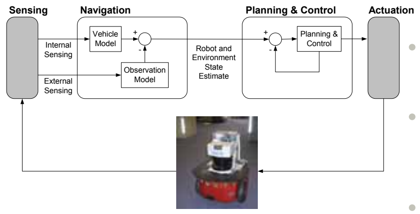
Mtrx 4700 : Experimental Robotics

Robotic Architectures and Multi Robot Systems

Dr. Robert Fitch
Dr. Stefan B. Williams

Mobile Robot Architectures



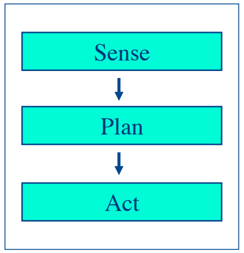
- Fundamental building blocks include sensors, actuators navigation, planning and control modules
- Specific instances are application and platform dependent
- Sensor information is used by navigation algorithms to generate an estimate of the system state
- System state estimate is used to plan the robot's actions and to generate commands for the system's actuators.

Definitions [Arkin]

- For computer architecture:
  "Computer architecture is the discipline devoted to the design of highly specific and individual computers from a collection of common building blocks." [Stone, 1980]
- By analogy, robotic architecture:
  " Robotic architecture is the discipline devoted to the design of highly specific and individual robots from a collection of common software building blocks." [Arkin, 1998]

Sense-Plan-Act

- First notable robot architecture
- Shakey – SRI, 1960s



Sense-Plan-Act

- Drawbacks? (discussion)

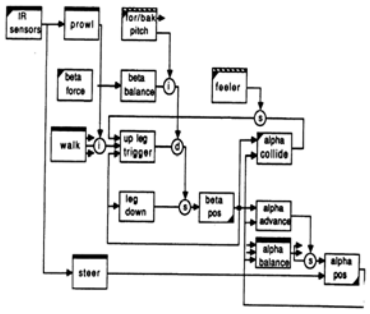
Reaction: Behaviour-Based Control

- “Planning is just a way of avoiding figuring out what to do next.” [Brooks, 1987]

Behaviour Based Control

- In the 1980s and early 1990s Behaviour Based control found favour in many robotic circles
- The central idea of Behaviour Based Robotics is to compose complex robotic systems from a series of simple primitive behaviours
- Behaviours such as ‘follow wall’, ‘avoid obstacles’, etc. are combined to create robotic systems
- ‘Emergent behaviours’ were described as the result of the interaction of these lower level primitives

Subsumption Architecture



From: R.A. Brooks, “A Robot that Walks: Emergent Behaviours From a Carefully Evolved Network”, MIT AI Memo 1091, 1989

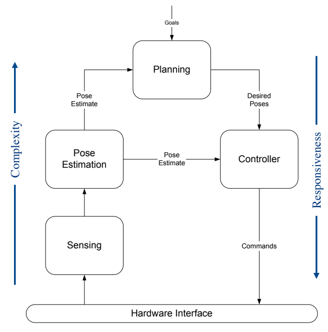
- One of the most popular Behaviour Based paradigms at the time was the Subsumption Architecture proposed by R. Brooks at MIT
- Behaviours were made up of a collection of Finite State Machines which exerted supportive and inhibitory effects on other behaviours
- Reasonably complex behaviours, such as walking, were demonstrated

Subsumption Architecture

- Drawbacks? (discussion)

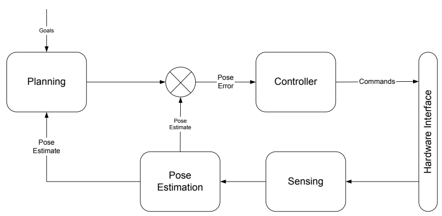
Modern Architectures: Deliberative AND Reactive Components

- The components of a robotic system can often be broken down into a hierarchy
- Sensing and interfacing to hardware is done at a low level and demands a high degree of responsiveness
- Estimation and control rely on interfaces to the mechanism
- Planning of paths and reasoning can be done at lower rates but is often more complex



