**University of California, Santa Cruz**

**COSMOS Cluster 6:**

**Networking and Robotics**

**Final Project: Volcano Escape**

**(DRAFT)**

## Introduction:

On a seemingly ordinary Friday afternoon in Summer 2019, your robotics team receives a late-night call from the director of the National Emergency Management Administration. They believe they have detected a supervolcano in the heartland of America, and it’s about to blow, taking half of the continent with it. They have deposited an autonomous probe robot in the chamber of the volcano to take critical seismic data, but they have a problem: No one ever programmed the robot to exit the volcano.

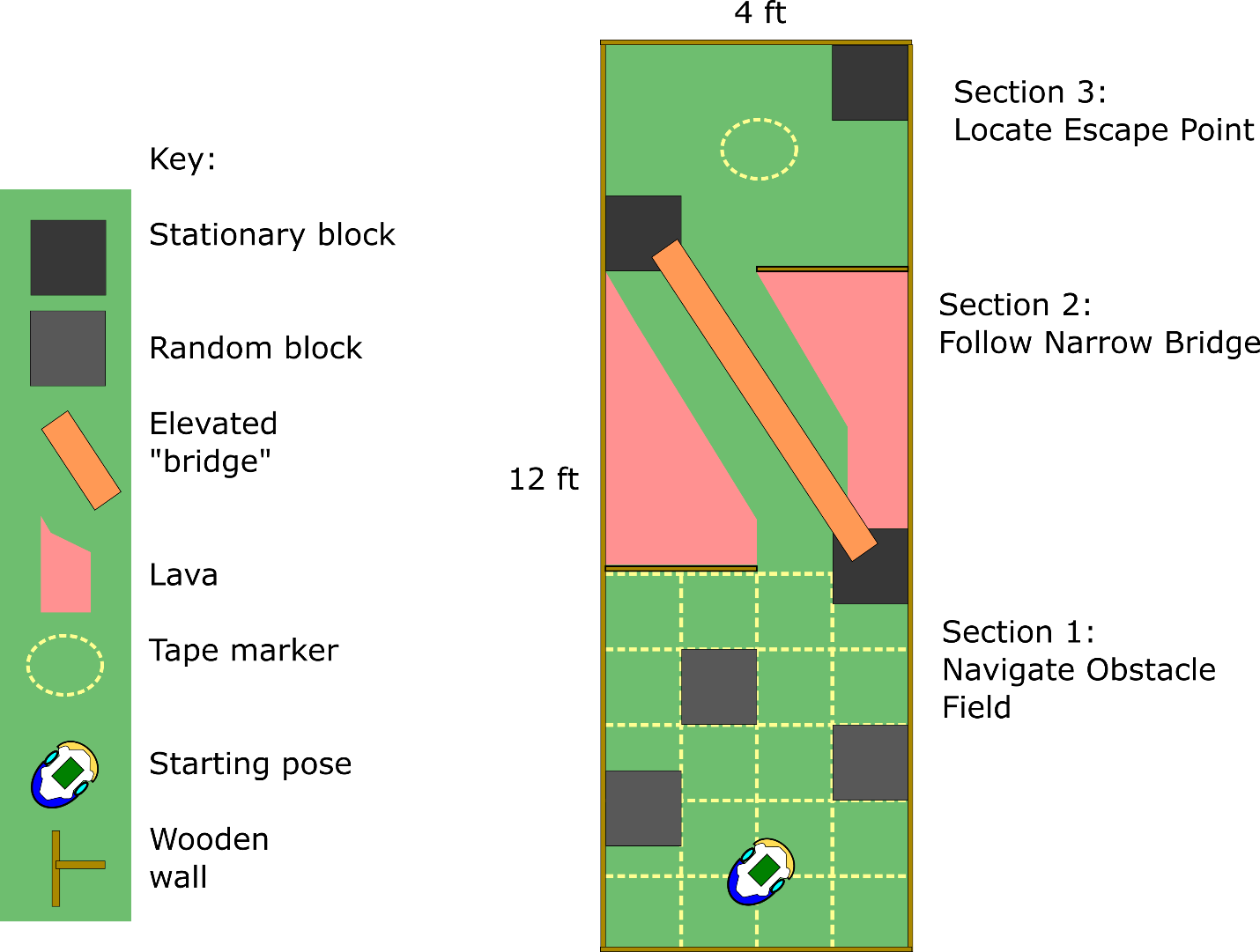
Now, the country depends on your skilled team of engineers. Your mission is to design an escape algorithm for the probe. Fortunately, the probe is very similar to a RoachBot, so you and your team can test your algorithms in the safe environment of Baskin Engineering 113.

