Lecture 15: Coordination I

Autonomous Robots, 16-200

Carnegie Mellon University Qatar

Overview

- Logistics
- Path Planning with A*
- Movie
- Coordination

Logistics

- HW #3 is due tonight at midnight
- Graded research assignments will be available outside Bernardine's office by 6p.m. today
- Lab #4 is posted and is due November 17th
- We will be out of Qatar over Eid, but you can contact us via email
- Submit your project topics TODAY if you haven't done so already. We will send you feedback shortly. Start your projects today!!!

Plan

- Today:
  A* and Coordination I
- November 9th:
  Overall comments on research assignment and Coordination II
- November 10th Lab:
  Work on projects and Lab 4

A*

- Often used to search for the lowest cost path from the *q_start* to the *q_goal* location in a graph of cells/voronoi/visibility/quadtree
- Problem setting:
  - A graph of nodes, n_s is start, n_g is goal
  - A cost metric for moving from node to node
  - Want to find the path from n_s to n_g with the minimum cost

Remember Best-first Search?

- Define a *heuristic h(n)*
  - Estimates cost of shortest path from node *n* to the goal
  - E.g. Euclidean distance to goal
- Greedy search always chooses to follow child with lowest h(.) value
  - Can produce non-optimal paths!
- Also called depth-first

9	8	7	6	5	6
8	S			4	5
7	6			3	4
6	5			2	3
5				1	2
4	3	2		G	1
5	4	3		1	2
6	5	4	3	2	3

Remember Best-first Search?

- Define a *heuristic h(n)*
  - Estimates cost of shortest path from node *n* to the goal
  - E.g. Euclidean distance to goal
- Greedy search always chooses to follow child with lowest h(.) value
  - Can produce non-optimal paths!
- Also called depth-first

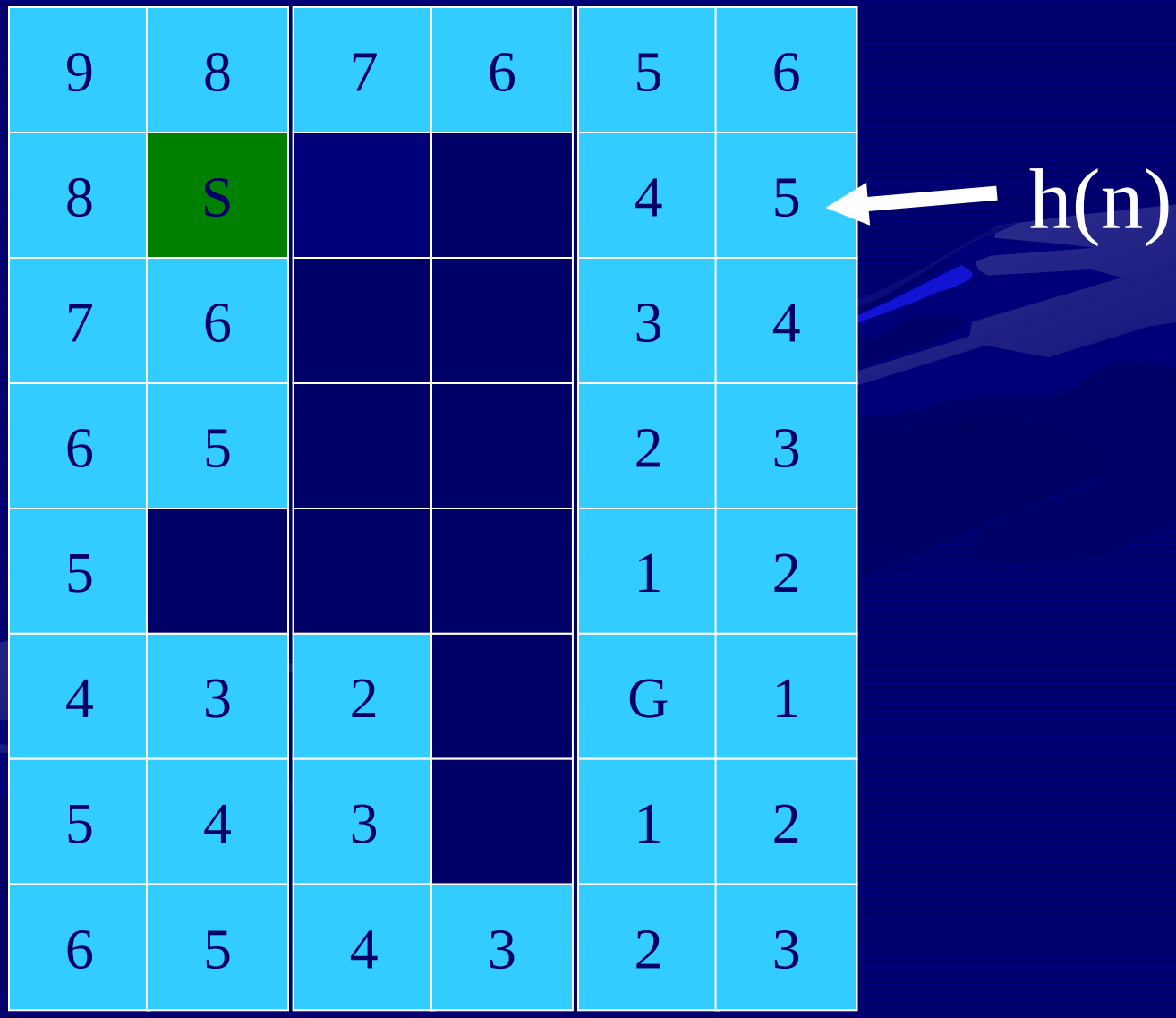
9	8	7	6	5	6
8	S			4	5
7	1			3	4
3	2			2	3
4				1	2
5	6	7		G	1
5	4	8		12	2
6	5	9	10	11	3

