Harvard CS 182: Artificial Intelligence September 21, 2022

Lecture 6: Multi-robot Systems

Lecturer: Stephanie Gil *Author: Mujin Kwun*

This lecture will serve as a practical overview on the current state of multi-robot systems and how it relates to topics covered in this class.

1 Introduction

1.1 Background

"A robot is a machine that carries out actions in the physical world using computer algorithms." It often uses sensors to capture information about the world, and uses these sensory inputs to inform its actions.

Examples of multi-robot systems:

- Manufacturing Robots - must work in cohesion to complete tasks in a factory
- NASA terrestrial planet finder - composed of modules that are deployed independently but together create a synthetic aperture for a telescope
- Disaster Relief - robots might be deployed into a disaster zone to perform reconnaissance or evacuation, particularly in areas that may be inaccessible to human responders
- Google Project Loon - on demand communication after disasters
- Lincoln Lab Perdix Swarm - autonomous sensors deployed that can work together to map an environment
- Delivery Drones
- Autonomous Vehicles - route autonomous vehicles, understand and predict the behavior of other agents
- Medical Robots - ingestible robots to return sensory information for accurate medical diagnoses

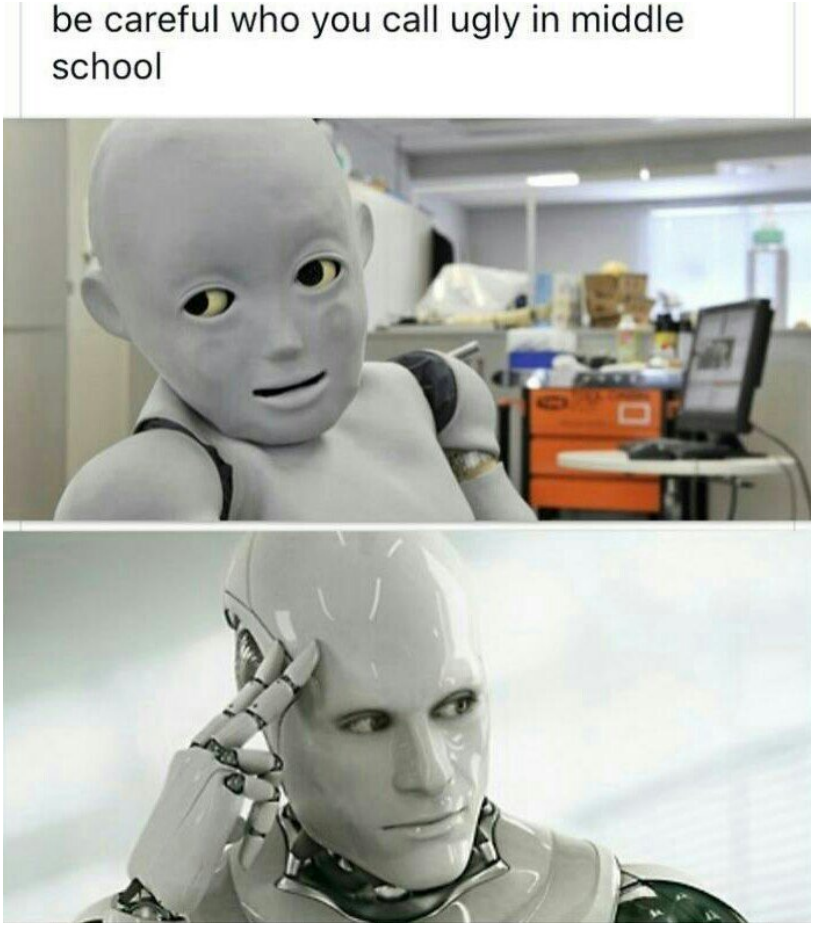

Figure 1: Robots

1.2 Types of Multi-robot Systems

1. Collective Swarm
    - Agents act independently
    - Minimal need for global information about other agents in the system
2. Intentionally Cooperative
    - Agents act together based on the state, actions, and capabilities of teammates in the system to accomplish a task
    - Need to know information about other agents in the environment

2 Multi-robot System Representation

2.1 Encoding Robot State

How do we make robots do what we want them to do?

