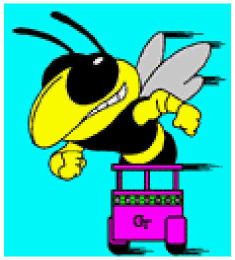
Multi-Level Learning in Hybrid Deliberative/Reactive Mobile Robot Architectural Software Systems

DARPA MARS Review Meeting - January 2000

Approved for public release: distribution unlimited

Personnel



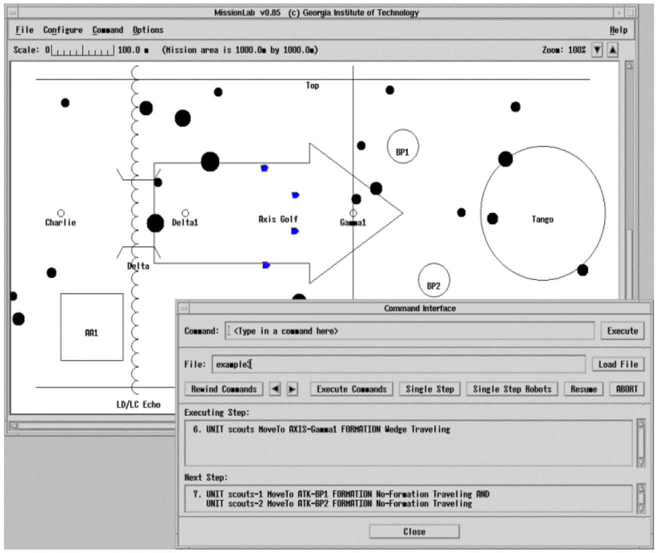
- Georgia Tech
  - College of Computing
    - Prof. Ron Arkin
    - Prof. Chris Atkeson
    - Prof. Sven Koenig
  - Georgia Tech Research Institute
    - Dr. Tom Collins
- Mobile Intelligence Inc.
    - Dr. Doug MacKenzie

- Students
  - Amin Atrash
  - Bhaskar Dutt
  - Brian Ellenberger
  - Mel Eriksen
  - Max Likachev
  - Brian Lee
  - Sapan Mehta

Adaptation and Learning Methods

- Case-based Reasoning for:
  - deliberative guidance ("wizardry")
  - reactive situational-dependent behavioral configuration
- Reinforcement learning for:
  - run-time behavioral adjustment
  - behavioral assemblage selection
- Probabilistic behavioral transitions
  - gentler context switching
  - experience-based planning guidance

Available Robots and *MissionLab* Console



Command: <Type in a command here> Execute

File: example3 Load File

Executing Step:

```
6. UNIT scouts MoveTo AXIS-Gamma1 FORMATION Wedge Traveling
```

Next Step:

```
7. UNIT scouts-1 MoveTo ATK-BP1 FORMATION No-Formation Traveling AND
   UNIT scouts-2 MoveTo ATK-BP2 FORMATION No-Formation Traveling
```

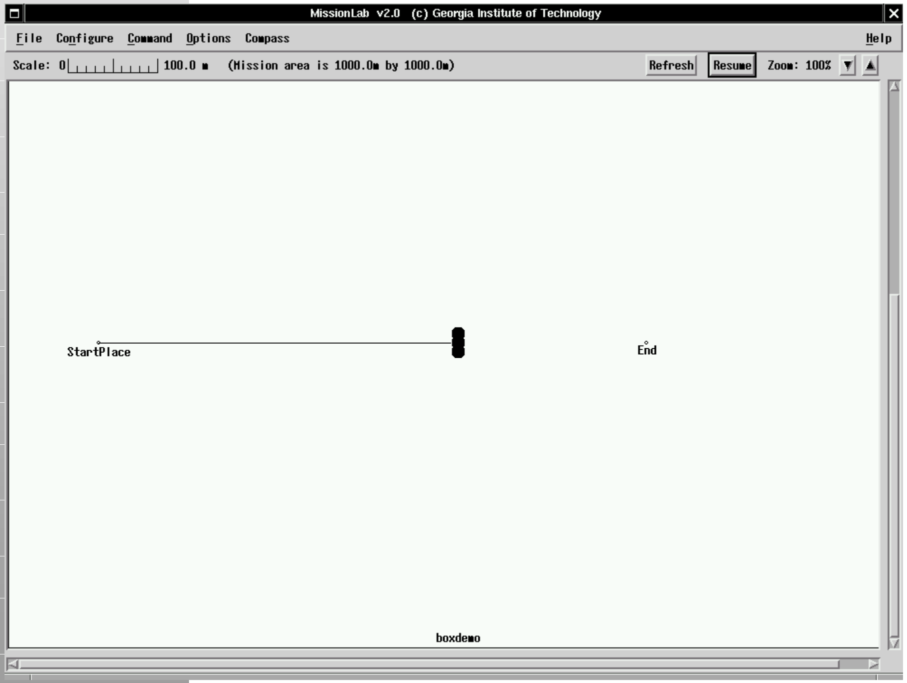
1. Learning Momentum

- Reactive learning via dynamic gain alteration (parametric adjustment)
- Continuous adaptation based on recent experience
- Situational analyses required
- In a nutshell: If it works, keep doing it a bit harder; if it doesn’t, try something different

Learning Momentum - Design

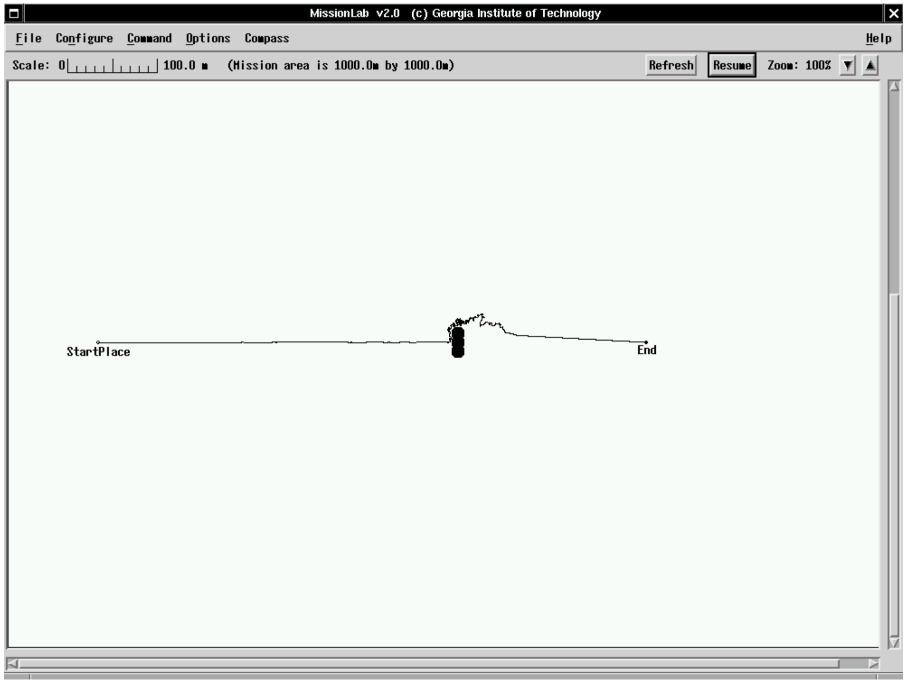
- Integrated into MissionLab in CNL Library
- Works with MOVE_TO_GOAL, COOP, and AVOID_OBSTACLES
- Has not yet been extended to all behaviors

