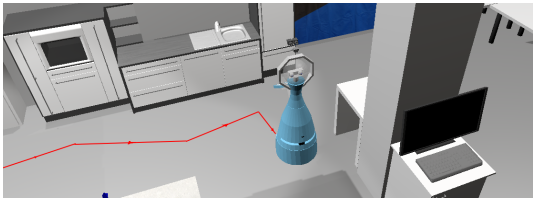
Learning and Optimizing Probabilistic Models for Planning under Uncertainty in Mobile Robot Navigation



Rob van Bekkum

Delft University of Technology

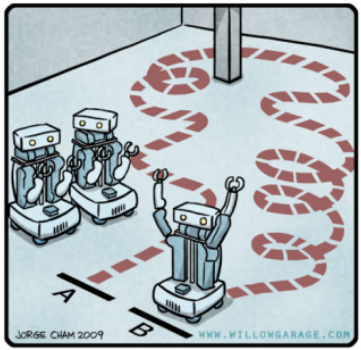
April 5, 2017

Decision-Theoretic Planning

DTP Problem Scope

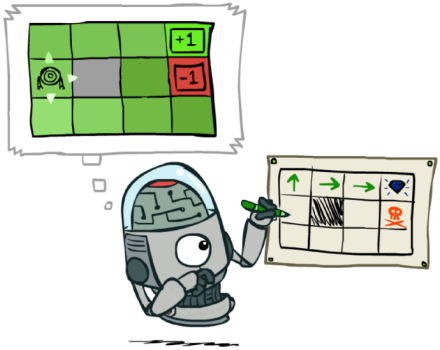
Systems:

- Dynamics are stochastic, i.e., involve uncertainty
  - Execution of actions (e.g., robot may slip)
  - Exogenous events (e.g., doors open/closed)
  - Observations (e.g., sensor noise)
- Controlled by one or more agents
- Sequential decisions of actions to execute

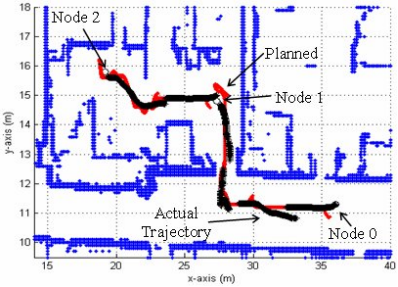


Decision-Theoretic Planning

Example Domain: Path Planning



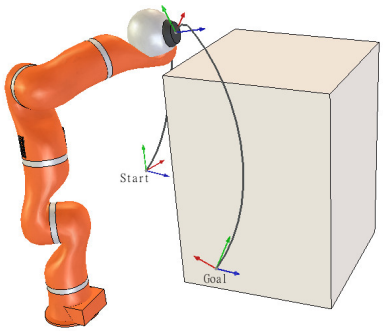
Source: Berkeley CS188 sheets



Source: `http://www.cs.cmu.edu/~gunhee/r_psr.html`

Decision-Theoretic Planning

Example Domain: Motion Planning



Source: http://www.coppeliarobotics.com/helpFiles/en/motionPlanningModule.htm

Decision-Theoretic Planning

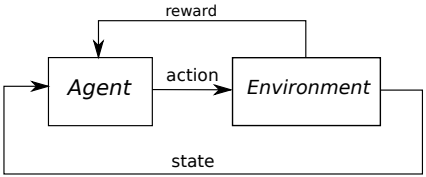
Markov Decision Processes (MDPs)

In DTP systems are modeled by probabilistic models, e.g. MDPs:

Definition

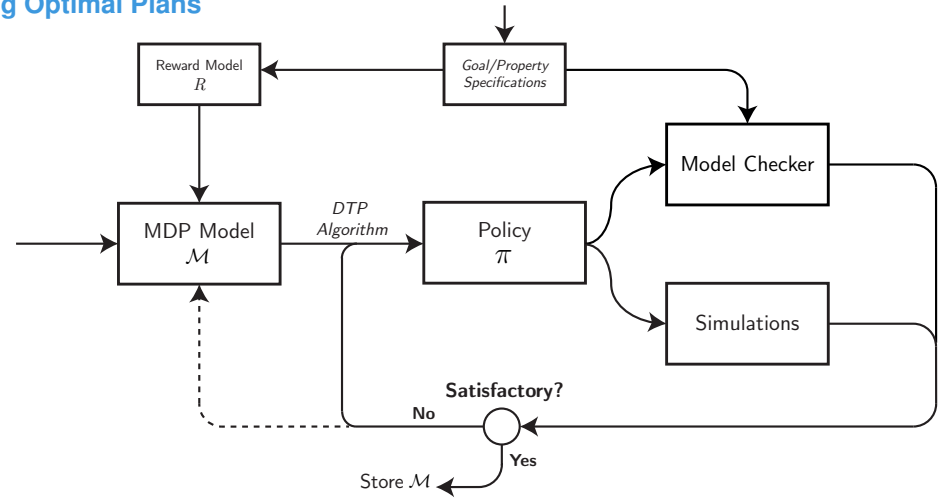
An MDP is a 5-tuple $\mathcal{M} = (\mathcal{S}, s_0, A, \delta, R)$:

- $\mathcal{S}$ is the state-space, $s_0 \in \mathcal{S}$ the initial state
- A is the action-space
- δ is the transition function (N.B.: accounts for uncertainty)
- R is the reward function (N.B.: defines the goals)



Decision-Theoretic Planning

Learning Optimal Plans



Decision-Theoretic Planning

Model Development

Problem: How to find a suitable MDP model?

Classical approach: Model development by a *human designer*, however:

- Requires significant effort (e.g., trial-and-error)
- Typically demands knowledge/experience, accompanied by high costs

Alternative: Use Reinforcement Learning instead of Planning, however:

- Requires direct interaction with environment
- One might require *reusable* models, applicable for multiple tasks

Decision-Theoretic Planning

Model Development

Problem: How to find a suitable MDP model?

Idea: Automate the model building process by learning algorithms

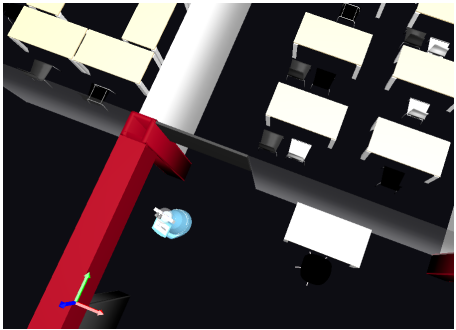
- Learn from (exploration) data about the environment
- Optimization for the best MDP model

Mobile Robot Navigation

Goal: Move robot to area while minimizing traveled distance

MDP Model $\mathcal{M}$ should define:

- States $\mapsto$ locations
- Actions $\mapsto$ robot translations
- Rewards $\mapsto$ goal areas
- Transition-function



Model-Learning for Mobile Robot Navigation

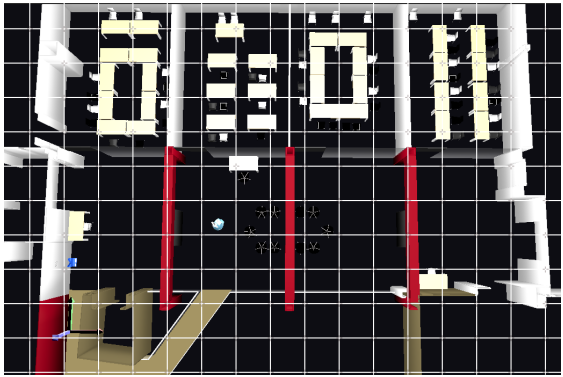
Learning State-Spaces

Given: Execution trace(s) from exploration-stage

Existing Approaches:

- Grid-discretization
- Best-First Model Merging
- State Merging by Trajectory Clustering

