Integrating Soar with Physical Robots and Behavior-Based Control

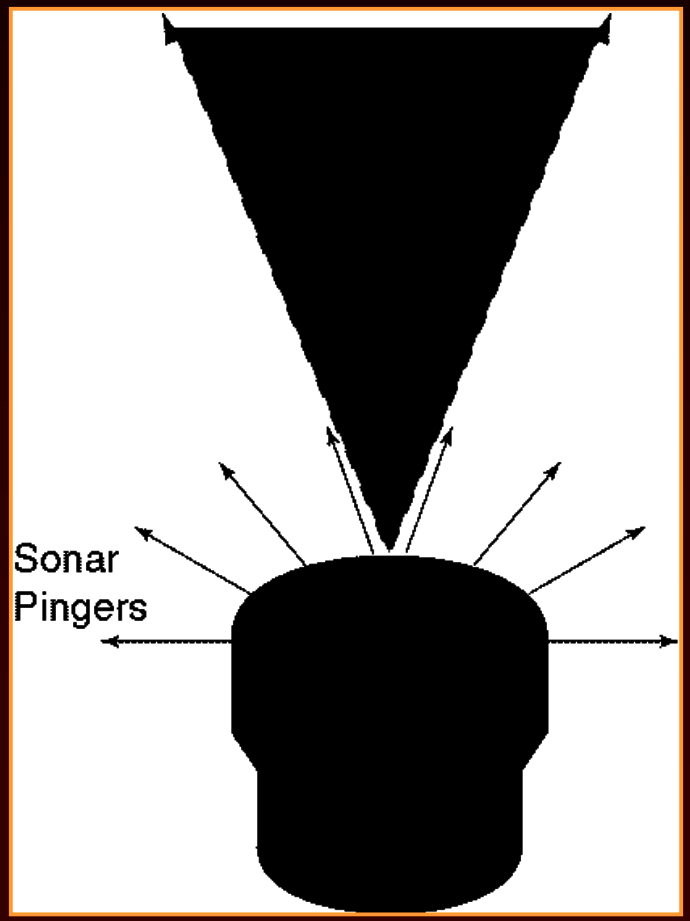
Barry Brian Werger

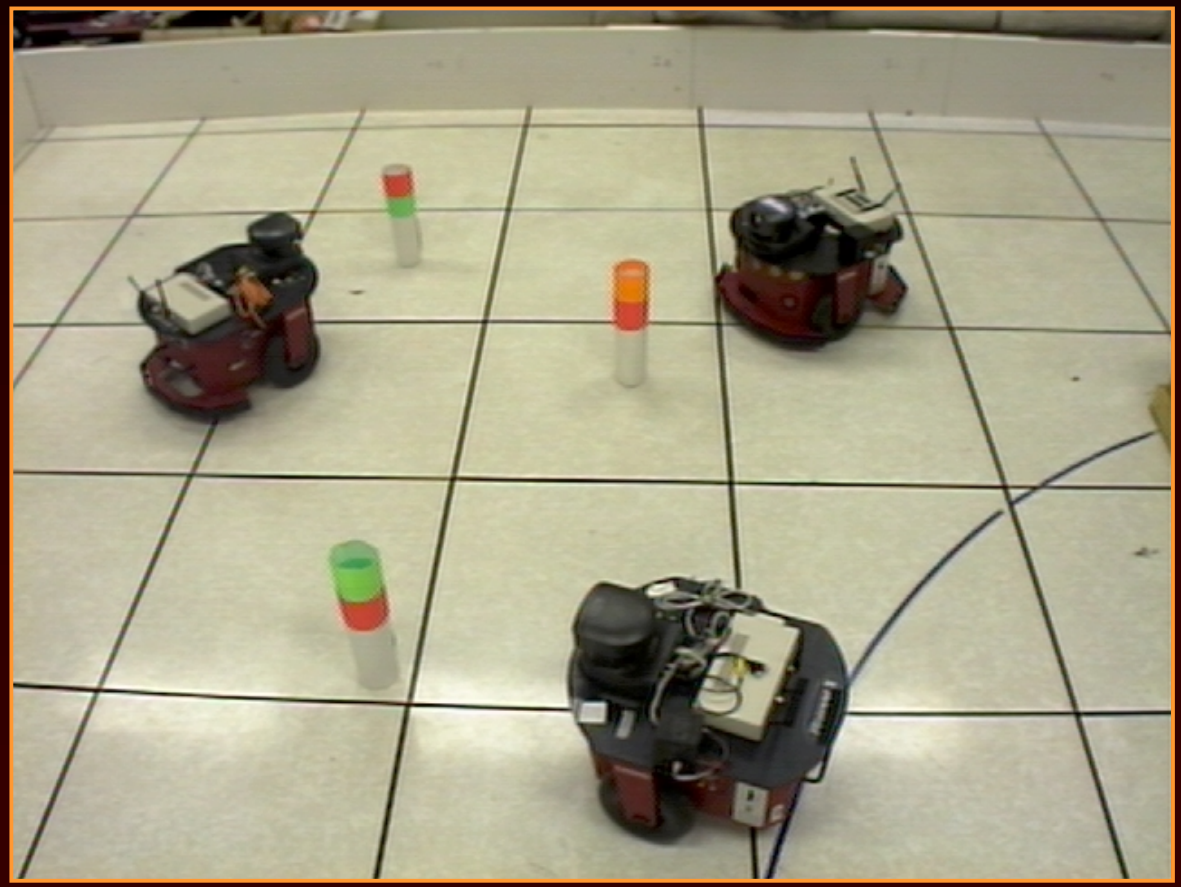
Young Jun Kim

USC Robotics Research Labs and ISI

USC Robotics Research Labs

Robots



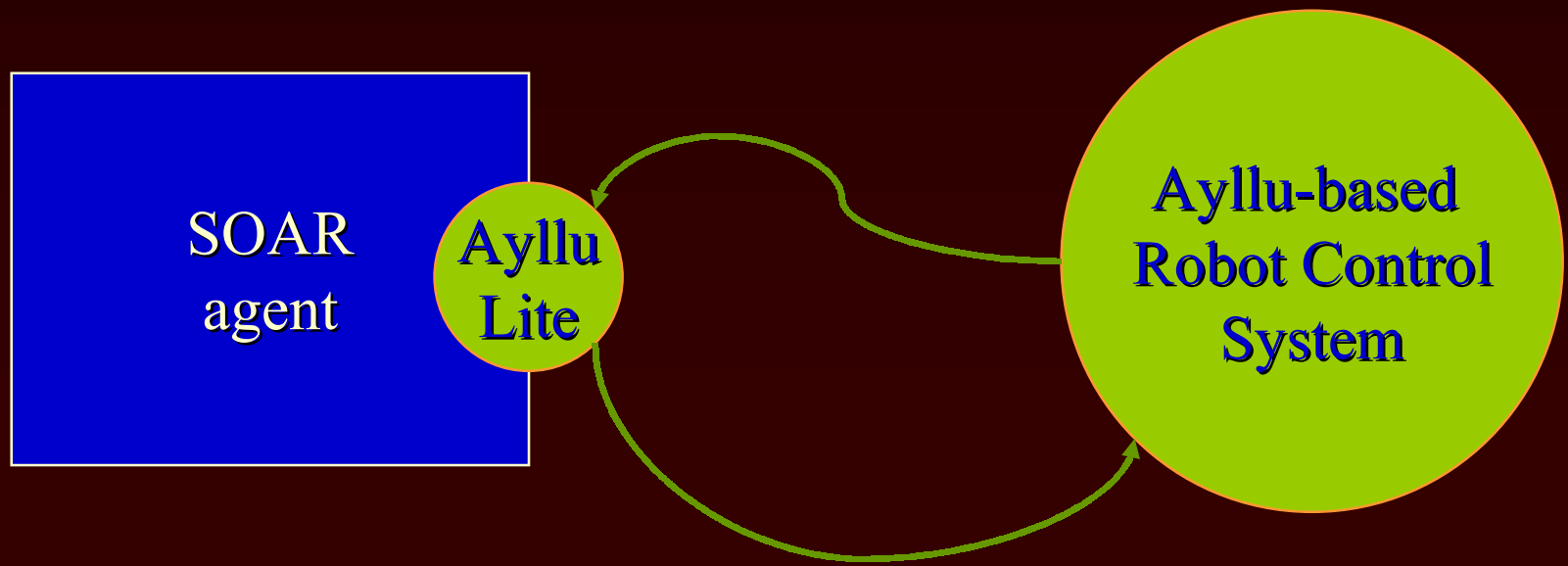


Port-Arbitrated Behavior-Based Control

- Control for environmental interaction
- Behavior: black box w/public interface of *ports*
  - All inter-behavior communication through ports
  - Port data propagates along *connections*
  - Connections can inhibit/suppress ports
- Data driven approach
- Data has limited lifetime

Integration

- SOAR
- Ayllu
  - Distributed behavior-based control for mobile robots (esp. teams)