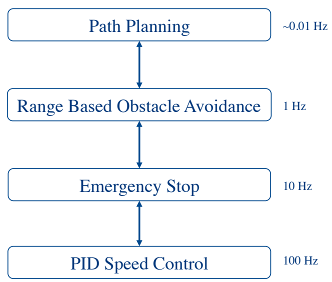
Mobile Robot Architectures

- You may recognize the diagram recast in a traditional control layout
- There are effectively two control loops here
  - The inner loop achieves particular poses (note: there is often a rate controller in addition to the pose controller shown here)
  - The outer loop is concerned with trajectory control



Temporal Decomposition

- Each level in the hierarchy is usually characterised by different temporal requirements
- Low level motor control requires high responsiveness
- As the complexity of tasks increase, they are usually required less often
- Global Path Planning can occur on the order of minutes depending on the application



Classic Three-Tier Architecture

- Task-planning
- Executive
- Behavioural (real-time) control

Example: Office Delivery Robot [Springer Handbook of Robotics]

- Behavioural Control
  - Move down hallway
  - Find door
  - Announce delivery
  - …
- Executive
  - Decompose delivery task into subtasks
  - Triggers replanning, e.g. if encounters locked door
- Planning
  - Scheduling
  - Route planning

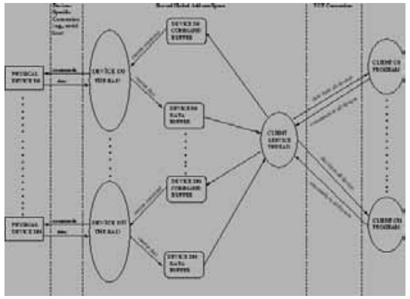
Composing Robotic Systems

- As you can probably appreciate, robotic systems composed of many interacting parts – including hardware and software – are often very complex and difficult to assemble, debug and maintain
- There has been a push of late to standardise on architectures or frameworks for describing the manner in which these components interact
- The related field of Component Based Software Engineering describes individual modules, or components, and defines the interactions between them
- A number of related software projects have appeared in the robotics community

Player/Stage

- Player/Stage is arguably the most popular robotic toolkit
- It is available for free download and has support for many sensors and mobile robotic platforms
- There are a large (and increasing) number of algorithms supported by player. These include modules for
  - Interfacing to platforms (stage, pioneer, Segway RMP, Nomad, …)
  - Interfacing to sensors (sonar, laser, cameras, gps, …)
  - Obstacle avoidance
  - Localisation
  - Mapping
  - Planning

Player/Stage



- Player/Stage is essentially a client/server architecture
- Sensors present server interfaces which are connected to via TCP/IP by clients which receive and process information
- Inter process communication is somewhat difficult in this model

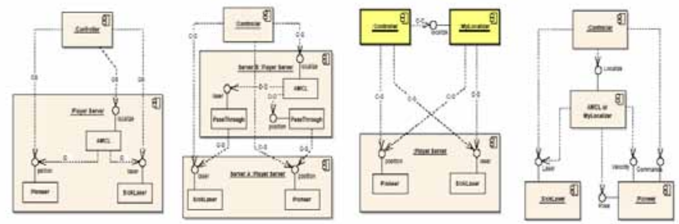
ROS (Robot Operating System)

- Willow Garage (CA, USA)
- Originated at Stanford AI Lab
- Provides standard operating system services
- Multi-process (graph architecture)
- Implements many robot algorithms
- BSD license (open source)
- Rapidly increasing popularity

ORCA Framework

- The ORCA Robotics Project, being developed jointly by us here at CAS and researchers from KTH in Sweden, proposes a framework for the interaction of robotic components together with a repository of reusable components
- This Framework provides patterns of communication which can be used to connect components
- There is no explicit division between clients and servers – any component can present servers and act as a client for other components

ORCA Framework



(a) The setup using standard Player devices and a controller in client space.

(b) Moving AMCL to a different host requires extra PassThrough devices.

