Learning in Robotic Systems

Harsha Cuttari
Manny Crespi
Nikhil Badami

Introduction

- Learning in Robotic Systems
- Behavioral-based Control Architectures
- Hybrid Coordination of Behaviors
- Reinforcement Learning
    - Semi-Online Neural-Q Learning
- Conclusions

What is Learning in Robotic Systems?

Autonomous robots: a research subject that combines topics such as sensing, actuation, powering, communication, control theory and artificial intelligence

Autonomous robots must be able to integrate two concepts: sensing and acting

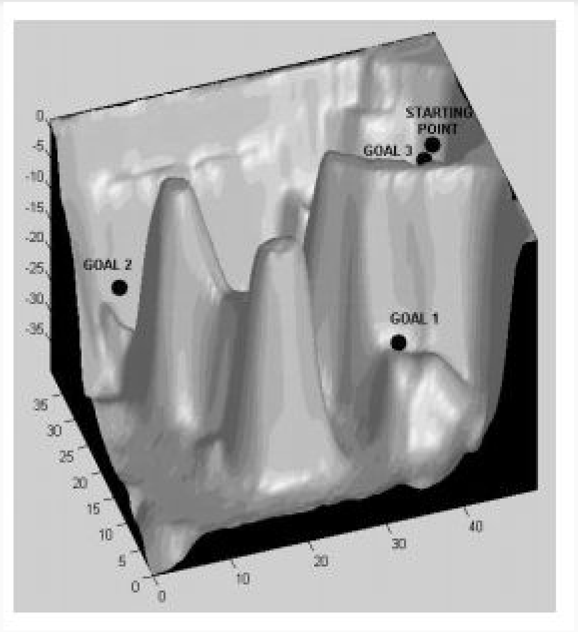
Main Goals of an Autonomous Robot

- Ability of a machine to fulfill a task similar to a human in the real world
- Be able to act in the real world by changing states or moving itself through it
- Be able to sense the state of the world by interpreting sensor information
- Be able to decide what to do, how to relate the state of the environment to its action possibilities in order to achieve a predefined goal

Application: Personal Assistants

- The robot is first taught the skill
- First few trials are a failure
- With the use of motion capture, gyroscopes and accelerometers, as well as reinforced learning, it becomes better to a point of almost no failure.

Application: Autonomous Underwater Vehicle (AUV)



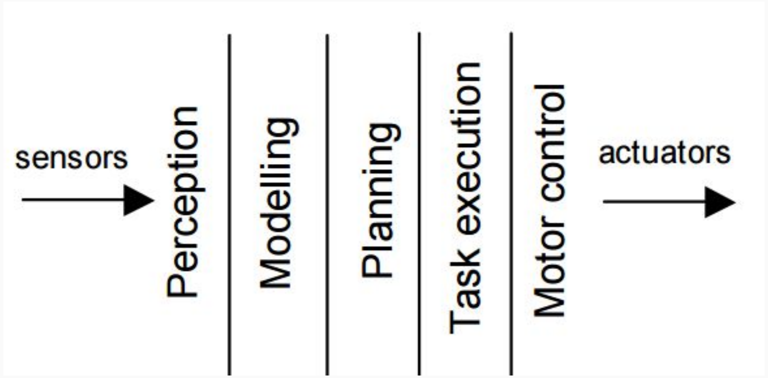
- Simulated underwater environment where the AUV must reach three goal points
- Obstacle avoidance behavior
- Avoid trapping behavior
- Relevance to current project

Behavioral-Based Control Architectures

Behavioral-Based Control Architectures

- Quick definition: The verification of sensory information to generate actions in the real world
- Full definition: A set of simple parallel behaviors that react to the environment to compose an appropriate response the robot must take in order to accomplish a goal
- Design of these control architectures is based on top-down philosophy
- Design is broken down into orderly sequence of functional components
- User formulates explicit tasks and goals for the system