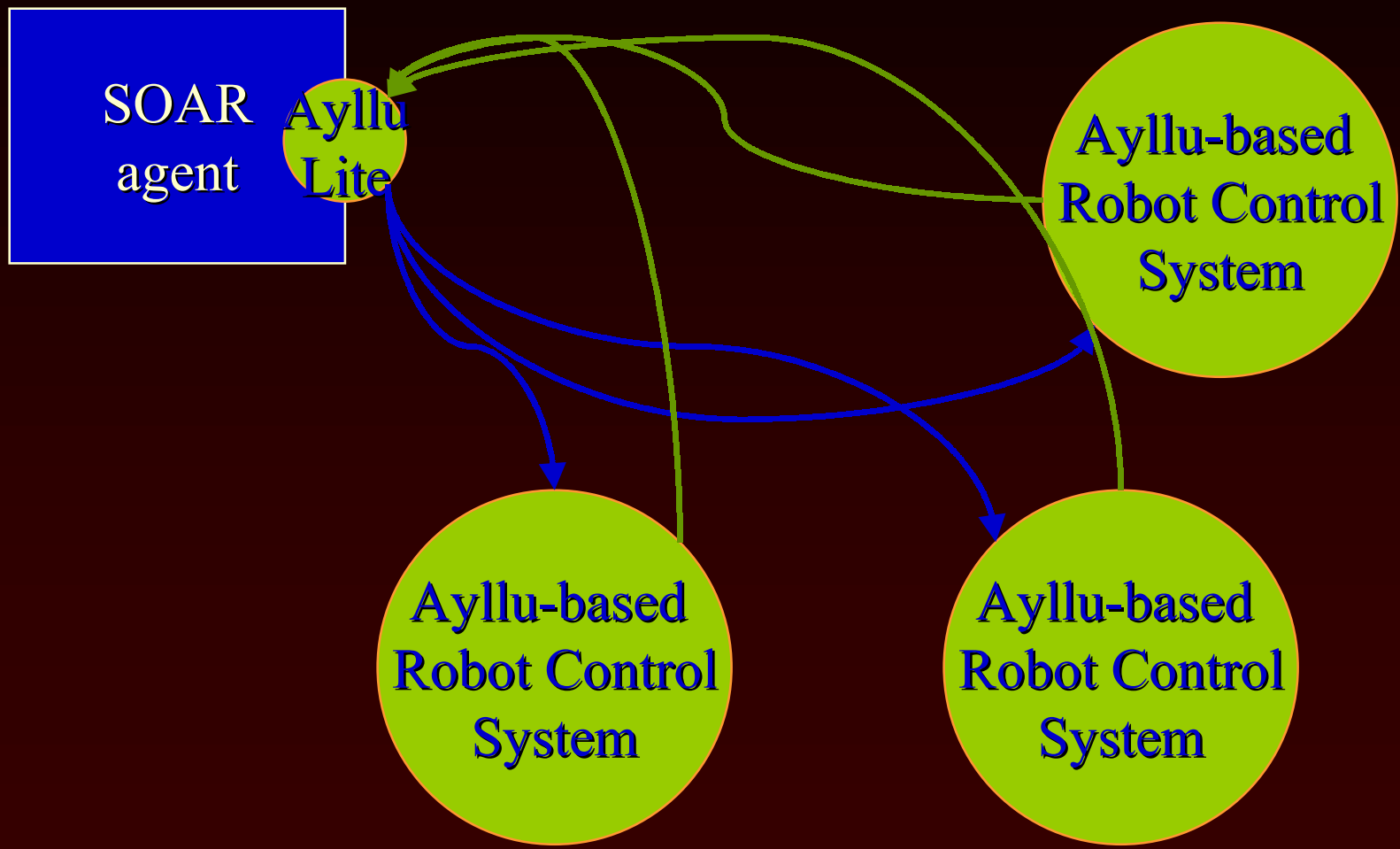
SOAR AGENT

SOAR Agent can do better than anything the real robots do!!!

- Modified Soar (7.0.4) Source Code

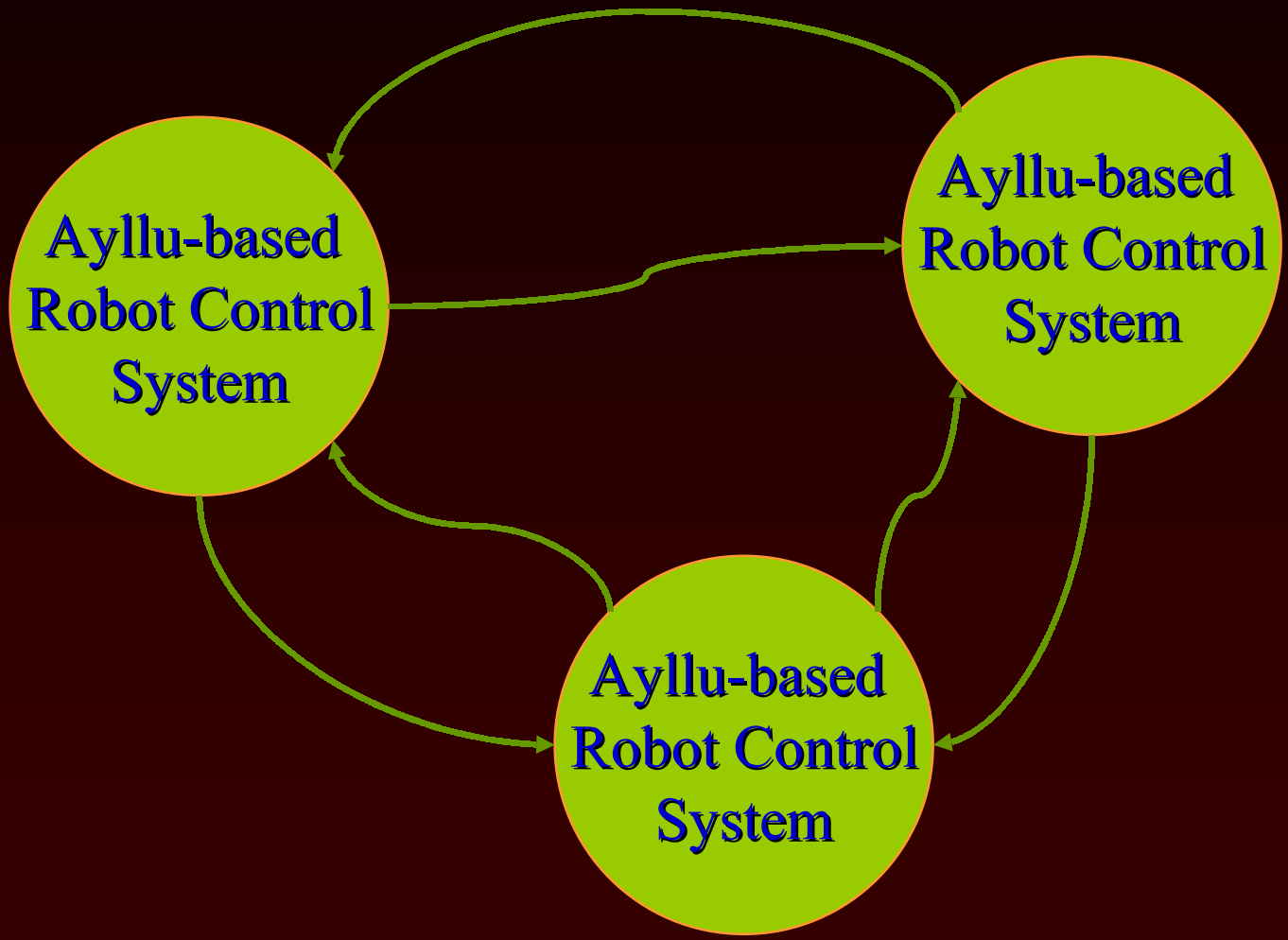
```
Add Input-link
(<s> ^input-link.robot <il>)
(<il> ^robot1 <r1> ^robot2 <r2> ^robot3 <r3>)
(<r1> ^target1 <v1> ^target2 <v2> ^target3 <v3> ^target4 <v4>)

Add Output-link
(<s> ^output-link <ol>)
(<ol> ^command task
        ^robot1 <target1> ^robot2 <target2> ^robot3 <target3>)
```

