Robotics: Science and Systems II
Challenge

Introduction

In this second half of Robotics: Science and Systems, we will continue to address the Grand Challenge of "Building a Shelter on Mars." Recall that in the spring, you built a robot that explored, searched, gathered and built a structure in a dynamic partially-known environment. You studied issues of robot control, visual servoing, motion planning, position estimation and manipulation in the laboratory setting, in a moderately controlled environment. We ended the semester with a challenging but abstract version of the problem, the Course Challenge.

This semester, we will, as a group, scale this task up to something that approximates a real-world challenge task. You will have to address the challenges of unstructured, outdoor environments, unknown world models, real-world navigation and locomotion, and real-world manipulation. These are all open research topics in the literature, so as a class you will be right at the cutting edge of robotics research. Our ultimate goal is for each team to generate a publishable result, although publication is by no means a requirement. In addition to the research challenges in autonomy, you will have to address the engineering challenges of developing in collaborating teams, developing across multiple platforms, and developing software that is reliable enough for others to use.

Project Goal

On December 6th, the Aero/Astro hangar door will open. The robot should, over a long period of time, collect building components located on campus, return them to the hangar, and and assembling as large a structure as possible in the hangar. Your objective, as a class, is to develop this robotic system running your collective software to control this robot to leave the hangar, find as many of the building components as possible, return them to the hangar and assemble them into a structure.

Your robot will need skills to do the following:

- navigate in the environment
- locate good construction objects
- identify the location of the construction site
- retrieve, carry the objects to and place the components at the construction site
- create a structure at the construction site

Your objectives as students of Robotics: Science and Systems is to apply the lessons you learned in the first semester to some real-world problem.

Robot Platform

The "Splinter" is a larger, more capable version of the robots that you built in R:SS I. The major differences are:

- SICK laser range finder
  We have provided a laser range finder and interface software to allow you to acquire precise range measurements to objects in the world. We expect the laser to be useful in obstacle detection, mapping and localization. The laser is *not* capable of sensing the building components.
- A more powerful arm
  We have provided a more powerful manipulator, capable of lifting more and bigger building components.
- More powerful on-board computation
  We expect that all necessary processing for carrying out the task will occur on board the robot. We have provided two single-board computers and two orc boards on each splinter, giving you separate microcontrollers and processors for different tasks.

You will continue to have cameras, and should expect the cameras to be an important component of obstacle detection, component detection and tracking. If you choose to use GPS, you must be robust to loss of GPS signal. You do have 802.11 wireless, however. A motivated team may choose to implement mapping and localization using 802.11 wireless signals. Of course, just as there are GPS deadspots, there are multiple 802.11 deadspots in the outdoor areas of campus.

We have built two (identical) Splinters for you. We expect that each team will carry out design, implementation and testing of their software on their R:SS I robots. An important issue that each team must address is how to design their algorithms and implementations such that their implementations are general across robot form factors (R:SS I-bot and Splinter), and how their implementations will interact with other team tasks. Regular design reviews to minimize the likelihood of inconsistencies will be a major component of this course.

We will give you source code to the Carmen robot control platform, including the laser range daemon, the localization software and navigation. This software is *not* capable of performing outdoor navigation, but may provide a useful framework for development.

Map

You will be given an electronic file with an *a priori* metric map of the environment, containing the locations of campus structures and the locations of the building component locations. This map will be very approximate, at the level of precision and accuracy that you could expect from some overhead observers. We have not specified the map format yet, but you should expect it to be ASCII and easily parsed.

You can expect the campus structures to be given metrically as low-order polygons, with small features not represented. Additionally, the locations of the components to be found will be given approximately, representing the uncertainty that would result from materials dropped from an orbiting craft.

Two graphical renderings of a possible map is shown in figure 1. Clearly, such maps will not be useful for localization or motion planning. But, such maps may be useful for constraining exploration strategies. (We'd prefer not to see any robots driving down Vassar St. at 5pm; your control strategies should use knowledge of the start location and the map to avoid doing so.)

Building Components

The building materials are plastic garden blocks. Some of the blocks have interlocking teeth, and some of the blocks may be double height. Additionally, some blocks may be "intelligent", in that the block itself contains some computation and sensing.

Constraints

You should assume that software reliability is part of the course. You are responsible for being robust to any errors in software you write as a class. You are not responsible for robustness to errors in software you are given (although it's probably a good idea.) You are also not responsible for devising algorithms that are robust to hardware failures (e.g., sensor loss, motor failure), although you have of course seen issues of calibration error and sensor noise, and should expect such issues to continue to be a problem. Finally,

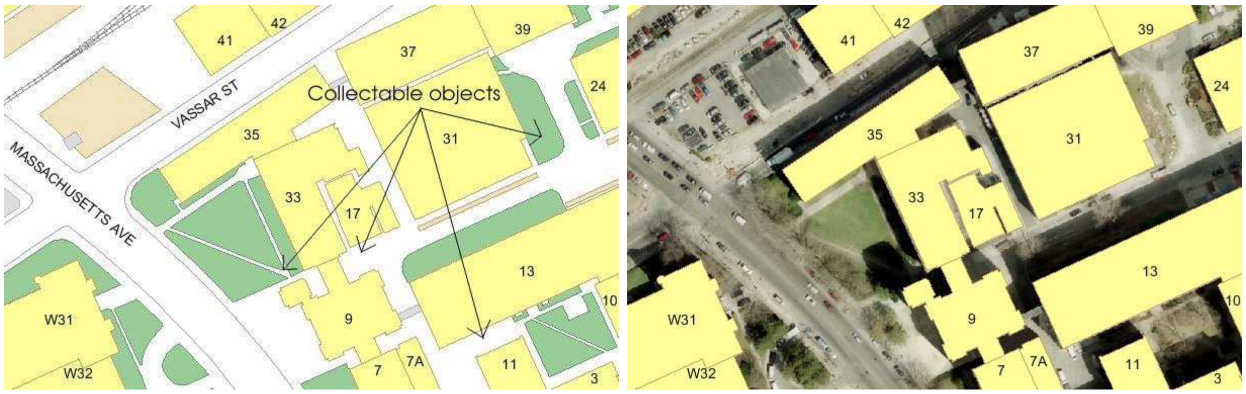

Figure 1: Example map and object locations.

you are responsible for devising algorithms that are robust to loss of internet connectivity. We will disable networking periodically to test how reliant you are on the wireless.