Phases of a deliberative control architecture



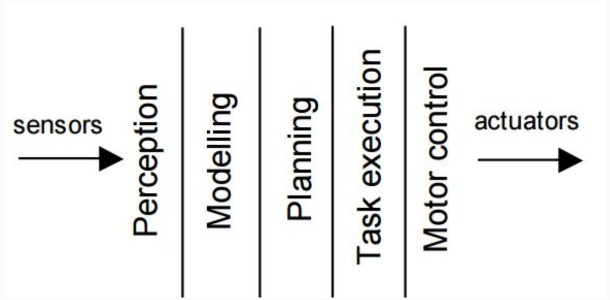
Phases of a deliberative control architecture

1. Perception - Sensor interpreter resolves noise and conflicts in the sensor data while the perception algorithms are used to find characteristics and objects within the environment
2. Modelling - Data is used to build symbolic representation of the world. Geometric details of all objects in the robot’s world with their positions and orientations
3. Planning - Operate on symbolic descriptions of the world and produce a sequence of actions to achieve the goal given

Phases of a deliberative control architecture

4. Task execution - controls the execution of the planned tasks generating the set-points for each actuator

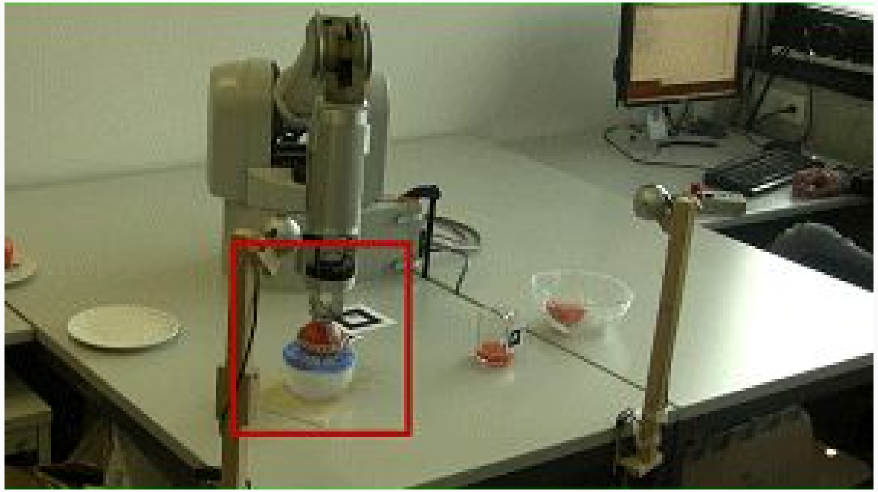
5. Motor Control - control system used to control the actuators in accordance with the set-points



Application: Personal Assistant

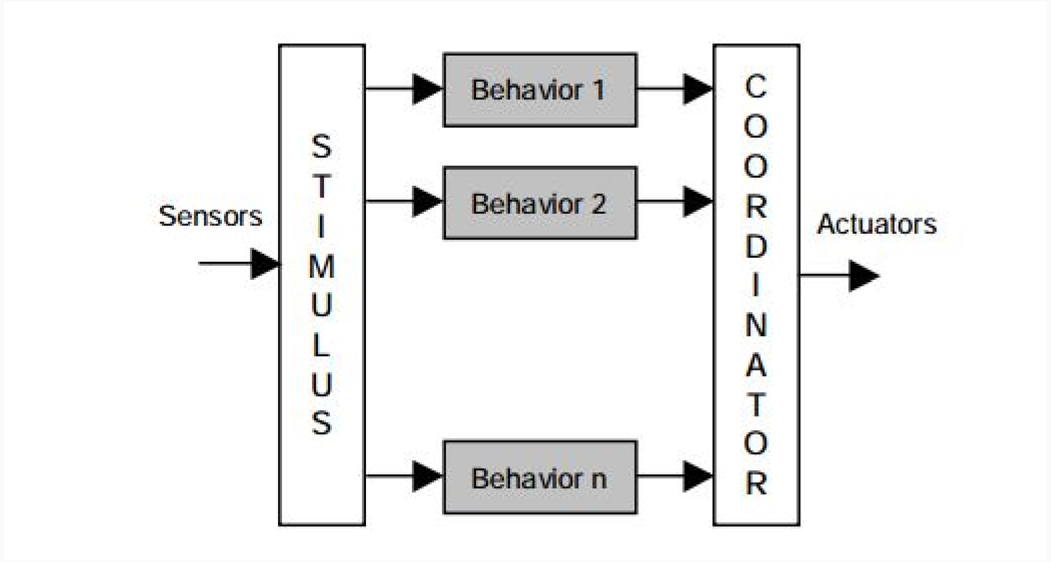
The teacher performs several demonstrations of the same task, changing the location of each item in between to allow the robot to generalize correctly. From observing these changes the robot can infer that the relative positions of the objects matter, but that their absolute positions do not.

