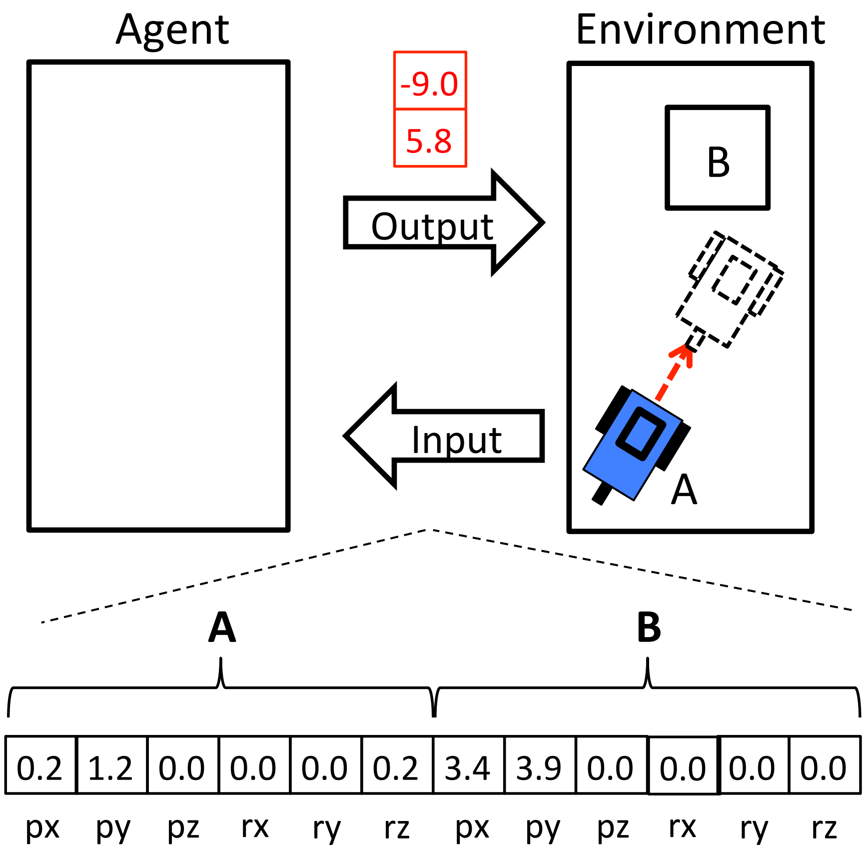
Learning Modal Continuous Models

Joseph Xu

Soar Workshop 2012

Setting: Continuous Environment



- Input to the agent is a set of objects with continuous properties
  - Position, rotation, scaling, ...
- Output is fixed-length vector of continuous numbers
- Agent runs in lock-step with environment
- Fully observable

Levels of Problem Solving

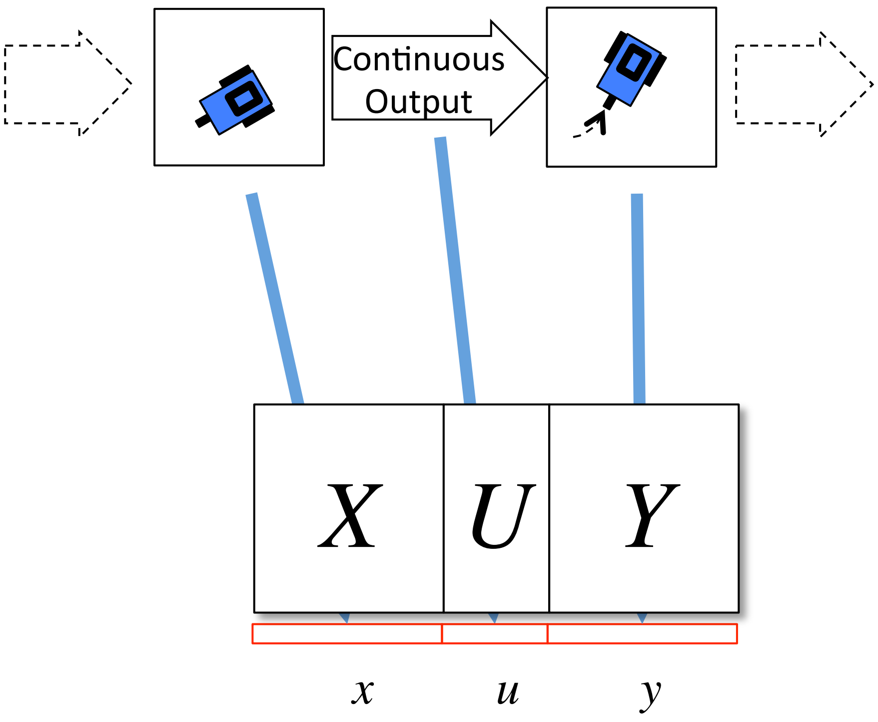
Characteristics	Problem Solving Method	Knowledge Required
Faster Task Completion General Solutions	Symbolic Planning	Symbolic Model
	Symbolic Model Free Methods (RL)	Symbolic Abstraction
	Continuous Sampling Methods (RRT)	Continuous Model
Slower Task Completion Specific Solutions	Motor Babbling	Goal Recognition

