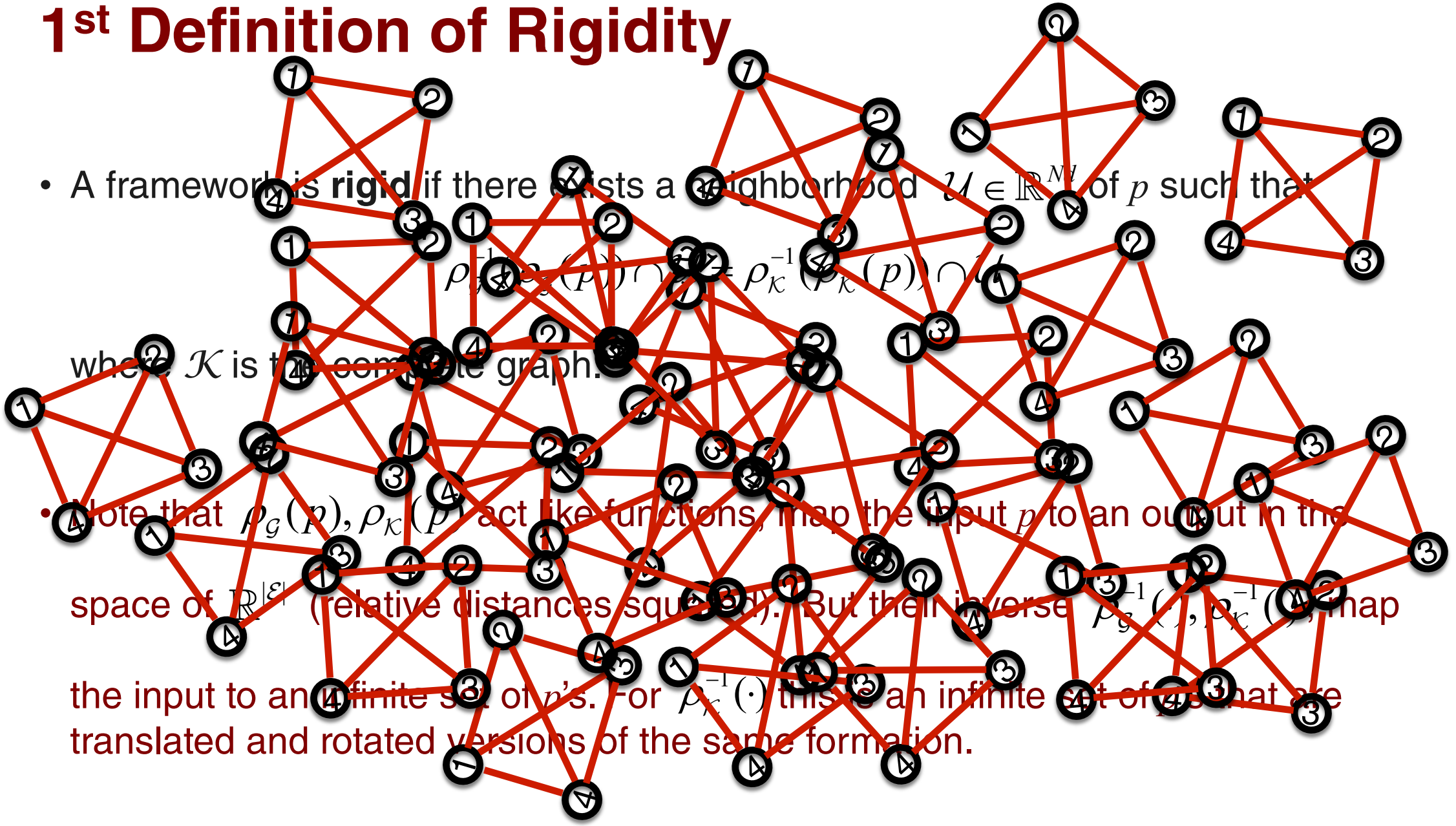
- A framework is **rigid** if there exists a neighborhood $\mathcal{U} \in \mathbb{R}^{Nd}$ of p such that

$$\rho_{\mathcal{G}}^{-1}(\rho_{\mathcal{G}}(p)) \cap \mathcal{U} = \rho_{\mathcal{K}}^{-1}(\rho_{\mathcal{K}}(p)) \cap \mathcal{U}$$

  where $\mathcal{K}$ is the complete graph.

- Note that $\rho_{\mathcal{G}}(p), \rho_{\mathcal{K}}(p)$ act like functions, map the input p to an output in the space of $\mathbb{R}^{|\mathcal{E}|}$ (relative distances squared). But their inverse $\rho_{\mathcal{G}}^{-1}(\cdot), \rho_{\mathcal{K}}^{-1}(\cdot)$, map the input to an infinite set of p's. For $\rho_{\mathcal{K}}^{-1}(\cdot)$ this is an infinite set of p's that are translated and rotated versions of the same formation.

1st Definition of Rigidity



- A framework is **rigid** if there exists a neighborhood $\mathcal{U} \in \mathbb{R}^{Nd}$ of p such that

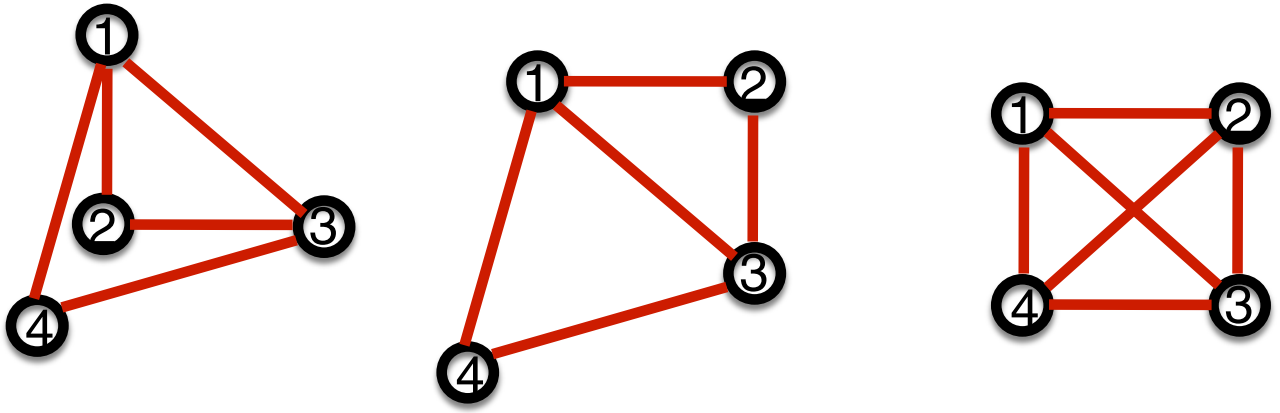
$$\rho_{\mathcal{G}}^{-1}(\rho_{\mathcal{G}}(p)) \cap \mathcal{U} = \rho_{\mathcal{K}}^{-1}(\rho_{\mathcal{K}}(p)) \cap \mathcal{U}$$

where $\mathcal{K}$ is the complete graph.