Application: Personal Assistant



After learning, the robot successfully reproduces the task even when all objects are in novel positions.

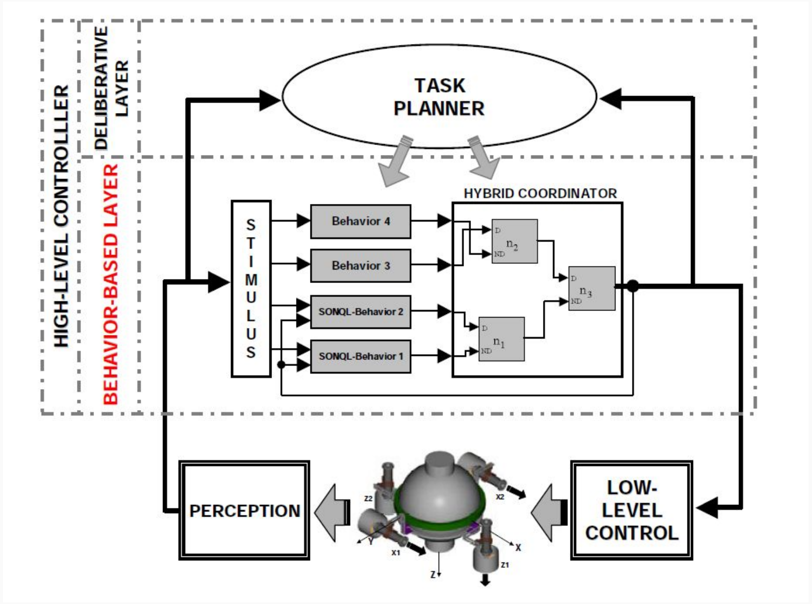
Structure of Behavior-Based Control Architecture



Hybrid Coordination of Behavior

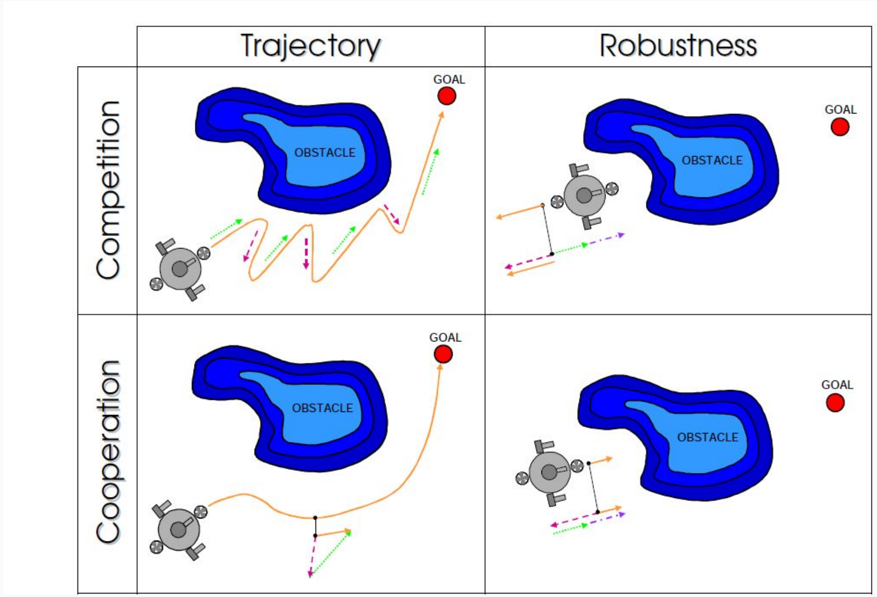
Behavior Based Control-Layer

- High and Low Level Controller for Autonomous Robot
- Deliberative layer: Breaks down overall goal into smaller tasks, picks best behaviour for task
-
- Behavior Based Layer: Compound set of behaviors and coordinator
- Hybrid Control focuses on Behavior-Based Layer
- Competitive vs. Cooperative Control