(c) Using one's own localizer in client space requires the use of client-client communication.

(d) The Orca method: only one type of communication is required.

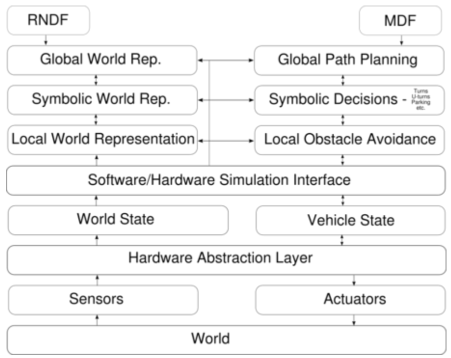
Fig. 3. Various configurations for a system with hardware servers, a localizer (either Player's Adaptive Monte Carlo Localizer (AMCL) or a hand-written localizer) and a central controller, using either (a)-(c) Player or (d) Orca. The map (required for AMCL) is omitted for clarity. For Player, communication links are labelled Client-Server (C-S), Device-Device (D-D) or Client-Client (C-C). The use of non-standard Client-Client communication in (c) affects the re-useability of both the 'MyLocalizer' and 'Controller' implementations.

Autonomous Systems

Case Study: DGC

Autonomous Systems

Rav4 Software Architecture



Autonomous Systems

Rav4 Planning Architecture

Town level Planner

Road segments

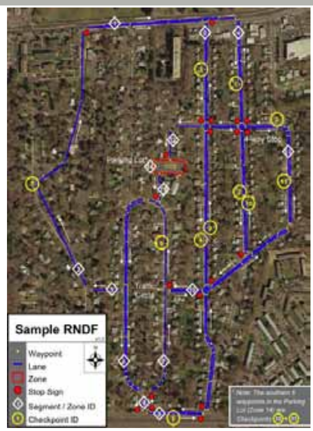
Street level Planner

Waypoints

Low level path planner

Speed / Steering angle

Low level controllers



Autonomous Systems

Multi-Vehicle Architectures

- As if single vehicles weren't complex enough, there is real interest in the deployment of multi-vehicle systems

Image courtesy of SRI International

Autonomous Systems

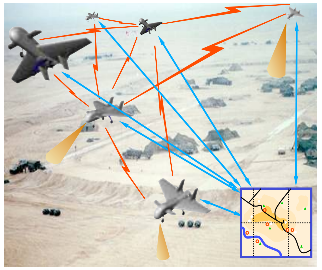
Why Multi-Robot Systems?

Why worry about *multiple* robots when ***one*** is hard enough?

- **Synergy**: combining unreliable robots can yield a reliable system
  - Hugh's simple two-sensor example
- **Redundancy**: two is better than one
  - Robust against failures
- **Scale**: you can't be large and small at the same time
  - Wheeled robot/aibo search & rescue example

Autonomous Systems

Example: Distributed Data Fusion



- We spoke about methods for data fusion in a single vehicle context
- When multiple vehicles are involved, techniques for consistently fusing information from different platforms is required
- Distributed Data Fusion techniques investigate the mathematical foundation required for sharing information

Autonomous Systems

Examples of Multi-Robot Systems

Autonomous Systems

Swarm Robotics

- There is also work looking at developing swarm robotics
- Based loosely on the study of insect colonies such as ants and bees
- The philosophy here is similar to Behaviour Based systems - take a large number of simple robots, give them some primitive behaviours and allow them to adapt their behaviour until some useful behaviour emerges

Swarm Robotics

Key Issues [Bekey]

- Homogeneous vs. heterogeneous
- Centralized vs. distributed
- Loosely-coupled vs. tightly coupled tasks
- Ensemble vs. individual performance
- Communication
- Task Assignment
- Specialization

