Memory-based exploratory online learning of simple object manipulation

Master thesis

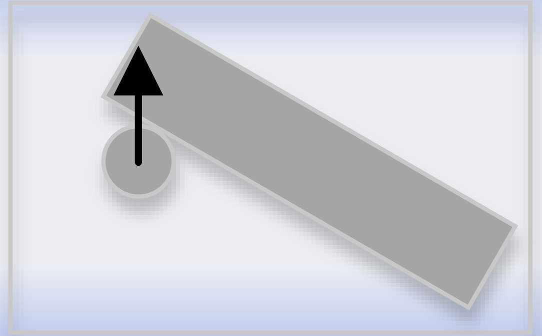
Jan Pöppel

05.10.2015

Outline

1. Task
2. Current Results
3. Methods
4. Issues
5. Technologies
6. Evaluation

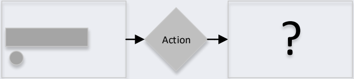
Scenario



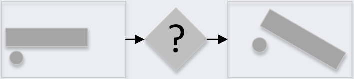
Task

Given an environment and a set of action primitives to control some actuator, incrementally and interactively learn a

Forward Model



Inverse Model



Current Results

Video(s)

Methods

Created two models

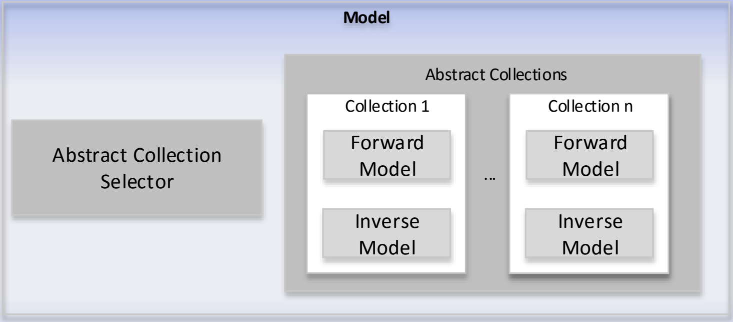
Interactions

- Represent interaction states
- Split space into subspaces along different interactions

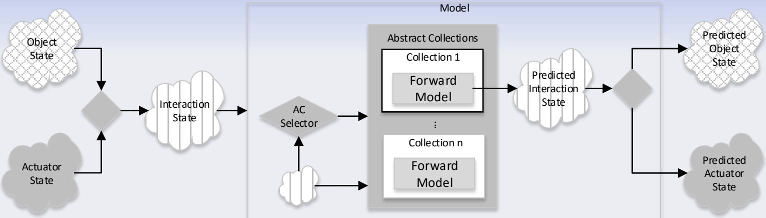
Objects with gate

- Represent individual object states
- Introduce gate, distinguishing interaction from no interaction

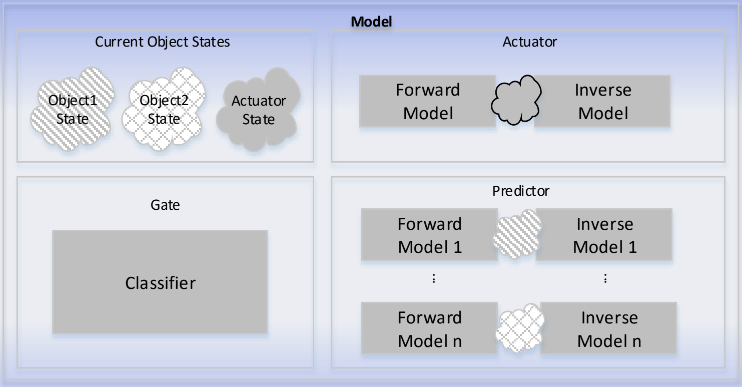
Method - Interactions



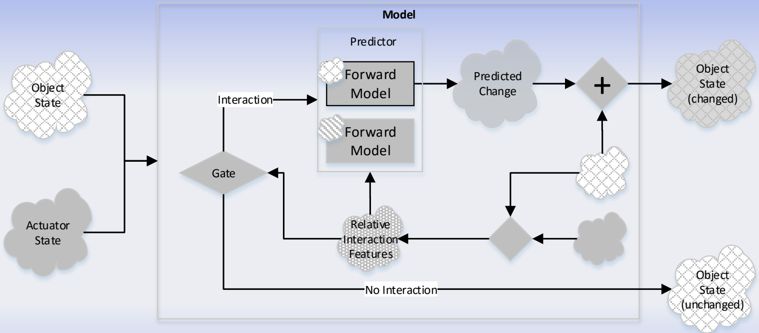
Method - Interactions Prediction



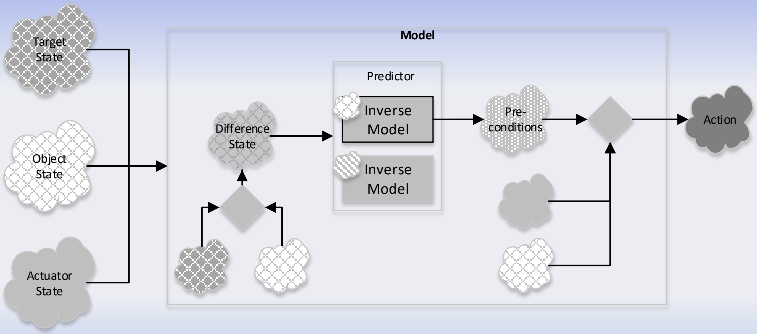
Method - Gate


Model
Current Object States
Object1 State
Object2 State
Actuator State
Actuator
Forward Model
Inverse Model
Gate
Classifier
Predictor
Forward Model 1
Inverse Model 1
⋮
⋮
Forward Model n
Inverse Model n

Method - Gate Prediction



Method - Gate Planning



Issues

- Incremental update
- Time constraints for updates and queries due to environment loop
- Metric problem
  - Features may have different orders of magnitudes (e.g. orientation/position)
  - Features may have different importances

Technologies

- Regression and classification models are memory/instance based
- Adapted the instantaneous topological map (ITM) as main model
- Developed abstract inverse model to avoid extrapolation and metric problems

Technology - ITM

Technology - Inverse Model

Inhalt...

Evaluation

Two main tasks in order to test forward and inverse model

PushTaskSimulation (Video 1)

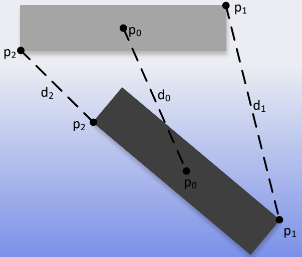
- Train at different starting positions and push the object
- Use previous prediction as input when testing
  → No environment feedback

MoveToTarget (Video 2)

- Show all feature changes during training
- Give fixed target position for one object
- Operate in open loop → Constant environment feedback

Evaluation - PushTaskSimulation

- A run is defined as the number of frames until the actuator traveled a certain distance
- Testruns start at varying starting points
- Calculate difference to actual object at end of each run
- Difference defined over keypoints in order to combine position and orientation
- Test against different number of training runs



Evaluation - MoveToTarget

Multiple possibilities

- Measure number of frames/actions required to reach target
- Measure distance after fixed number of allowed actions
- Measure average distance reduction per action
- Try reduced number of training examples/interactions
  - e.g. only show position change and no orientation

Further Evaluations and further features

- Evaluate subparts by swapping them out where possible
    - E.g. gate classifier/actuator forward model
- Distinguish testpositions that were not included in trainset and those that are
- Multiple objects
- Self-exploration

Thank you for your attention!