- Note that $\rho_{\mathcal{G}}(p), \rho_{\mathcal{K}}(p)$ act like functions, map the input p to an output in the space of $\mathbb{R}^{|\mathcal{E}|}$ (relative distances squared). But their inverse $\rho_{\mathcal{G}}^{-1}(\cdot), \rho_{\mathcal{K}}^{-1}(\cdot)$, map the input to an infinite set of p's. For $\rho_{\mathcal{K}}^{-1}(\cdot)$ this is an infinite set of p's that are translated and rotated versions of the same formation.

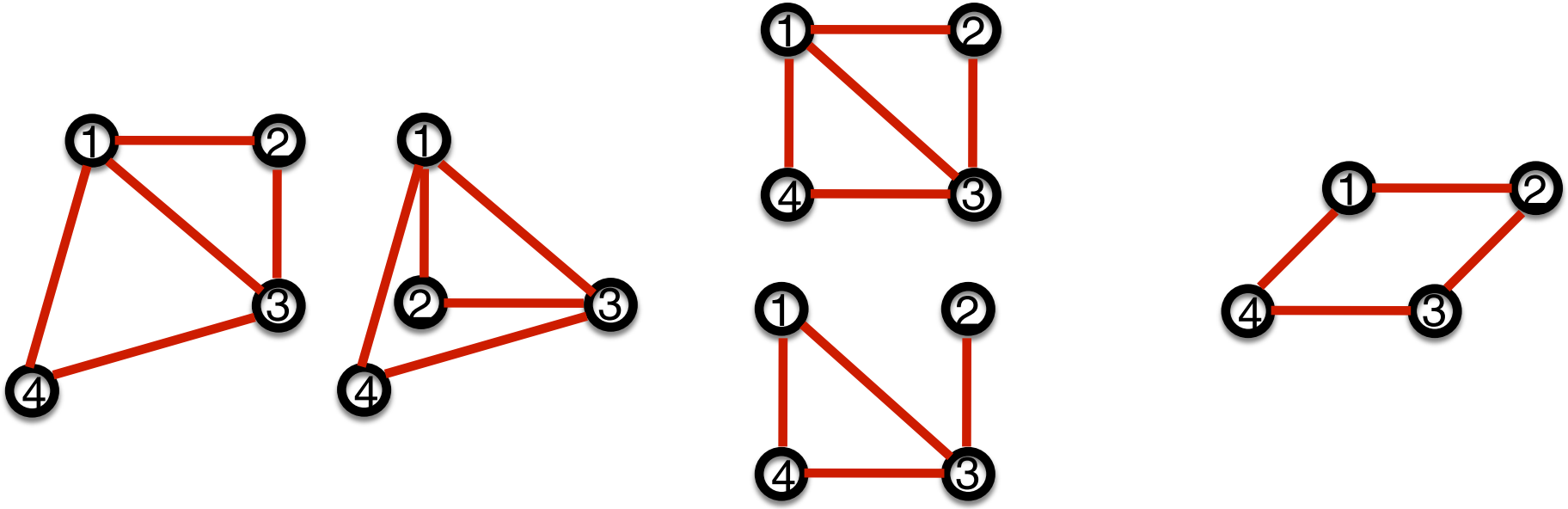
1st Definition of Rigidity

- Based on that 1st definition of rigidity,
  - A framework is **globally rigid** if $\mathcal{U} = \mathbb{R}^{Nd}$
  - A framework is **minimally rigid** if the removal of any edge yields a non-rigid framework



Equivalence and Congruence of Frameworks

- Two frameworks $(\mathcal{G}, p_1)$ and $(\mathcal{G}, p_2)$ are **equivalent** if $\rho_{\mathcal{G}}(p_1) = \rho_{\mathcal{G}}(p_2)$ (constraints are satisfied over the edges in $\mathcal{E}$)
- Two frameworks $(\mathcal{G}, p_1)$ and $(\mathcal{G}, p_2)$ are **congruent** if $\rho_{\mathcal{K}}(p_1) = \rho_{\mathcal{K}}(p_2)$ (constraints are satisfied over all possible edges)



Alternative (2nd) Definition of Rigidity

- Two frameworks $(\mathcal{G}, p_1)$ and $(\mathcal{G}, p_2)$ are **equivalent** if $\rho_{\mathcal{G}}(p_1) = \rho_{\mathcal{G}}(p_2)$ (constraints are satisfied over the edges in $\mathcal{E}$)
- Two frameworks $(\mathcal{G}, p_1)$ and $(\mathcal{G}, p_2)$ are **congruent** if $\rho_{\mathcal{K}}(p_1) = \rho_{\mathcal{K}}(p_2)$ (constraints are satisfied over all possible edges)

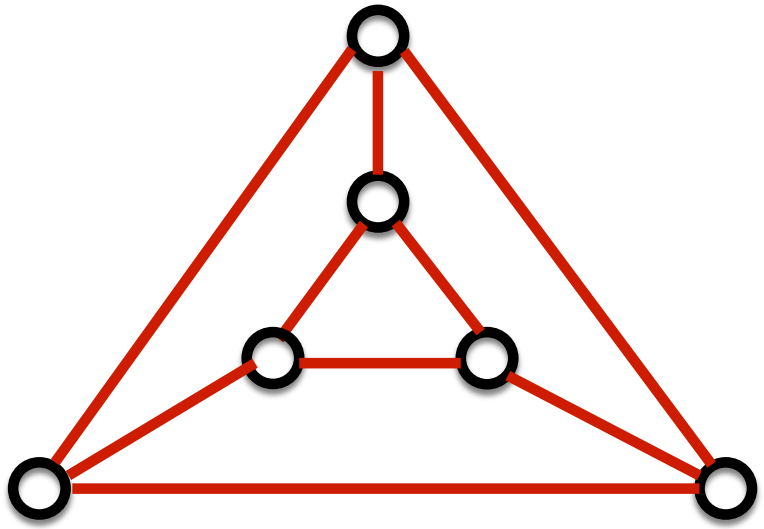
- **Alternative definition of rigidity:** A framework $(\mathcal{G}, p_1)$ if rigid if all the frameworks $(\mathcal{G}, p_2)$ and $p_2 \in \mathcal{U}(p_1)$, which are equivalent to $(\mathcal{G}, p_1)$ are also congruent to $(\mathcal{G}, p_1)$.