Continuous Model Learning

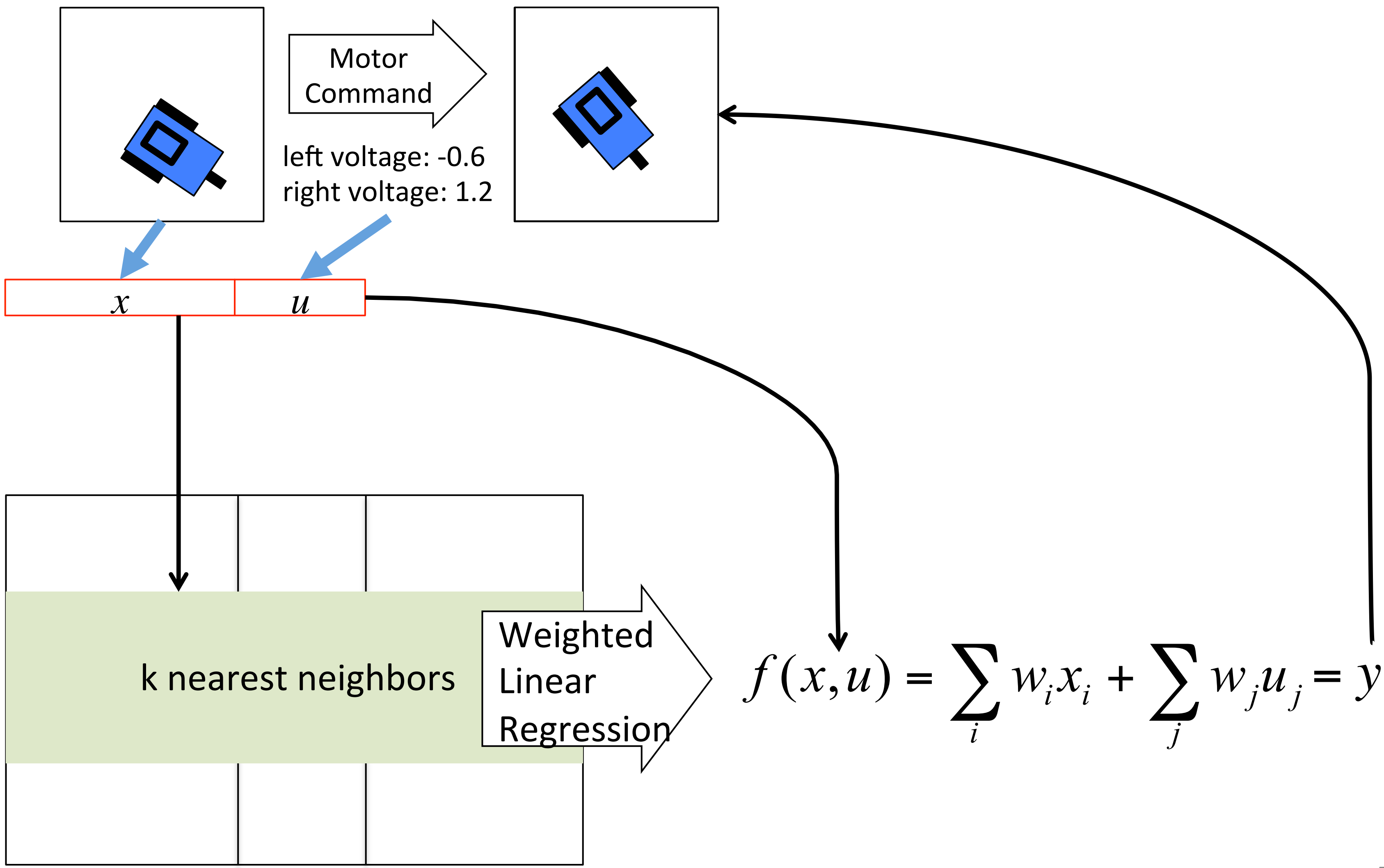
- Learn a function

$$f(x,u) \rightarrow y$$

- x: current continuous state vector
- u: current output vector
- y: state vector in next time step