Simple Example



Scale: 0 100.0 m (Mission area is 1000.0m by 1000.0m)

Zoom: 100%



Scale: 0 100.0 m (Mission area is 1000.0m by 1000.0m)

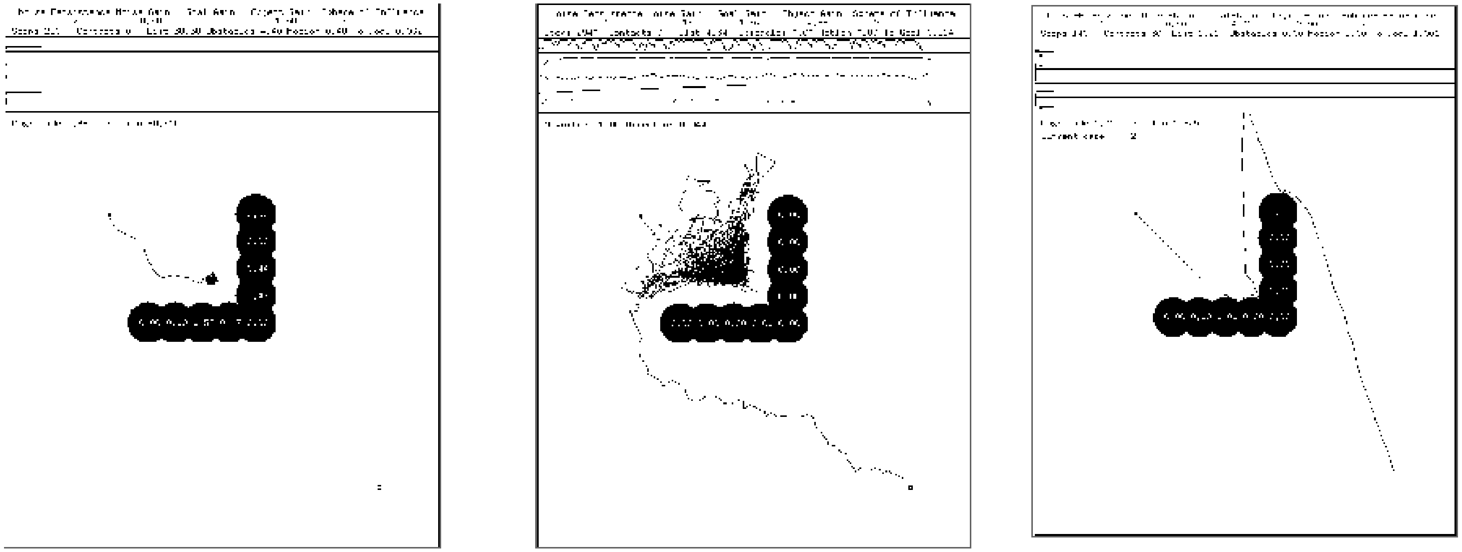
Zoom: 100%

Learning Momentum - Future Work

- Extension to additional CNL behaviors
- Make thresholds for state determination rules accessible from cfgedit
- Integrate with CBR and RL

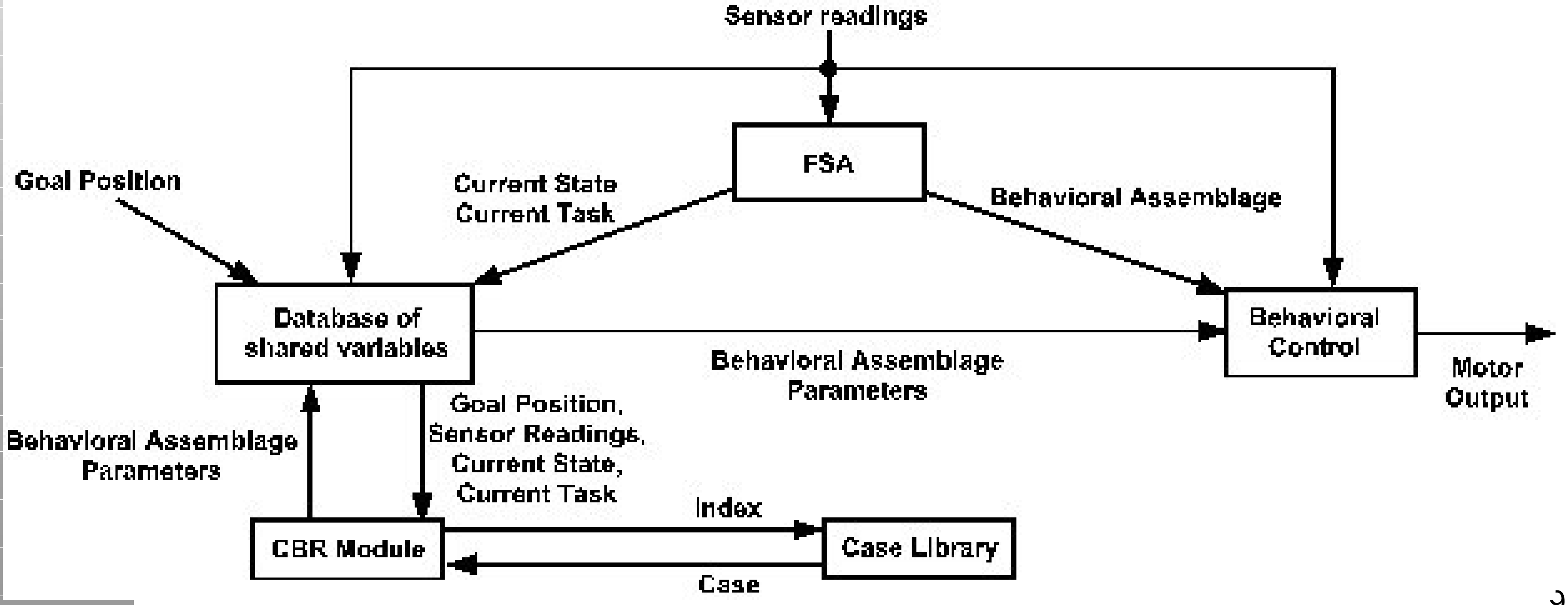
2. CBR for Behavioral Selection

- Another form of reactive learning
- Previous systems include: ACBARR and SINS
- Discontinuous behavioral switching