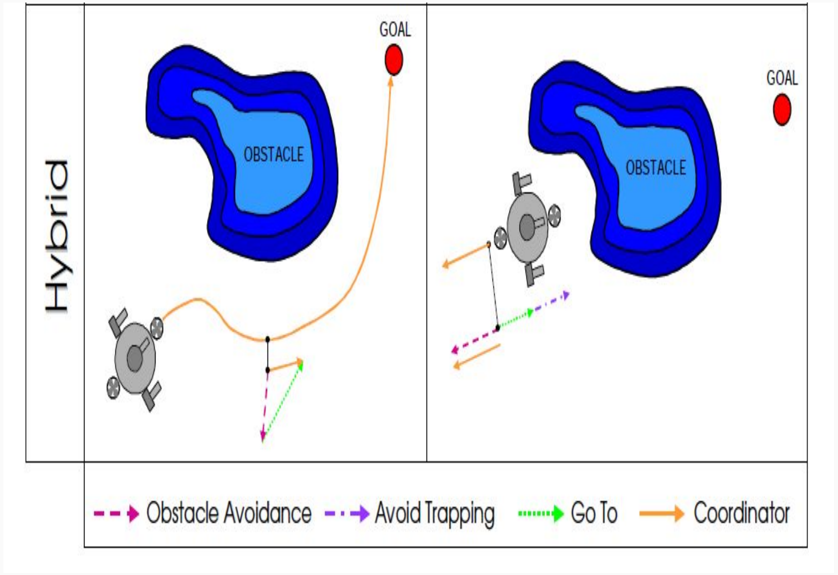
Competitive vs. Cooperative Control

- Competitive Behavior:
  - Only highest priority behavior controls robot per time step
  - Very robust behavior
  - Non-optimal trajectory
- Cooperative Behavior
  - Can select optimal trajectory
  - Susceptible to wrong control action



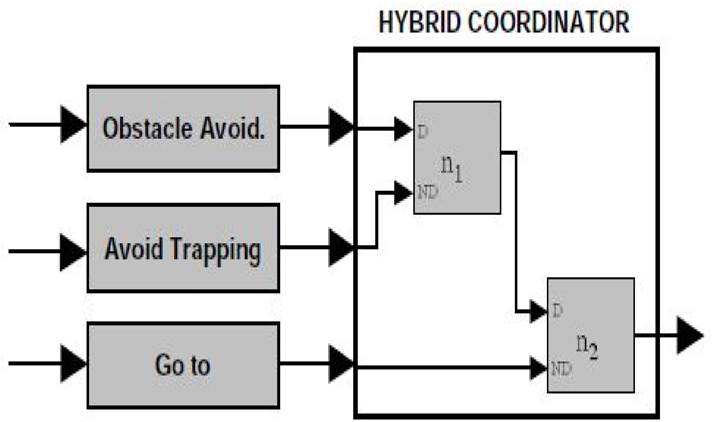
Hybrid Coordination of Behaviors

- Best of both Competitive and Cooperative control
- Can select optimal trajectory as well as avoid critical situations
- Uses a hierarchy of control nodes to determine behavior



Hybrid Coordination of Behaviors cont.

- Activation levels from 0 -> 1: ai
- Hybrid Coordinator uses nodes to decide behavior
- Outputs also have activation levels
- Dominant and Non-Dominant input
- If ad is 1, output is dominant behavoir
- If a(nd) is 1 and ad is 0, output is non-dominant behavior
- Hybrid coordinators consists of many of these blocks



Reinforcement Learning (RL)

Game Playing: Best move at a given time

Control Problems: Elevator scheduling

Robotics:

Uses a method by which a robot is able to discover an optimal behavior through trial-and-error interactions in the environment.