1st , 2nd, and 3rd Definitions of Rigidity

- 1st and 2nd are both pretty much useless for us
  - No obvious method to go from definition of rigidity to a linear/nonlinear control law
- We will now consider a 3rd Definition of rigidity, referred to as, **infinitesimal rigidity**
  - To avoid confusion, let's think of it first as an entirely different approach to defining rigidity

Infinitesimal Rigidity

- In some sense, infinitesimal rigidity is stricter than the 1st definition of rigidity.
- Under the 1st definition of rigidity, this framework is **rigid**, **globally rigid**, AND minimally **rigid**.

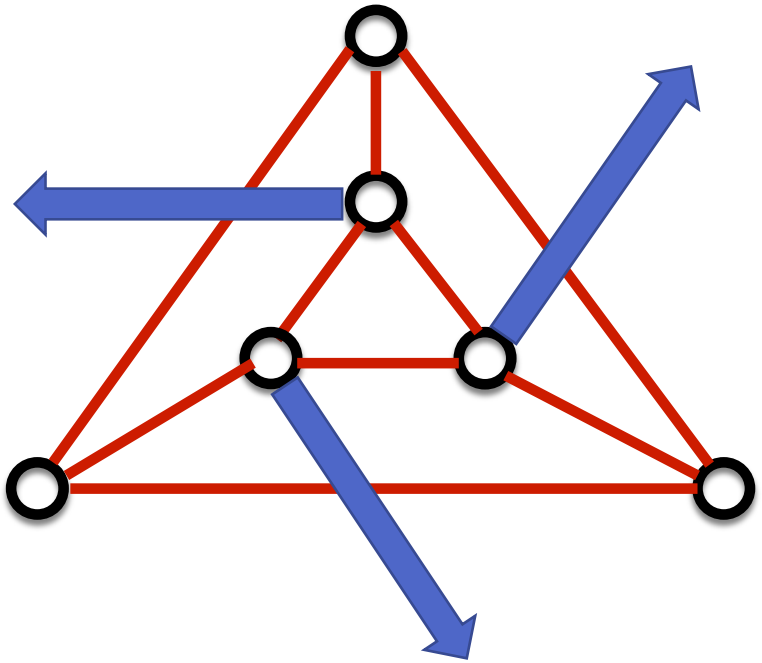


Checking this equation also works out

$$2N - 3 = 2 \cdot 6 - 3 = 9$$ edges

Infinitesimal Rigidity

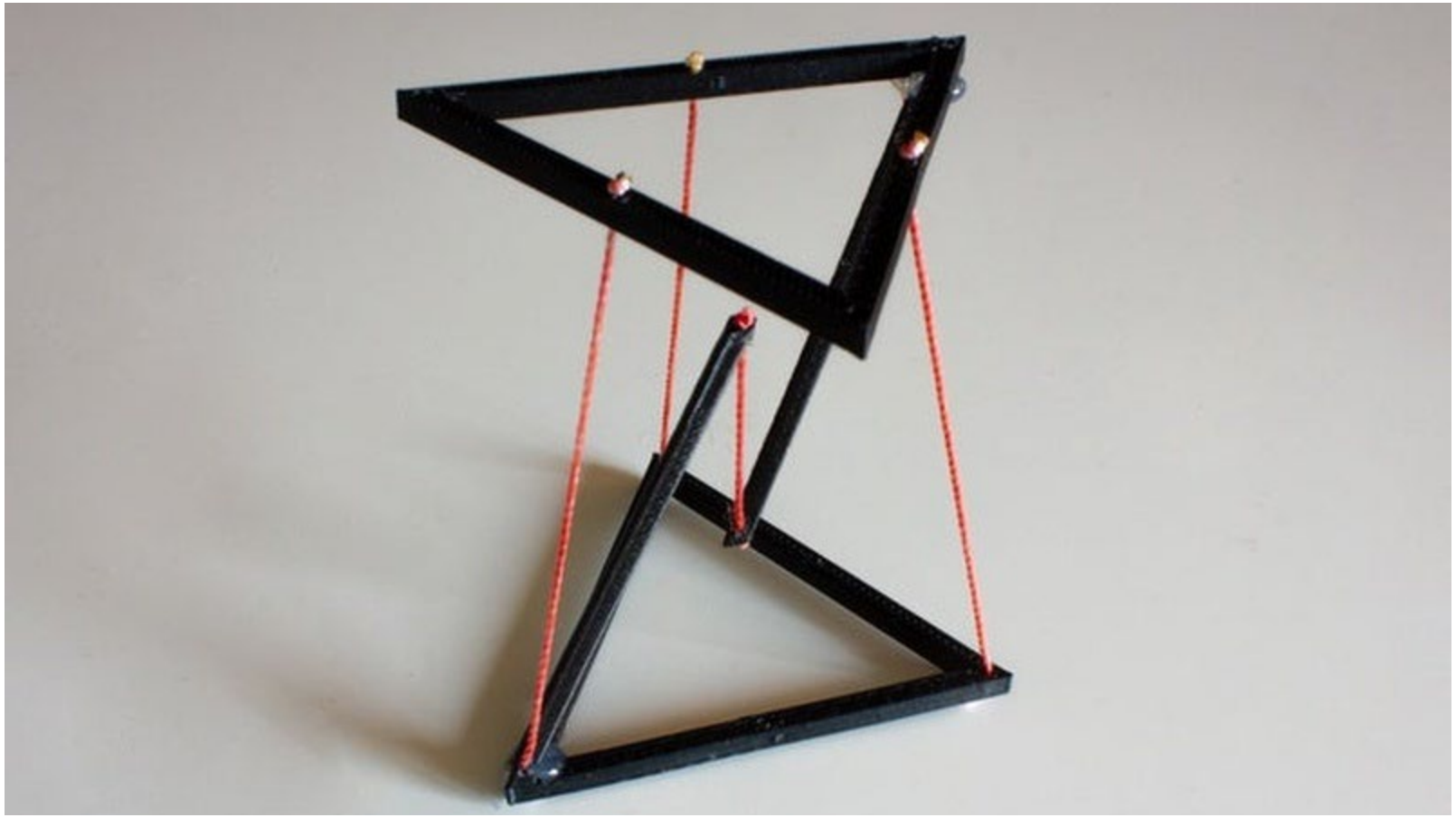
- In some sense, infinitesimal rigidity is stricter than the 1st definition of rigidity.
- Under the 1st definition of rigidity, this framework is **rigid**, **globally rigid**, AND minimally **rigid**.