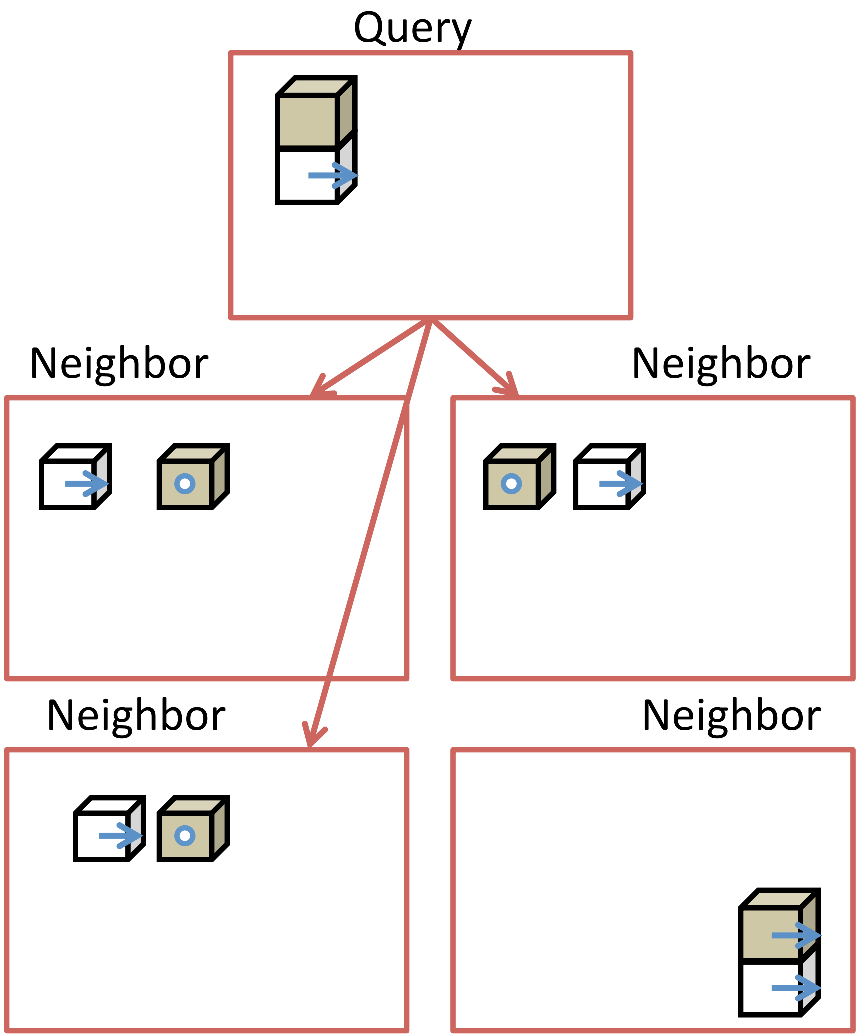
Locally Weighted Regression



$$f(x,u) = \sum_i w_i x_i + \sum_j w_j u_j = y$$

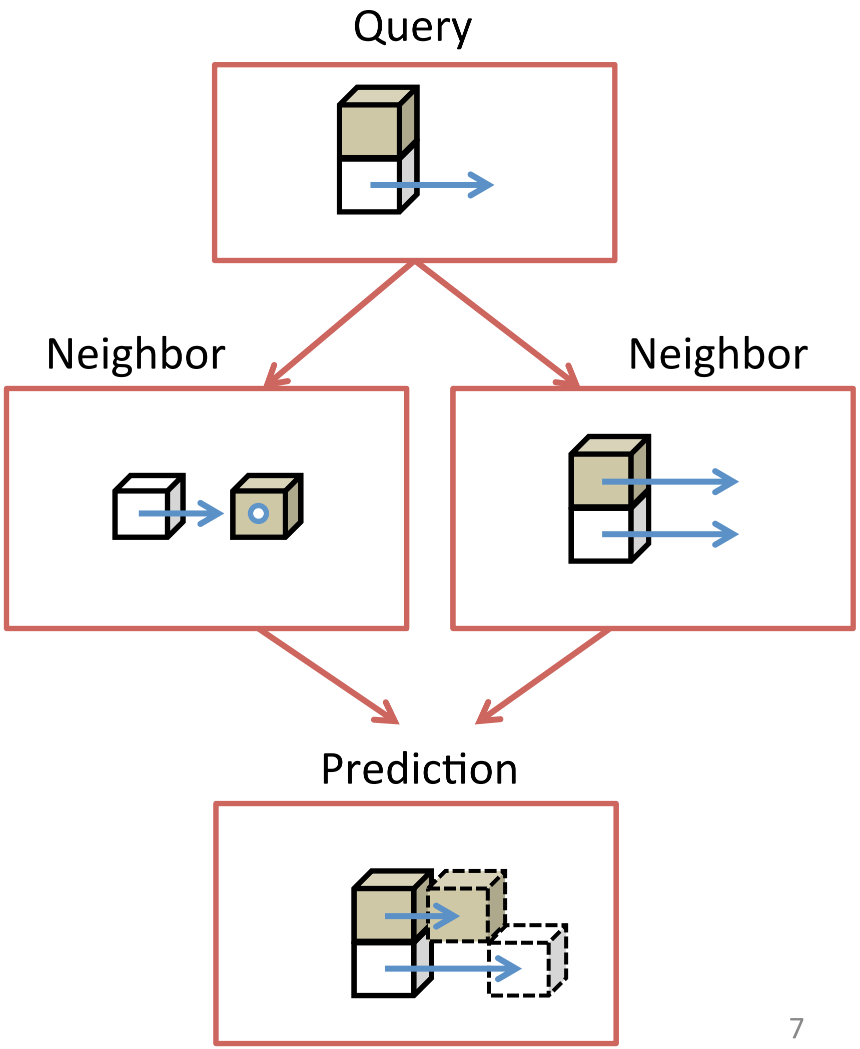
Problems with LWR

- Euclidean distance doesn't capture relational similarity
- Averages over neighbors exhibiting different types of interactions



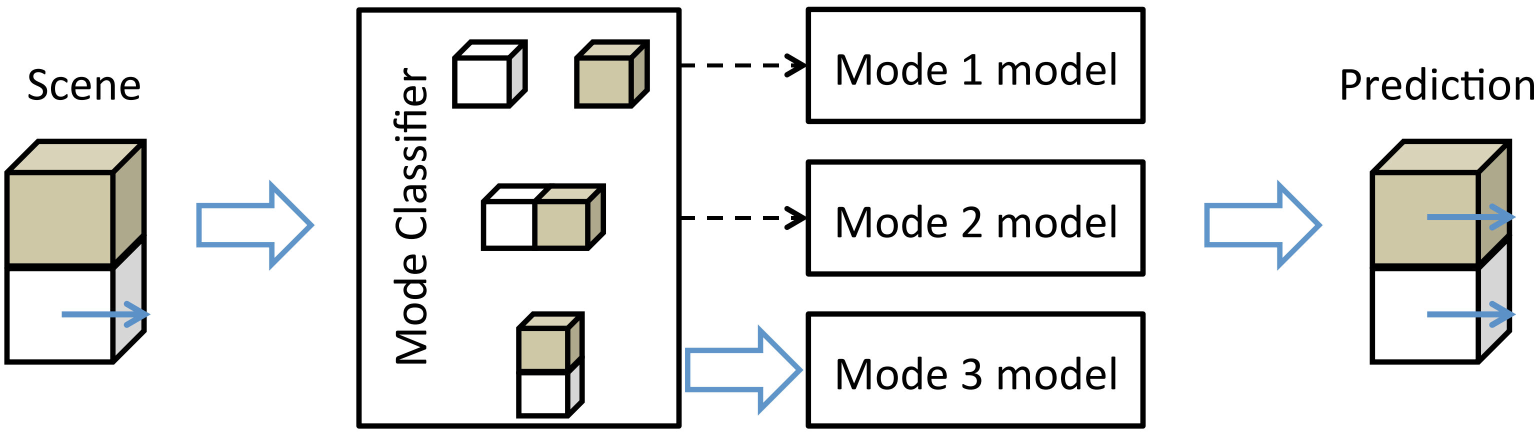
Problems with LWR

- Euclidean distance doesn't capture relational similarity
- Averages over neighbors exhibiting different types of interactions