Integration



USC Robotics Research Labs

Integration

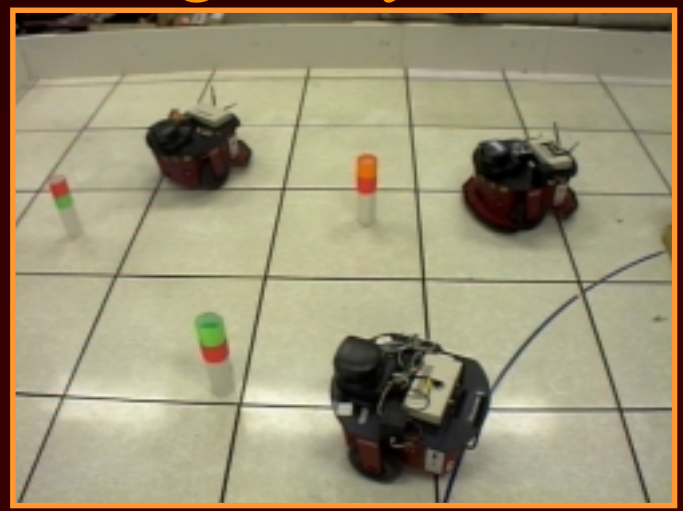


CMOMMT

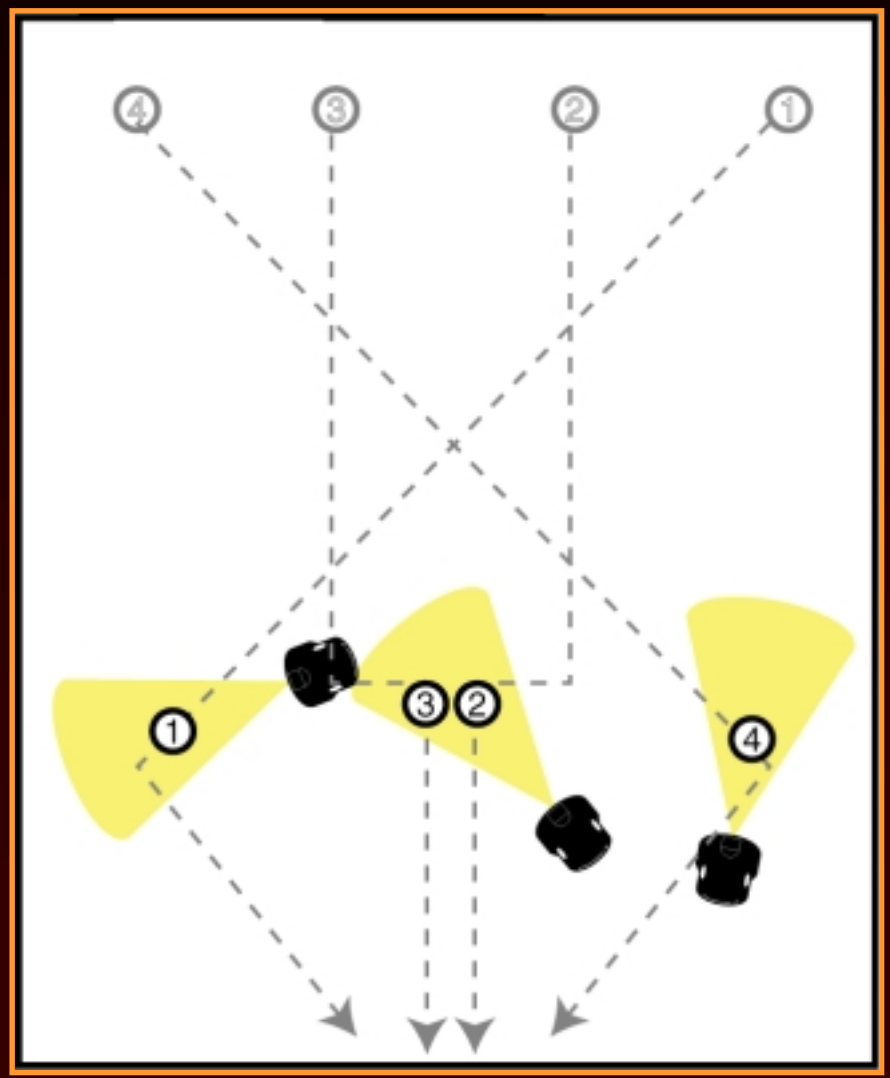
Cooperative Multi-robot Observation
of Multiple Moving Targets

Experiments with CMOMMT

- Cooperative Multi-Robot Observation of Multiple Moving Targets (Parker)
- Simple example of the class of problems BLE is designed for: multiple concurrent roles (target observers) with clear eligibility function
- Simple evaluation and
- systems for comparison
- (Werger & Mataric 2000)



