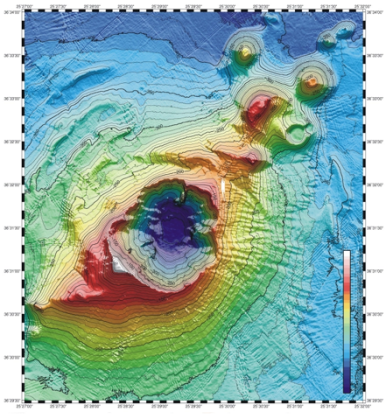
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
16.412J / 6.834J – COGNITIVE ROBOTICS
SPRING 2018

GRAND CHALLENGE

INTRODUCTION



You will culminate Cognitive Robotics this semester with a Grand Challenge: a showcase of the technology you've developed and taught in your Advanced Lecture groups, applied to a real (or simulated) robotic platform. This is the final project for this course; we hope you have **fun** with it!

While the goal of the advanced lecture was for you to gain insight into an innovative method, including its theory and implementation, the goal of the grand challenge is to gain insight into its appropriate application in the context of a significant real-world problem. Like the advanced lectures, this project is **flexible**, in that you will use your creativity to decide innovative ways to meet the grand challenge, using what you learned during your advanced lecture. This innovation could include, for example, an innovative application of the methods from your advanced lecture, an elegant analysis and selection of the most appropriate method, or a novel extension or combination of these methods.

Nominally, you'll be working with the same team as your advanced lecture. Feel free to break into sub-teams of for example two to three, in order to tackle the grand challenge from different angles. If you decide to form a sub-team from your advanced lecture team, or a whole new team, then please discuss it with the course TA, and leave the staff a note on Piazza, with the members of your new team by Apr 18th. All team members should divide up the work amongst themselves, and each must make an equitable contribution to the effort. Each individual will write up their contribution separately in their final report.

THE CHALLENGE SETUP

Each team will implement a module and together the teams will integrate and finish the Grand Challenge. We will provide both a simulation environment in Gazebo and a hardware testbed for testing and demonstration. Our challenge will be based on a real-life field robotics scenario that resembles the science mission the course staff did during their January 2018 Hawaii trip. That is, our challenge theme is coordinating robot teams to explore an uncertain environment in order to maximize science goals.

Figure 1. X-WAM

For our challenge, we will flip the mission from underwater and surface, to air and ground. During the Hawaii trip, the science team on board the research vessel Falkor uses a fleet of heterogeneous underwater vehicles to "scout" the ocean. To reflect a similar setting, we will provide a mobile base vehicle, either a X-WAM or a Turtlebot 3 (to be decided), as the ship Falkor, and up to 4 Bebop2 UAVs as the underwater vehicles.

Figure 1 shows the X-WAM. This mobile platform is rather powerful and has four mechanum wheels – making it omnidirectional (it can move sideways in addition to forward and back). Additional capabilities and sensing on this robot include a Hokuyo LiDAR, an onboard camera with point, tilt, and zoom capabilities, a Kinect 3D camera, the ability to lift its top up and down vertically via a scissor lift mechanism, and an onboard quad-core computer with wireless networking. (We might change to a Turtlebot 3 Burger or Waffle. To be announced.)

Figure 2 shows the Bebop2 UAV. It is a quadcopter with a 1080p HD camera on board, with a battery life of 25 minutes. We will be running our software on these robots, and also on a base station computer that can wirelessly communicate to the onboard computer using ROS.

Figure 2. Bebop 2 UAV

The location of the grand challenge is to be announced. We will construct a 3-dimensional testing environment with Legos, but your implementations can make use of a 2D map or a 3D map, as you see fit. Discuss with the course staff early on if you have specific needs on the setup for your demonstration.

The simulator and the hardware testbed will be provided and announced later.

GUIDELINES

For the final project, you need to demonstrate your ability to understand, extend, implement and integrate novel techniques from your advanced lectures. The implementation for your advanced lecture will likely be oriented to be more pedagogical. You should think through how you want to leverage this implementation towards your final project, versus an implementation that is tailored towards the challenge, and its performance needs. Your implementation should push along at last one dimension of excellence and innovation, which could be one of the three following:

> **Innovative application.** You come up with a clever idea for applying one or more methods from your advanced lecture to the Grand Challenge. You focus on turning this into reality, by designing your solution and integrating it with RMPL/Enterprise, and methods provided by other teams. This showcases your talent for innovative systems.
>
> **An extension.** You can pick one of the papers that describe a method and re-implement it (you may have done that for your advanced lecture), reporting on the surprises that emerge as you understand the work at a deeper level, or thinking about the limitations of your work as you apply it to the grand challenge. You then create and report upon a novel extension to that approach, reporting on the improvements you make. Alternatively, you can propose a novel integration of two methods.
>
> **A systematic benchmark and comparison of a number of existing methods.** You can pick two or more methods described in papers (preferably more recent ones), implement the methods and carry out a performance benchmark of the methods. You should pick a benchmark that can be carried out fairly rigorously, and one which will yield some interesting results, e.g. something that demonstrates the difference between the methods. Try to pick some dimensions along which the performance of the different methods can be compared (time vs. space usage, complexity of problem, accuracy, optimality and risk).