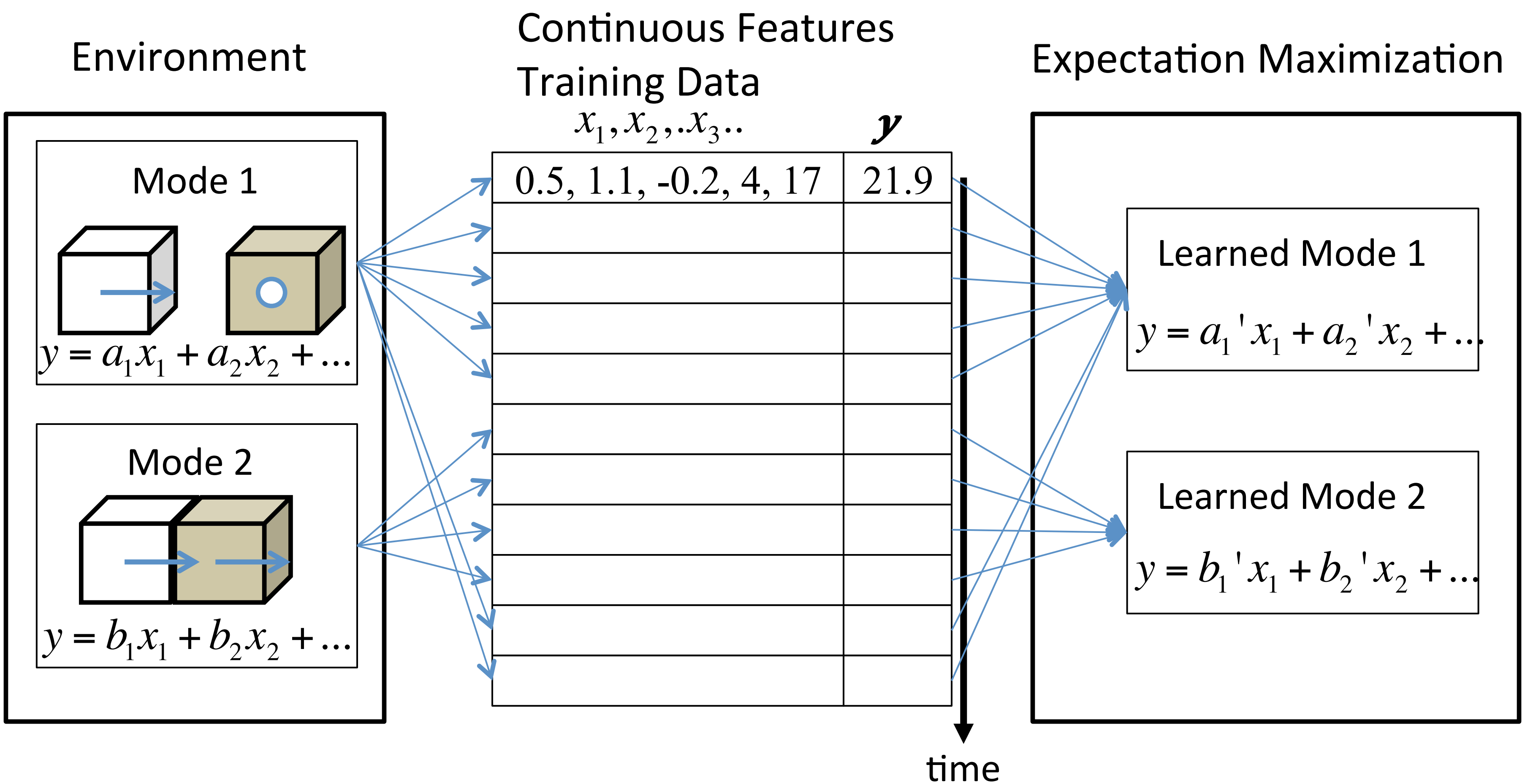


Modal Models

- Object behavior can be categorized into different **Modes**
  - Behavior within a single mode is usually simple and smooth (inertia, gravity, etc...)
  - Behaviors across modes can be discontinuous and complex (collisions, drops)
  - Modes can often be distinguished by discrete spatial relationships between objects
- Learn two-level models composed of:
  - A classifier that determines the active mode using spatial relationships
  - A set of linear functions (initial hypothesis), one for each model



Unsupervised Learning of Modes From Data



Expectation Maximization

- Expectation

  Assuming your current model parameters are correct, what is the likelihood that the model m generated data point i?

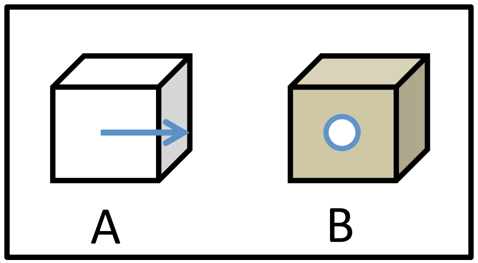
- Maximization

  Assuming each data point was generated by the most probable model, modify each model's parameters to maximize likelihood of generating data

- Iterate until convergence to local maximum

Learning Classifier

Scene




```
left-of(A,B)  = 1
right-of(A,B) = 0
on-top(A,B)   = 0
touch(A,B)    = 0
```

1000101011011

Spatial Relations Training Data

attributes	class
1000101011011	1
0101011010100	1
1100101100000	1
1010111010100	1
0010100010101	1
1110100010100	2
0001010100111	2
1111010101010	2
1010100001001	2
1010101010011	1
0100110010101	1