Issues: Dimensionality, grid resolution



Model-Learning for Mobile Robot Navigation

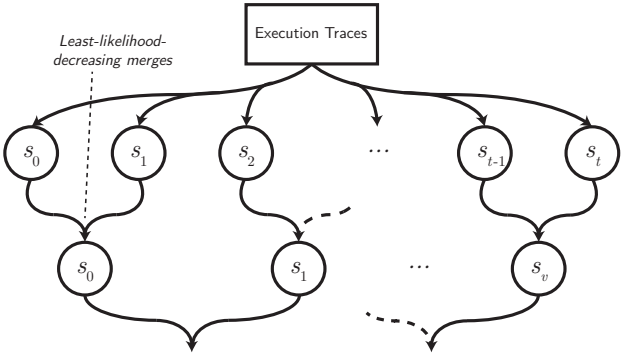
Learning State-Spaces

Given: Execution trace(s) from exploration-stage

Existing Approaches:

- Grid-discretization
- Best-First Model Merging
- State Merging by Trajectory Clustering

Issues: Greedy merging, $O(t^3)$ algorithm



Model-Learning for Mobile Robot Navigation

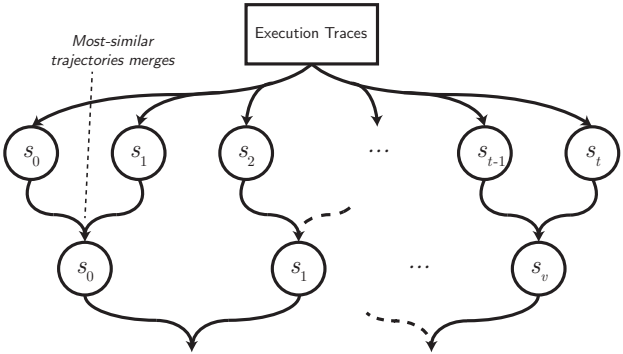
Learning State-Spaces

Given: Execution trace(s) from exploration-stage

Existing Approaches:

- Grid-discretization
- Best-First Model Merging
- State Merging by Trajectory Clustering

Note: For learning POMDPs facing *perceptual aliasing*



Model-Learning for Mobile Robot Navigation

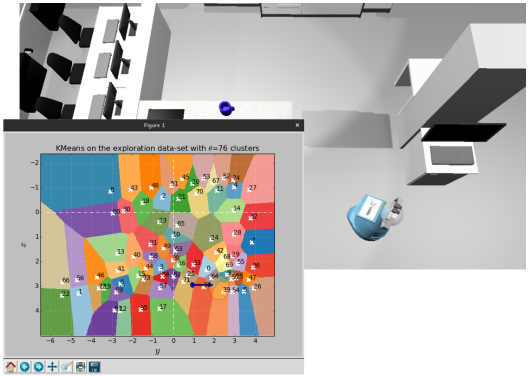
Learning State-Spaces

Given: Execution trace(s) from exploration-stage

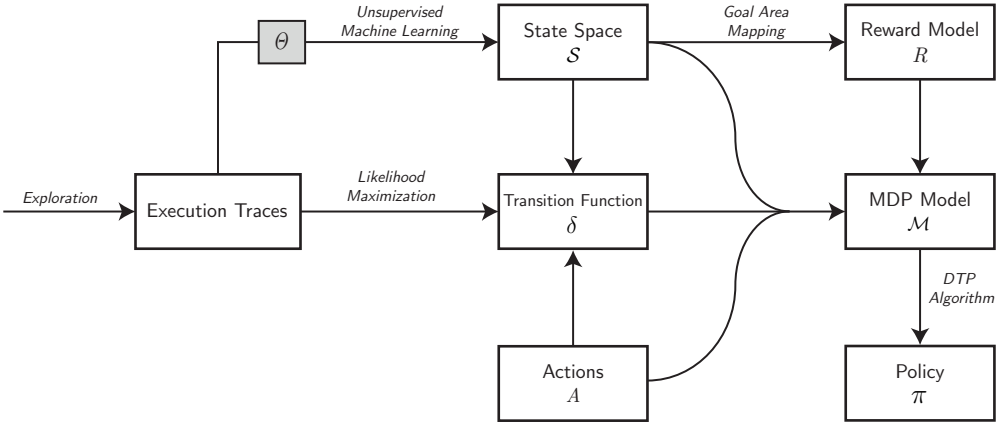
Approach: Unsupervised ML for state-spaces, i.e.:

- K-Means Clustering
- Gaussian Mixture Models
- . . .

