MASSACHUSETTS INSTITUTE OF TECHNOLOGY
16.412J / 6.834J – COGNITIVE ROBOTICS
SPRING 2016

ADVANCED LECTURES & IMPLEMENTATIONS

INTRODUCTION

A goal of Cognitive Robotics is to give you a taste what it's like doing research in autonomy and/or robotics. A big part of this is self-learning topics, and being able to teach those topics as well as implement them. That's what you'll be in this project.

The course staff will sort students into teams of about 5-6. Each team will then be asked to:

1. **Self-learn** an advanced, cutting-edge topic related to cognitive robotics
2. Prepare a clear, polished 80-minute **lecture**
    a. Do a dry run with the course staff (about a week before)
    b. Give final lecture during class period
3. **Implement** the algorithm / technique in Python & prepare a short IPython notebook **mini-problem set / tutorial**
4. Do one other group's **mini problem set**

The deliverables for this project will be:

1. Complete a survey selecting lecture topics by Monday, March 14th (groups will be released on shortly afterwards)
2. A brief team proposal due by Friday, March 18th
3. Your advanced lecture (and dry run to course staff the week before)
4. Your implementation & mini-problem set (due same day as lecture)
5. Completing another groups mini problem set (due by 1 week after their lecture, unless extended [see section])

ADVANCED LECTURE

You'll first have to learn the advanced lecture topic yourself, by diving into the literature. For some of the topics below, we recommend some places to start. Feel free to ask the course staff for guidance.

Your lecture will be a full period, 80 minutes long. You'll also be required to give a dry run to the course sometime about a week or so the lecture, to be scheduled later. This means that teams going earlier, while they'll have a more relaxed time later, might need to do some work over March break so that they are prepared to give a dry run in the first week back.

IMPLEMENTATION

Please implement a technique from the topic you're lecturing on. Note that some of the topics could be pretty complex to implement, too much so for a course project. As such, you're free to implement a reasonable subsection, so long as you have a clear way to test everything. You're also encouraged to use third party libraries or implementations as part of your project. Please discuss any questions with the course staff. Some groups' implementations may also be used as part of the Grand Challenge at the end of the class. All implementations must be in Python.

You must also document your implementation, through comments in the code, as well as by putting together a short tutorial / mini-problem set. About 5-6 other students in the class will be doing this tutorial / mini-problem set, so please make it polished and non-painful for them! It should take the other students about 1-2 hours to complete. This mini problem set should discuss how to use your API, with a short tutorial / example, and also a short exercise where the students can do it themselves. This tutorial / mini-problem set should be implemented as an IPython Notebook. You should submit two nearly-identical IPython notebooks: one with the full solutions, and the second without them (but with `#YOUR CODE HERE` type comments), which will be given to other students.

PROPOSAL

The proposal, due on Friday March 18th, is intended to get teams thinking about their projects. The proposal should be fairly short (1-2 pages max). It should include:

- Division of labor (who's doing what)
- Thoughts on the topics you plan to lecture about
- Thoughts about what you'll implement
- A few references (papers, tutorials, etc.)
- Any questions you have

DIVISION OF LABOR

Feel free to divide up the teamwork on your team as you see fit. Come up with a plan early on, because managing larger groups of 5-6 is a lot harder than smaller teams of 2. In fact, a division of labor section will be required in the brief proposal due on March 18th.

It's fine to have some of your team doing a little bit everything (but again, make sure that too many chefs aren't spoiling the soup). It's also fine to have part of your team tackle the lecture, while the other half does the implementation. Just as long as you everyone knows and understands the subject material.

EXTENSIONS ON OTHER ASSIGNMENTS

Putting together both an advanced lecture and a high-quality implementation is a lot of work, and the course staff realizes this! As such, you'll have an automatic extension on any assignments due around the time of your advanced lecture as follows:

- Any problem set (or mini problem set from another group) spanning part of the 2 "business weeks" before your lecture, will be due for you 1 week after your lecture. Spring break does not count as a business week. Further extensions would require the use of your "get out of jail free" card.

Example 1. Suppose you're in the first group, so you're advanced lecture is on 4/4. Then all problem sets you could be working on in the interval between 3/14 – 4/4 (longer because it goes through Spring break!) will all be due for you on 4/11.

Example 2. Suppose you're in the last group, so your advanced lecture is on 4/27, but you've signed up to to do the mini-problem set for the group before you (which will be available around 4/25). Then any problem set (including the mini problem set) overlapping the range of 4/13 – 4/27 will be due for you the following week on 5/4.

TOPICS

Here are the advanced lecture topics for this year, and their due dates (both lecture and implementation):

1.) **Incremental Path Planning (4/4)**
"Changing paths when the world changes"

Incremental path planning is a technique in which new paths can be generated efficiently under a changing world. For instance, if the robot discovers an obstacle in rough terrain that it didn't know about before, it's more efficient to modify the results of previous searches to come up with a new solution instead of doing a full replan from scratch. Topics in this lecture should include D* Lite, and could include other such topics as Lifelong Planning A*, D*, incremental APSP algorithms, or other. Suggested authors of papers include Sven Koenig and Maxim Likhachev to start. The lecture should also discuss its application to mobile robot path planning.