time

Expectation Maximization

Learned Mode 1

$$y = a_1' x_1 + a_2' x_2 + \ldots$$

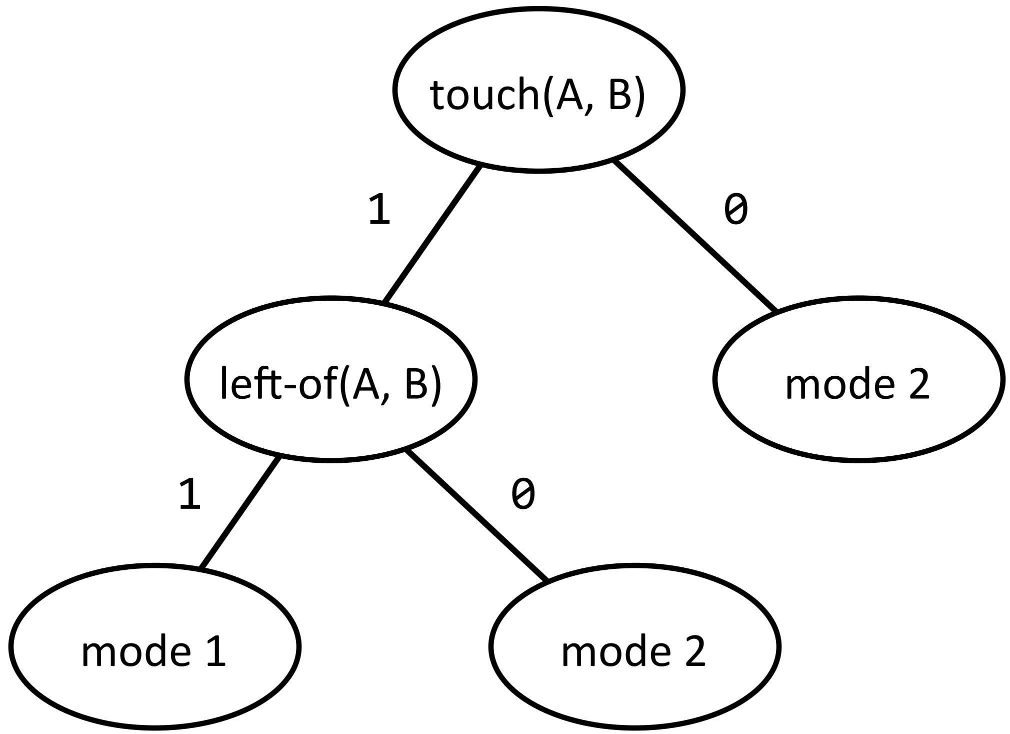
Learned Mode 2

$$y = b_1' x_1 + b_2' x_2 + \ldots$$

Learning Classifier

Classifier Training Data

attributes	class
1000101011011	1
0101011010100	1
1100101100000	1
1010111010100	1
0010100010101	1
1110100010100	2
0001010100111	2
1111010101010	2
1010100001001	2
1010101010011	1
0100110010101	1