Relevance of RL in Robotics

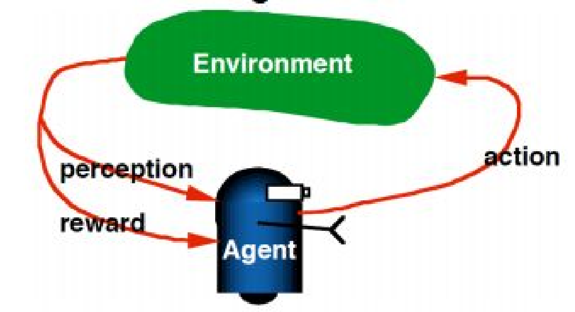
Generalization problem - need the robot to be good at “generalizing” new data

Programming/calibrating actions for a controlled environment gives the robot deterministic movement

But the real world is not a controlled environment

So we consider how we ourselves learn, and how this process of “rewarded actions” can be applied to robotics behavior

Goal of RL

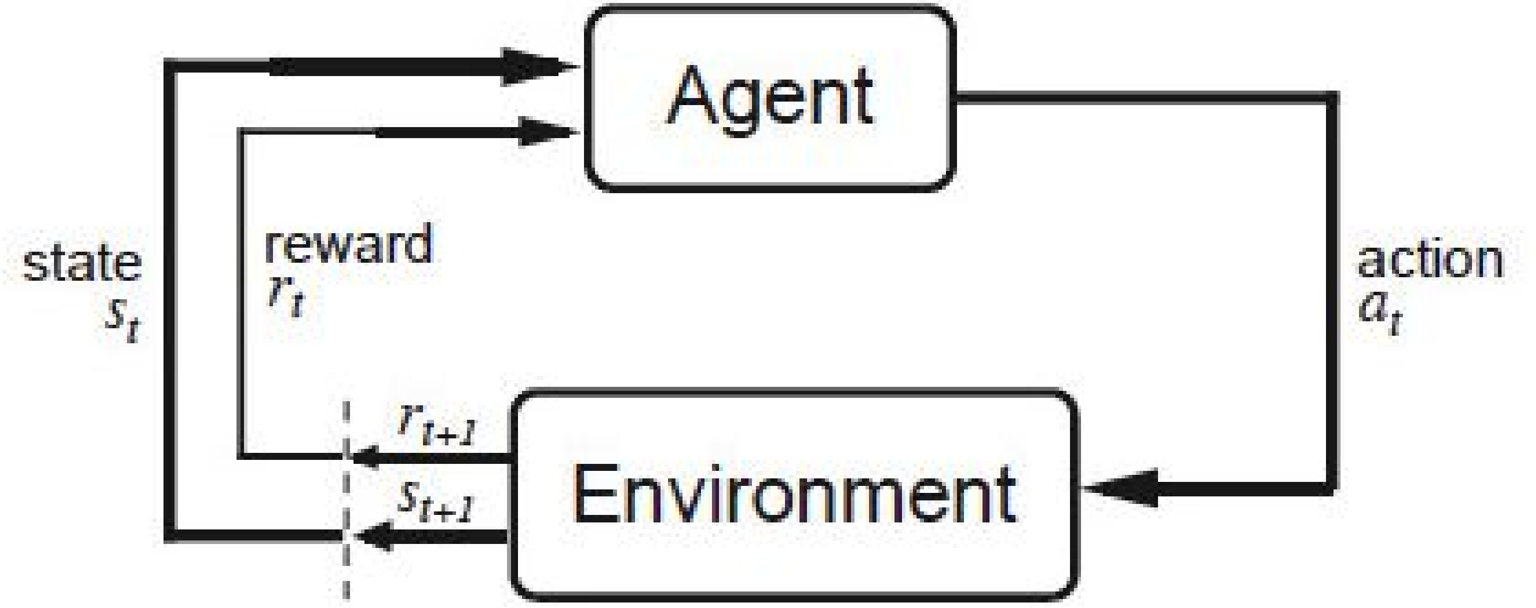


To find a mapping from states to actions that maximize the (cumulative) expected reward