CMOMMT Scenario



Broadcast of Local Eligibility

applied to the

CMOMMT

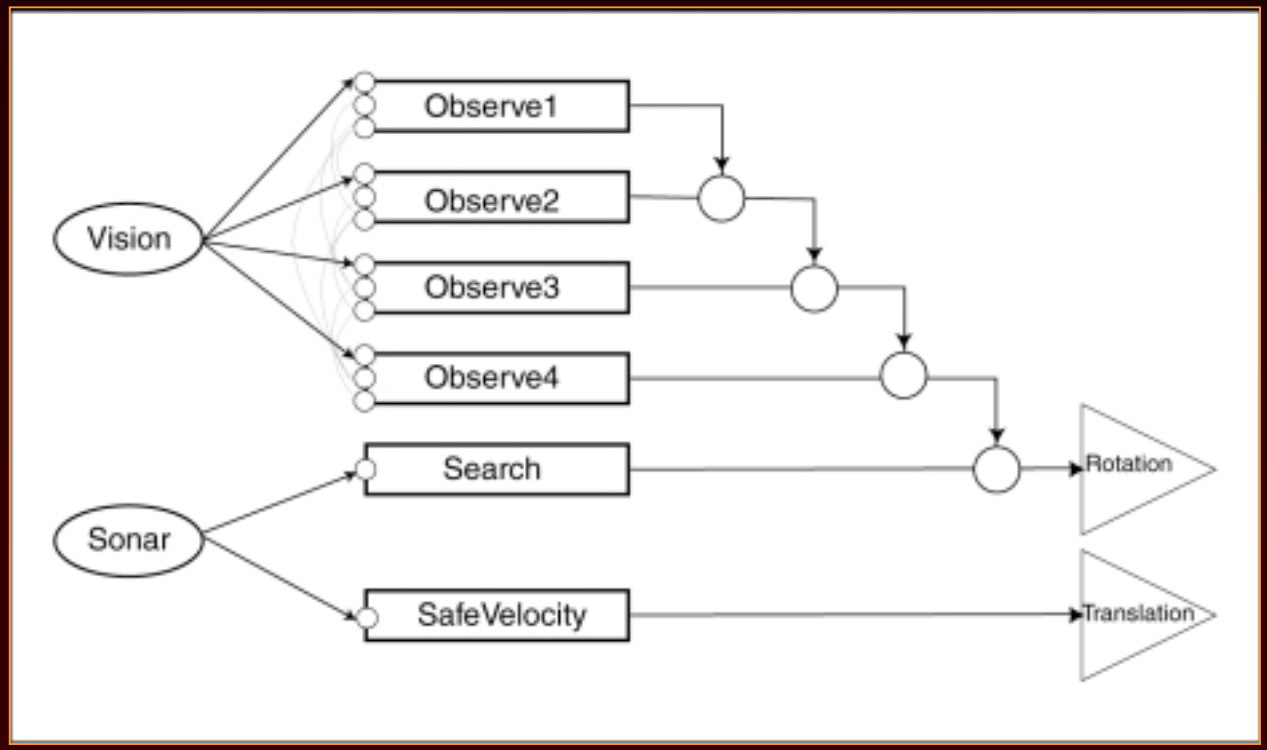
Cooperative Observation Task

Barry Brian Werger

Robotics Research Lab
University of Southern California

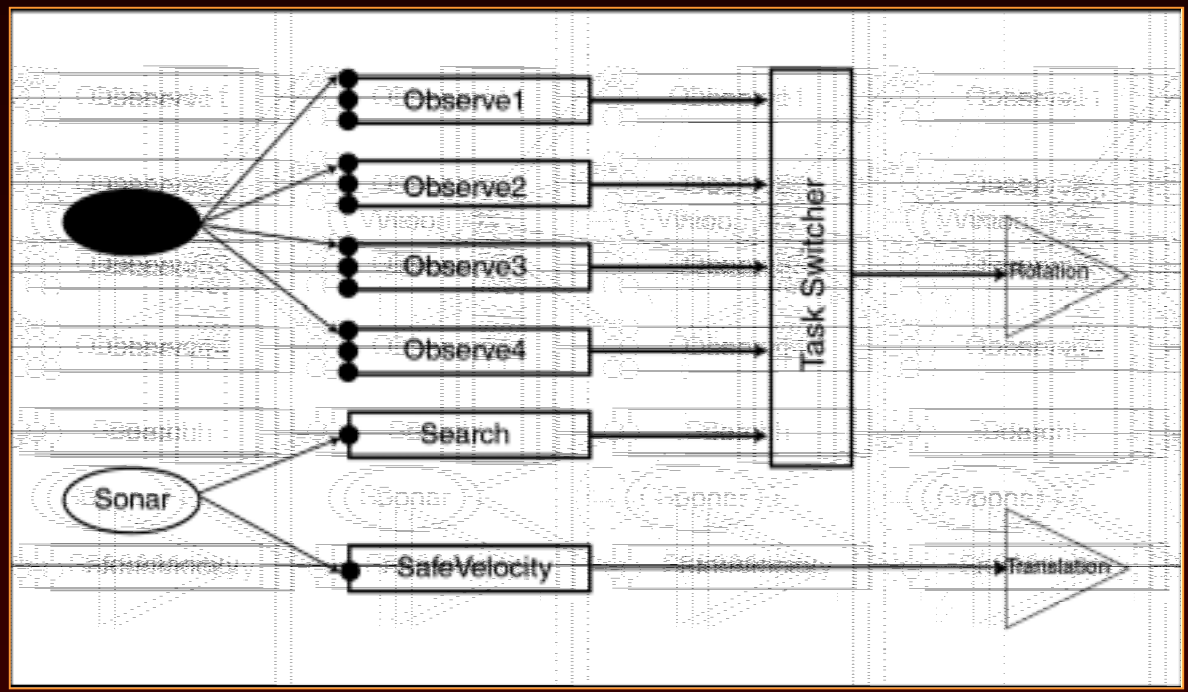
Behavior-Based Control for CMOMMT

- Subsumption hierarchy prioritizes targets
- "Local broadcast" spreads target distances
- Defaults to *Search*