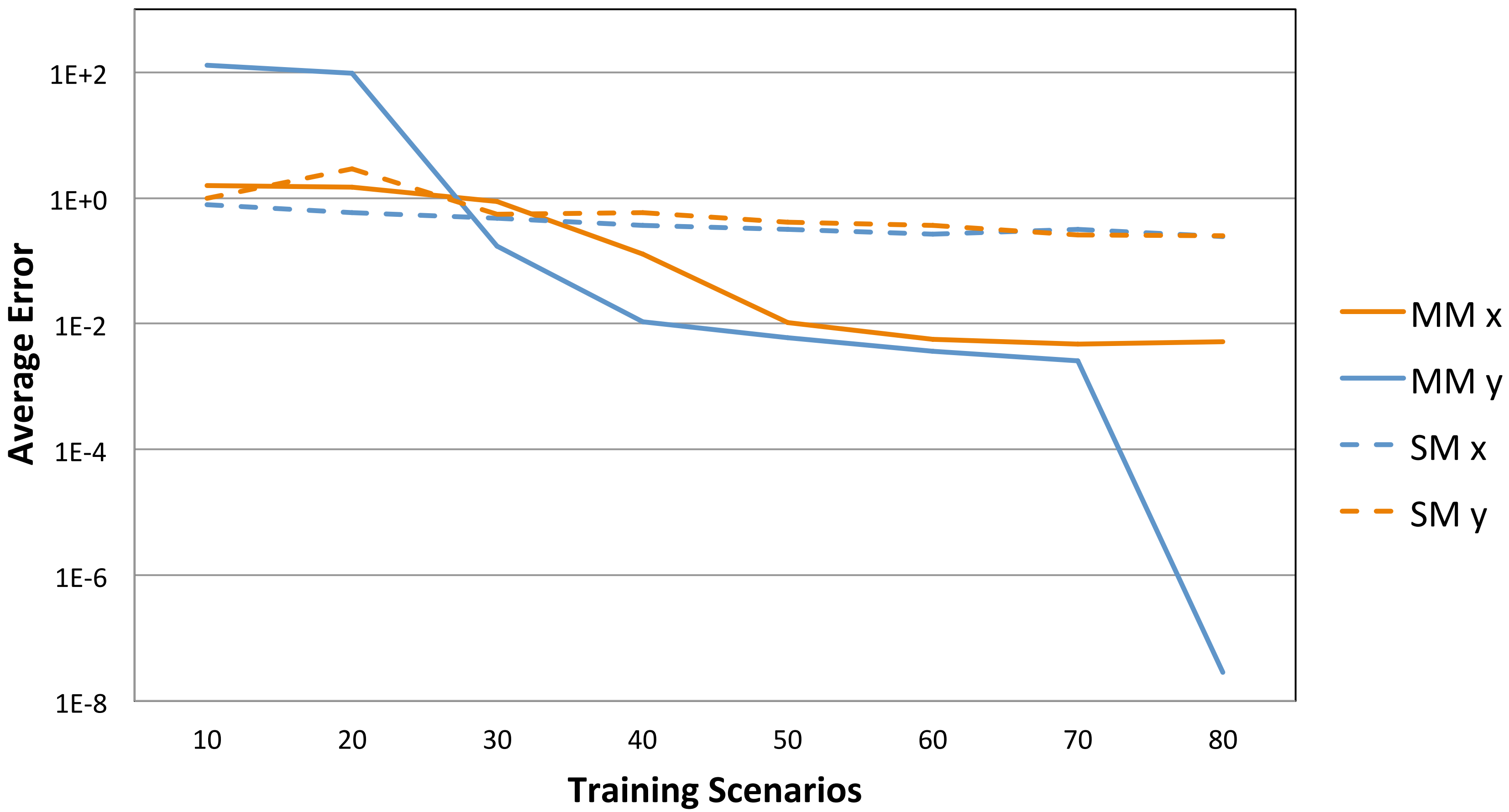


Use linear model for
items in same model

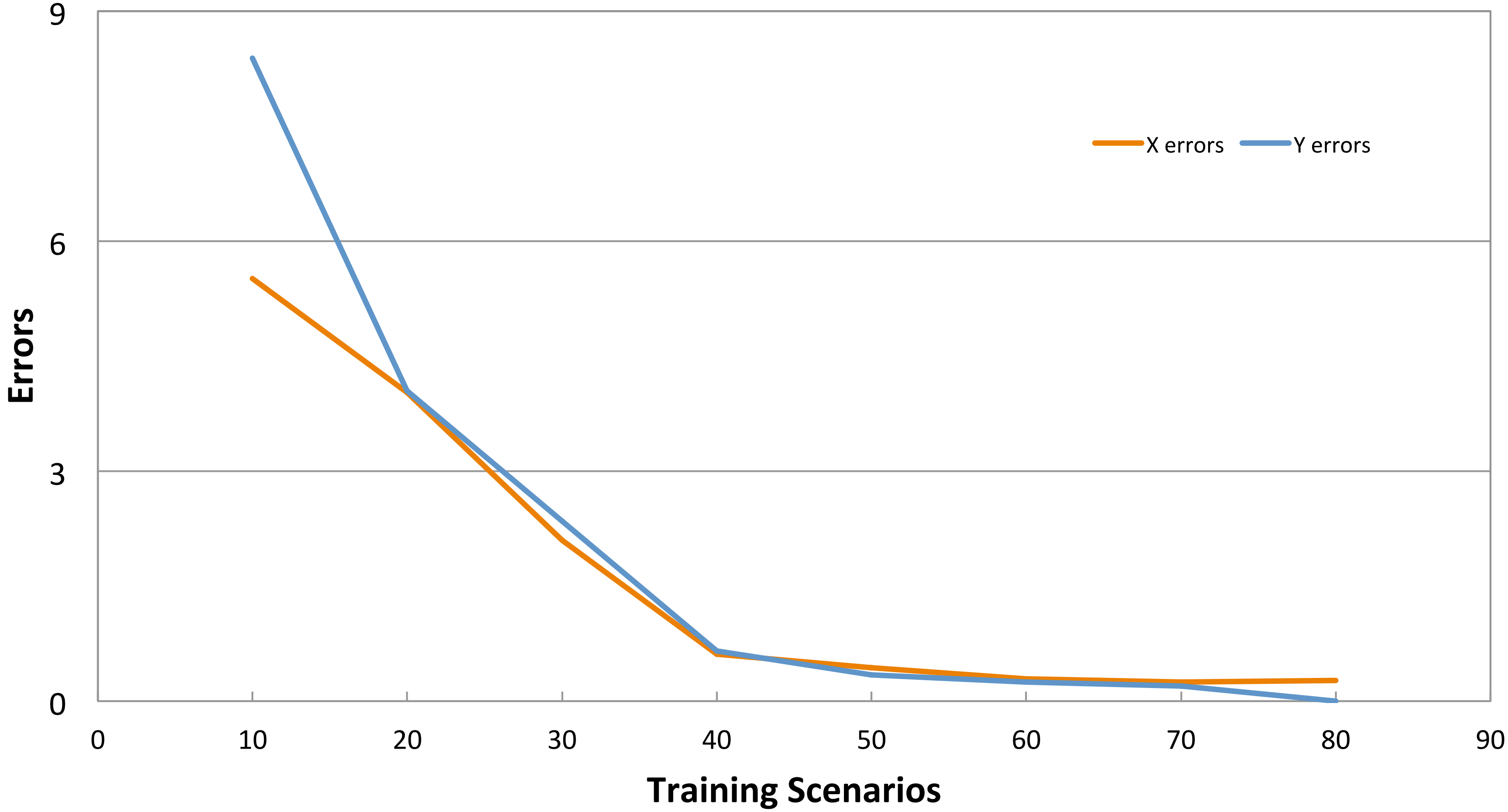
Prediction Accuracy Experiment

- 2 Block Environment
  - Agent has two outputs (dx, dy) which control the x and y offsets of the *controlled block* at every times tep
  - The *pushed block* can't be moved except by pushing it with the controlled block
  - Blocks are always axis-aligned, there's no momentum
- Training
  - Instantiate Soar agent in a variety of spatial configurations
  - Run 10 time steps, each step is a training example
- Testing
  - Instantiate Soar agent in some configuration
  - Check accuracy of prediction for next time step