ACFR
AUSTRALIAN CENTRE FOR FIELD ROBOTICS

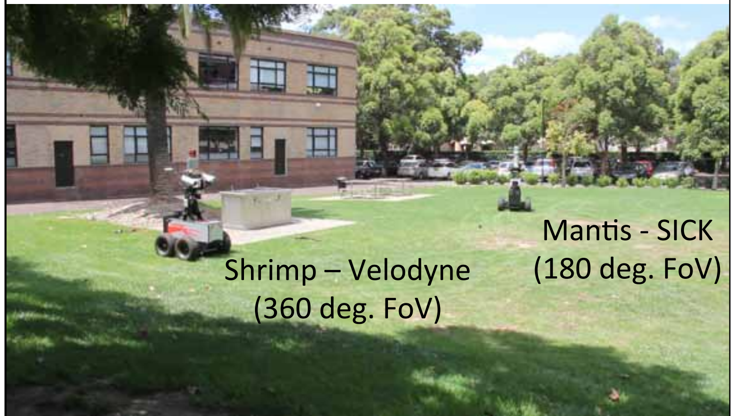
Case Studies: Coordinated Information Gathering with Decentralised Multi-Robot Systems

Authors:
Robert Fitch
Zhe Xu
Jason Gan
Salah Sukkarieh

Decentralised Information Gathering

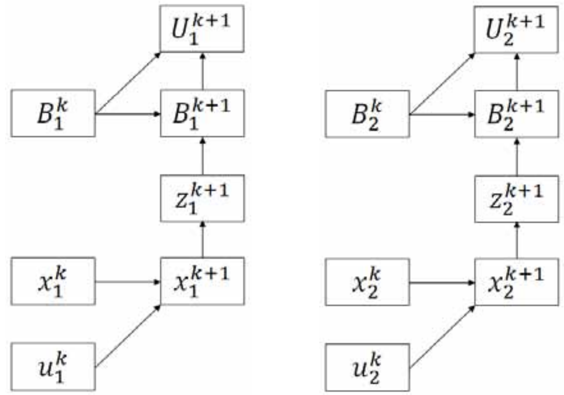
We'd like robots to move in such a way as to gain the most information over time

- **Information** refers to Shannon Information (entropy)
- **Mutual Information** measures information gained by making an observation
- Predicting the value of future observations allows us to do ***planning***
- The ability to plan cooperatively rests on ***communication***

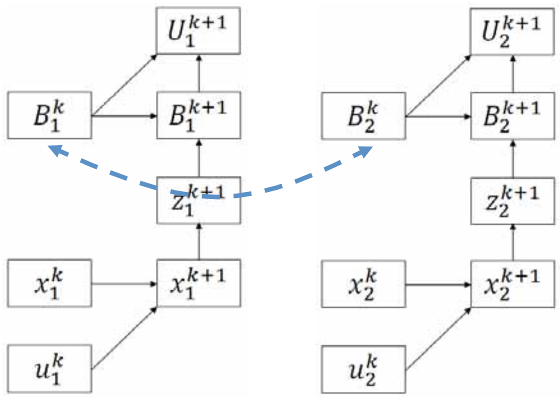


[Xu 2011, 2012]