## Outline

Your task is to program the roach robot to autonomously navigate an obstacle course and successfully reach its destination (the food source).

The obstacle course is a 4ft x 12ft field that consists of three main challenges. In the first challenge, your roach will have to bypass several roadblocks obstructing its path to the food source. These roadblocks will be placed on the field in a semi-random layout. Once the roach has successfully cleared the way, its next challenge will be to crawl its way through in the dark under a narrow bridge that will help shield it from pools of molten lava. But be careful! If the roach strays too far away from the path under the bridge, it can fall into the scorching lava that surrounds the area. When your roach emerges from under the bridge, its last challenge will be to orient itself in a way that enables it to drive to the extraction point (an area marked by tape in the middle of the field) and stop there.

In the first two sections, the roach will rely on its built-in bumper and light sensors. The final section is extremely challenging with dead reckoning, but is dramatically simplified by adding a rangefinder or IMU sensor.



## Project Structure:

The project will be completed in two phases. The first phase is a “small group” phase, in which groups of 2-5 students will design, implement, and test their own solutions to subsets of the full challenge. The second phase is a “large group” phase, in which larger groups will collectively decide how to integrate solutions from the first phase into a system that can complete the full challenge.

### Small Group phase:

For the small group phase, each student will use a Google form to indicate their preferences for their small group goal, and teammate preferences. Then, the staff will assign small groups. The categories for small group goals are as follows:

1. Section 1: Obstacle navigation state machine
   1. Students will design a state machine that can reliably exit section 1.
2. Section 2: Narrow bridge
   1. Students will design a state machine that can reliably exit section 2 without falling into lava.
3. Section 3: Locate escape point, rangefinder version
   1. Students will test and implement software to interface with one or two rangefinding sensors (infrared and ultrasonic) that can be added to the roach.
   2. Students will then implement a state machine that uses these sensors to reliably bring the roach to the extraction point in section 3.
4. Section 3: Locate escape point, IMU version
   1. Students will test and implement a service to perform communication with an Inertial Measurement Unit (IMU) sensor.
   2. Students will then implement a state machine that uses these sensors to reliably bring the roach to the extraction point in section 3.
5. Section 5: Improve ES framework
   1. Students will improve the ES framework by:
      1. Moving event checking systems to an interrupt-based system for improved responsiveness
      2. Implementing an event queuing system to leverage benefits of interrupt-based event checkers
      3. Any other improvements the students deem useful

This portion of the project will be allotted 4 days. Students are expected to carefully document their design process, including:

1. At least 3 distinct proposed initial state machines
   1. Each SM should include a plain-English description of overall strategy
   2. Each SM should be as different from the others as possible
2. Daily reports of progress
   1. Logged on student website
3. Methodical test reports
   1. At various points throughout the process, students should perform a sequence of at least 5 test runs, and record success rate and average time-to-completion.
   2. These reports should be logged on student website
4. A finalized state machine

During this phase, regular progress report meetings will be conducted with staff.

At the completion of this phase, each team will write a complete report of their design. This report could be a webpage, a poster, or a slideshow.

### Large Group Phase:

In this portion of the project, groups will be merged into groups of about 5 students, for a total of 4-5 groups. Each group will evaluate the results of other groups from the small group phase, and integrate them into a full roach hex file.