We can formulate the behavior of these robots as an optimization problem, describing the change in state of a multi-robot system over time as a mathematical equation:

$$f : X * U \rightarrow X$$

where X is the state space and U is the input space. We can write this equation differently depending on whether this evolution of states is continuous/discrete and time dependent/independent.

1. Discrete, no time dependence: $x(l+1) = f(x(l), u(l))$
2. Discrete, time dependent: $x(l+1) = f(l, x(l), u(l))$
3. Continuous: $\dot{x}(t) = f(t, x(t), u(t))$

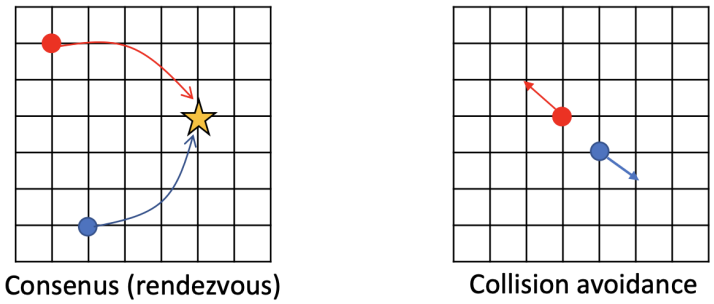

Figure 2: State Evolution

We are interested in how a state x changes over time according to $f(x)$. Furthermore, we are interested in equilibrium points where $f(x^*) = x^*$ as well as some **performance guarantees**:

- Conditions under which x^* exists
- Conditions under which x^* can be characterized or controlled

These performance guarantees depend on:

1. the function f - convex or nonconvex
2. connectivity - how one agent's behavior affects the behavior of other agents

2.2 Encoding Relationships

How do we represent agents and their relationships?

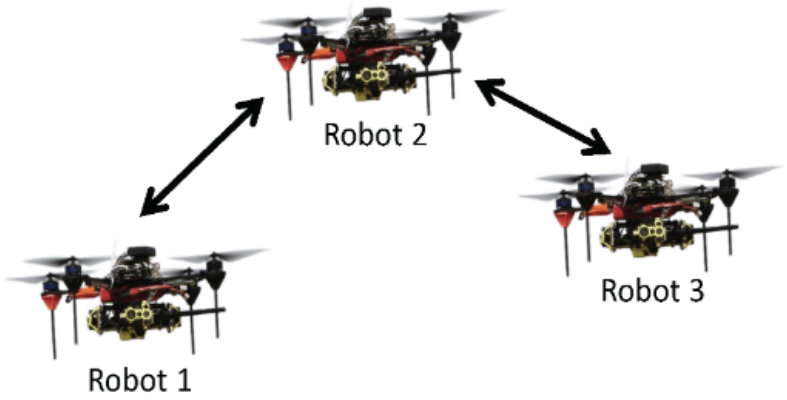

Figure 3: Graph Representation

Graphs: $G = (V, E)$

- Vertices - agents
- Edges - relationships between agents

We can use control graphs to depict different kinds of control architectures:

1. Centralized - one leader node organizes and gives commands to other agents in the system
2. Hierarchical - a hierarchy of agents that control different parts of the system
3. Decentralized - local clusters in which agents communicate with neighboring nodes
4. Hybrid - any mix of the above

In addition to describing control architectures, graphs can also be used to convey the flow of information in a system.