Path length=13!!!

A*

- Define *g(n)*
  - Cost to reach node *n* from start
- Greedy search by following child with lowest *f(n)*
  - $f(n) = g(n) + h(n)$
- Provided *h(n)* is *admissible*, A* will be *complete* and *optimal*
  - Performance will depend on how good *h(n)* is
- Admissibility:
  - *h(n)* must never overestimate the cost

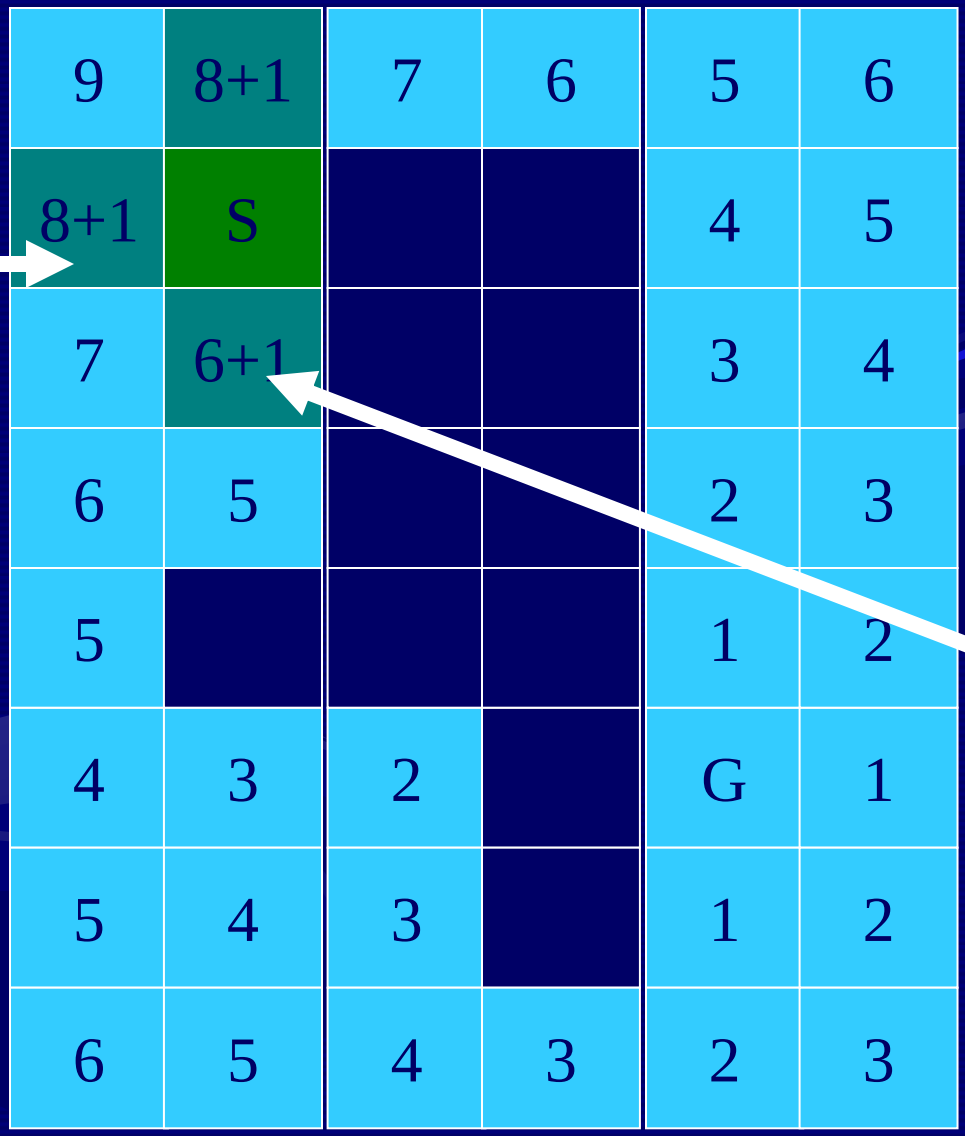
A* Example



A* Example

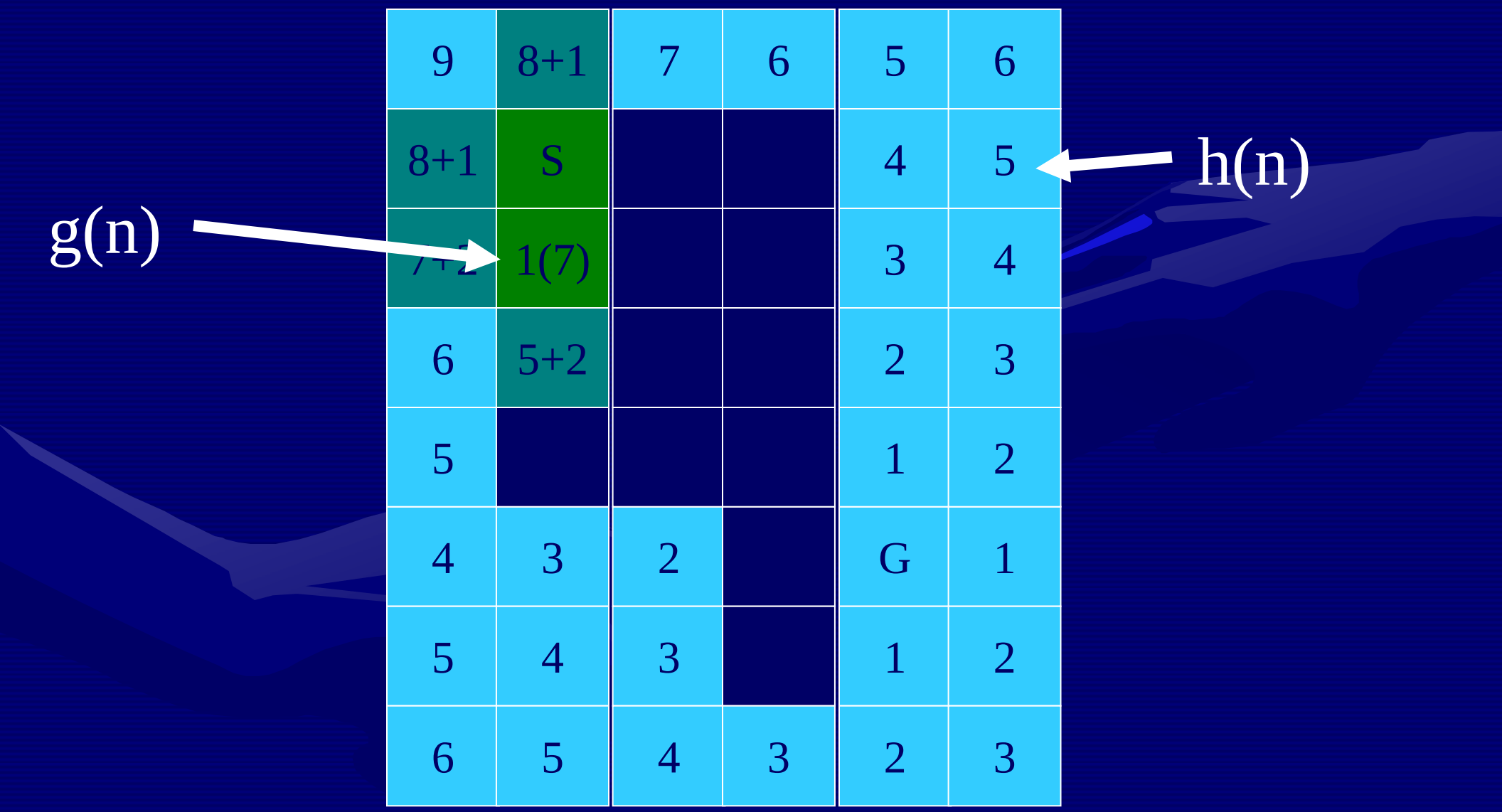
f(n)=

h(n)+g(n)



Queue of children to consider

A* Example



A* Example