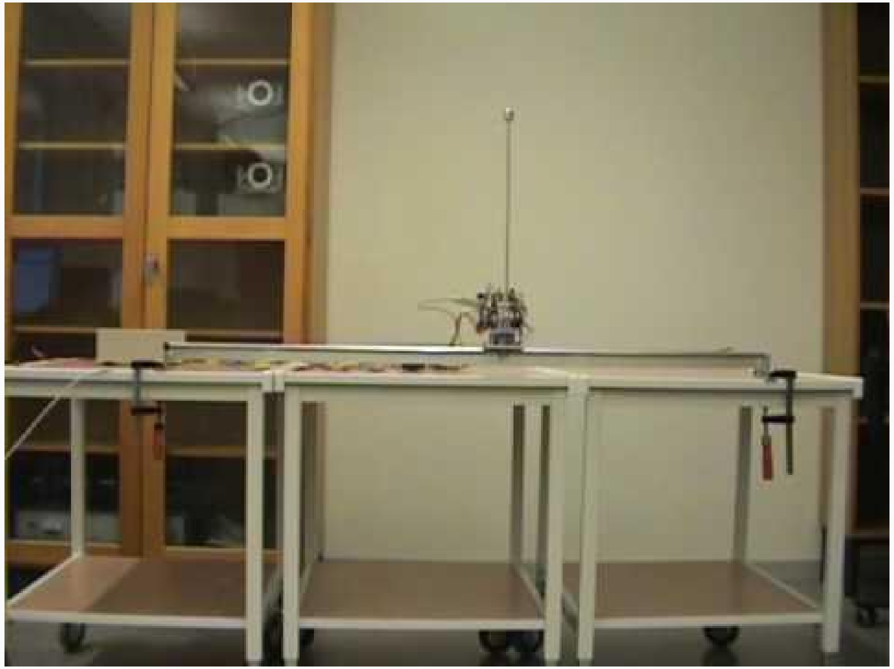
Most approaches assume the use of Markov Decision Processes (MDPs) where the next state depends on the previous one in a non-deterministic way

This is a way to model predictive behavior that is capable of generalizing

Generalized MDP



Classic Example (RL for cart-pole balancing)



Problems/Challenges in RL

Open Problems to consider in robotics design

→ Curse of dimensionality (Bellman 1957)