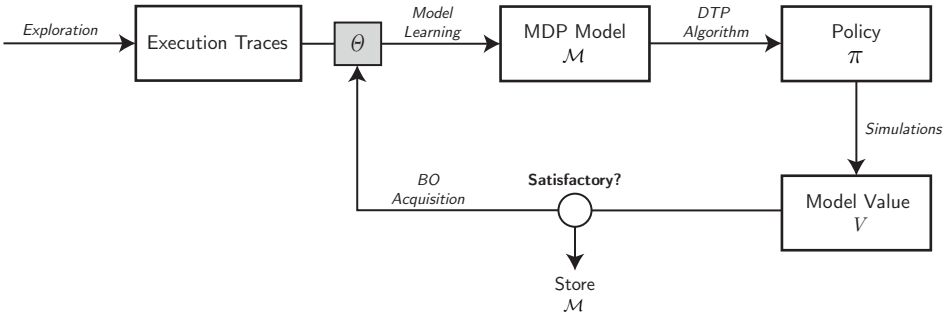
Variable: θ parameter of ML-algorithm



Model-Optimization Routine



Model-Optimization Routine



Bayesian Optimization

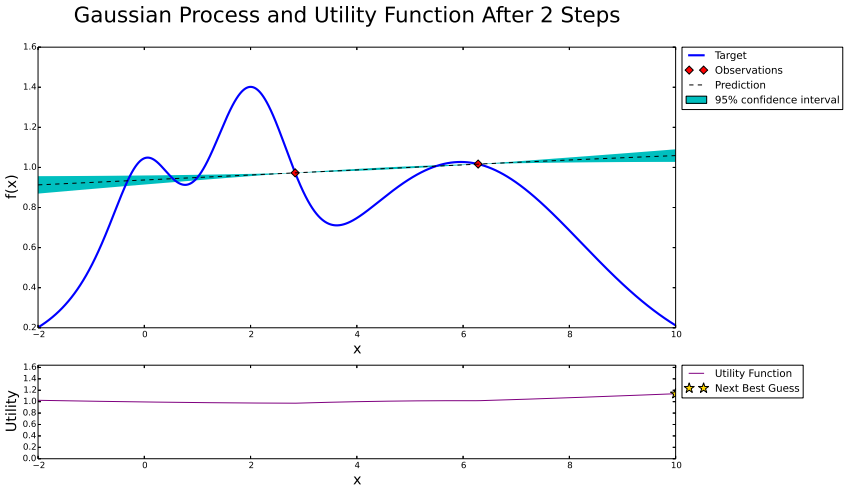
- Given an expensive, unknown objective function $f : \mathcal{X} \mapsto \mathbb{R}$ ($\mathcal{X} \subset \mathbb{R}^m$)
- Find global maximizer $x^* \in \mathcal{X}$ s.t.:

$$x^* = \arg\max_{x \in \mathcal{X}} f(x)$$

- Bayesian Optimization components:
  - *Prior over* f: Gaussian Process $f \sim GP(\mu(\cdot), K(\cdot, \cdot))$
  - *Evidence set*: $\mathcal{D}_{1:t} = \{(x_i, y_i) \mid i = 1 \dots t\}$
  - *Posterior* or *Surrogate function* ('estimate of f')
  - *Acquisition* or *Utility function*: $u : \mathcal{X} \mapsto \mathbb{R}$

