Autonomous Robotic Systems

CS 685

Jana Kosecka, 4444 Research II
kosecka@gmu.edu, 3-1876
Office hours Tue 2-3pm

Logistics

- Grading: Homeworks + Project 65% Exam: 35%
- Prerequisites: basic statistical concepts, geometry, linear algebra, calculus, CS 480, CS 580
- Course web page
  https://cs.gmu.edu/~kosecka/cs685/

- Homeworks about every 2 weeks, Midterm, Final Project
- Choose from the list of projects, suggest your own
- Implement one of the covered methods on robot/robot simulator, come up with new ideas of robotics tasks
- Write a report and prepare the final presentation
Recommended Text


Overview of the topics

- Kinematics, Kinematic Chains, Mobile Robot kinematics
- Notion of state, sensing state, elementary control
- Motion planning, Graph Based Methods, Potential Field Based methods, Sampling Based Methods, Configurations Space

- Robot Perception – Image Features, Stereo, Motion Estimation and 3D reconstruction, Object Detection, Semantic Segmentation

- Foundations of Probabilistic Robotics
- State estimation and Tracking
- Localization using Particle Filters
- Simultaneous Localization and Mapping using vision and RGB-D data

- Dynamic Programming and Markov Decision Processes
- Learning how to act – Reinforcement Learning
Course Logistics

• Required language Python
• Robot simulators, real robots

• Availability of robotics platforms
• Pioneers with range sensors, cameras

• Turtlebot Pyrobot open source robotics platform http://pyrobot.com
• Humanoid – Small soccer league
• Simulators – AI Habitat AI-Thor https://ai2thor.allenai.org/
• CARLA Autonomous Driving Simulator https://carla.org/
• List of resources for mobile robotics http://www.mobilerobots.org/

• Possibilities of programming real robots equipped with range sensors, RGB-D cameras
• Current trends and areas of robotic technologies
- House 3D (Facebook Research)
- Rich simulated environments
- Navigation, Perception, Visual Question Answering

Applications History - Robots in manufacturing/material handling

Manhattan project (1942) – handling and processing of radioactive materials – Telemanipulation

Manufacturing
- storage, transport delivery
- table top tasks, material sorting, part feeding – conveyor belt
- microelectronics, packaging
- harbor transportation
- construction (automatic cranes)

Suitable for hard repetitive tasks – heavy handling or fine positioning
Successful in restricted environments, limited sensing is sufficient – limited autonomy

Autonomous Robotic Systems
AGV’s - automated guided vehicles
AUV’s - automated unmanned vehicles
Grasp planner. One failure mode occurred when the RGB-D
I. Failure Modes
attempts.

successfully placed the correct objects in the box on 4 out of
the object to either the shipping box or a reject box, depending
demonstrations mapping binary images to push locations [31].

first separated the objects using a policy learned from human
on a planar worksurface, illustrated in Fig. 7. Since the Dex-
three distractor objects when starting with the objects in a pile
of three target objects to a shipping box in the presence of
planner, we used it in an order fulfillment application with
H. Application: Order Fulfillment
took an average of 2.5
2.0 grasp planner achieved 94
finds in the supplemental file. The CEM-augmented Dex-Net
maxima of the robust grasping policy. More details can
with the highest predicted robustness, in order to find better

Transport three target objects to a shipping container (box on right).

Fig. 7: (Left) The test set of 40 household objects used for evaluating the
image-based grasp quality metrics (IGQ) and point cloud registration (REG).

GQ performs best in terms of success rate and precision, with 100
details on the methods used for comparison can be found in Section VI-C.

TABLE IV: Performance of grasp planning methods on our grasping bench-

Rob
Dexnet
Planning
Local site
Teleoperation
Need for (semi)

Applications History - Space Robotics

50-ties US space program, exploration of planets, collecting samples
Astronauts bulky space suits – difficult
NASA, JPL, DARPA – sponsoring agencies
Space programs, military application – surveillance, assistance
Planetary Rovers – initially controlled by humans
- large time delays,
- poor communication connections
Need for (semi) – autonomy

Teleoperation – Mars Rover

Human operator controls the robot
Local site – human views the sensory data, sends the commands
Remote site – sensors acquire the information
Lasers, camera, radar, GPS, compass, antenna, IMU,
Steer by wire system, PC’ s with Ethernet for processing information from sensors

Grand Challenges

DARPA

Learn policies

Supervised learning paradigm
training data $o_t, a_t$

Learn the policy $\pi_0(a_t|o_t)$

End-to-end interpretable trainable motion planner

Optimizing perception, motion planning and control jointly integrating map data, predictions of the object detectors

End-to-end interpretable neural planner, CVPR 2019
Example 1: Building Virtual Models of Mars

Example of stereo pipeline, from raw data, preprocessing, meshes, texture maps

See http://schwehr.org/photoRealVR/example.html

Apollo

Lunar Rovers

Current NASA Prototype
Applications: Underwater robotics

- Sensor network
- Remotely Operated robot for ocean exploration

Robots in the service of humans

- Robotic surgery - DaVinci robotic surgery robot – human assisted
- Robotics in rehabilitation surgery (Hocomo Inc)
- Mobile Robots
  - courier in buildings and hospitals, vacuum cleaners,
Variety of domains and tasks