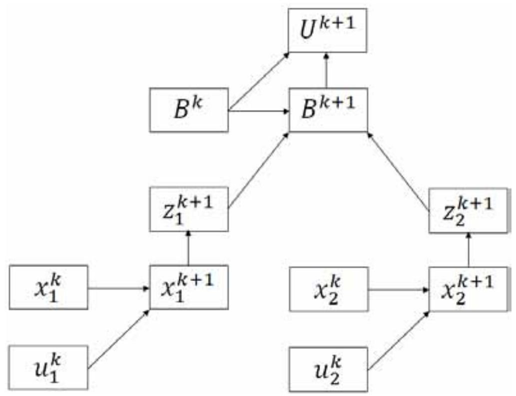
Implicit Coordination



Implicit Coordination



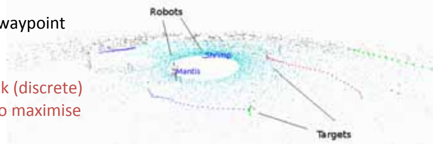
Explicit Coordination

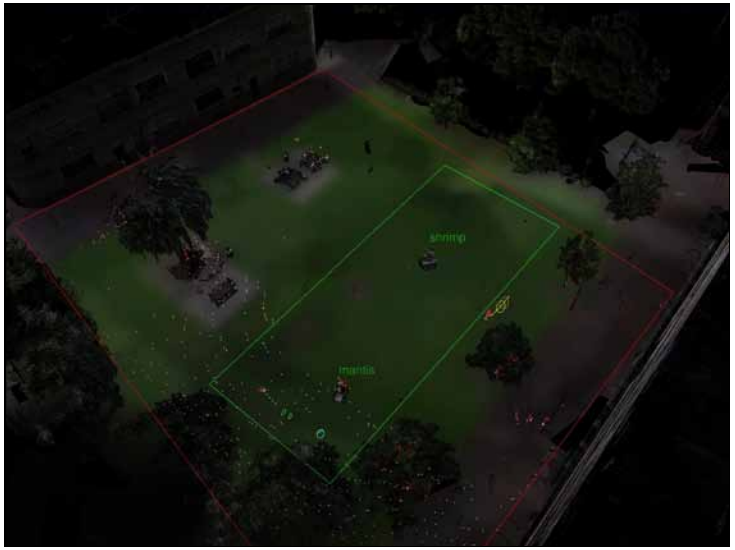


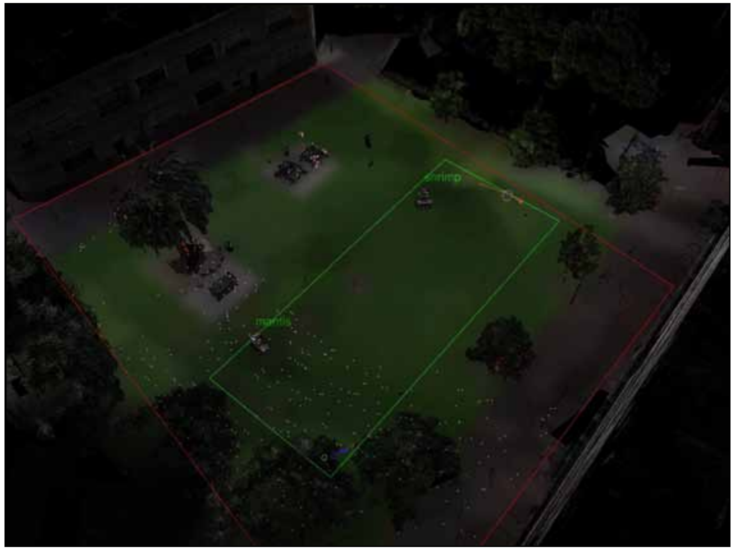
Multi-UGV Multi-Target Tracking

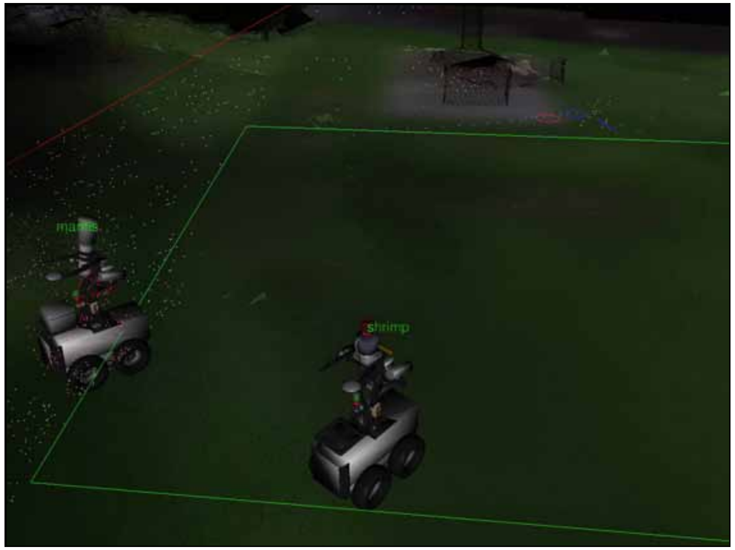
Each robot chooses a single waypoint
Re-plans continuously