- However, in this current configuration, it is not **infinitesimally rigid** because I could apply **infinitesimal motions**, i.e., can apply angular velocities, at some vertices while still preserving all the distance constraints.

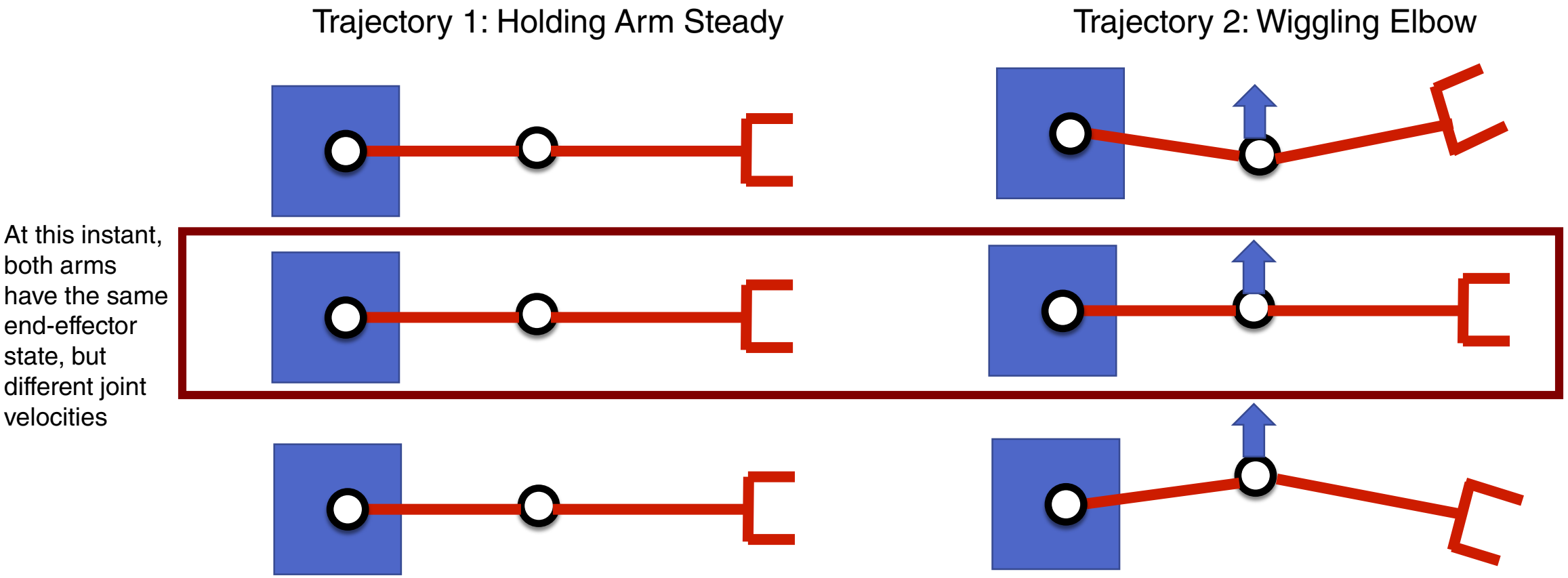
Infinitesimal Rigidity

- Shows up in the design of tensegrity structures



Analogy to Manipulator Jacobian in Singular Pose

- In a robotic arm in a straight-arm configuration, I can actually apply an angular velocity upwards at the elbow, while preserving the end-effector position and velocity



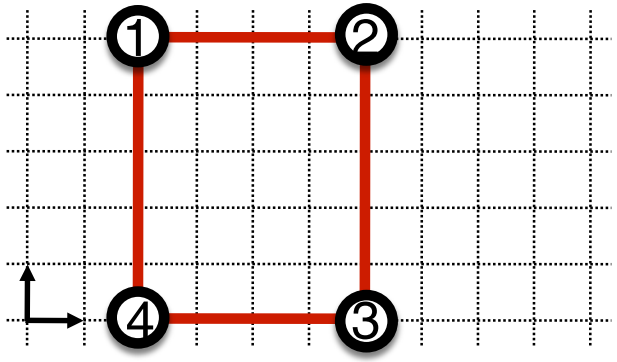
Analogy to Manipulator Jacobian in Singular Pose

- As seen in our crude “animation” in the previous slide,
  - When the manipulator Jacobian is singular, the robot can “wiggle” while satisfying whatever end-effector position and velocity constraint at a particular instant in time.
    - The null-space of the Jacobian at a particular singular pose, is the set of all possible wiggling motions!
  - Checking whether a Jacobian is singular, is a simple exercise in linear algebra, to check the *rank* of the Jacobian
  - Hopefully, the analogy to robotics will help with your intuition for when we formally introduce the definition of **infinitesimal rigidity** coming up soon

Rigidity Matrix

- **Definition:** The rigidity matrix $R_{\mathcal{G}} \in \mathbb{R}^{|\mathcal{E}| \times Nd} \rightarrow \mathbb{R}$ is defined as $R_{\mathcal{G}} = \dfrac{\partial \rho_{\mathcal{G}}(p)}{\partial p}$

- Think about computing it entry by entry as a matrix derivative (i.e., Jacobian)