9	8+1	7	6	5	6
8+1	S			4	5
7+2	1(7)			3	4
6+3	2(7)			2	3
5				1	2
4	3	2		G	1
5	4	3		1	2
6	5	4	3	2	3

h(n)

A* Example

9	8+1	7	6	5	6
8+1	S			4	5
7+2	1(7)			3	4
3(9)	2(7)			2	3
5+4				1	2
4	3	2		G	1
5	4	3		1	2
6	5	4	3	2	3

A* Example

9	8+1	7	6	5	6
8+1	S			4	5
7+2	1(7)			3	4
3(9)	2(7)			2	3
4(9)				1	2
5+4	3	2		G	1
5	4	3		1	2
6	5	4	3	2	3

A* Example

9	8+1	7	6	5	6
8+1	S			4	5
7+2	1(7)			3	4
3(9)	2(7)			2	3
4(9)				1	2
5(9)	3+6	2		G	1
5+6	4	3		1	2
6	5	4	3	2	3

A* Example

9	8+1	7	6	5	6
8+1	S			4	5
7+2	1(7)			3	4
3(9)	2(7)			2	3
4(9)				1	2
5(9)	6(9)	2+7		G	1
5+6	4+7	3		1	2
6	5	4	3	2	3

A* Example

9	8+1	7	6	5	6
8+1	S			4	5
7+2	1(7)			3	4
3(9)	2(7)			2	3
4(9)				1	2
5(9)	6(9)	7(9)		G	1
5+6	4+7	3+8		1	2
6	5	4	3	2	3

An interesting thing just happened!!!

A* Example

9+1	1(9)	7+2	6	5	6
8+1	S			4	5
7+2	1(7)			3	4
3(9)	2(7)			2	3
4(9)				1	2
5(9)	6(9)	7(9)		G	1
5+6	4+7	3+8		1	2
6	5	4	3	2	3

A* Example

9+1	1(9)	2(9)	6+3	5	6
8+1	S			4	5
7+2	1(7)			3	4
3(9)	2(7)			2	3
4(9)				1	2
5(9)	6(9)	7(9)		G	1
5+6	4+7	3+8		1	2
6	5	4	3	2	3