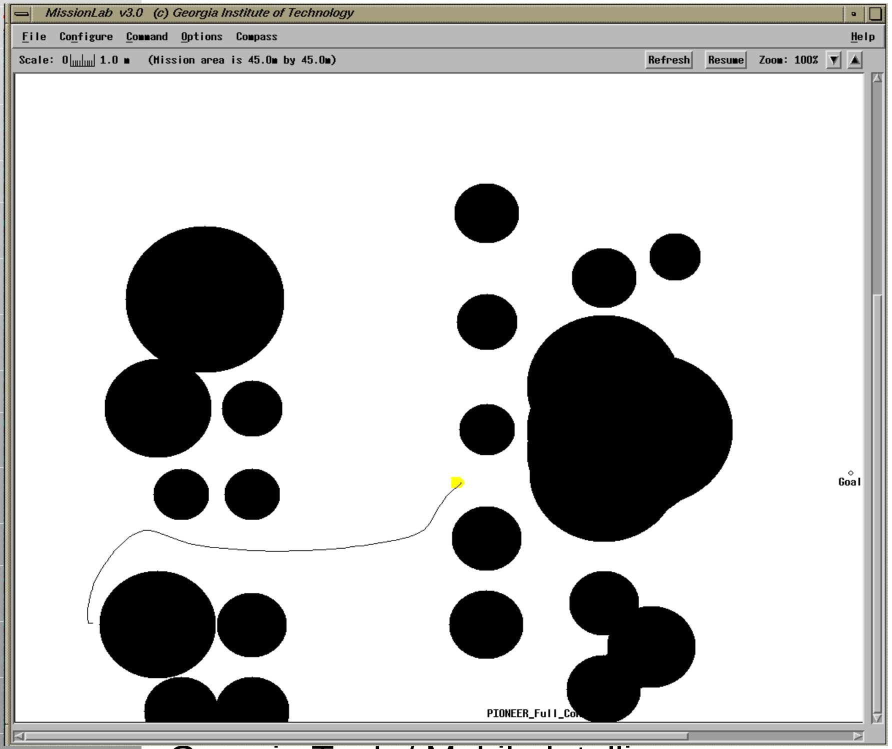
Case-Based Reasoning for Behavioral Selection - Current Design

- The CBR Module is designed as a stand-alone module
- A hard-coded library of eight cases for MoveToGoal tasks
- Case - a set of parameters for each primitive behavior in the current assemblage and index into the library

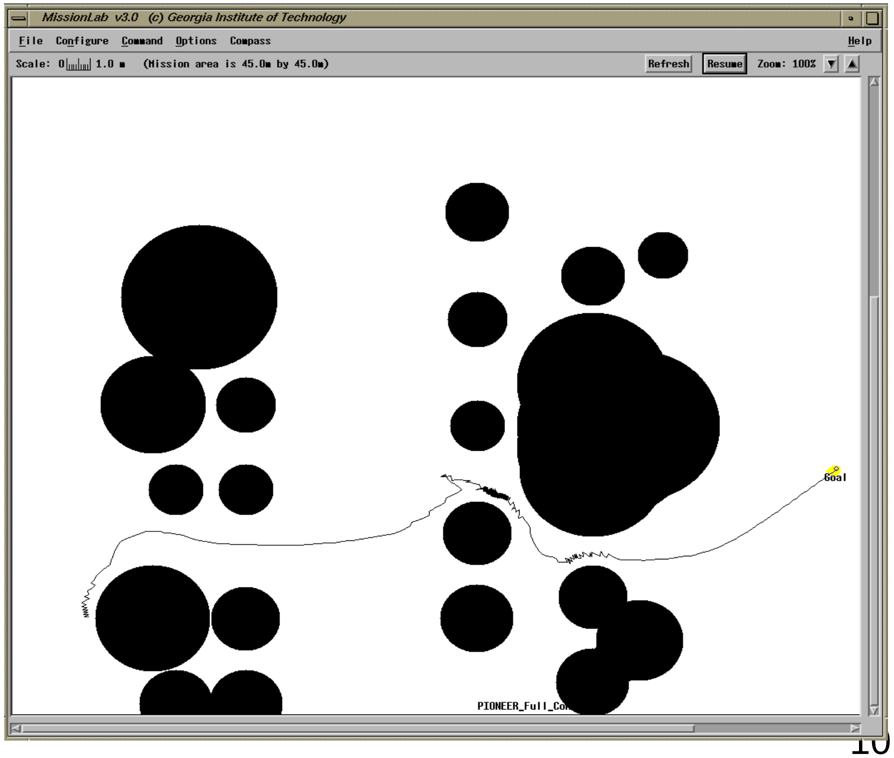

Sensor readings
FSA
Goal Position
Current State
Current Task
Behavioral Assemblage
Database of shared variables
Behavioral Control
Behavioral Assemblage Parameters
Motor Output
Goal Position, Sensor Readings, Current State, Current Task
Behavioral Assemblage Parameters
Index
CBR Module
Case Library
Case

Case-Based Reasoning for Behavioral Selection - Current Results

- On the Left - MoveToGoal without CBR Module
- On the Right - MoveToGoal with CBR Module



(Mission area is 45.0m by 45.0m)



(Mission area is 45.0m by 45.0m)

Case-Based Reasoning for Behavioral Selection - Future Plans

- Two levels of operation: choosing and adapting parameters for selected behavior assemblages as well as choosing and adapting the whole new behavior assemblages
- Automatic learning and modification of cases through experience
- Improvement of case/index/feature selection and adaptation
- Integration with Q-learning and Momentum Learning
- Identification of relevant task domain case libraries