$$\mathcal{V} = \{v_1, v_2, v_3, v_4\}$$

$$\mathcal{E} = \{d_{12} = 4, d_{23} = 5, d_{34} = 4, d_{41} = 5\}$$

$$p = \{(v_1, (2,5)), (v_2, (6,5)), (v_3, (6,0)), (v_4, (0,2))\}$$

$$R_{\mathcal{G}} = \frac{\partial \rho_{\mathcal{G}}(p)}{\partial p} = \begin{bmatrix} \frac{\partial \rho_{12}}{\partial x_1} & \frac{\partial \rho_{12}}{\partial y_1} & \frac{\partial \rho_{12}}{\partial x_2} & \frac{\partial \rho_{12}}{\partial y_2} & \frac{\partial \rho_{12}}{\partial x_3} & \frac{\partial \rho_{12}}{\partial y_3} & \frac{\partial \rho_{12}}{\partial x_4} & \frac{\partial \rho_{12}}{\partial y_4} \\ \frac{\partial \rho_{23}}{\partial x_1} & \frac{\partial \rho_{23}}{\partial y_1} & \frac{\partial \rho_{23}}{\partial x_2} & \frac{\partial \rho_{23}}{\partial y_2} & \frac{\partial \rho_{23}}{\partial x_3} & \frac{\partial \rho_{23}}{\partial y_3} & \frac{\partial \rho_{23}}{\partial x_4} & \frac{\partial \rho_{23}}{\partial y_4} \\ \frac{\partial \rho_{34}}{\partial x_1} & \frac{\partial \rho_{34}}{\partial y_1} & \frac{\partial \rho_{34}}{\partial x_2} & \frac{\partial \rho_{34}}{\partial y_2} & \frac{\partial \rho_{34}}{\partial x_3} & \frac{\partial \rho_{34}}{\partial y_3} & \frac{\partial \rho_{34}}{\partial x_4} & \frac{\partial \rho_{34}}{\partial y_4} \\ \frac{\partial \rho_{41}}{\partial x_1} & \frac{\partial \rho_{41}}{\partial y_1} & \frac{\partial \rho_{41}}{\partial x_2} & \frac{\partial \rho_{41}}{\partial y_2} & \frac{\partial \rho_{41}}{\partial x_3} & \frac{\partial \rho_{41}}{\partial y_3} & \frac{\partial \rho_{41}}{\partial x_4} & \frac{\partial \rho_{41}}{\partial y_4} \end{bmatrix}$$

Nothing hard, just annoying. Taking many partial derivatives of $(x_i - x_j)^2 + (y_i - y_j)^2$

Back to Infinitesimal rigidity

- A framework is **infinitesimally rigid** if $\ker(R_{\mathcal{G}}(p)) = \ker(R_{\mathcal{K}}(p))$ or, equivalently, $rank(R_{\mathcal{G}}(p)) = rank(R_{\mathcal{K}}(p))$ *
  - $\ker$ refers to null-space
  - *Exception when $R_{\mathcal{K}}(p)$ is not a regular point (see next slide about definition of a regular point and also the Brain Teaser at the end)
- Note that infinitesimal motions (velocities) $\dot{p}(t)$ will preserve all of the relative distance constraints (satisfy $R_{\mathcal{G}}\dot{p} = 0$) if and only if $\dot{p}(t) \in \ker(R_{\mathcal{G}}(p))$
  - Proof outline:
    - Satisfying constraints means that the rigidity mapping is constant $\rho_{\mathcal{G}}(p)$, i.e., doesn't change with time. Hence, by chain rule:

$$\frac{d}{dt}\rho_{\mathcal{G}}(p) = 0$$

$$\frac{\partial \rho_{\mathcal{G}}(p)}{\partial p}\frac{dp}{dt} = 0$$

Comparing Definitions

- **1st definition of rigidity:** Somewhat intuitive but absolutely useless
- **Infinitesimal rigidity:** Somewhat less intuitive. Also restrictive because it does not allow infinitesimal motion. However, it is useful because it results in algebraic conditions for rigidity (checking rank of a matrix).
- An **infinitesimally rigid** framework will also satisfy the **1st definition of rigidity**, but the converse is not always true!
- Satisfying the **1st definition of rigidity** *and* being a regular point is equivalent to **infinitesimal rigidity**.
  - **Definition:** A point $\overline{p}$ is a regular point if $rank(R_{\mathcal{G}}(\overline{p})) = \max_{p} rank(R_{\mathcal{G}}(p))$

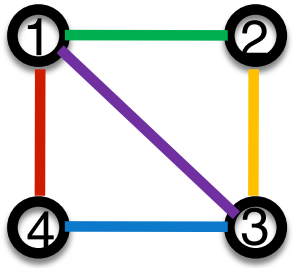
Regular if rank does not decrease at that point!

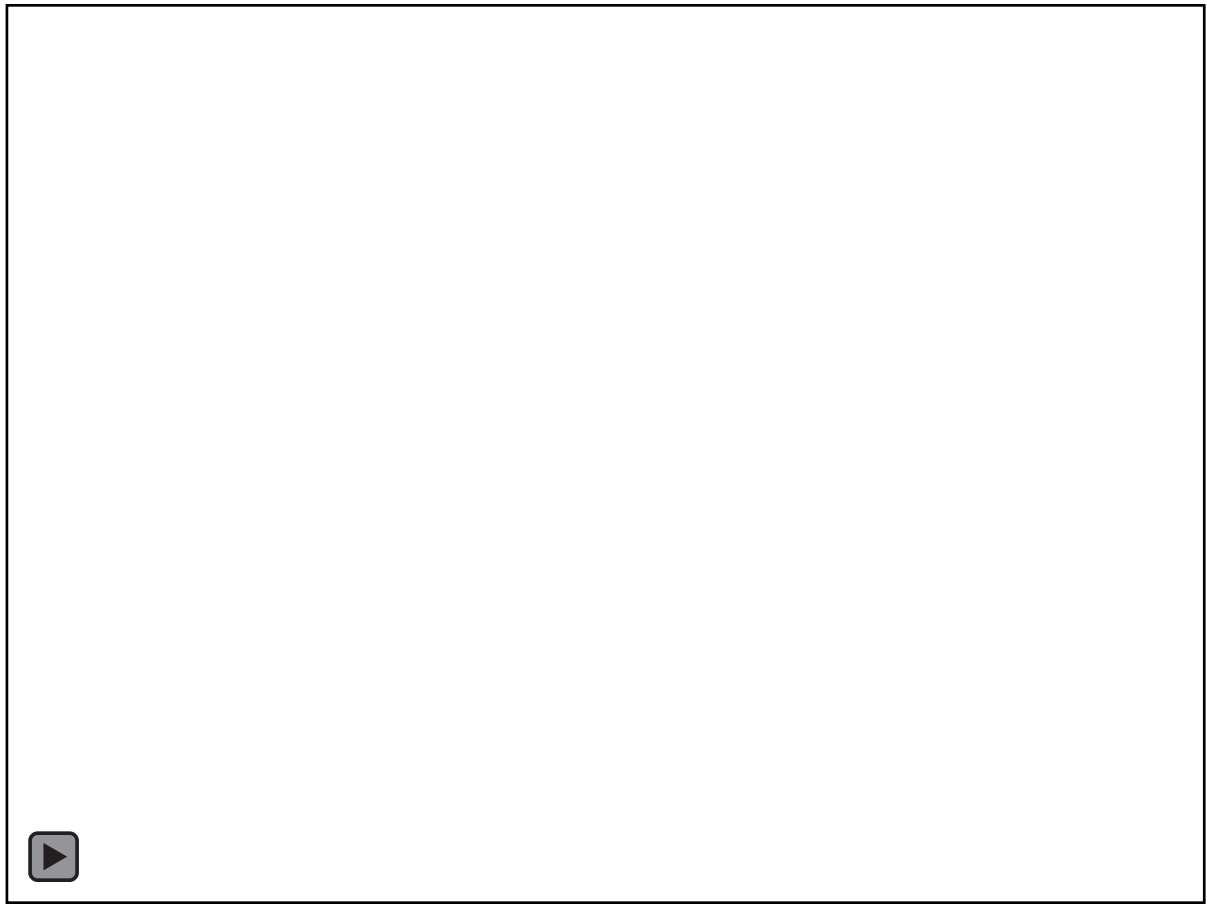
Back to Rigidity Matrix

- The rigidity matrix is fundamental for control:
  - Link between agent motions and constraints
  - Its null space is the space of all constraint-preserving motions
  - Rigidity of a framework is equivalent to a rank condition