Communication:

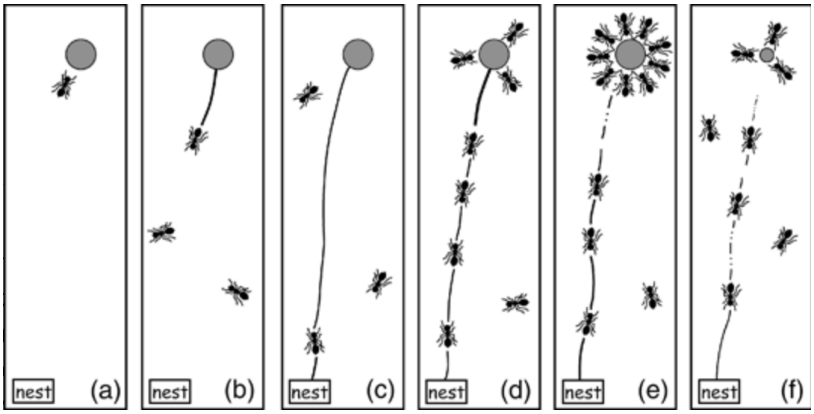

Figure 4: Stigmergy

1. Stigmergy - agents communicate by sensing and changing the environment, indirectly influencing the behavior of other agents.
   (a) For example, ants deposit pheromones on the environment for other ants to follow.
   (b) We can encode this as a graph by creating an "environment node" with edges to all other nodes on the graph
2. Passive - agents sense each other but don't directly communicate with one another.
3. Explicit - agents communicate directly with each other

Motivating Question: How can we combine control and information architectures for different tasks?

1. Foraging or Coverage
   - Weakly cooperative
   - Control: Decentralized
   - Communication: Stigmergy
2. Flocking or Formations
   - Control: any
   - Communication: Passive
3. Box-pushing, Cooperative Manipulation
   - Communication: Explicit
4. Traffic Control, Multi Robot Path Planning
   - Communication: Explicit

2.3 Main Research Challenges

The broad goal of research in autonomous networked robots is to combine communication, perception, and control to enable new capabilities.

1. Networking - What kind of information exchange can our system support?
2. Communication - What kind of information *should* we exchange?
3. Control - Execution of commands
4. Perception
5. Decision Making - What commands should we execute to achieve a given goal?
6. Adapting

3 Controlling Group Behavior

3.1 Dynamics

Motivating Question: What is the difference between robot networks and computer or sensor networks?

Robots can be controlled to physically interact with, and even change, its environment. Therefore, the information acquired from the environment and from neighbors changes as well as the actions taken. Dynamic equations like those shown earlier can capture information about individuals, but interactions are still better modelled with graphs.

3.2 Graph Theory

Figure 5: Starlings

Motivating Observation: Swarms of starling birds - Individual birds achieve aggregate motion and shape

Adjacency matrices to encode information on how agent states influence each other and tell us about underlying graph structure. Given adjacency matrix A, we can exponentiate A, and A^k_{ij} tells us the number of paths composed of directed edges from i to j.

Connectivity:

1. Globally Reachable: A vertex is globally reachable if it can be reached from any other vertex via a directed path
2. Strongly Connected: A directed graph is strongly connected if every vertex is globally reachable. Strongly connected and connected are the same for undirected graphs.

Laplacian: The Laplacian matrix for a graph is the degree matrix minus the adjacency matrix.

3.3 Mappings

Controlling a large group of agents, like the swarm of starling birds in the previous motivating observation, may be difficult and computationally intensive as you need to track and control each individual agent. To mitigate this issue, we find a mapping or abstraction from the high dimensional construction of the problem to a lower dimension space.

For example, we may choose to represent the swarm of starlings with its centroid and track/plan the swarm's trajectory using a single point. If we wish to retain information

about the shape of the group, we may use something like an ellipsoid to represent our swarm instead.

4 Gradient Based Control

Start by constructing a functions f

$$x(t+1) = x(t) + \alpha u(t)$$

$$u(t) = -\frac{\partial f(x(t))}{\partial x(t)}$$

We can control the state of our team by performing gradient descent on our potential function f. For example, in a navigation task, we might construct a potential field where obstacles and natural bounds in the environment are assigned high potential while points we want to visit are assigned low values. Performing gradient descent will lead us to avoid obstacles while navigating toward the sinks.