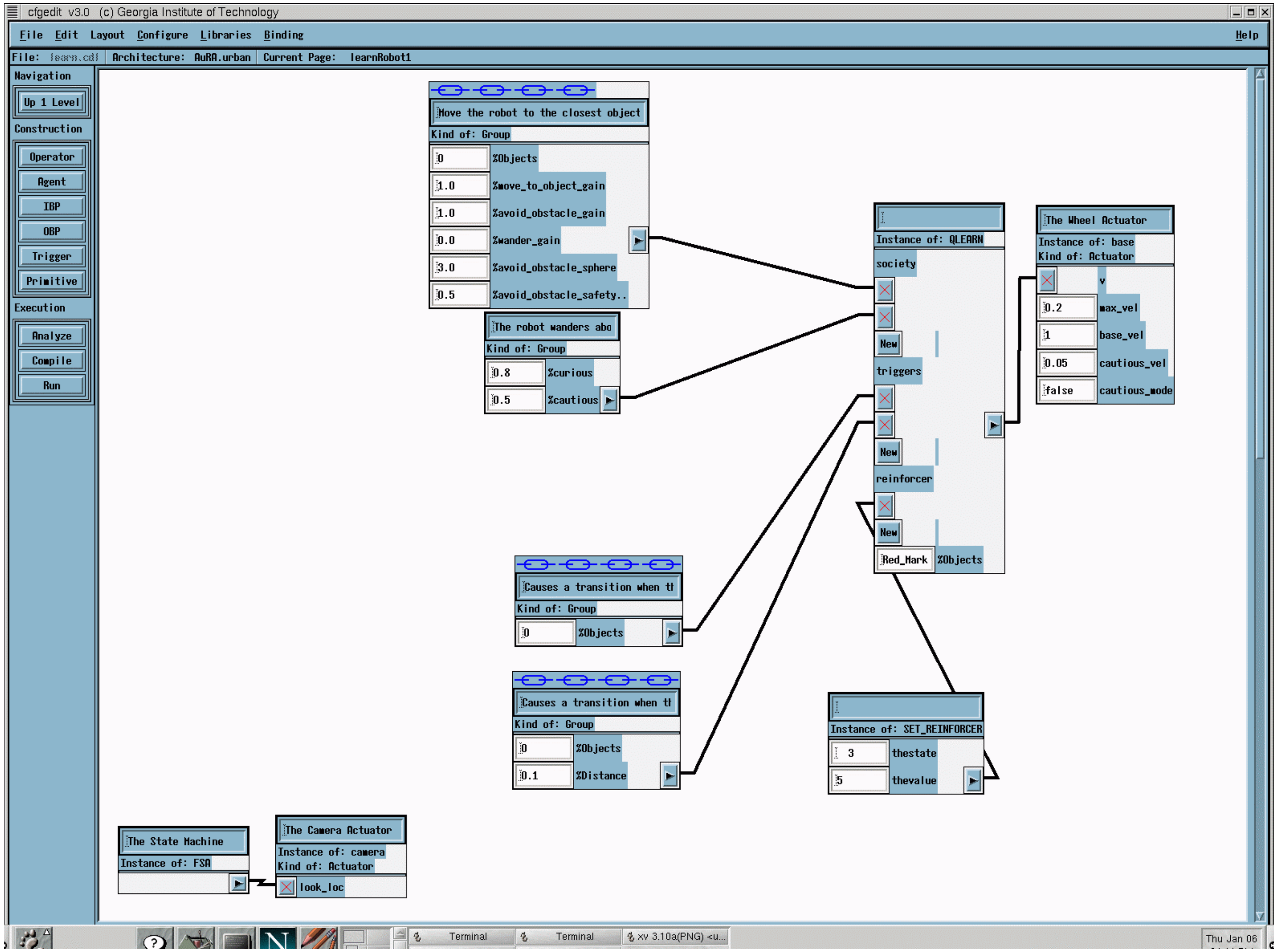
3. Reinforcement learning for Behavioral Assemblage Selection

- Reinforcement learning at coarse granularity (behavioral assemblage selection)
- State space tractable
- Operates at level above learning momentum (selection as opposed to adjustment)
- Have added the ability to dynamically choose which behavioral assemblage to execute
- Ability to learn which assemblage to choose using wide variety of Reinforcement Learning methods: Q-learning, Value Iteration, (Policy Iteration in near future)

Selecting Behavioral Assemblages - Specifics

- Replace the FSA with an interface allowing user to specify the environmental and behavioral states
- Agent learns transitions between behavior states
- Learning algorithm is implemented as an abstract module and different learning algorithms can be swapped in and out as desired.
- CNL function interfaces robot executable and learning algorithm

Integrated System



cfgedit v3.0 (c) Georgia Institute of Technology

File: learn.cdl | Architecture: AuRA.urban | Current Page: learnRobot1

Move the robot to the closest object
Kind of: Group
0 %Objects
1.0 %move_to_object_gain
1.0 %avoid_obstacle_gain
0.0 %wander_gain
3.0 %avoid_obstacle_sphere
0.5 %avoid_obstacle_safety..

The robot wanders abo
Kind of: Group
0.8 %curious
0.5 %cautious

Instance of: QLEARN
society
triggers
reinforcer
Red_Mark %Objects

The Wheel Actuator
Instance of: base
Kind of: Actuator
v
0.2 max_vel
1 base_vel
0.05 cautious_vel
false cautious_mode

Causes a transition when th
Kind of: Group
0 %Objects

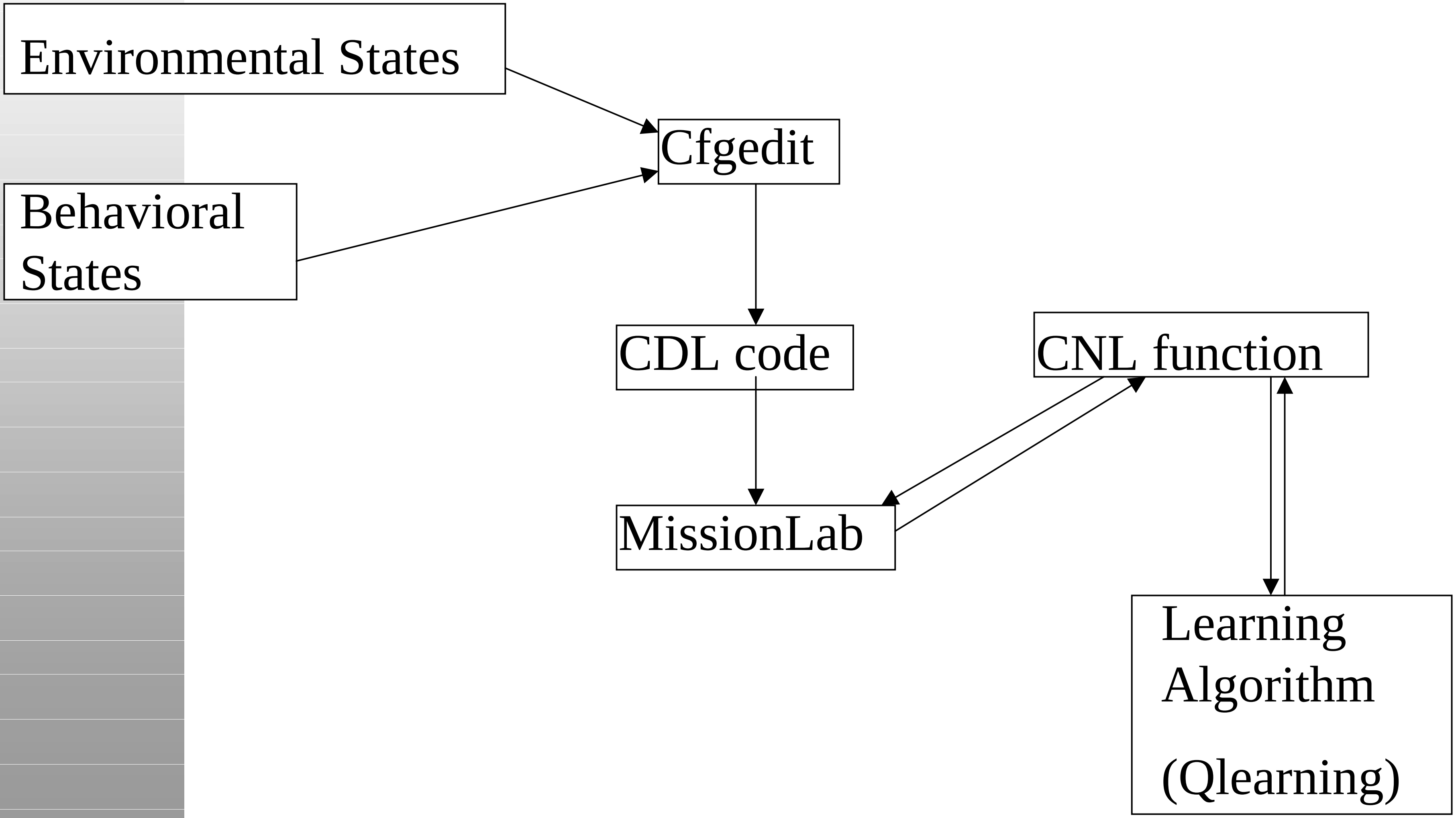
Causes a transition when th
Kind of: Group
0 %Objects
0.1 %Distance