By breaking your advanced lecture team into smaller sub-teams, you have a better opportunity to collectively explore your lectured methods along each of these dimensions, with each sub-team focused on one facet.

At the same time, you will use your implementations, combined with those of other lecture teams, to solve the grand challenge. In our cognitive robotic architecture Enterprise, we have path planners, activity planners and state estimators modularized. This allow us to plug and play different planners and components for different applications. For the final project, your implementations need to be packaged into a module following the same criteria. Your module will interface through ROS (more details below) to talk to Enterprise and the hardware. All implementations should be in Python. This means that you also need to do the following:

Document your module/package. You should give a high-level overview of what functionalities and features your package provides. You should record the API of any exported functions, and be specific about input / output format, the assumptions of your method, and what your functions do. We recommend using a Git repository and writing your documentation in README.md.

Model and solve a real problem. Each team will model the problem for their responsible part of the Grand Challenge. That is, you will extract from the scenario what the input / output specification should be for the problem. Then, you will identify any gaps between the problem that you're solving and what your implementation provides, and you will make adaptations to your functions accordingly. Your modelling of the problem should consider other teams, since you are jointly solving a challenge together. In the end, with the help of the course staff, teams will together model the challenge scenario using RMPL and the Enterprise/ROS architecture.

SCHEDULE

3/7.	Lecture on grand challenge scenario: deep sea exploration.
3/14.	Project guidelines out.
4/4 11:59 pm.	Proposal due.
4/25 11:59 pm.	Progress report due.
5/11 11:59 pm.	Final report due.
	Implementation due.
5/16.	Grand challenge during class.

The grand challenge itself will be on Wednesday, May 16th during the normal class period, from 10:30am – 12pm. Additionally, many groups will need plenty of lab time with the hardware. This will be arranged on an as-needed basis; just email the course TA.

Your final implementation and report will be due on Friday, May 11th by 11:59:59 PM. The time between the due date and the actual demonstration time will be used by the course staff to coordinate with each team to make sure things actually run for the Grand Challenge.

DELIVERABLES

We list the deliverables for the final project. The final project will consist of 100 points in total and takes up 25% of your total grade. The team will share the same score for 40 points, and 60 points will be evaluated individually.

Proposal (due Wed 4/4 11:59pm, max 6 pages, 10 pts):

1. A short motivation and problem statement, related to your advanced lecture.
2. Background, introducing the methods that your implementation will be based on.
3. Proposed API
4. Your innovations and proposed implementations
5. Your planned evaluation, including any data that you will use

6. Your plan for the Grand Challenge
7. Planned schedule
8. Division of work (see details later)
9. Any questions or what help you want the staff to provide

We need one proposal for the whole team.

If you plan to break into sub-teams to implement multiple innovations, each sub-team should write a sub-section and state your innovation and planned evaluation separately.

Make sure you carve out time early on in order to perform any needed back ground research, to brain storm strategy, and to make a solid plan, before you write your team proposal. As you go along the way, some parts of your plan may change, but it will help you make steady progress towards getting things done.

Think carefully about your API. Where you plan to integrate with other groups (details below), talk to them about how you will interface through ROS. We recommend designing both a ROS-independent API and a separate ROS interface (e.g. you can write wrapper functions to interface with ROS if appropriate), so that your evaluation does not have to go through ROS.

Plan ahead on what your evaluations will be. They should help you validate your novel extension or help lend insight into the analytical questions you want to answer. Find data that you need early on in the project.

For the schedule, you should plan the project in two spirals. For the first spiral, you should focus on getting your basic implementations solid. Identify the basic components and the difficult ones. Stub out any difficult components, but make sure you have place holders for all the components. In addition, make sure that you have exercised all interfaces. During your second spiral, elaborate the module to its full capability in light of the time available, and provide a good set of evaluations to support your innovation.

Think carefully about how you divide your work. See more details below.

Progress report (due Wed 4/25 11:59pm, max 3 pages, 5 pts):

1. Current Progress.
2. Any surprises or interesting findings during your implementation.
3. Any changes towards your planned API & implementations.
4. Rest of the schedule.