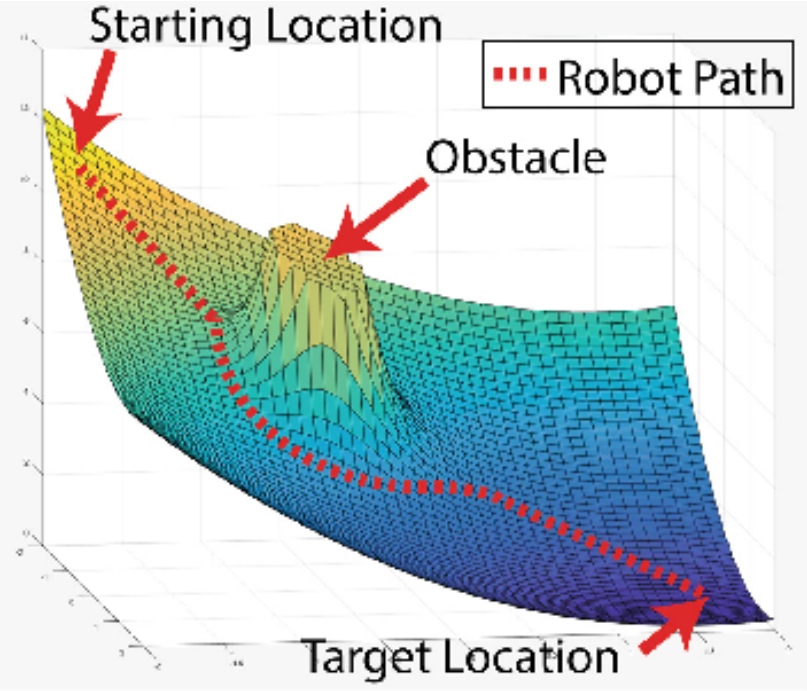

Figure 6: Potential Field for Navigation

4.1 Multi Agent Consensus

One example of a gradient based control in a multi agent system is in tackling the rendezvous problem in which multiple agents must converge on the same point in the environment.

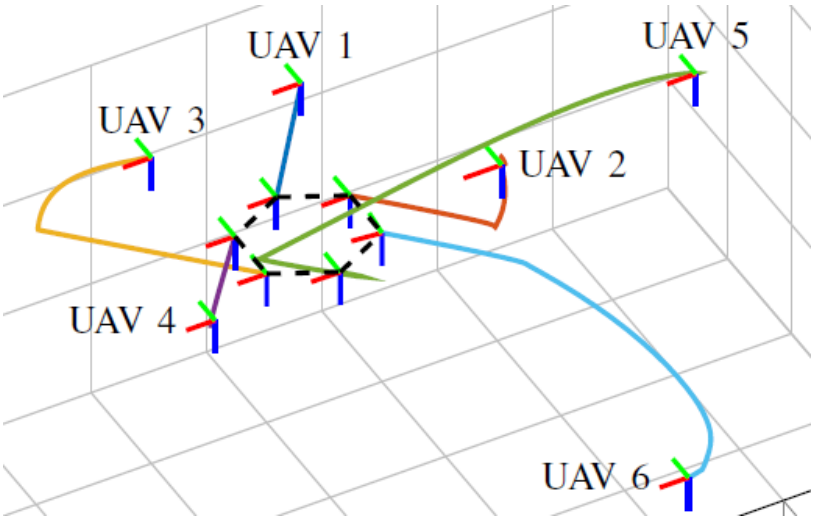

Figure 7: Potential Field for Navigation

To solve this problem, gradient descent is performed on the "disagreement function" below

$$\Phi_G(x) = \frac{1}{2}\sum_{i,j=1}^{n} a_{ij}(x_j - x_i)^2$$

This disagreement function describes the sum of squared distances between values of neighboring agents in our system.

A couple good references on the problem:

https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.119.2353rep=rep1type=pdf
https://dspace.mit.edu/bitstream/handle/1721.1/79093/Slotine$_U nifying$