Instance of: SET_REINFORCER
3 thestate
5 thevalue

The State Machine
Instance of: FSA

The Camera Actuator
Instance of: camera
Kind of: Actuator
look_loc

Architecture



Environmental States

Behavioral States

Cfgedit

CDL code

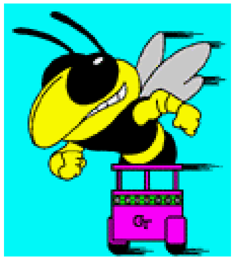
MissionLab

CNL function

Learning Algorithm

(Qlearning)

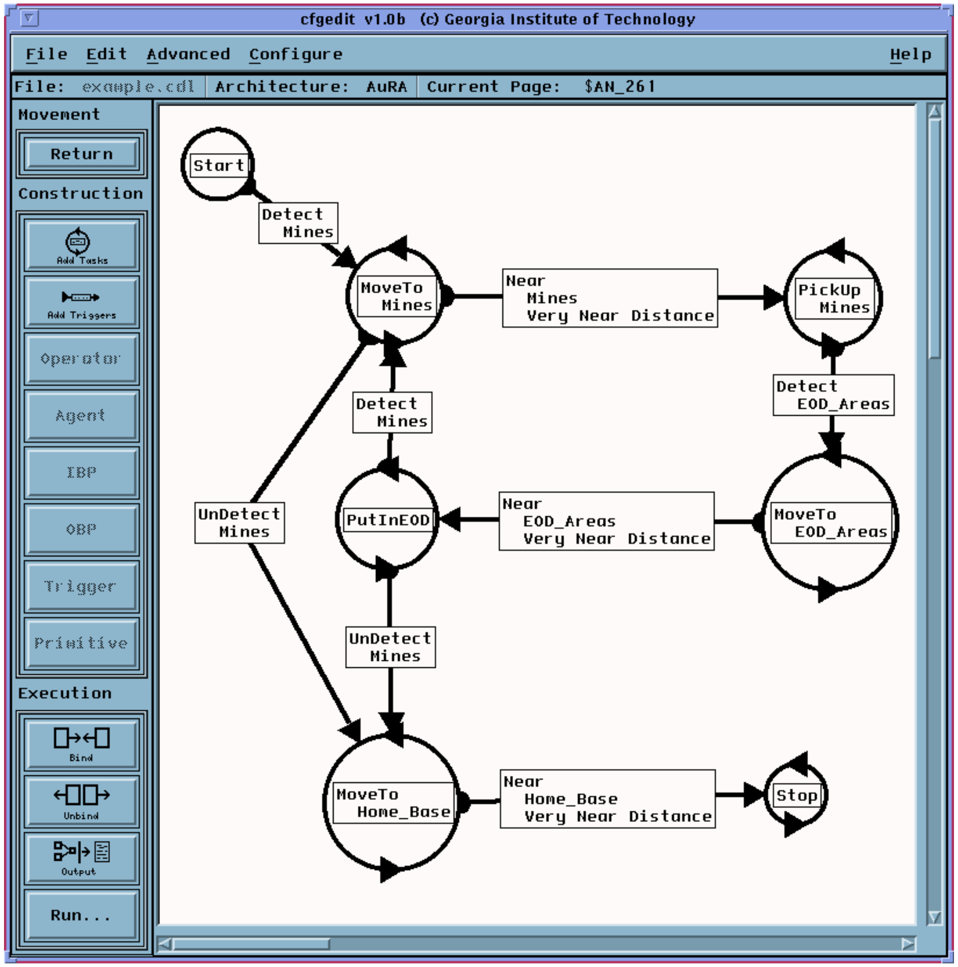
RL - Next Steps



- Change implementation of Behavioral Assemblages in *Missionlab* from simply being statically compiled into the CDL code to a more dynamic representation.
- Create relevant scenarios and test *Missionlab*'s ability to learn good solutions
- Look at new learning algorithms to exploit the advantages of Behavioral Assemblages selection
- Conduct extensive simulation studies then implement on robot platforms

4. CBR "Wizardry"

- Experience-driven assistance in mission specification
- At deliberative level above existing plan representation (FSA)
- Provides mission planning support in context