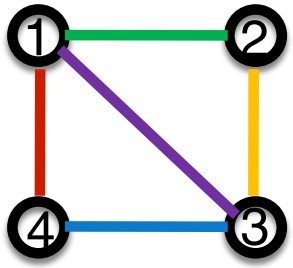
- Like the Laplacian, the structure of the rigidity matrix is very adaptable to some local protocol

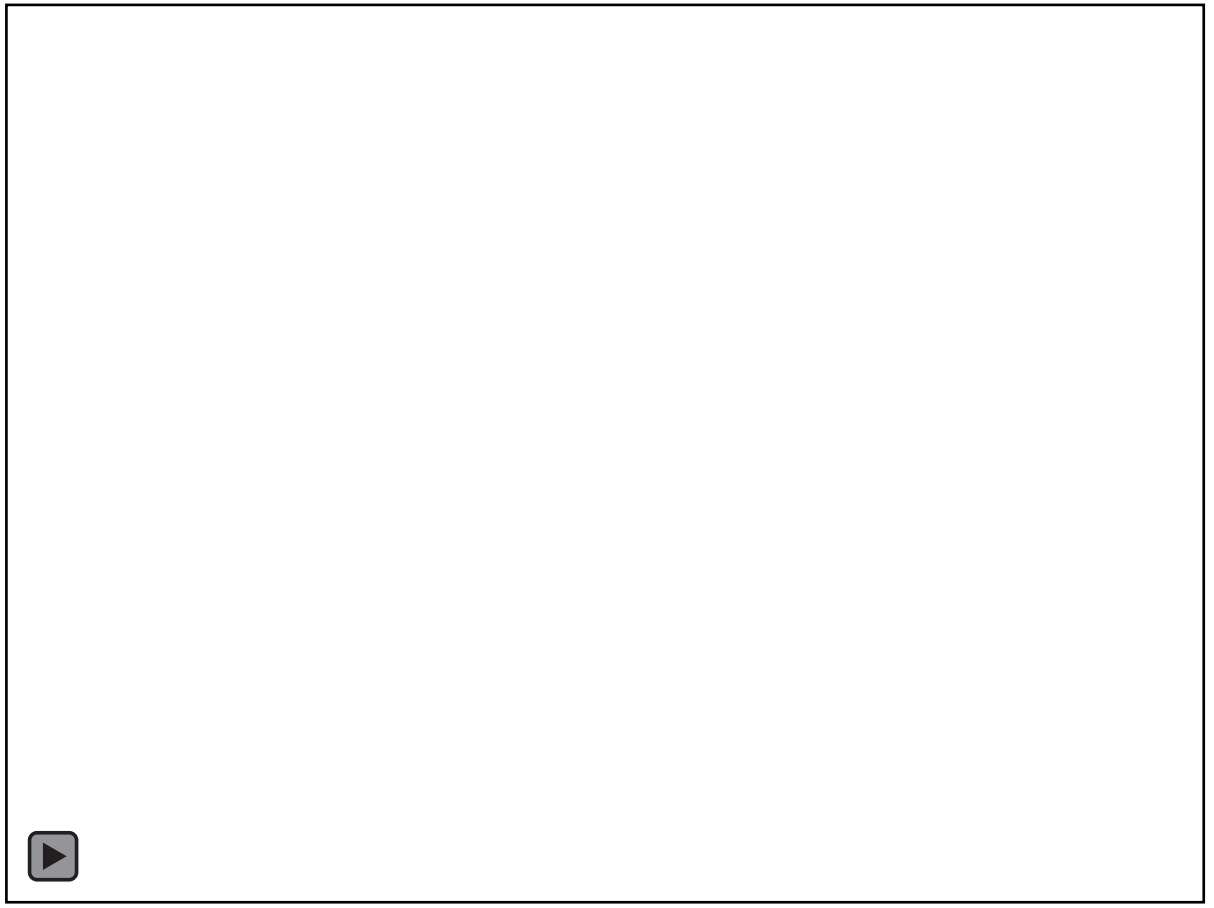
Rigidity-based control of formations



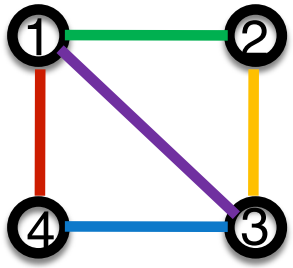


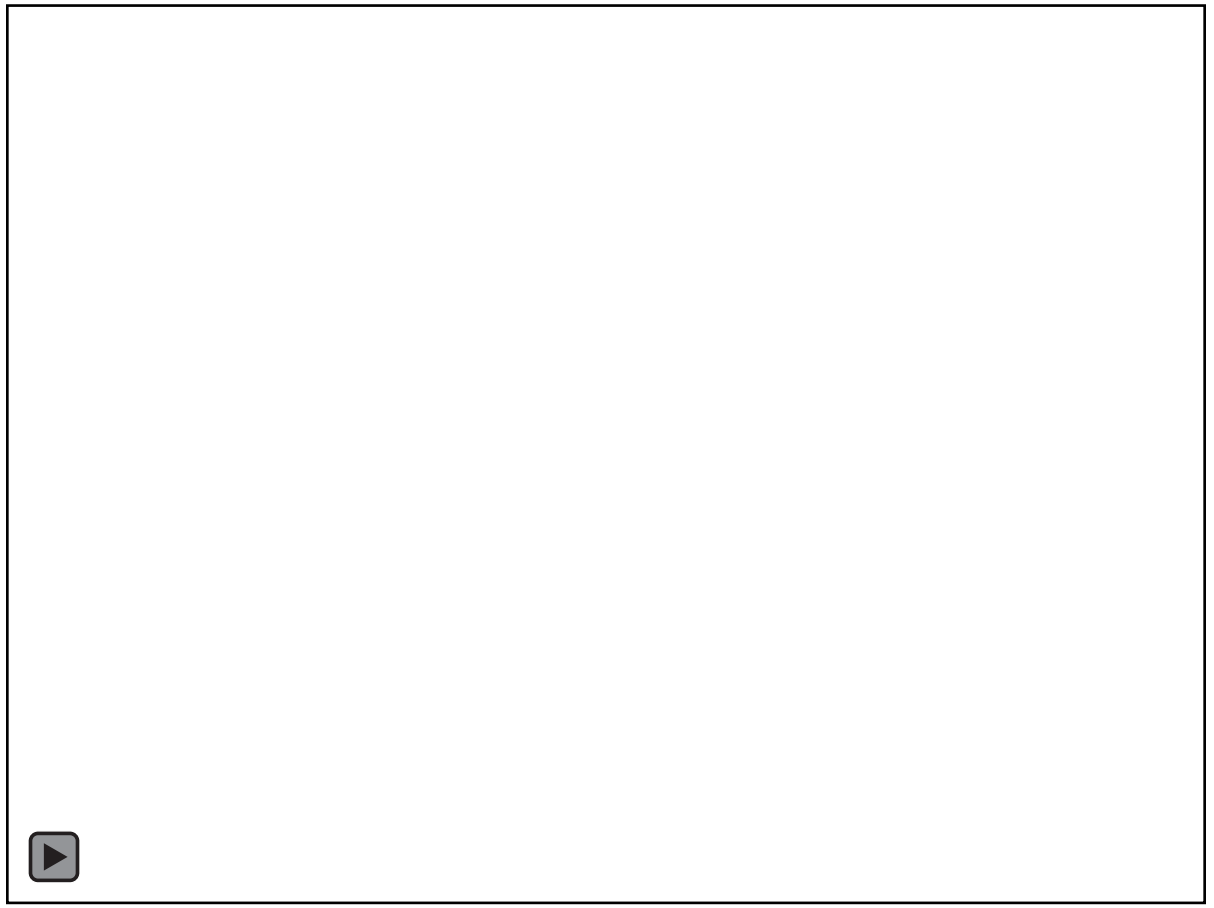
Rigidity-based control of formations



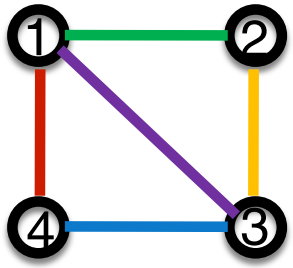


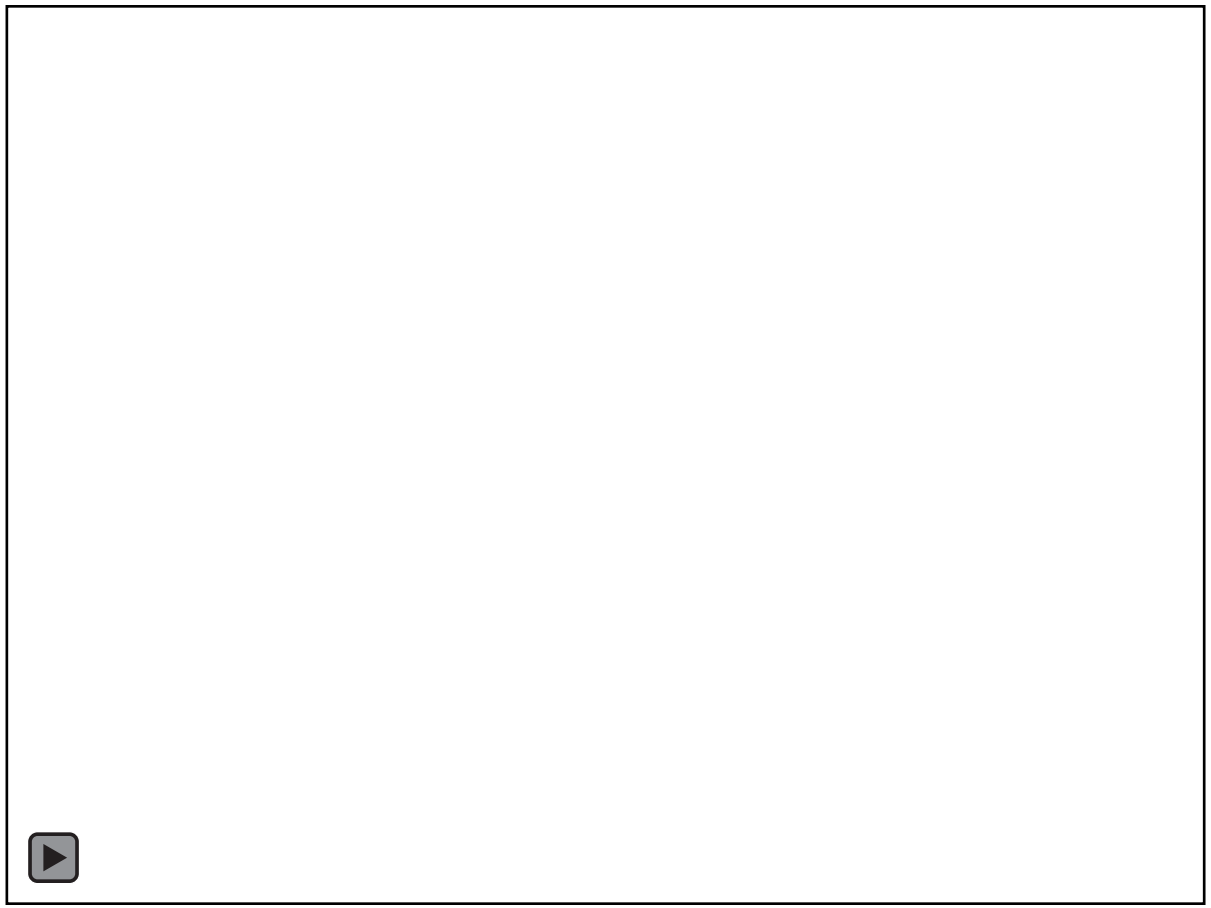
Rigidity-based control of formations



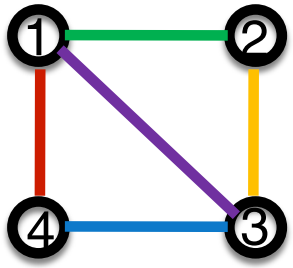


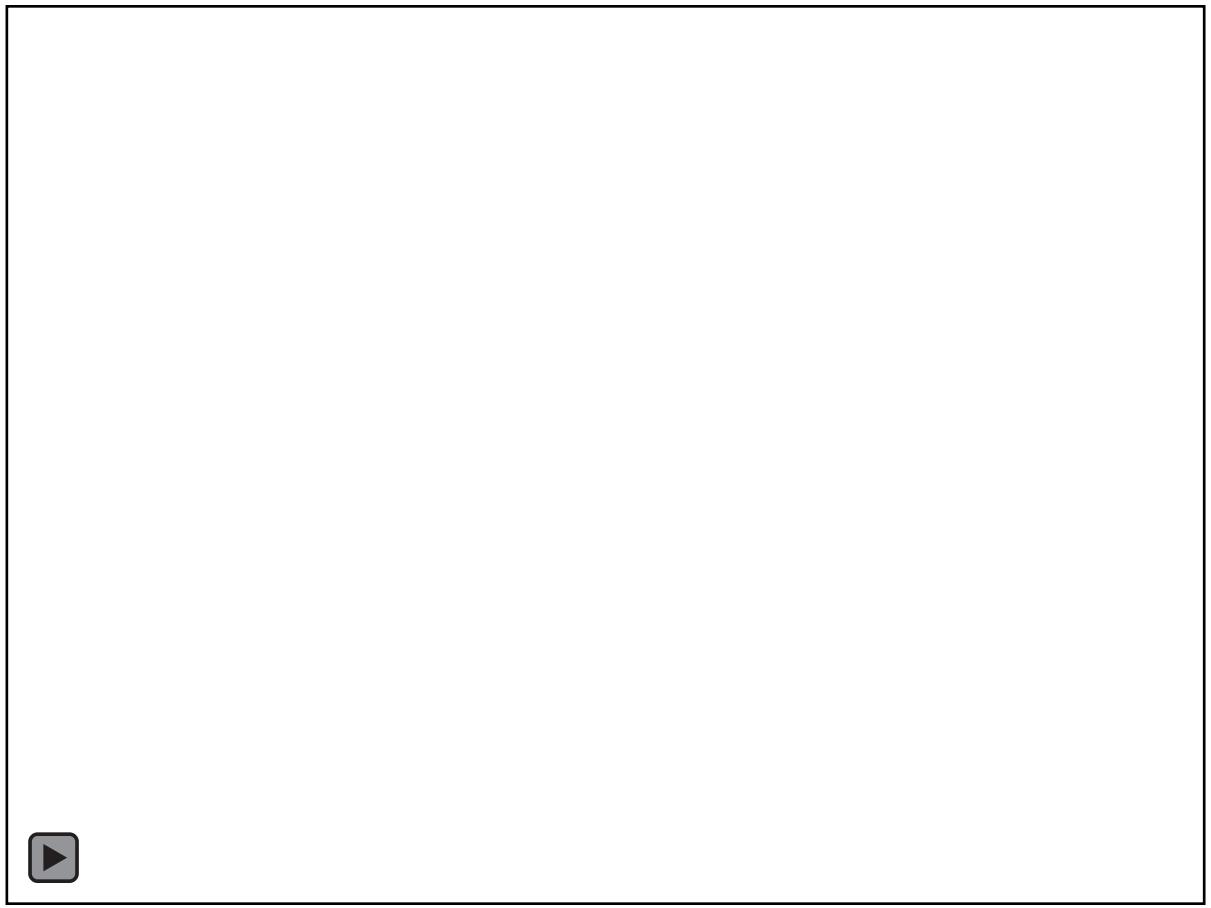
Rigidity-based control of formations





Rigidity-based control of formations





Perspectives Gained From Rigidity Matrix

- The rigidity offers a reason for why we need ***at least*** $2N–3$ edges.
- In $\mathbb{R}^2$, there are always ***at least*** 3 possible rigid motions while satisfying distance constraints (translation in 2 directions and 1 rotation). Hence, the kernel of the rigidity matrix, which is the space of all these possible formation-preserving motions, is at least 3.