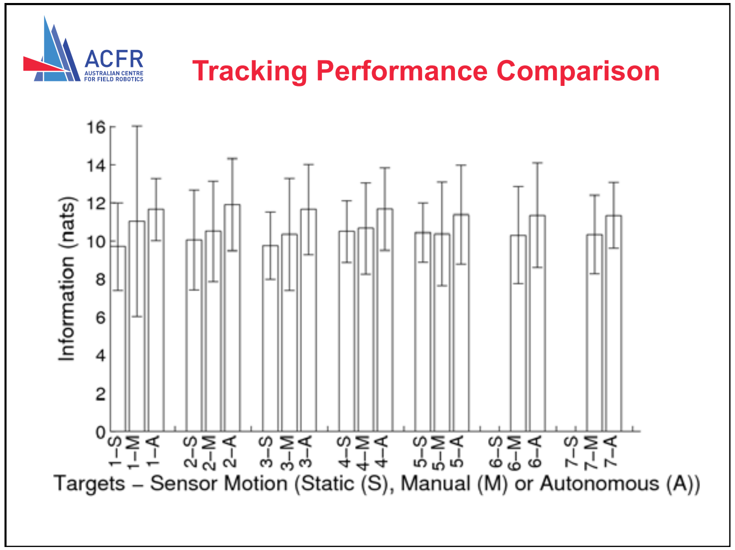
Chooses which target to track (discrete) and waypoint (continuous) to maximise information gain



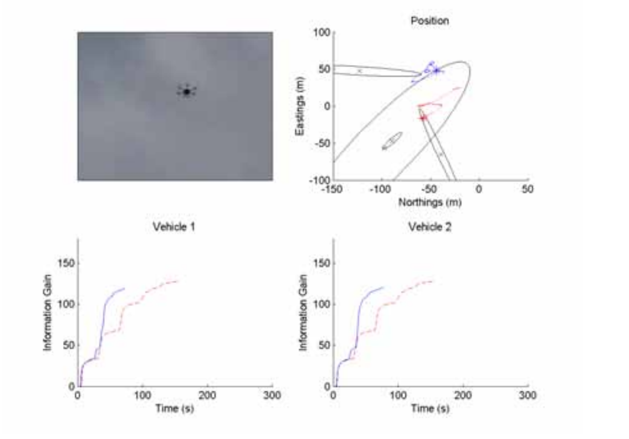




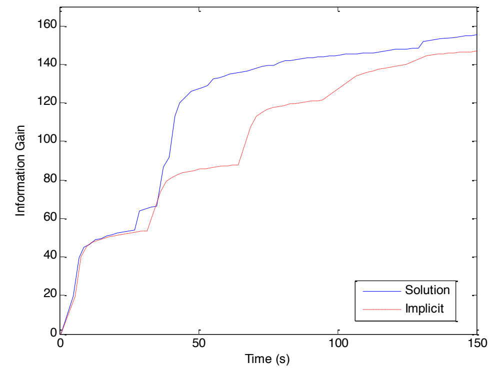

mantis
shrimp


ACFR
AUSTRALIAN CENTRE FOR FIELD ROBOTICS
Tracking Performance Comparison
Information (nats)
0
2
4
6
8
10
12
14
16
1–S
1–M
1–A
2–S
2–M
2–A
3–S
3–M
3–A
4–S
4–M
4–A
5–S
5–M
5–A
6–S
6–M
6–A
7–S
7–M
7–A
Targets – Sensor Motion (Static (S), Manual (M) or Autonomous (A))