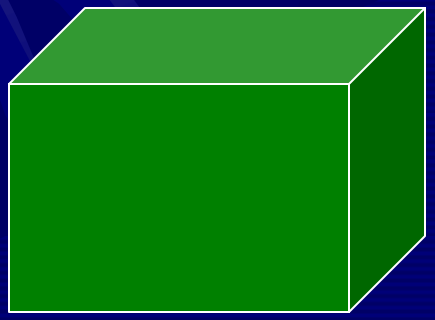
A* Example

9+1	1(9)	2(9)	3(9)	4(9)	6+5
8+1	S			5(9)	5+6
7+2	1(7)			6(9)	4+7
3(9)	2(7)			7(9)	3+8
4(9)				8(9)	2+9
5(9)	6(9)	7(9)		G	1
5+6	4+7	3+8		1	2
6	5	4	3	2	3

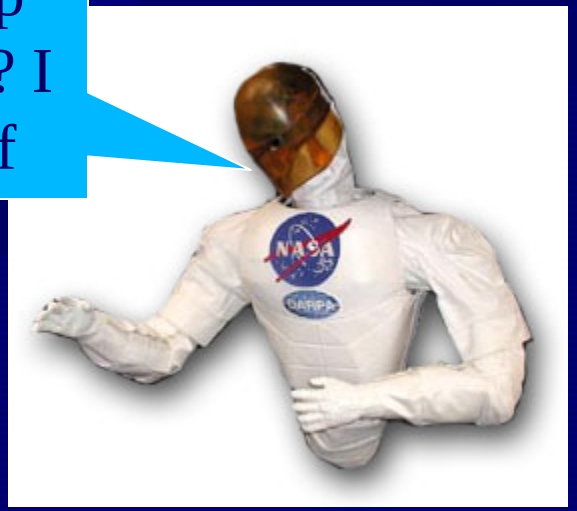
Path Length = 10!!

Coordination

- The problem:
  - How do we make 2 or more robots work together to execute a task?
- Want to execute the task better than if robots do not coordinate



Hey, can you help me move this box? I can't do it myself



NASA's RoboNaut

Movie

USC – Interaction Lab – Maja Mataric and Brian Gerkey

Highlights from experiments performed with the pusher-watcher algorithm, implemented using the MURDOCH task allocation system