We need one progress report for the whole team.

The end of the first spiral should correspond roughly with the deadline for the progress report. At that point, carefully reassess where you stand, and the difficulty of the remaining work for each component. Then adjust the scope of your project accordingly.

Your progress report should report surprises or findings during your implementation, if any, and describe how they will affect the direction of your remaining work.

Final report (due Fri 5/11 11:59pm):
The team as a whole provides **(5 pts)**:

1. Cover page

2. Summary of motivation, problem statement and contribution of work (**~1 page**).

Each team member provides a final report that highlights what each member has learned. A final report should include the following components, depending on your work division (**4 two-column conference pages per person, 60 pts per team member, evaluated separately**):

1. Abstract
2. Motivation
3. Problem statement
4. Background
5. Method
6. Any notable implementation details
7. Evaluation, analytical and/or empirical
8. Demonstration, i.e. your simulation/hardware demonstration during Grand Challenge
9. Discussion, including any surprises that discovered during implementation or integration with hardware/simulation, any major insights you gained along the way and lessons learned
10. Conclusion
11. Self-evaluation of your contribution to the team

Everyone in the team should write a separate 4-page final report in conference format.

We will use the final report to independently assess your understanding of the subject. In your team 1-page summary, you should provide the high-level motivation and problem statement. In your individual write-ups you should emphasize the work that you have done, and most importantly the insights that you gained from this work, instead of trying to cover the whole project. This allows you to show your deeper understanding and your unique view. If 2 or 3 of you are working closely on one part of the project, you can talk about the same content, but please offer your own insights and provide separate reports.

In your abstract, include the problem you solved, how it fits into the team and grand challenge, your approach and insights. Your motivation and problem statement should elaborate upon the specific challenge that you personally address. Ask yourself one of the following questions, as appropriate: What is the challenge that your novel extension is addressing? Or what is the question you are trying to answer with your comparison? Or what is the application need within the Grand Challenge that caused you to adapt your lectured method?

Implementation (due Fri 5/11 11:59pm, 20 pts):

1. Code
2. Documentation

We need one repository for the whole team.

Your code should be cleaned up. You should include a simple example for how to use it. Include any evaluation scripts. Include any modellings you've done for the Grand Challenge. If you use large amount of data, do not include them, but record the sources of the data.

PROJECT GRADING

Final report:

> A – represents mastery: mastery illustrates the ability to analyze, extend and apply existing methods in a way that is novel and insightful, and the ability to explain and motivate in a manner that is particularly intuitive.

B – represents solid competence: illustrates the ability to motivate, explain, implement and evaluate a focused set (i.e., 1 or 2) of methods.

C – represents partial competence of the above.

Others:

A – you have satisfied the requirements we gave, and you have shown that you gave some thoughts to the questions we asked.

B – basic satisfaction of the requirements.

C – partial satisfaction of the requirements.

DIVISION OF WORK

The course has large teams of 4-6 people. To avoid free-loading and ensure everyone has the full learning experience, you should divide your work reasonably. The most effective way is often to have 1-3 people working closely as a sub-team on one part of the project. Here are some suggestions for how you might divide the work.

The team as a whole works under the same topic, which you explored in your advanced lecture. Together, you have given a remarkable advanced lecture, have understood the major paradigms, and have developed a pedagogical implementation. Collectively, you have done an extensive survey of the subject, and each of you may have some ideas about the limitations, extensions, and novel application of the subject's methods. The team can brainstorm about these ideas and think about what applications, extensions, and comparisons you want to do. You then pick a couple that interest you most and break off into smaller sub-teams to implement them. Together, the team decides on the API, taking into consideration the Grand Challenge application, and the extensions or comparisons each of you want to do. (Of course, you can have extensions that require different inputs and output, but you should have a consistent set of API.) In the end, you draw insight from these implementations, and decide how they will be used towards the Grand Challenge.

The discussion above is inward facing. In parallel, each team should send out a "scout party" to brainstorm with other advanced lecture teams about a collective vision of how these methods combine to confront the grand challenge. A key to class success is to scope out early on an overall scenario, an architecture, well defined interface to the modules of each team. It is also important to implement the architecture at the very beginning, with most or all modules implemented as "dummy" stubs, to check that you have the interfaces right. It is also important to carefully differentiate minimum functionality from "nice to haves", and to implement, integrate and demonstrate the minimum functionality first, so that you have the flexibility of throwing out "nice to haves" rather than working late at night.

More specifically, here is an example of division of work.