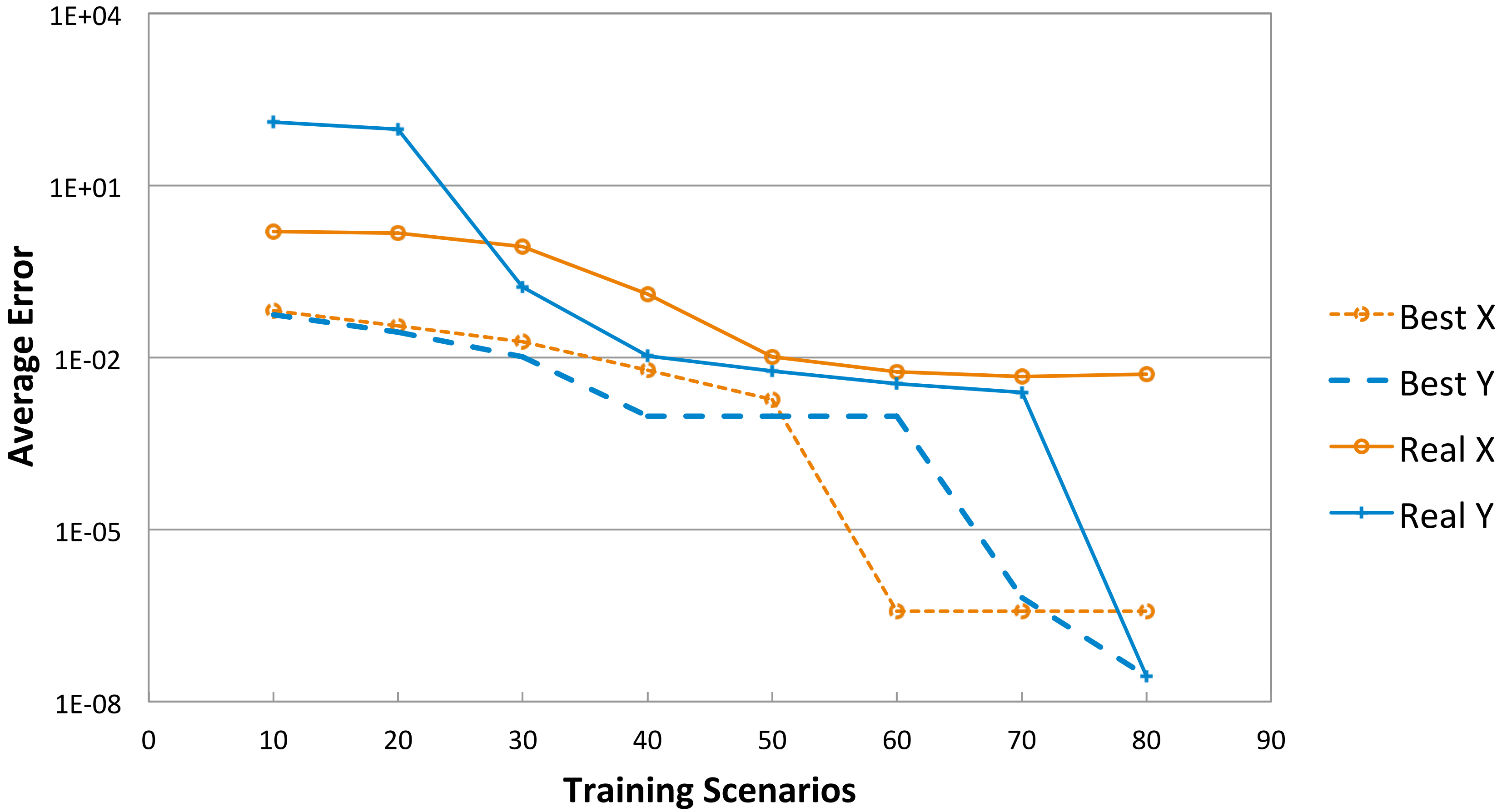
Prediction Accuracy – Pushed Block



Classification Performance



Prediction Performance Without Classification Errors

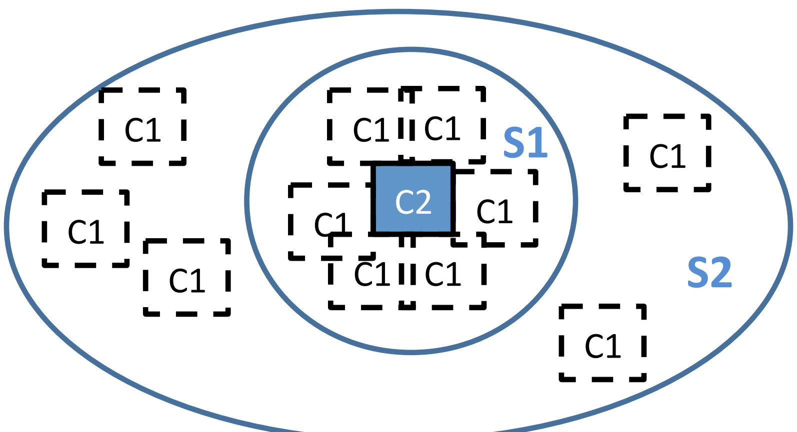


Levels of Problem Solving

Characteristics	Problem Solving Method	Knowledge Required
Faster Task Completion General Solutions	Symbolic Planning	Symbolic Model
	Symbolic Model Free Methods (RL)	Symbolic Abstraction
	Continuous Sampling Methods (RRT)	Continuous Model
Slower Task Completion Specific Solutions	Motor Babbling	Goal Recognition

Symbolic Abstraction

- Lump continuous states sharing symbolic properties into a single symbolic state
- Should be Predictable
  - Planning requires accurate model (ex. STRIPS operators)
  - Tends to require more states, more symbolic properties
- Should be General
  - Fast planning and transferrable solutions
  - Tends to require fewer states, fewer symbolic properties



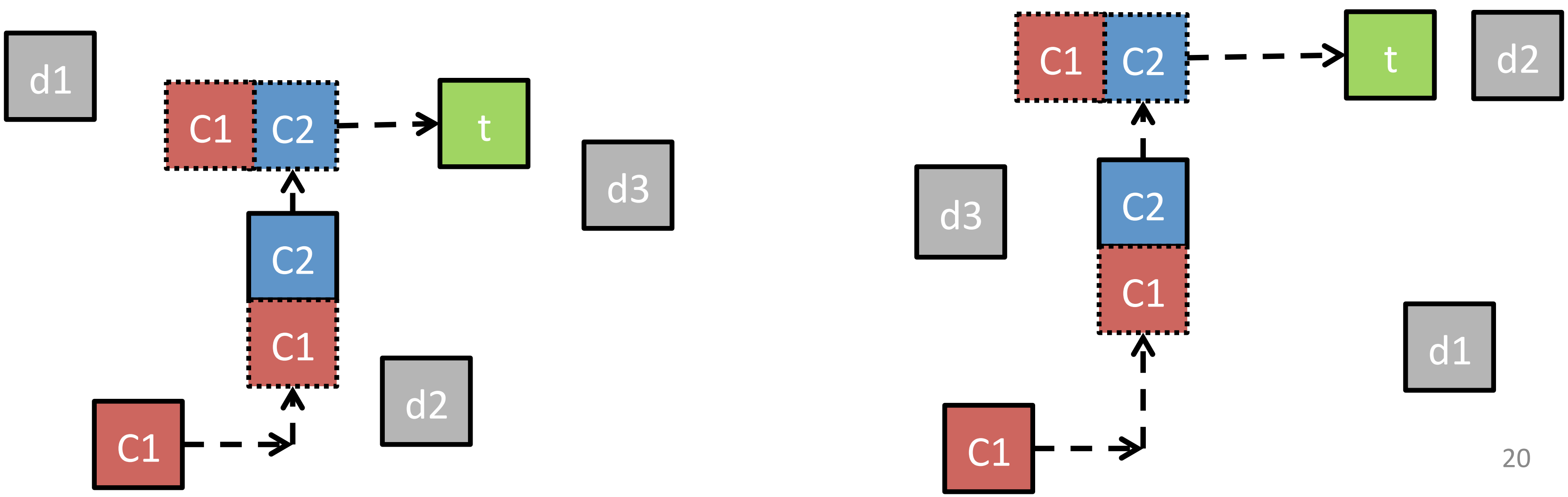
S1: intersect(C1, C2)
S2: ~intersect(C1, C2)