Motivation

Multirobot applications

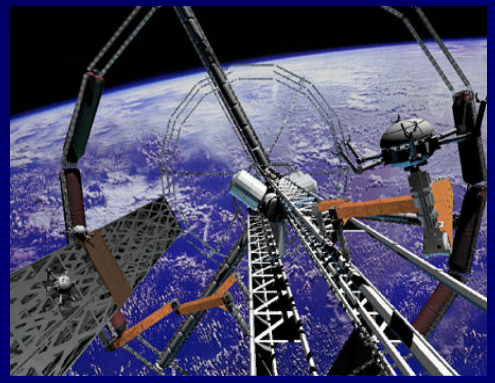
Remote Operations

Production Plants

Education and Entertainment

Warehouse Management

Intelligent Environments



Automated Construction

Motivation

Multirobot applications

Urban Reconnaissance

Agriculture

Urban Search
And Rescue

Hazardous Exploration

Hazardous
Clean-Up

Motivation

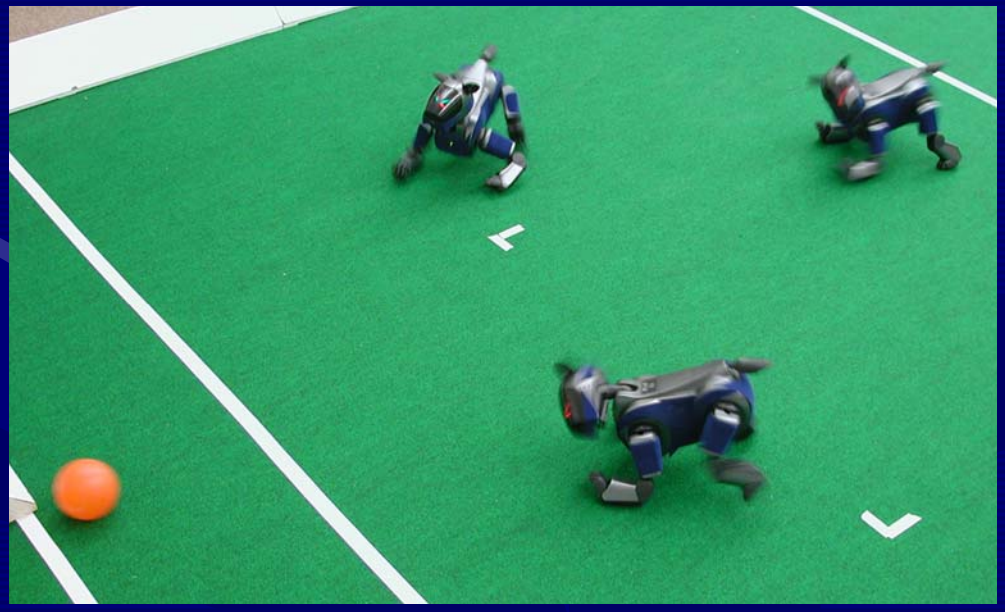
Why Multiple Robots?

?

- Task requirement
- Improved efficiency
- Simplified robot design (cooperating specialists)
- Improved localization
- Improved robustness (redundancy)
- Wider variety of possible solutions

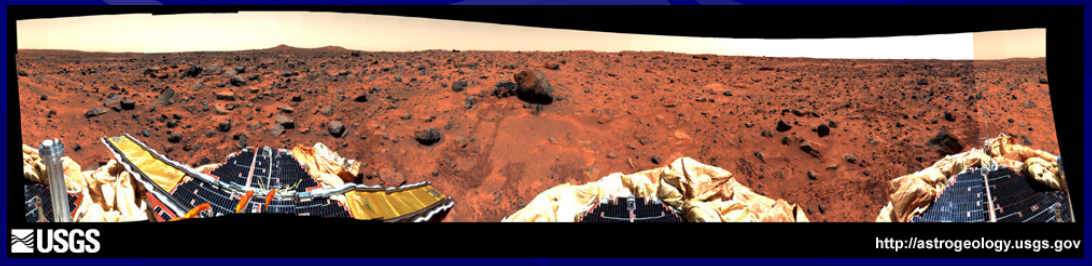
Motivation

Coordinating multiple robots to cooperate in dynamic environments is a difficult problem to solve!



Dynamic Environments

- Many unknowns
- Changing obstacles
- Changing task requirements
- Changing resources
- Limited resources



Required Characteristics

1. **Robustness** to Failures
2. Response **Speed**
3. Solution **Efficiency**
4. Handling imperfect **Information**
5. Handling Imperfect **Communication**
6. Efficient **Resource** Utilization
7. Efficient Task **Allocation**
8. Efficient **Role** Adoption
9. Handling **New Input**
10. **Fluid** Addition/Loss of Robots
11. Handling **Heterogeneous** Teams
12. **Extensibility**
13. **Flexibility**/Easily Applied to New Applications
14. Handling **Tight-Coordination**
15. **Scalability**
16. **Learning** and On-line Adaptation
17. Proven **Implementation** on Robots