Games and Entertainment

Furbies

Aibos Latter & Macaron

Aibo soccer league - RoboCup
Environment percepts

Models

actions

Architecture

Deliberative decision making

Interface/Language

Task planner

Map Builder

Path Planner

Collision Detection/Kinematics Dynamics Control

Semantic Parser

Perception

Localization

Action

Feedback/Reactive control
Architecture

Deliberative Control and decision making

- Semantic Parser
  - Interface/Language
  - Task planner
  - Map Builder
  - Collision Detection/Kinematics Dynamics Control
  - Localization
  - Perception
  - Path Planner
  - Action

Robotics and AI

Knowledge representation
how to represent objects, humans, environments
symbol grounding problem

Computer Vision, Pattern Recognition, Perception
recognition, vision and motion, segmentation and grouping representation

Natural Language Processing
provides better interfaces, symbol grounding problem

Planning and Decision Making
How to make optimal decision, actions give the current knowledge of the state, currently available actions

Learning in Robotics
Learning to plan, learning to explore, learning to perceive, visual dialog
Learning to grasp, end-to-end learning, modular learning
Autonomous Robotic System

- Three Basic Components of the Robotic System
  - SENSE – process information from the sensors
  - PLAN – compute the right commands/directives
  - ACT – produces actuator commands

- Different organization of these functionalities gives rise to different robot architectures
**Autonomous Driving**

- DARPA Grand Challenge 2005
  - 2004 CMU vehicle drove 7.36 miles out of 150
  - 2005 5 teams finished, Stanford won
- DARPA Urban Challenge 2007
  - Urban environment other vehicles present
  - 6 teams finished
- Google Self-Driving Car
  - By July 2015 1M miles, 14 minor accidents
- Ernst Dickmans / Mercedes Benz 1987
  - 1758 Km, 60 miles per hour
- Parking maneuvers, overtake maneuvers, skidding

---

**Robotic Navigation**

- Stanford Stanley Grand Challenge
- Outdoors unstructured env., single vehicle
- Urban Challenge
- Outdoors structured env., mixed traffic, traffic rules
Robot Components (Stanley)

- Sensors
- Actuators-Effectors
- Locomotion System
- Computer system – Architectures – (the brain)

- Lasers, camera, radar, GPS, compass, antenna, IMU,
- Steer by wire system
- Rack of PC’s with Ethernet for processing information from sensors

Stanley Software System
• Terrain mapping using lasers

• Determining obstacle course

**Autonomous Driving**

• Reduction of the number of accidents
• 5 million accidents and
• > 30,000 fatalities due to traffic accidents
• Time recovered due to commuting,
• Improved parking in the cities,
• New models of personal mobility
Mapping, Control, Planning for autonomous driving

- Navigation strategies
- Trajectory following, planning, (overtake, lane change)

Data Driven, Machine Learning Techniques

Autonomous Helicopter Flight

[Abbeel, Coates & Ng]

Kalman filtering, model-predictive control, LQR, system ID, trajectory learning

Slide courtesy P. Abbeel
Four-legged locomotion

- value iteration
- receding horizon control
- motion planning
- inverse reinforcement learning
- no learning
- learned

Kolter, Abbeel & Ng

Slide courtesy P. Abbeel

Mobile Manipulation

- [Maitin-Shepard, Cusumano-Towner, Lei, Abbeel, 2010]

- localization
- motion planning for navigation and grasping
- grasp point selection
- visual recognition

Slide courtesy P. Abbeel
Visuomotor Learning

[Levine*, Finn*, Darrell, Abbeel, 2015]

Learn policies

Supervised learning paradigm
training data $o_t$, $a_t$

Learn the policy $\pi_\theta(a_t|o_t)$
End-to-end interpretable trainable motion planner

optimizing perception, motion planning and control jointly integrating map data, predictions of the object detectors

End-to-end interpretable neural planner, CVPR 2019

Robots @ GMU

Pioneer, Pybot, Flockbots, RoboPatriots
• 100 Billion neurons
• On average, connected to 1 K others
• Neurons are slow. Firing rates < 100 Hz.
• Can be classified into
  • **Sensory** – vision, somatic, audition, chemical
  • **Motor** – locomotion, manipulation, speech
  • **Central** – reasoning and problem solving

---

**Trends in biological and machine evolution**  
**Hans Moravec: Robot**

• 1 neuron = 1000 instructions/sec  
• 1 synapse = 1 byte of information  
• Human brain then processes $10^{14}$ IPS and has $10^{14}$ bytes of storage  
• In 2000, we have $10^{9}$ IPS and $10^9$ bytes on a desktop machine  
• In 25 years, assuming Moore’s law we obtain human level computing power
Course Overview – PART I
Modeling Geometric transformation