Symbolic Abstraction

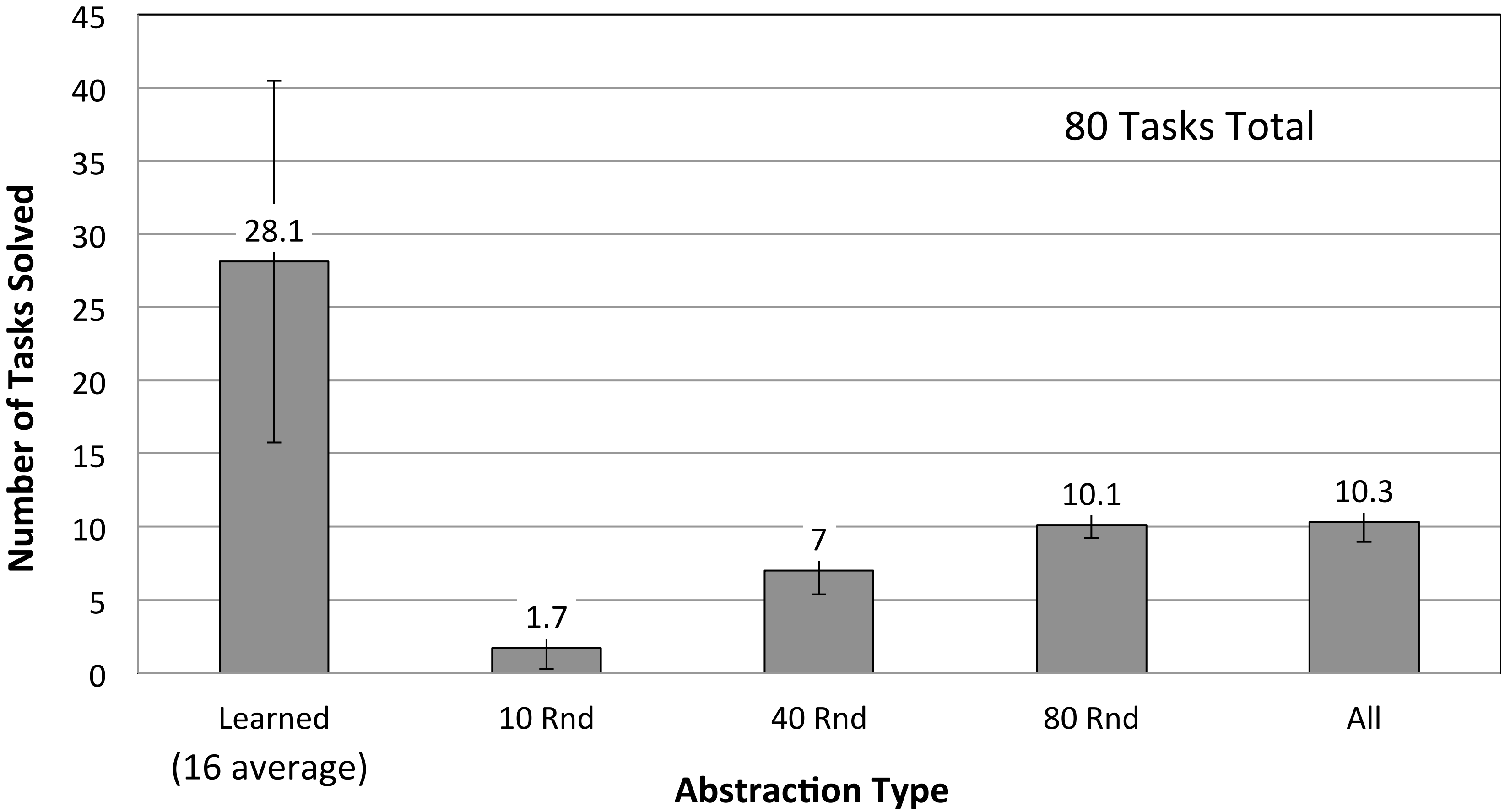
- Hypothesis: contiguous regions of continuous space that share a single behavioral mode is a good abstract state
  - Planning within modes is simple because of linear behavior
  - Combinatorial search occurs at symbolic level
- Spatial predicates used in continuous model decision tree are a reasonable approximation

Abstraction Experiment

- 3 blocks, goal is to push c2 to t
- Demonstrate a solution trace to agent
- Agent stores sequence of abstract states in solution in epmem
- Agent tries to follow plan in analogous task
- Abstraction should include predicates about c1, c2, t, avoid predicates about d1, d2, d3



Generalization Performance



Conclusions

- For continuous environments with interacting objects, modal models are more general and accurate than uniform model
- The relationships that distinguish between modes serve as useful symbolic abstraction over continuous state
- All this work takes Soar toward being able to autonomously learn and improve behavior in continuous environments

Evaluation

Coal

- Scaling issues: linear regression is exponential in number of objects
- Linear modes is insufficient for more complex physics such as bouncing -> catastrophic failure

Nuggets

- Modal model learning is more accurate and general than uniform models
- Abstraction learning results are promising, but preliminary