$$\dim \ker(R_G) = 3 \rightarrow rank(R_G) = 2N - 3$$

Brain Teaser

- Consider a framework with 3 vertices and edges such that they form a complete graph

$$p : \mathcal{V} \to \mathbb{R}^2 = \{(v_1, (0,0)), (v_2, (1,d)), (v_3, (2,0))\}$$

- What is the rigidity matrix?
- For which values of *d* is the framework:
  - Infinitesimally rigid?
  - Rigid?

Brain Teaser Solution

- Rigidity matrix

$$\rho = \{\rho_{12}, \rho_{23}, \rho_{31}\} = \begin{bmatrix} (x_1 - x_2)^2 + (y_1 - y_2)^2 \\ (x_2 - x_3)^2 + (y_2 - y_3)^2 \\ (x_3 - x_1)^2 + (y_3 - y_1)^2 \end{bmatrix}, p = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \end{bmatrix}$$

$$R_g = \frac{\partial \rho}{\partial p} = \begin{bmatrix} 2(x_1 - x_2) & 2(y_1 - y_2) & -2(x_1 - x_2) & -2(y_1 - y_2) & 0 & 0 \\ 0 & 0 & 2(x_2 - x_3) & 2(y_2 - y_3) & -2(x_2 - x_3) & -2(y_2 - y_3) \\ -2(x_3 - x_1) & -(y_3 - y_1) & 0 & 0 & 2(x_3 - x_1) & 2(y_3 - y_1) \end{bmatrix}$$