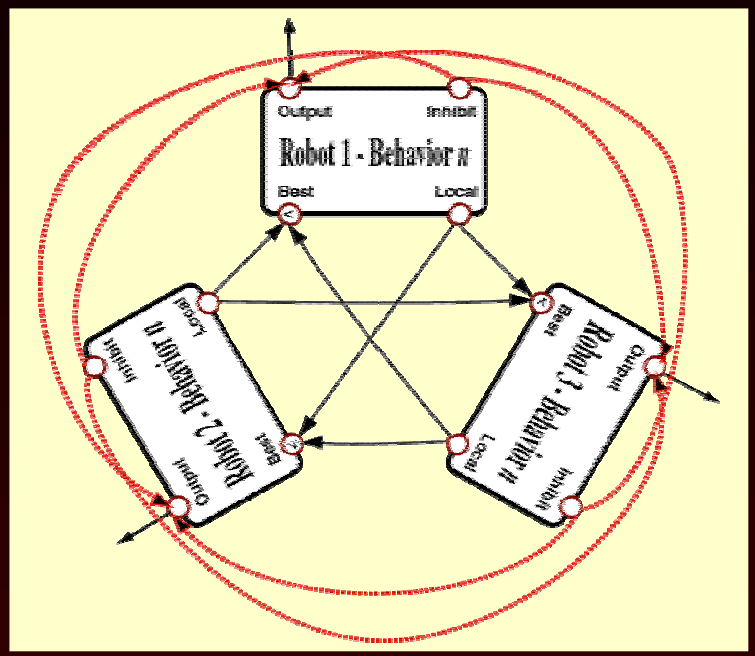


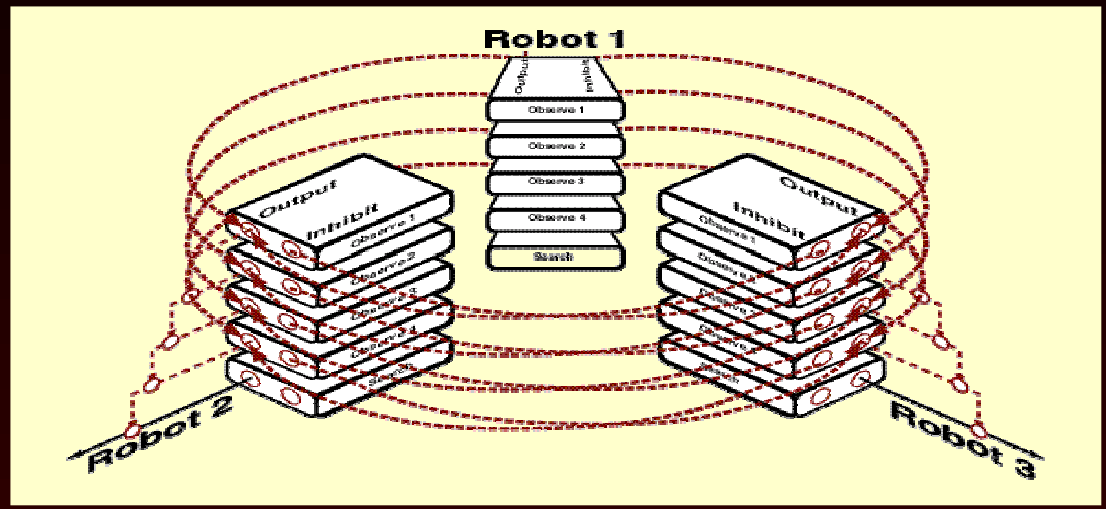
Soar Control for CMOMMT

- Task switching behavior replaces subsumption hierarchy



Behavior-Based Coordination

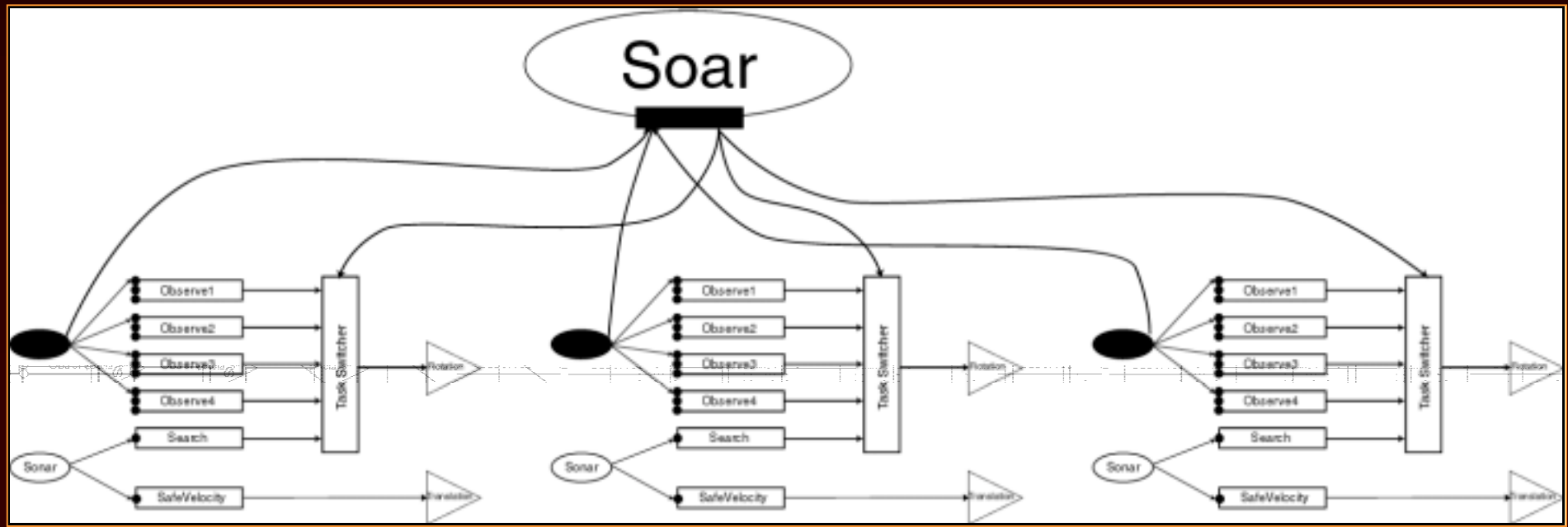




- Peer groups
- Each robot controlled by its highest priority non-cross-inhibited behavior - the highest priority active behavior which is not claimed by a more eligible robot.