→ Curse of Real world samples

→ Curse of under-modeling model uncertainty

→ Curse of Goal Specification

Principles of Reinforced Learning

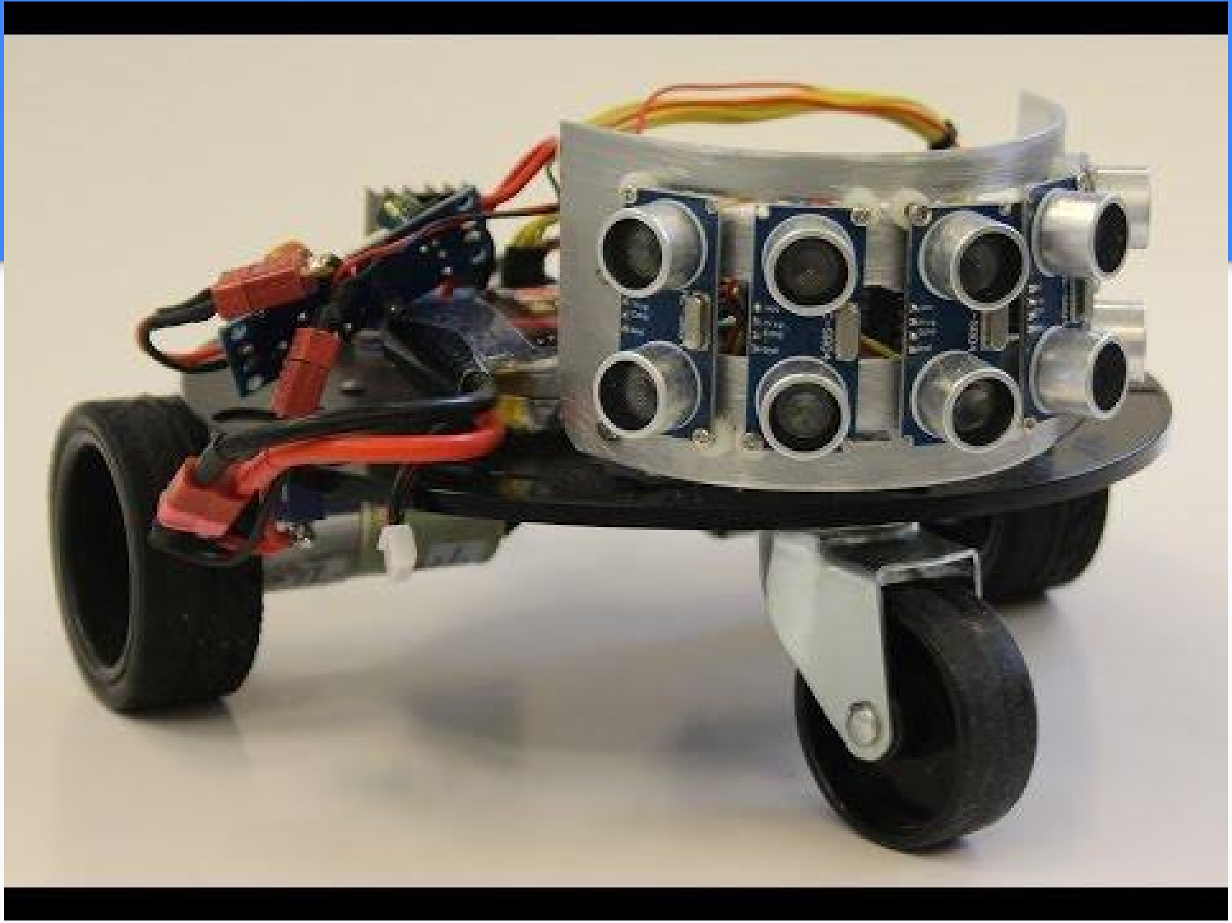
- Effective Representation
  - Reducing dimensionality of states
  - Accurate and robust value functions
  - Search policies to enhance learning speed
- Approximate Models
  - Forward Models/Simulations
  - Reduce learning as much as possible
- Prior Knowledge/Information
  - Demonstration (provide partially successful results)
  - Task Structuring (sub-tasks)

Semi-Online Neural-Q-Learning

Keeping track of the most representative learning samples in a state table with corresponding actions

Makes sure to update this table, reflecting whether the reward was good (+) or poor (-)

Can be implemented as a greedy algorithm (taking the best reward for a state and action)



Relevance to Current Project

Obstacle avoidance behavior - Keep the robot from crashing into obstacles. Uses sensors to detect obstacles to build a table of rewards as well as using the appropriate sensor information to build speed vectors in the proper directions

Avoid trapping behavior - Uses previously used paths to go back and make acute adjustments until trapping is overcome. This essentially “builds a map” for future use (based on the paths taken and sensor information)

References

J. D. Ratcliffe, P. L. Lewin, E. Rogers, J. J. Hätönen, and D. H. Owens, “Norm-optimal iterative learning control applied to gantry robots for automation Applications,” IEEE Trans. Robot., vol. 22, no. 6, pp. 1303–1307, 2006.

J. Kober, J. Andrew (Drew) Bagnell, and J. Peters, “Reinforcement Learning in Robotics: A Survey,” International Journal of Robotics Research, July, 2013. (pdf)

K. Goldberg and B. Kehoe, “Cloud Robotics and Automation:A Survey of Related Work,” 2013.

M. Carreras-Perez, “A Proposal of a Behavior-based Control Architecture with Reinforcement Learning for an Autonomous Underwater Robot,” University Girona, 2003.

"Reinforcement Learning." - *Csewiki*. N.p., n.d. Web. 05 Nov. 2016. (link)

Vilches, Víctor Mayoral. "Reinforcement Learning in Robotics." *RSS*. N.p., n.d. Web. 04 Nov. 2016. (link)