- Modeling Rigid Body Motion
- Modeling Kinematic Chains
Modelling motion

- Notion of state, state evolution
- Systems view vector $\mathbf{x}$ denotes the state of the system, vector $\mathbf{u}$ types of controls/actions the system can carry out we will discuss ways of characterizing the motion of the system

\[
x_{t+1} = f(x_t, u_t) \\
x(t) = f(x(t), u(t))
\]
Motion Control: Open Loop Control

- trajectory (path) divided in motion segments of clearly defined shape:
  - straight lines and segments of a circle.
- control problem:
  - pre-compute a smooth trajectory based on line and circle segments

Motion Control: Feedback Control, Problem Statement

- Find a control matrix $K$, if exists
  $$K = \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \end{bmatrix}$$
- with $k_{ij}=k(t,e)$
- such that the control of $v(t)$ and $\omega(t)$
  $$\begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix} = K \cdot e = K \cdot \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$
- drives the error $e$ to zero.

$$\lim_{t \to \infty} e(t) = 0$$
Motion Planning: Graph Based Methods

Single shortest path – single destination \( t \) (single source)
Given pair of vertices – what is the shortest path from \( u \) to \( v \)

Example:

Starting from the source \( s \), the Grassfire algorithm can be represented as follows:

- **Discretize the space**: Create a queue \( Q \) of all pixels at the boundary of obstacles. For each, set the boundary to 1 and the free space to 0.

- **For each element in the queue**:
  - If \( d(q) = 0 \) set \( d(q) = 1 + \min d(q') \) of the neighbors which differ from 0.
  - Add all neighbors to the queue with \( d(q) = 0 \).

Resulting map – distance to the nearest obstacle.
Motion Planning: Roadmap Methods

Capture the connectivity of $C_{\text{free}}$ with a roadmap (graph or network) of one-dimensional curves.

Configuration Space

Workspace

Configuration Space

- C-obstacle is a polygon.
Probabilistic Roadmap Methods

To Navigate Narrow Passages we must sample in them
• most PRM nodes are where planning is easy (not needed)

Idea: Can we sample nodes near C-obstacle surfaces?
• we cannot explicitly construct the C-obstacles...
• we do have models of the (workspace) obstacles...

Robot Perception: Feature Matching

Original image

Strong + connected weak edges

Interest points
Perception: Mapping and localization

- Visual odometry
- 3D reconstruction
Perception: 3D mapping

http://www.cs.unc.edu/Research/urbanscape

Perception for Autonomous Driving

Drivable Areas

Lane Markings

Car detections

Semantic Segmentation
Localization from Semantic Observations via the Matrix Permanent N. Atanasov, M. Zhu, K. Daniilidis and G. Pappas

Scannet Challenge, Dai, Sava, Niessner, Chang, CVPR 2019

NYU v2 - Ground Truth

Semantic Segmentation
Probabilistic Robotics

- Taking into account uncertainty of sensors and actions
- Localization in the presence of uncertainty,
- Map building

Robot Perception

- How to process information from sensors
- Visual Sensing
- Range Sensing

- MDP’s
- POMDP’s

Markov Localization:
Applying probability theory to robot localization

- Bayes rule: \[ p(A|B) = \frac{p(B|A)p(A)}{p(B)} \]
  - Map from a belief state and a action to new belief state (ACT):
    \[ p(l_t|o_t) = \int p(l_t|l_{t-1}, o_t)p(l_{t-1})dl_{t-1} \]
  - Summing over all possible ways in which the robot may have reached \( l \).
- Markov assumption: Update only depends on previous state and its most recent actions and perception.
Probabilistic Robotics: MDP

- “Markov” generally means that given the present state, the future and the past are independent.

- For Markov decision processes, “Markov” means action outcomes depend only on the current state.

\[
P(S_{t+1} = s' | S_t = s_1, A_t = a_1, S_{t-1} = s_{t-1}, A_{t-1} = a_{t-1}, \ldots, S_0 = s_0)
\]

\[
= P(S_{t+1} = s' | S_t = s_1, A_t = a_t)
\]

- This is just like search, where the successor function could only depend on the current state (not the history).

Andrey Markov (1856-1922)
Probabilistic Robotics: Policies

- In deterministic single-agent search problems, we wanted an optimal plan, or sequence of actions, from start to a goal.

- For MDPs, we want an optimal policy \( \pi^* : S \rightarrow A \):
  - A policy \( \pi \) gives an action for each state.
  - An optimal policy is one that maximizes expected utility if followed.
  - An explicit policy defines a reflex agent.

- Expectimax didn’t compute entire policies:
  - It computed the action for a single state only.

Optimal Policies

- Expected rewards for different states:
  
  - \( R(s) = -0.01 \)
  
  - \( R(s) = -0.03 \)
  
  - \( R(s) = -0.4 \)
  
  - \( R(s) = -2.0 \)
Robot Learning, Reinforcement Learning

- How to improve performance over time from our own/systems experience
- Goal directed learning from interaction
- How to map situations to action to maximize reward

![Diagram](image)

- **Agent**
  - Input: state(t)
  - Output: action(t)
  - Feedback: reward(t+1)

- **Environment**
  - Input: state(t+1)
  - Feedback: state(t+1)