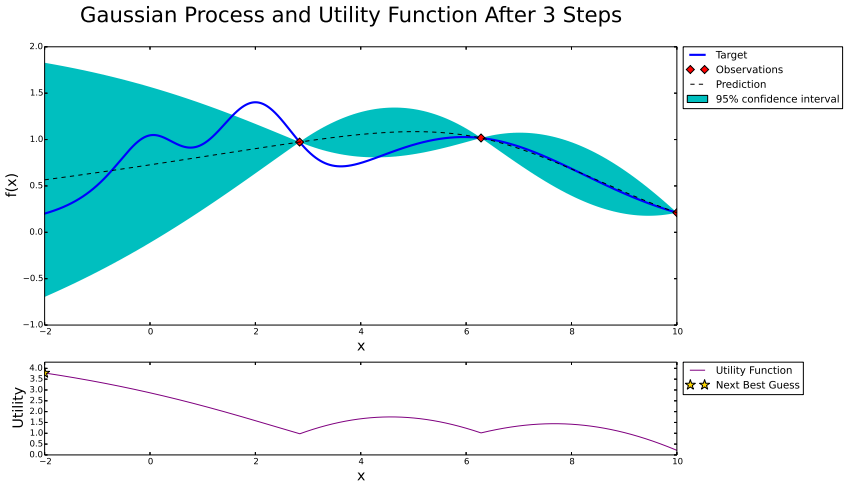
Bayesian Optimization

Example



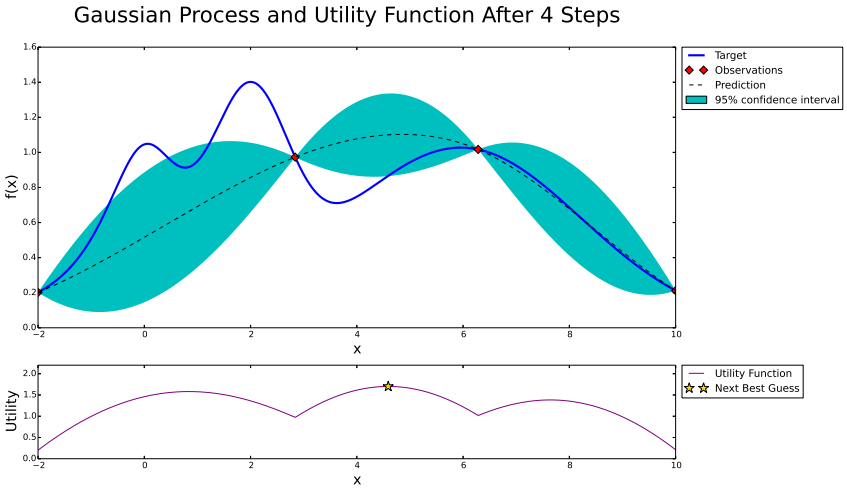
Bayesian Optimization

Example



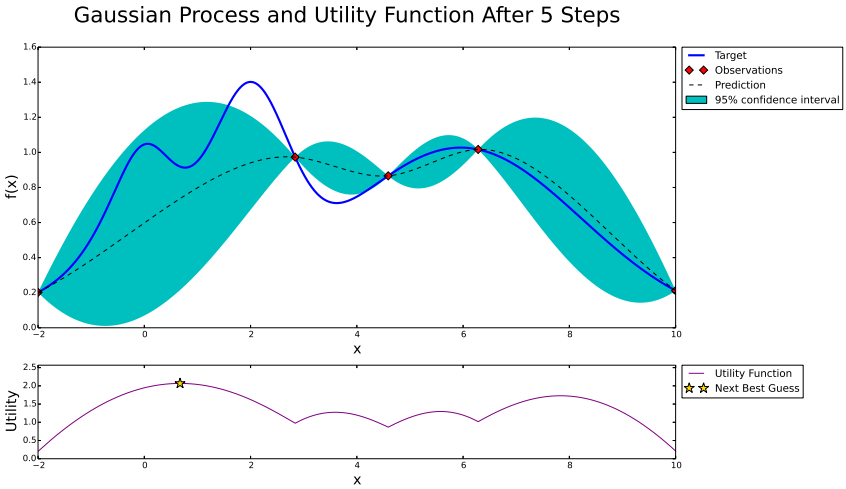
Bayesian Optimization

Example



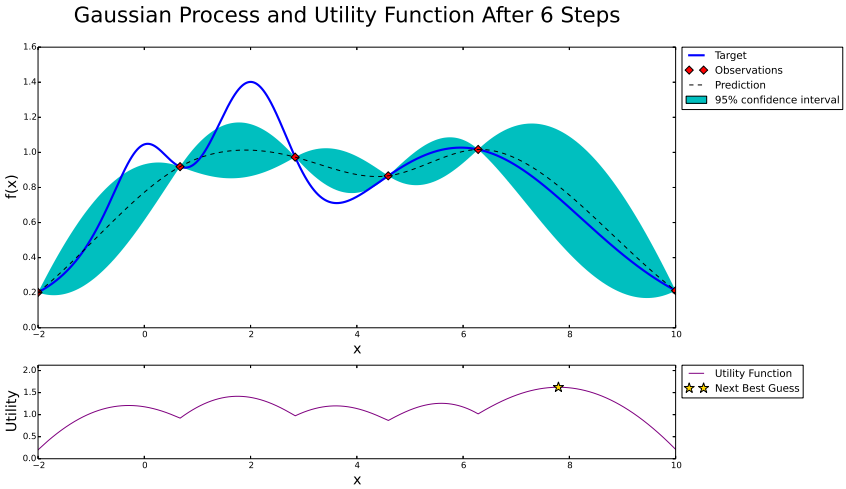
Bayesian Optimization

Example



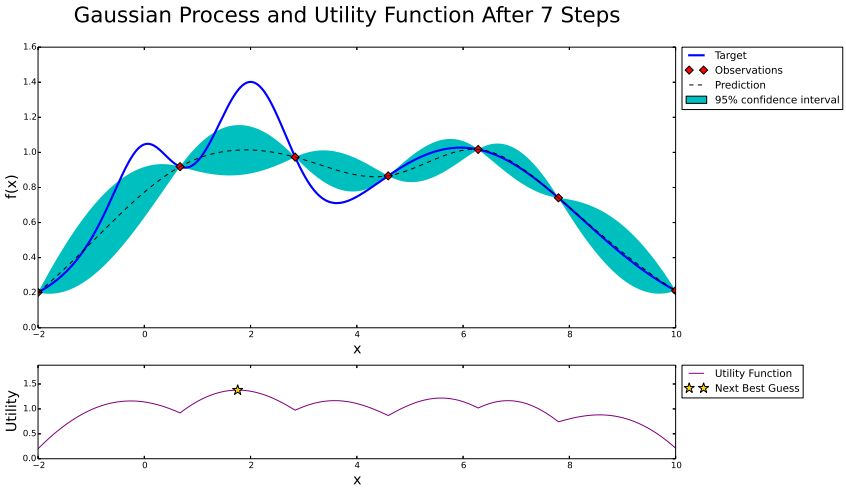
Bayesian Optimization

Example



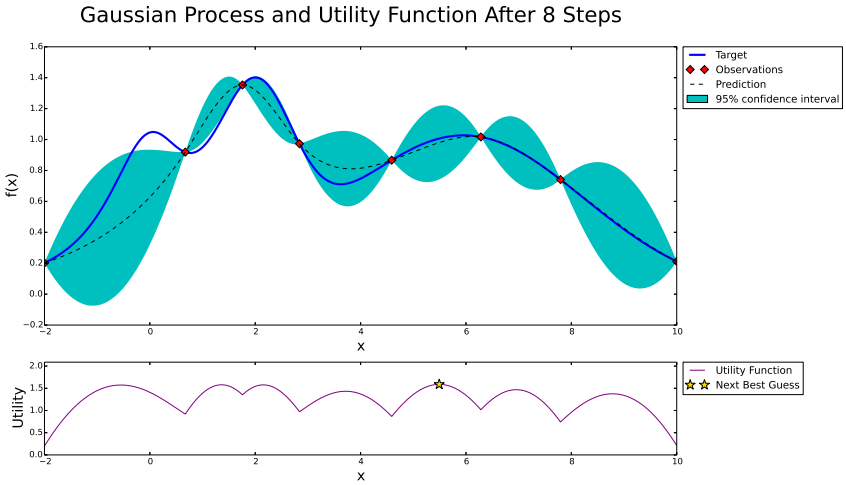
Bayesian Optimization

Example



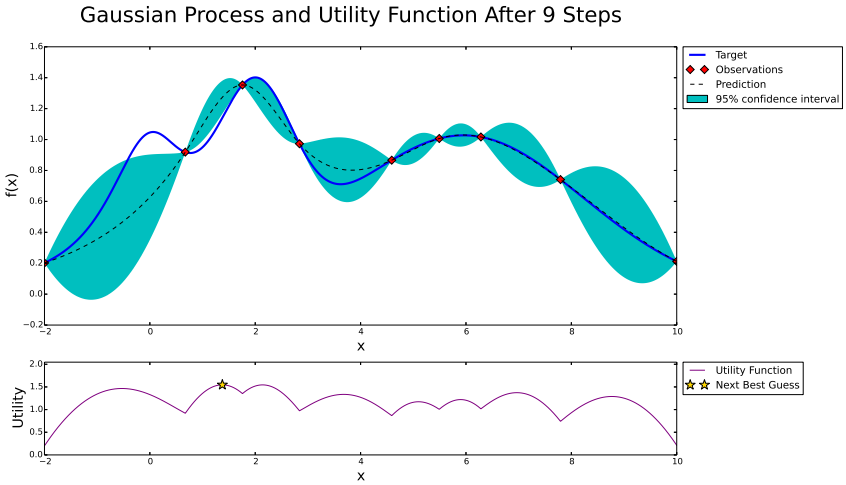
Bayesian Optimization

Example



Bayesian Optimization

Example



Implementation of Model-Optimization Routine

- **Exploration:** 'Randomly' navigating, writing execution trace to files