Soar Coordination

- Centralized task allocation
- Soar receives observation data, sends task assignments



TankAyllu

Hard Tank Teamwork

USC Robotics Research Labs

Introduction

- Inspiration
  - QuakeBots, TankSOAR
  - Need to scale up tasks for behavior-based approaches
- Goals
  - Develop an interesting, robust framework for conflicting-subgoal tasks
  - As similar as possible to TankSOAR, with SOAR interface
  - Develop and compare “situated” team strategies
  - Get Soar community involved

Differences from TankSOAR (1)

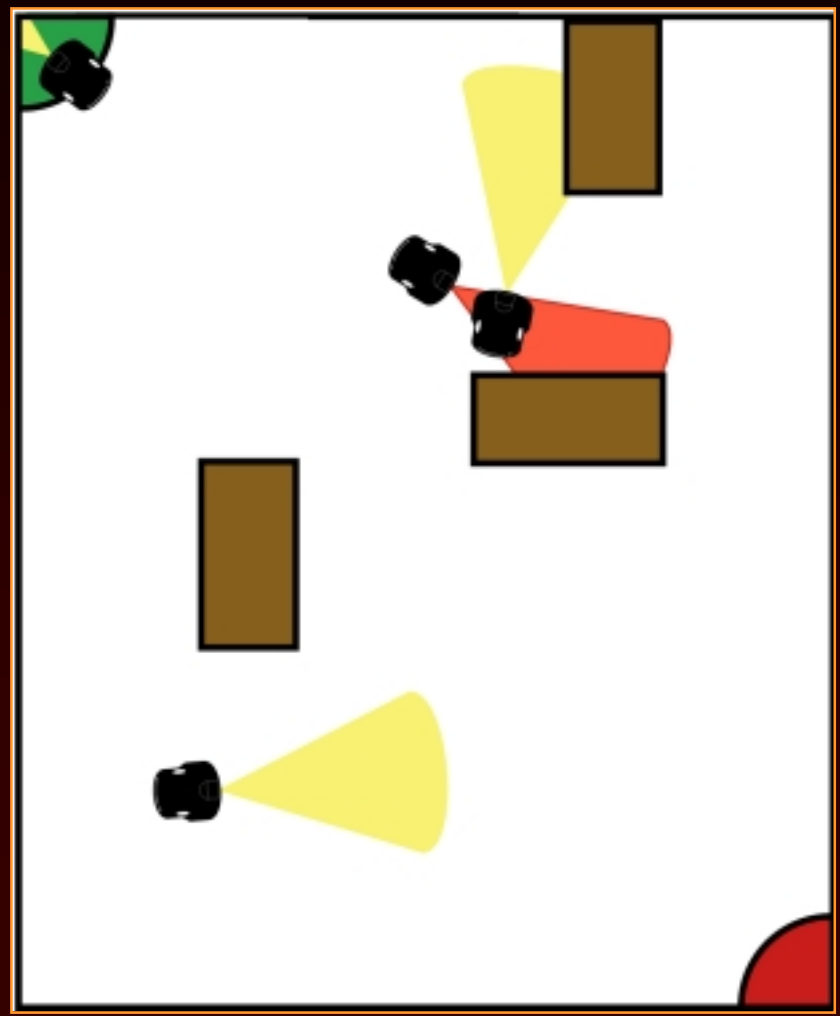
- Physical robots
  - Uncertain sensors, actuators
  - Really bad positioning
  - Limited information
- Continuous
  - Actions
  - Time
  - Life (focus away from reincarnation)
- Distributed
- Focus on multi-robot teams

Differences from TankSOAR (2)

- No missiles (energy attacks)
  - Proximity-based, multi-target
- Vision rather than radar, no cost (?)
  - Thus resource maintenance a bit simpler
- Sonar model rather than "blocked"
- No omnidirectional sensing (smell, motion, incoming direction, sound, rwave direction)
- Chargers not "instant death"

Physical Entities

- Tanks
  - Teammates
  - Opponents
- Chargers
  - Health
  - Energy
- Obstacles
- 18x22 ft. corral



Robot Resources

- Health
  - Maximum of 1000
  - Decreases when attacked and not shielded
  - Increased 100/sec on charger
- Energy
  - Maximum of 1000
  - Decreased by shield use, attacks
  - Increased 100/sec on charger
- THESE VALUES WILL HAVE TO BE PLAYTESTE□D!

USC Robotics Research Labs

Robot Capabilities

- Attack
  - Robot decides amount of energy to use (max 250/sec)
  - Scaled by distance, must be in view, all in view
- Move
  - Translation, TrasVel (safe), Rotation, RotVel
- Shields
  - Absorb 2 x energy expended
  - Cost 20 units/second
- Camera Pan

USC Robotics Research Labs

Robot Sensors

- Sonars (8 or 16)
  - Distance, angle
- Vision
  - Teammate (dist, angle)
  - Opponent (distance, Angle)
  - Healthcharger (dist, angle)
  - Energycharger (dist, angle)
  - Direction marker(s)
- Incoming (how many)
- Rwaves (how many) (?)
- Charger in use (?)
- Auxiliary
  - Energy
  - Health
  - Shield-status
  - On charger (energy, health)
  - Heading (error-prone)
  - Coordinates (error-prone)