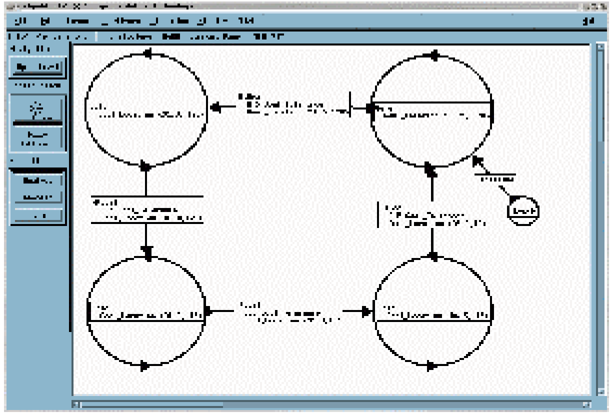


cfgedit v1.0b (c) Georgia Institute of Technology

File: example.cdl Architecture: AuRA Current Page: $AN_261

CBR Wizardry / Usability Improvements

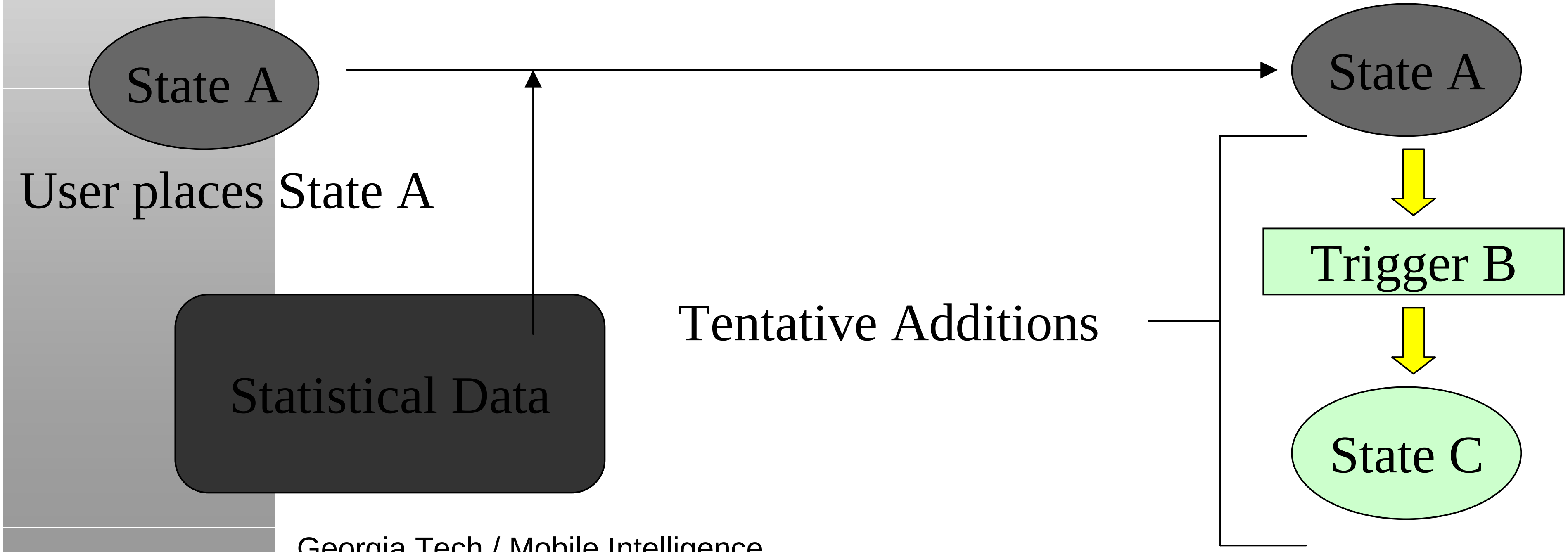
- Current Methods: Using GUI to construct FSA - may be difficult for inexperienced users.



- Goal: Automate plan creation as much as possible while providing unobtrusive support to user.

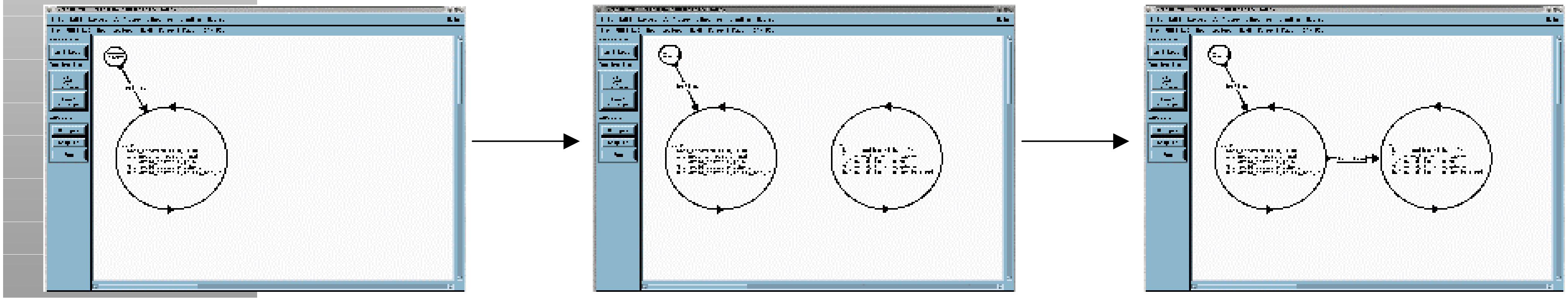
Tentative Insertion of FSA Elements:
A user support mechanism currently being worked on

- Some FSA elements very often occur together.
- Statistical data on this can be gathered.
- When user places a state, a trigger and state that follow this state often enough can be tentatively inserted into the FSA.
- Comparable to URL completion features in web browsers.



Recording Plan Creation Process

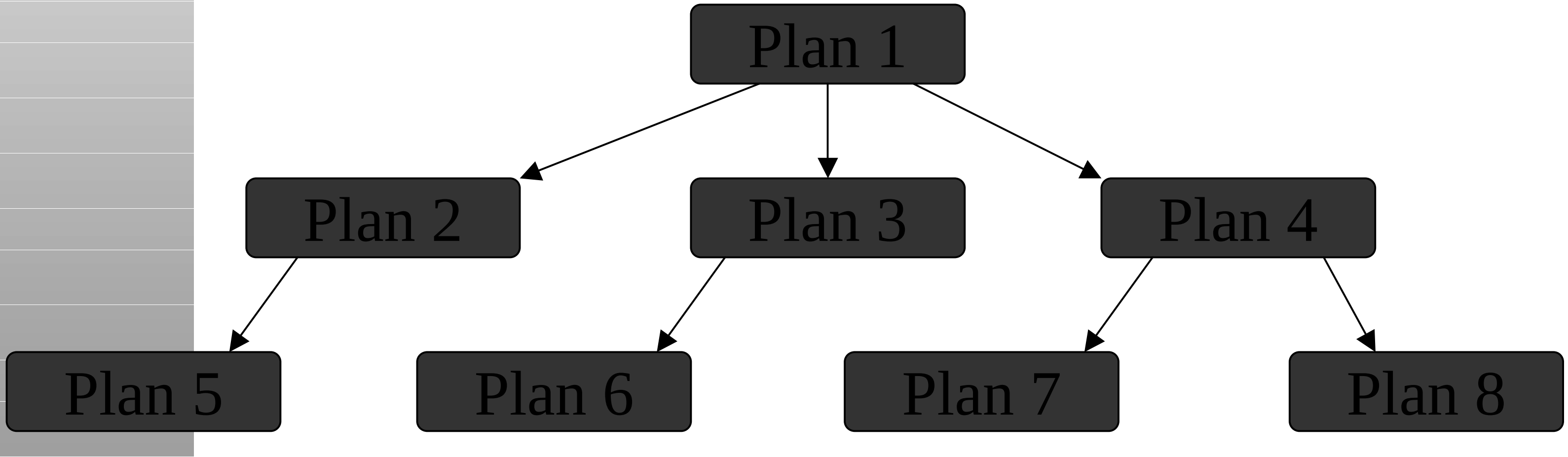
- Pinpointing where user has trouble during plan creation is important prerequisite to improving software usability.
- There was no way to record plan creation process in MissionLab.
- Module now created that records user’s actions as (s)he creates the plan. This recording can later be played back and points where the user stumbled can thus be identified.