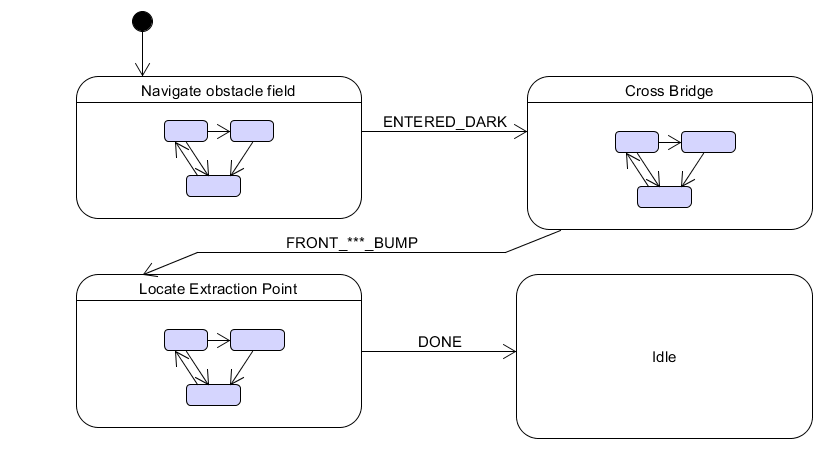
Each group will prepare a presentation for COSMOS Presentation Day (the final Friday of COSMOS).

## Project details:

In all cases, students are required to implement their work using an event-driven state machine following the provided events-and-services framework[[1]](#footnote-1).

These state machines will eventually be linked by a hierarchical state machine (HSM). A hierarchical state machine is a way of nesting a state machine inside the state(s) of another state machine. Events that are not “consumed” by a transition on the higher level SM are passed to the lower SM. This recursion does not necessarily have a depth limit, so you may utilize HSMs in your SMs if you like.

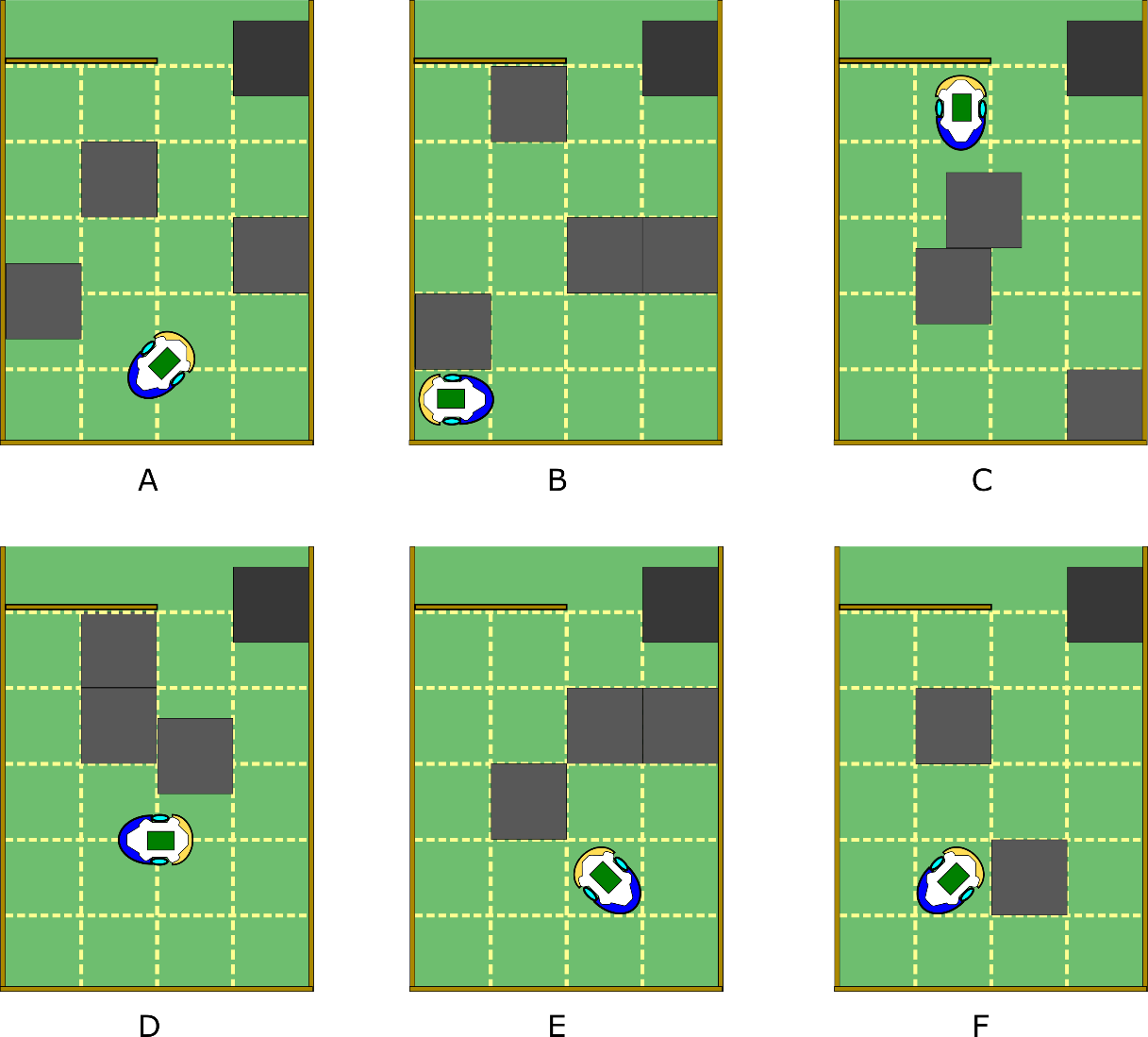
The top-level SM will look like this:



Below, we explain each of the project sections in more detail:

### Section 1: Obstacle Field

The obstacle field consists of a series of immovable square boulders that impede your egress. The boulders will be placed in one of the following configurations:



There is a time limit of 2 minutes for escaping the obstacle field.

Note that it is not necessarily required that your SM completes all 6 configurations – you may decided it is better to solve 4 of 6 configurations reliably.

### Section 2: Crossing Narrow Bridge:

Your robot must cross between two pools of “lava” by following a narrow bridge. The bridge is a piece of cardboard raised 6 inches off the ground, just enough for a roach to slip under. Then, your roach may pass underneath, using its light sensor to navigate. This is an inverted version of the common robotics task of line-following.

The bridge may take one of the following shapes:

A close up of a sign

Description automatically generated

Usually, line-following utilizes multiple sensors at different places around the robot. The roach has only one such sensor[[2]](#footnote-2), so more creative approaches are required. Note that the sensor is not at the center of the robot, so rotation about the robots sensor results in a “sweep” effect.

You may find it useful to add an event checker to the framework.

It is acceptable for the roach to touch the lava, but at no point should both wheels be in the lava.

### Section 3: Locating Extraction Point

You must bring your roach to an oblong space in the center of the final section, and bring all motors to a halt, and flash your LEDs to signal that your roach is ready for extraction.

This section requires precise localization (the process of finding your robot’s pose). By the time your roach has reached section 3, it is unlikely that its orientation will be known, so dead reckoning is unlikely to work. For this reason, a sensor must be added to the robot. Three sensors are available:

1. An HC-SR04 ultrasonic rangefinder:
   1. <https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf>
   2. This sensor uses echolocation to determine the distance to an object.
2. A Sharp 2YA02 infrared rangefinder:
   1. <https://www.sparkfun.com/datasheets/Sensors/Infrared/gp2y0a02yk_e.pdf>
   2. This sensor uses a beam of infrared light to determine the distance to an object.
3. An Adafruit LSM9DS1 9-DOF Accel/Mag/Gyro Breakout Board:
   1. <https://cdn-learn.adafruit.com/downloads/pdf/adafruit-lsm9ds1-accelerometer-plus-gyro-plus-magnetometer-9-dof-breakout.pdf>
   2. This sensor contains an accelerometer, gyro, and magnetometer. By monitoring the gyro constantly, it is possible to maintain an estimate of the Roach’s attitude (ie angle)

Since the extraction point is in the exact center of Section 3, rangefinding allows for a (relatively) simple state machine. If a gyro monitoring service is implemented, then section 3 can be solved using partial dead reckoning.

For each sensor, the students will read the sensor’s datasheet with assistance from staff to understand what code is required to communicate with the sensor. Additionally, students will read datasheets from the Pic32 to write services that perform communication.

This section is easily the most challenging, but will also yield a new level of understanding into embedded systems and electronics for students who undertake it.

## A final note:

It is possible that these challenges are too easy or too hard for our class. If this is the case, we can modify the challenges to reduce or increase the difficulty.

Similarly, if you have suggestions for ways to improve this challenge, please notify the staff. We can work with you to produce a more fun or more edifying challenge!

1. This will be almost identical to the Lab4 framework, but with added tools for debugging and a slightly improved event checking system. [↑](#footnote-ref-1)
2. Adding a sensor is a possibility. Talk to Max. [↑](#footnote-ref-2)