- **Optimization:** Objective function defined by performance in simulations:
  - Goal area(s) reached (before time-out)?
  - Time passed
  - Part of goal area covered by corresponding final states

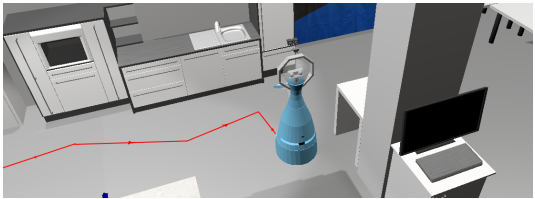
- **Software:** Python, ROS, Morse simulator
- **Robot:** Scitos A5, mobile service robot

DEMO

Next Steps

- Cost-Sensitive Optimization
  - BO acquisition based on *Expected Improvement per second*
- Identification of incomplete exploration data
  - Feedback from simulation trajectories
  - Use feedback in BO to reduce noise/uncertainty in observations
- Variable Resolution in State-Space Acquisition
- Inspect influence of other variables in the optimization loop
  - Discount factor γ in MDP solver
  - ML-algorithm (K-Means, GMM, ...)
  - Gaussian-Process hyper-parameters?

Learning and Optimizing Probabilistic Models for Planning under Uncertainty in Mobile Robot Navigation



Rob van Bekkum

Delft University of Technology

April 5, 2017

References

D. Nikovski and I. R. Nourbakhsh.
Learning probabilistic models for decision-theoretic navigation of mobile robots.
In *ICML*, pages 671–678. Citeseer, 2000.

A. Stolcke and S. M. Omohundro.
Best-first model merging for hidden markov model induction.
arXiv preprint cmp-lg/9405017, 1994.