The Creation of a Plan

Wizardry - Future Work

- Use of plan creation recordings during usability studies to identify stumbling blocks in process.
- Creation of plan templates (frameworks of some commonly used plan types e.g. reconnaissance missions)
- Collection of library of plans which can be placed at different points in “plan creation tree”. This can then be used in a plan creation wizard.



Plan 1

Plan 2

Plan 3

Plan 4

Plan 5

Plan 6

Plan 7

Plan 8

Plan Creation Tree

5. Probabilistic Planning and Execution

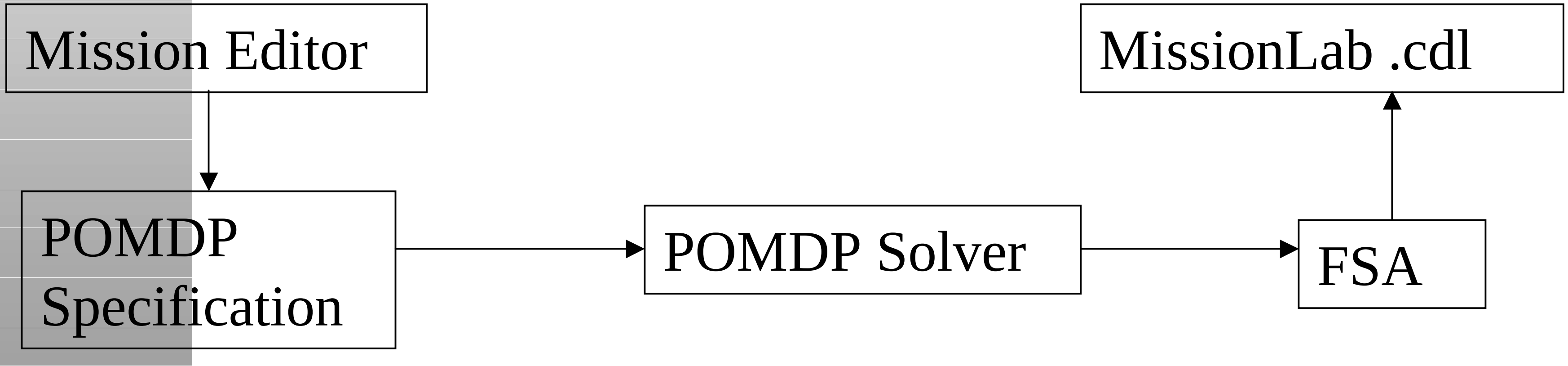
- "Softer, kinder" method for matching situations and their perceptual triggers

- Expectations generated based on situational probabilities regarding behavioral performance (e.g., obstacle densities and traversability), using them at planning stages for behavioral selection

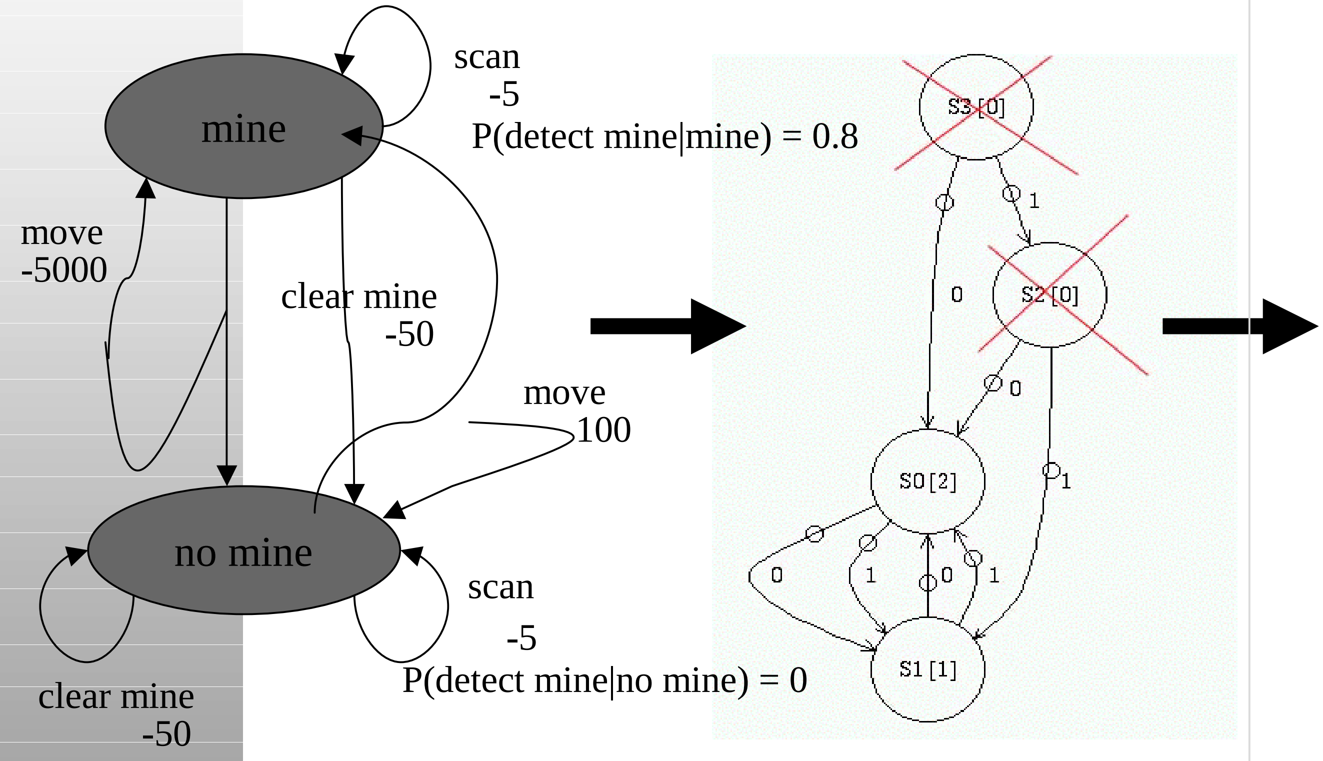
- Markov Decision Process, Dempster-Shafer, and Bayesian methods to be investigated