USC Robotics Research Labs

Recharging

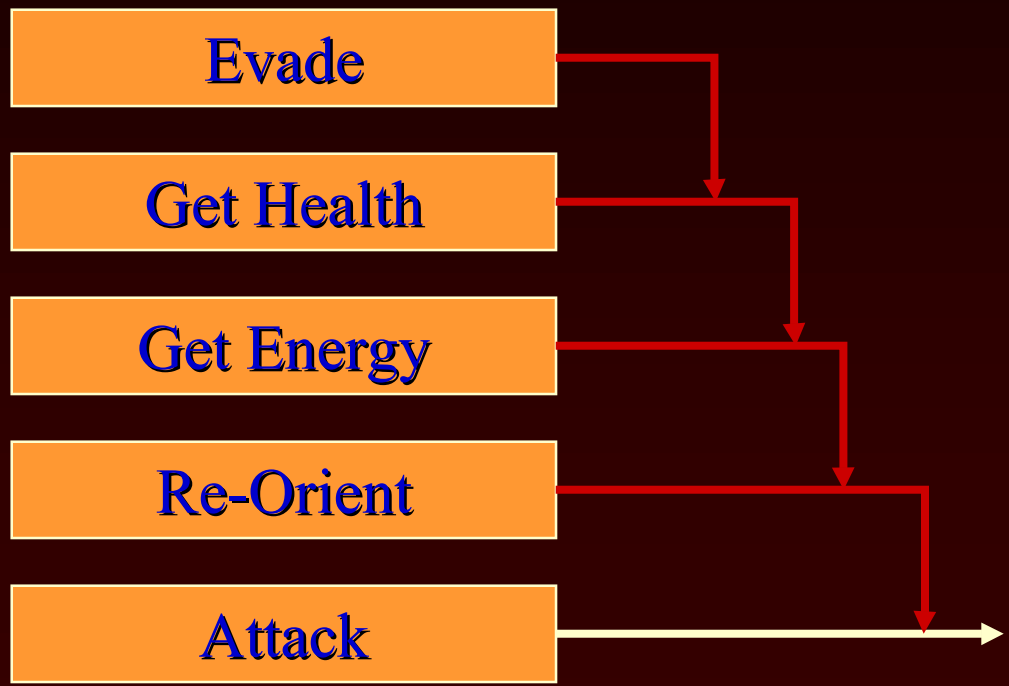
- 150 units/sec
- Incoming attacks at 250 units/sec when on charger
  - Regardless of distance
  - Regardless of strength
- Cannot attack from charger
  - Must be facing wall

Tank Sensing and Control Environment

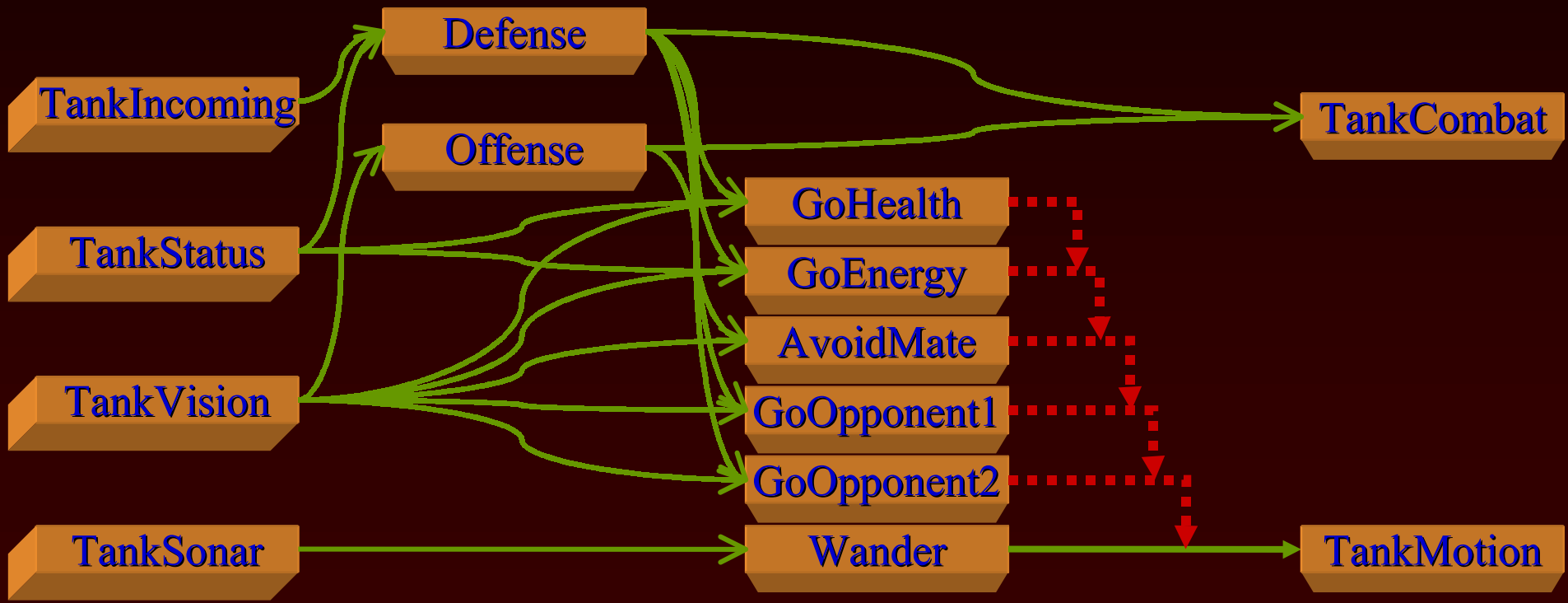
- Sensor and actuator interface behaviors

SimpleTank: Basis Behaviors

- Attack
  - TargetTank, strength, dist
- Evade
- GetEnergy
- GetHealth
- Re-Orient (team safety)



Simple BBC Controller



TankAyllu Status

- Foundation implemented and tested
  - Object sensing, attacking, re-charging, etc.
- Minimal 2-robot testing
  - With simple subsumption control
  - More bug shakeout than result
- SOAR integration issues solved
  - But not yet fully integrated

Generalized Ayllu Interaction

- We will ultimately have an interface such as:

```
^io
    ^input-link
        ^ayllu-message integer/float/string
^io
    ^output-link
        ^ayllu-message
            ^destination
                ^host
                ^behavior
                ^port
            ^value integer/float/string
```

Survey

- Would you be interested in using Soar-Ayllu / TankAyllu?
  - Would you go/send students to USC robotics lab
  - Would you invest in robots?
- Would you prefer a standard or flexible TankAyllu interface?
- How much would availability of a simulator influence your interest?