Project Tasks

There will be five teams consisting of 5 people per team. Each team will have a faculty advisor, and each team will meet with their faculty advisor once a week during the Tuesday lab sessions. Each team must choose, in conjunction with their advisor, an appropriate "critical task" in the grand challenge problem. Over the course of the semester, the team must then implement a solution to their chosen critical task.

During the first 17 days of the semester, each team must restore their microbot from R:SS I to working order; the microbot will serve as a test platform for each team's implementation in isolation of the other teams. Because the Splinters are limited resources, it is essential that teams be prepared to develop on the microbots. Implementations that generalize across multiple platforms will be valued by the teaching staff compared to robot-specific implementations that require ad-hoc patches to port from microbot to Splinter.

During the time that you are re-acquainting yourself with your microbot, each instructor will give one lecture outlining what they consider to be the key critical task, and what the interesting research questions are. Examples of critical tasks include (but are not limited to):

- **Large-scale motion planning and exploration**

  One critical task may be the development of a motion planner (possibly by extending the implementation from R:SS I) to allow precise motion planning across the large-scale space of MIT campus (rather than just the hangar, or the 3 rooms of Bldg. 35). This planner could also be extended such that new sensor information triggers replanning in order to allow intelligent exploration.

- **Vision-based Object Detection and Localization**

  One critical task may be the detection of and visual servoing towards important features in the environment, such as the building blocks, but also curb cuts, and obstacles not detectable by the laser range finder. The visual tracker might publish the type and location of each object that it recognizes and tracks in the visual field.

- **Navigation and Mapping**

  One critical task may be learning a globally consistent model of the environment from sensor data. This model could contain information about the relative positions of obstacles, building materials that have been seen but not yet collected, curb cuts for navigation onto sidewalks, and other relevant features.

- **Manipulation Planning**

  One critical task may be to generate plans for which bricks to collect and how to place bricks within the hangar. The intelligent building blocks may require development of specific placement strategies that allow the bricks to work together.

- **Reactive Robot Control**

  One critical task may be to convert high-level motion plans into local control strategies that react to environmental features. If the motion planner indicates that the next building brick to be collected is on a sidewalk, a reactive navigation system that finds a curb cut may be a required, while reacting to rapid changes in the environment such as pedestrians.

System Integration

The challenge is clearly too large a problem for any one team. Each team must choose one or two of the tasks and address these over the semester. The challenge will culminate with all teams running their tasks concurrently on a single robot. There is a clear issue of how to integrate the different systems cleanly. This will be addressed in several steps:

- The instructors must ensure that the tasks are sufficiently well-defined to allow clean modularity.
- By October 3rd, we will begin regular design reviews. One member of each team must present the current design, including specifics of interfaces, methods and parameters. You will be graded not only on your ability to present your designs, but also on your ability to react to criticism from others and your ability to incorporate useful criticism from others.
- By October 17th, each team must have defined a Java interface class for their task(s). This API allows all other teams to interact with it. By the end of the first month, this API must be frozen. A class library that adheres to the interface with stubbed out methods of the API must be provided by each team to all other teams, to allow testing.
- By November 7th, we will begin regular test reviews. One member of each team must present the current performance of the task.
- We will perform regular integration tests. This will (initially) ensure that the interfaces are at least being adhered to, and will eventually ensure that class libraries are being used correctly.
- The TAs must work closely with each team to minimize changes to the interfaces, and to minimize conceptual errors in how each interface is used.

Evaluation

Baseline (Must have)

- All building materials are located on the same plane as the hangar floor
- Robot is teleoperated around campus, and a map is built off-line using Carmen/Atlas-Moos/ DP-SLAM/etc and GPS.
- During teleoperation, building materials are detected using camera code from R:SS I, and map is annotated with building materials.
- Robot uses Carmen-based localization and Carmen-based navigation to drive to each site
- Robot reacts to intelligent bricks to plan brick collection and placement
- Robot uses R:SS I code to visually servo and pick up each brick
- Robot uses Carmen-based localization and navigation to return to hangar
- Robot uses R:SS I code to visually servo and place each brick in hangar

Extension I (Want to have)

- Building materials are located on different planes from the hangar floor (e.g., sidewalks)
- Robot performs exploration of campus, and a map is built off-line using Carmen/Atlas-Moos/ DP-SLAM/etc.
- Robot detects curb cuts and map is annotated with location
- During exploration, building materials are detected using camera code from R:SS I, and map is annotated with building materials.
- Robot uses new localization and new motion planner to drive to each site
- Robot incorporates knowledge of intelligent bricks in motion planning
- Robot uses R:SS I code to visually servo and pick up each brick
- Robot uses new localization and navigation to return to hangar
- Robot uses R:SS I code to visually servo in hangar
- Robot reacts to intelligent bricks and performs some degree of precise placement

Extension II (Nice to have)

- Robot performs online exploration and mapping
- Motion planner incorporates knowledge of wireless connectivity into motion plans
- Robot performs precise manipulation of bricks