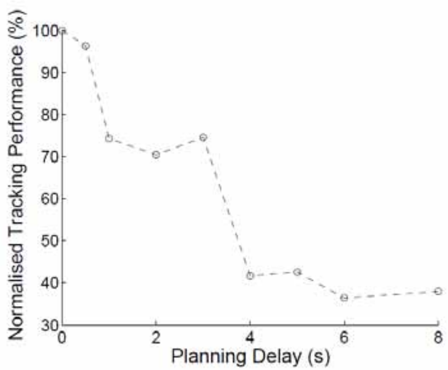
UAV Results



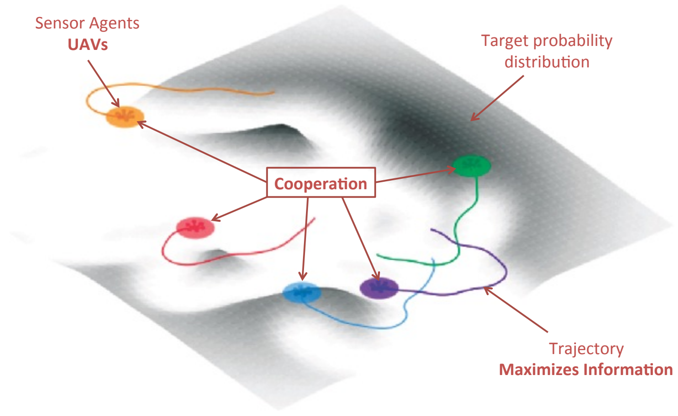
UAV Results: Implicit vs Explicit



Planning Time

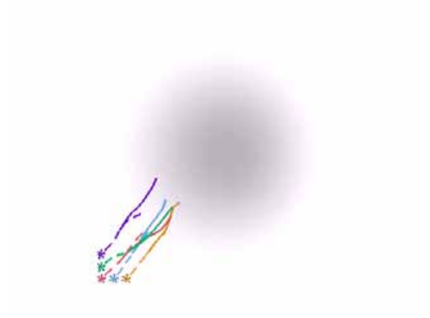


Multi-Agent Probabilistic Search

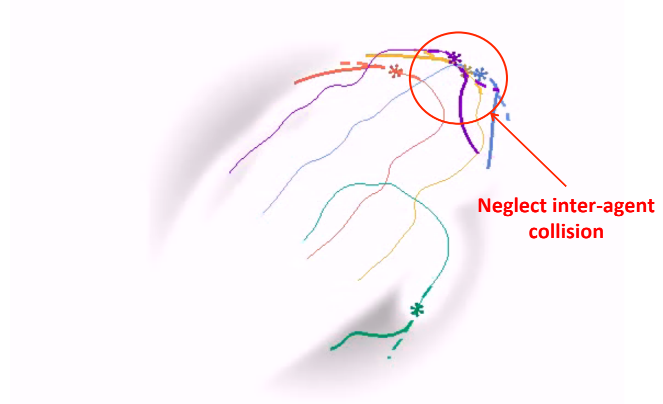


[Gan 2011, 2012]

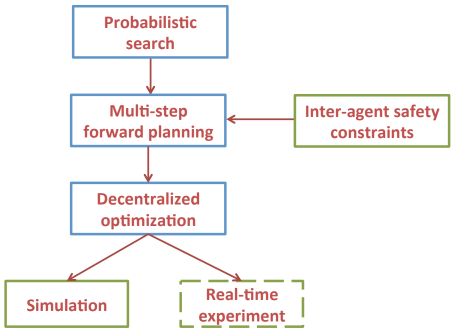
Multi-Agent Probabilistic Search



Multi-Agent Probabilistic Search



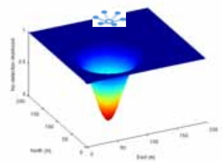
Incorporate Inter-Agent Constraint



Multi-Agent Probabilistic Search

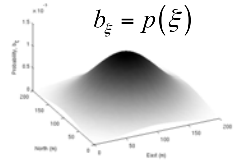
No-detection likelihood

$$p(z^k=\bar{D}\mid\xi)=1-P_{d_{max}}e^{-\sigma\left(\frac{d}{d_{max}}\right)^2}$$