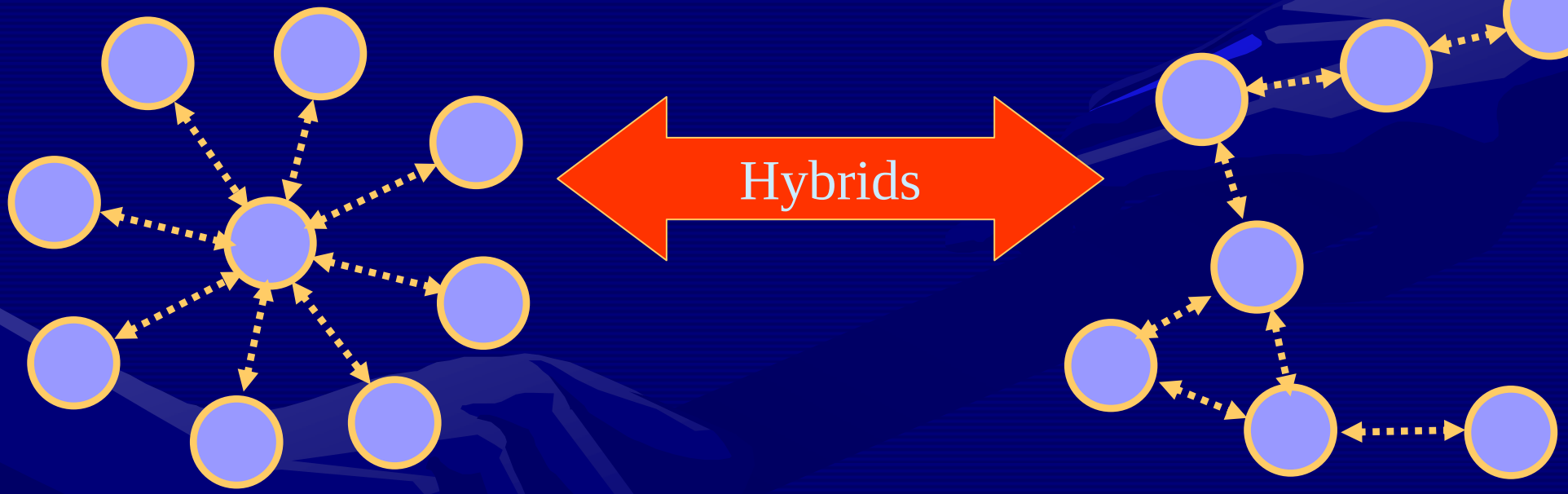
Coordination

- Robot A's choice of action depends on:
  - Robot B's actions
  - Robot B's sensing
  - Robot B's communication
- Examples:
  - Multi-robot exploration/mapping
  - Multi-robot surveillance/sensor networks
  - Construction
  - Passing in ball sports
  - Furniture moving, formation control

Coordination Approaches

Fully Centralized

Fully Distributed



Common Components of Coordination

- Robustness to failure
- Task allocation
- Role assignment
- Action synchronization (Temporal coordination)

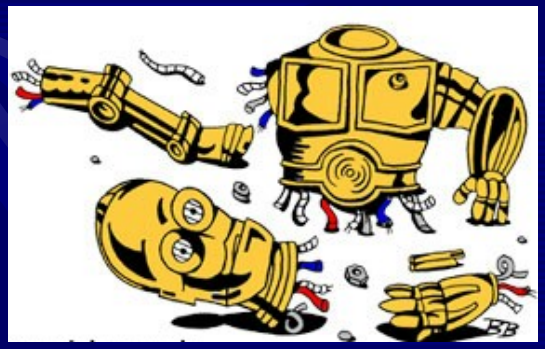

In the real world
things always
break!

Categories of Failure

1. Communication Failure

2. Partial Robot Malfunctions

3. Robot Death



Task Allocation

- Who is going to take responsibility for each task?
- Can one robot execute each task?
- Can the robots exchange tasks?
- Who is finally accountable to the operator?
- Some tasks can be decomposed into single-robot tasks:
  - Mapping
  - Exploration