Probabilistic Planning and Execution - Concept

- Find the optimal plan despite sensor uncertainty about the current environment

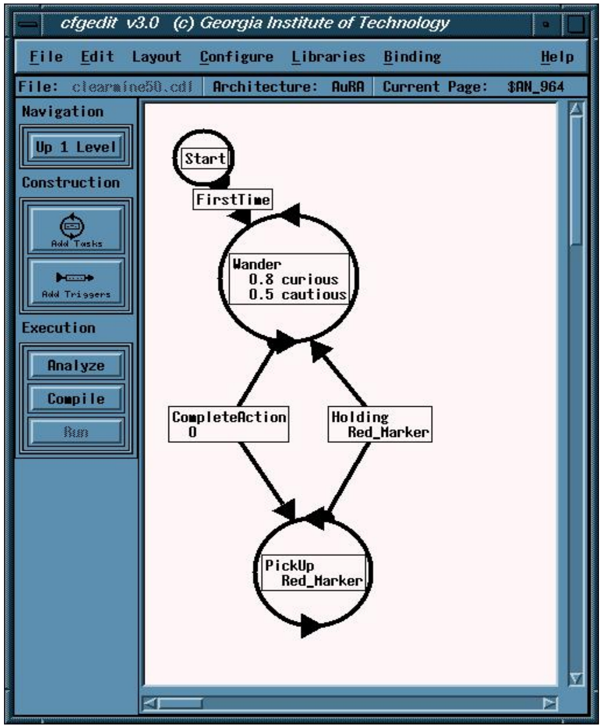


Probabilistic Methods: Current Status



POMDP

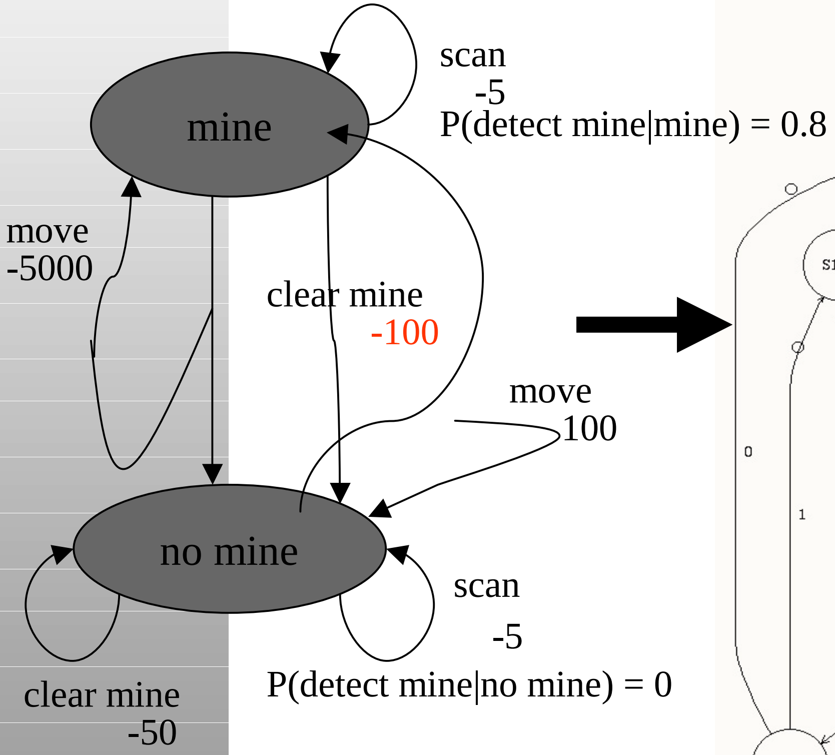
FSA



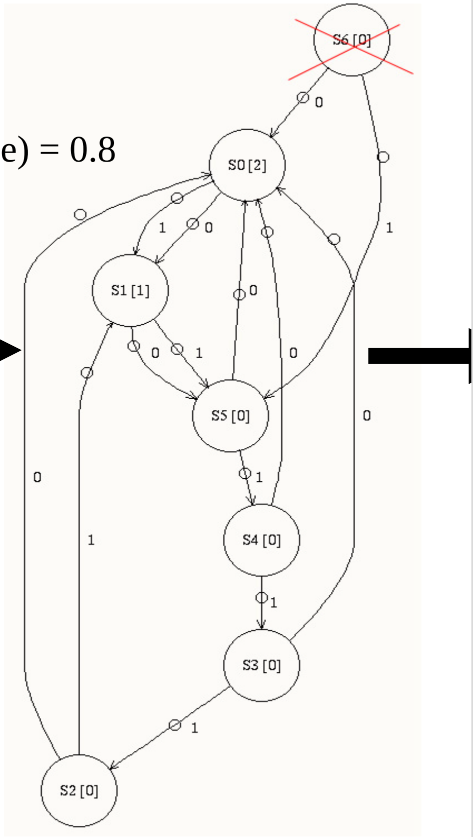
File: clearmine50.cdl | Architecture: AuRA | Current Page: $AN_964

MissionLab
(current work)

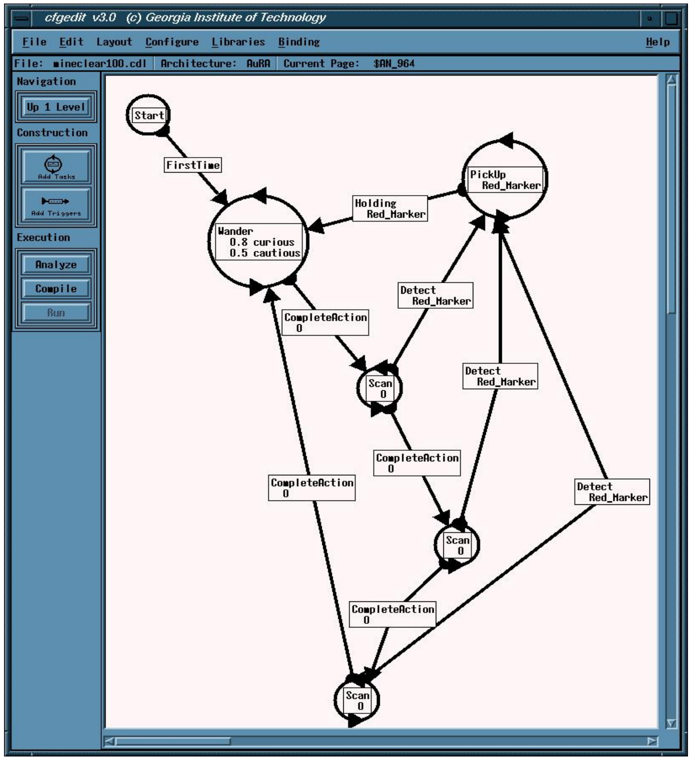
Varying Costs → Different Plans



POMDP



FSA



MissionLab (current work)

MIC’s Role

- Develop conceptual plan for integrating learning algorithms into *MissionLab*
- Guide students performing integration
- Assist in designing usability studies to evaluate integrated system
- Guide performance and evaluation of usability studies
- Identify key technologies in *MissionLab* which could be commercialized
- Support technology transfer to a designated company for commercialization

Schedule

Milestone		GFY01				GFY02				GFY03				GFY04		
	Jul	Oct	Jan	Apr	Jul	Oct	Jan	Apr	Jul	Oct	Jan	Apr	Jul	Oct	Jan	Apr
Demonstration of all learning algorithms in simulation			♦													
Initial integration within MissionLab on lab robots					♦											
Learning algorithms demonstrated in relevant scenarios									♦							
MissionLab demonstration on government platforms												♦				
Enhanced learning algorithms on government platforms														♦		
Final demonstrations of relevant scenarios with govt. platforms																♦