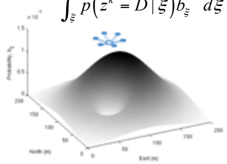
Prior target belief

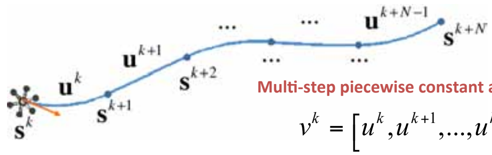
$$b_\xi = p(\xi)$$



Posterior target belief

$$b_{\xi,new}=\frac{p(z^k=\bar{D}\mid\xi)b_\xi}{\int_\xi p(z^k=\bar{D}\mid\xi)b_\xi\, d\xi}$$



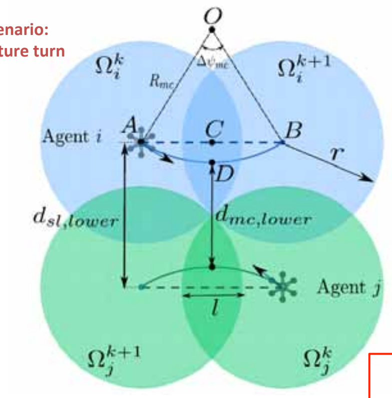


Multi-step piecewise constant action vector

$$v^k=\left[u^k,u^{k+1},\ldots,u^{k+N-1}\right]^T$$

Inter-Agent Safety Constraints

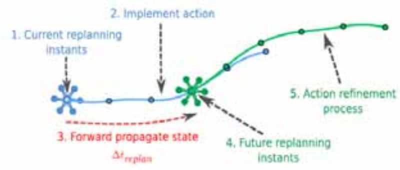
Worst case scenario:
Maximum curvature turn



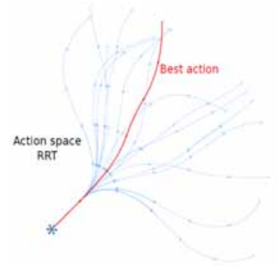
$$\boxed{d_{mc,lower} = 2R}$$

Real-time Decision Making

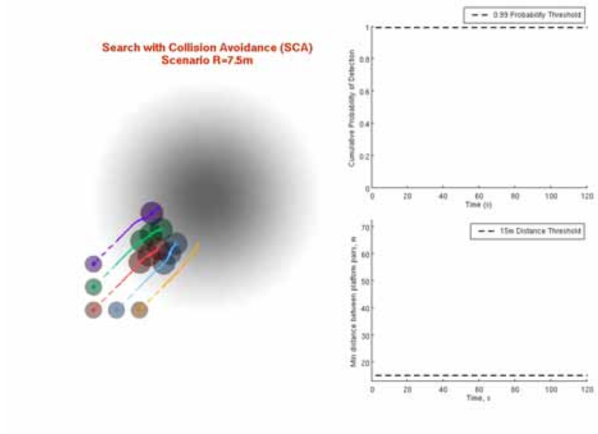
Real-time replanning and action refinement



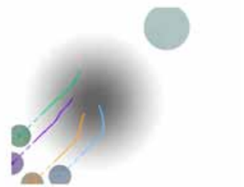
Sampling-based action initialization



Simulation - Video



Real-Time Experiment – Mobile Robots



- Validate algorithm in real-time
- Ground robot
- Decentralized
- Self computation and communication modules
- ZigBee mesh network
- EKF localization – acoustic sensors
- No reactive systems

Conclusions

- Robotic systems involve an integration of a number of different areas including hardware, electronic and software
- The complexity of these systems has motivated the development of a number of approaches to composing systems
- Standardization in the robotics community is still coming

Further Reading

- R. C. Arkin. *Behavior-Based Robotics*. MIT Press, 1998.
- R. Siegwart & I. R. Nourbakhsh. *Introduction to Autonomous Mobile Robots*. MIT Press, 2004.
- T. Balch & L. Parker, editors. *Robot Teams: From Diversity to Polymorphism*. A K Peters, 2002.