Recommended implementation: Implement one of the algorithms above such as D* lite (ideally keep it a general API / capability), make a small mobile robot simulator, and then apply your algorithm to the simulator.

2.) **Semantic Localization (4/6)**
"Finding where we are with respect to things we care about"

Localization in robotics is the task of figuring out where the robot is. Typically, this is reported in terms of geometry (i.e., the robot is at position *x, y* relative to some map). Semantic localization, however, may be more similar to what humans do. Semantic localization answers the "where we are" question not by returning numerical coordinates, but rather semantic tags, such as "I am in the kitchen". This lecture will introduce semantic localization, and discuss some techniques. Another related and possible topic of the lecture could be semantic mapping. The lecture should discuss applications to robotics.

Recommended implementation: Implement a semantic localization algorithm, ideally keeping the API / capability generic, and apply / test it on a small simulated domain.

3.) **Visual classification through deep learning (4/11)**
"Using deep neural nets to recognize images"

This lecture isn't just about visual classification, and isn't just about deep learning (though you'll need to discuss both of these), but rather about the two of them together. Deep learning and deep neural nets have recently made grade advances for recognizing images. Some suggested topics include convolutional neural nets, visual classification, various procedures for putting them together, and possibly other.

Recommended implementation: A system that can recognize from several different classes of images. Such as "plant", "flag", "human", etc.

4.) **Monte Carlo Tree Search (4/13)**
"Using randomness to improve some kinds of search"

Monte Carlo Tree Search is a technique that exploits random simulation within search. This lecture will discuss it algorithmically, as well as discuss some applications. Notably, you should end by giving an overview about how it's used in AlphaGo, which recently defeated the world's master Go player.

Recommended implementation: Monte Carlo tree search (ideally a general capability), and also an example / problem, possibly applying it to a simulated game.

5.) Computing Reachability for Planning, Control, Verification and/or Monitoring (4/20)
"Figuring out where our system could possibly end up"

Reachability, or computing reach sets, is the task of figuring out what states a dynamical system could possibly reach. They have applications for planning, control, verification, and monitoring of systems. This lecture should introduce the notion of reachability, and describe its application to different systems. Some possible topics include an elaboration on the "flow tubes" (introduced in lecture) and "funnels" (a similar topic), in which a a dynamic system and some control rules guarantee that you'll eventually reach a certain area of state space. This lecture should also discuss different ways of representing reach sets (such as possibly the polygonal representation introduced in lecture), and how to compute them, as well as applications to robotics.

Some suggested references include those of guest lecturer Andreas Hofmann, as well as papers from Prof. Russ Tedrake (who has used funnels extensively for the control of complex dynamica systems), and more.

Implementation recommendation: construct flow tubes and/or funnels for some simple dynamical systems.

6.) Temporal-based logic for expressing Plan Goals and/or Preferences (4/25)
"Making plans with not just goals at the end, but goals through time"

So far in lecture, we've discussed planning problems in which we want to reach some goal state by the *end* of the plan (ex., Levinia is saved by the end, we have a stack of blocks by the end, the package ends up at this city in the end, etc.). What if some of our goals go through time though? For instance, what if we want to encode that whenever a door is opened, it must be closed two time steps later? Or that some property will eventually hold, or always hold? This lecture will present some techniques for planning with temporally-extended goals. A related topic, that you can also include at their discretion, is that of temporally-extended preferences or "soft goals." These are goals that might be preferred (like, "try not to be holding heavy objects anywhere in the plan") but are not hard constraints in that the plan is allowed to violated these soft goals, but it's not as nice.

In the lecture, we recommend you introduce Linear Temporal Logic (LTL), which can be used to express goals and preferences over time. The lecture will then discuss techniques for using this in the planning process. We recommend you look at the work of Jorge Baier ("planning with temporally extended goals as heuristic search" and / or Emil Keyder (who has worked with soft goals / preferences, but less so with temporally extended goals).

The implementation for this project may be trickier, as it could involve understanding some complex automata theory and implementing a full-fledged planner may be too time consuming given the scope of this project. We therefore recommend, however, you could implement (potentially a subset) of the system described in Baier's work, where temporally extended goals are translated into automata, which are then translated to a new PDDL domain over which a standard classical planner can be called. Perhaps the LTL to automata translation could already be provided for you, you'd then need to translate it to PDDL.

7.) **Infinite Horizon Probabilistic Planning or Multi-Agent Planning as MDP's (4/27)**
"Planning with loops in uncertain domains" or "Planning probabilistically with multiple agents"

In lecture so far, we've discussed classical planning and planning with time. This lecture will extend these into probabilistic planning, and optionally, with a multi-agent twist. Specifically, the lecture will be about some advanced topics in probabilistic planning. These could include LAO* (which is a form of heuristic forwards search with loops, can be applied to Markov decision processes (MDPs)), or other advanced probabilistic planning techniques. Note that for these topics, you'll probably want to give a brief overview of MDP's and POMDP's (but the lecture should not focus on these, it should focus more on the advanced topics that build on them). The exact topic is up to the group however.