> On top of your pedagogical implementation, you have sub-teams implementing different novel extensions and evaluate them separately. Then, their results are compared on some common benchmarks, and you discuss where these different extensions performed well and poorly. A separate sub-team focuses on an integrated demonstration on the Grand Challenge, jointly with other lecture teams. This sub-team adapts an implementation of the lecture methods in order to work for the Challenge, and design and model demonstration scenarios to best illustrate these

methods. In the end, members of each sub-team write their individual final reports on the work they have done, and the overall team and sub-teams write connecting text (1-page summary) so that it hangs together.

SCENARIO

Here we provide a scenario that reflects what the expedition team did in Hawaii and translate this scenario to the setup that we are providing in the Grand Challenge. This should get you thinking about the API and the functionalities that your module will provide. For the Grand Challenge, you do not need to follow this scenario exactly. You should think about how you would like to design or modify this scenario to best demonstrate your capabilities.

During the Hawaii 2018 deployment, Falkor operates at night to map out the bathymetry data using multi-beam acoustic data. In the morning, the scientists get together and decide which areas are most interesting to explore, based on prior knowledge of where the coral reefs are and their predictions. Over the course of the day, multiple AUVs are deployed to explore those areas. The scientists plan the paths for the AUVs so that they do not collide with obstacles in the ocean, the shallow seafloor (obtained from the bathymetry data), or each other. The fleet of AUVs are heterogeneous. Some of them are more robust and almost fully-actuated using propellers, while others are under-actuated but more energy efficient, by taking advantage of the currents and buoyancy. Planning paths for these vehicles requires more careful reasoning over their dynamics and the environmental uncertainty. As the AUVs collect more samples or images of the coral reef, we update our prediction and adjust the vehicle paths to explore more rewarding areas. At night, the science team analyzes and classifies the data each AUV collected.

The X-WAM (or the Bebop2) can be used to map out the environment using SLAM with the sensors it has available. Given some labeling of interesting regions in the map, we can use Gaussian Process to reconstruct a map of interesting areas. Given the uncertainty information on the map, adaptive sampling can be used to predict where to explore next. We can then plan the paths for multiple Bebop2s to explore those regions simultaneously. To account for the dynamics of Bebop2, we can use path planners for under-actuated robotics to refine the trajectories. As the Bebop2s are deployed, they can create a semantic map along the way with the cameras on board. When the mission is finished, X-WAM will pick them up, and start another cycle of planning and exploration. Alternatively, we can use distributed path planners and update their plans along the way without picking them up in between. In the end, image classification techniques can be used to classifies some images taken by the Bebop2s.

GROUP INTERACTIONS

As shown by the scenario, there is a lot of potential for the groups to interact on the challenges. This sort of interaction is encouraged, but care must be taken to ensure that everyone can integrate together nicely. Below are some ways some of the groups can interact. Be sure to talk early on the API and throughout the project if you do want to work together.

- The SLAM team can produce the map and state estimates to be used by the path planning teams.
- The semantic mapping team can use the map and state estimates from the SLAM team, and the visual data layered on top to create a semantic map.
- The adaptive sampling team can use the uncertainty information from Gaussian process team.
- The under-actuated team can use the path from path planners as waypoints and refine the paths with under-actuated dynamics.

ABOUT THE ARCHITECTURE

For the Grand Challenge, we will program the challenge in RMPL, we will use Enterprise as our RMPL executive, and we will use ROS as the communication layer between team modules. Enterprise will dispatch the activities to be performed using the ROS interface. Upon receiving requests, your module will be run to perform their associated activities. Therefore, in order to integrate with our planning & execution framework, your software will need to use ROS (the Robot Operating System).

ROS is a framework that is well established in the robotics community. At its core, it provides message passing. Programs that can communicate with ROS (called ROS nodes) can send messages to each other over "topics." Additionally, nodes can call special functions in other nodes called "services," and can also call longer-running actions in other nodes called "actions." Please refer to the ROS documentation for more information; there are great tutorials online. **For this assignment, please be sure to use ROS Kinetic**. (There are different versions of ROS). Note that ROS is only supported on Linux (and best supported on Ubuntu). The course staff will be providing a virtual machine with ROS installed if you are unable to install it natively on your machine.

The staff recommends that you read through a ROS tutorial or two. When designing your API, you may want to look for some popular packages and see what input and output format they use, in order to have a more "standardized" API. For example, the path planning teams can look at the navigation stack package. The under-actuated team may take a look at the bebop_autonomy package (the package for the Bebop2 UAV). You may also refer to some ROS packages for API and documentation examples.

QUESTIONS?

If you have any questions, please be sure to ask them on Piazza, preferably earlier rather than later.