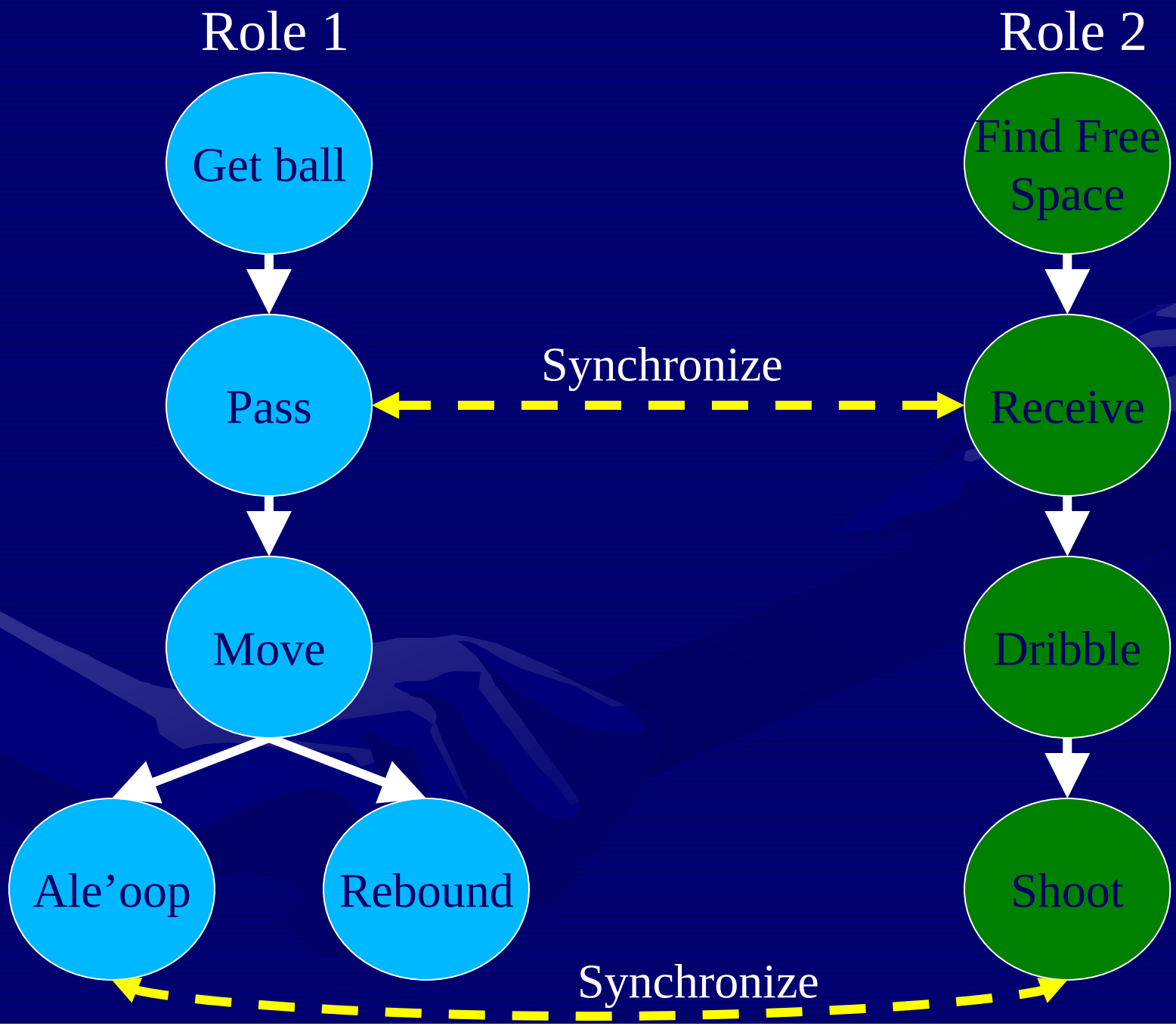
Role Assignment

- What happens when tasks are tightly coordinated?
- Then we need role assignment
  - Which robot will play each role?
- Static role assignment:
  - Locker Room Agreements [Stone, 98]
- Dynamic role assignment:
  - Work out *best* assignment given state of world
  - Can be a hard problem!
- Example domain: soccer

Action Synchronization

- What if the roles require robots to synchronize their actions?
- Tightly coupled
  - Many actions synchronized, short time ranges
  - E.g. Marching bands, passing
- Loosely coupled
  - Fewer actions, long time ranges
  - E.g. complete painting after house is built
- Can be modeled as coordinated state machines
  - We call these *Plays* [Bowling, et al. 04]

Plays: Coordinated State Machines



Putting It All Together

- Some tasks require coordination at all three levels!
- Example: Treasure Hunt!
  - Assign tasks: search for treasure in area “x”
  - Assign roles: Robot A should search while robot B makes sure they don’t get lost
  - Synchronize actions: Robots A and B need to work together!
- Next time we will talk about how to do this in more detail
- Also, what if your team consists of both humans and robots?

End of slides!

Happy Holidays!