$$= \begin{bmatrix} 2(0-1) & 2(0-d) & -2(0-1) & -2(0-d) & 0 & 0 \\ 0 & 0 & 2(1-2) & 2(d-0) & -2(1-2) & -2(d-0) \\ -2(2-0) & -(0-0) & 0 & 0 & 2(2-0) & 2(0-0) \end{bmatrix}$$

$$= \begin{bmatrix} -2 & -2d & 2 & 2d & 0 & 0 \\ 0 & 0 & -2 & 2d & 2 & -2d \\ -4 & 0 & 0 & 0 & 4 & 0 \end{bmatrix}$$

Rank is 3 (full-rank) when d is not zero
Rank drops to 2 when $d = 0$

Brain Teaser Solution

- Rigidity matrix

$$\rho = \{\rho_{12}, \rho_{23}, \rho_{31}\} = \begin{bmatrix} (x_1 - x_2)^2 + (y_1 - y_2)^2 \\ (x_2 - x_3)^2 + (y_2 - y_3)^2 \\ (x_3 - x_1)^2 + (y_3 - y_1)^2 \end{bmatrix}, p = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \end{bmatrix}$$

- Interestingly, this is a complete graph, but it is **not** infinitesimally rigid only when $d = 0$ while still being rigid (according to the 1st definition) for all values of d.

Bonus

- The $2N-3$ comes from a general formula $Nd - \frac{d(d+1)}{2}$ that comes from Maxwell's investigations into structural rigidity