Department of Electrical and Computer Engineering
Course ECSE 211 – Design Principles and Methods
Fall 2017 Project Description
Revision 1.2, Final, November 23, 2017

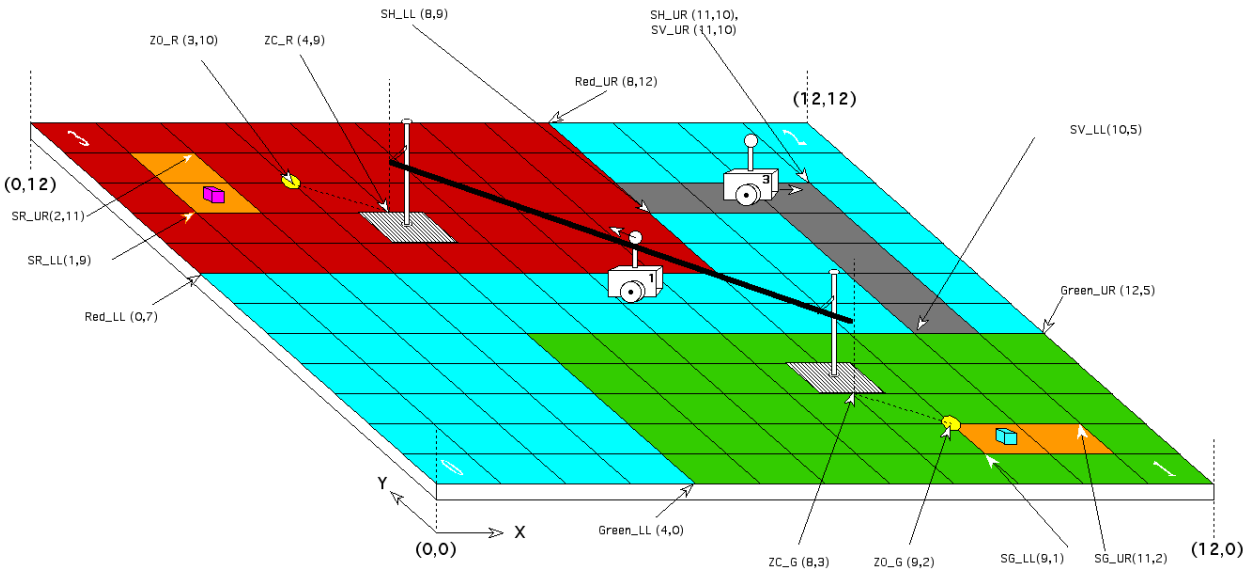

Figure 1

Overview

Note: In response to the results of Lab 5 and Q&A from the first week of meetings, project specifications will be incrementally modified to include any clarifications.

The goal of this project is to construct an autonomous machine that can play a one-on-one version of the game Capture the Flag. Consider the scenario depicted above in Figure 1, with two players labeled 1 and 3. The labels indicate the corners each machine started in, so Player 1 starts in Corner 1 (Green Zone) and Player 3 in Corner 3 (Red Zone). Each of the zones is surrounded by a virtual river (blue regions), with two methods of transiting from one zone to another – using the overhead zip line or rolling through shallow water (the gray region shown in the figure). Each zone corresponds to a rectangular region defined by its lower left (LL) and upper right (UR) corners relative to the origin. In the example shown in Figure 1, the red zone is defined as Red_LL (0,7) to Red_UR (8,12), and the green zone is defined as Green_LL (4,0) to Green_UR (12,5).

The playing field measures 12’ x 12’, with the origin located in the lower left hand corner, (0,0), as shown in Figure 1. At the start of a round, both players are placed in their respective corners at a random orientation and started. Each player waits for a set of game parameters to be downloaded from the game server (more about this later). Once the parameters are received (which describe the layout of the laying field), each player must cross the river over to the opponent side. The green zone player always transits using the zip line and returns over water; the red zone player transits over water and returns using the zip line (this is done to minimize the

possibility of collisions). For the green zone player, this amounts to a repeat of Lab 5. Here you are given the coordinates of two points: 1) ZC_G (8,3) corresponding to the projection of the zip line endpoint onto the grid, and 2) ZO_G (9,2) a second point that taken together with ZC_G, forms a vector that aligns with the zip line. Thus the robot aligns itself with the zip line by navigating to (9,2) and rotating so that it points to (8,3). The red zone player is given the location of a shallow path across the river in terms of the union of a vertical and horizontal box. As shown in the figure, the horizontal segment is defined by its lower left, SH_LL (8,9), and upper right, SH_UR (11,10) coordinates, and the vertical segment by its lower left, SV_LL (10,5), and upper right, SV_UR (11,10) coordinates respectively. You will need to devise an appropriate method to generate a path based on these parameters. *Note: the convention used here is that LL always corresponds to the point with the lower (X,Y) values.* Once the initial transit is completed, each robot must search for its opponent's flag, which consists of a colored Styrofoam block. Blocks will come in multiple colors; each color will be assigned a specific integer value in the range [0,5]. These will be made available to you at least 2 weeks before the competition. You will have to devise an appropriate method (e.g. using your color sensor) to figure out how to identify each of the blocks in the set. Thus in addition to the layout parameters shown in Figure 1, two parameters will be used to signify the color of the opponent's flag. Referring again to Figure 1, OG=3 (violet) and OR=4 (blue), meaning that the green zone player will look for a violet colored block, and the red zone player for a blue block.
To narrow down the search area, each player will be given the coordinates of a zone containing the opponent's block, e.g., for the red zone SR_LL (1,9), SR_UR (2,11), and SG_LL (9,1), SG_UR (11,2) for the green zone as shown in Figure 1. Note that there may be more than one block present in the search area, but only one will have a color matching OG or OR. Once the correct block is found, the robot must indicate a capture by beeping 3 times – it is not necessary to retrieve the block (e.g. picking it up). After completing the capture, each robot must return to their starting corner (and stop) using the opposite mode of transversal, e.g., the green zone returns via shallow river crossing and the red zone robot using the zip line.

The "winner" is the first player that makes it back to the start. In order to be successful, a player machine needs to do the following:

1. Receive parameters from the game controller. (You will be provided with an appropriate Java class to do this):
2. Localize – given the starting corner number, move to closest grid intersection and note initial position and alignment. This must be completed within 30 seconds.
3. Navigate to the ramp corresponding to the starting zone.
4. Traverse the river using the zip line or shallow crossing.
5. Search for opponent flag.
6. Indicate capture.
7. Navigate back to the start using the appropriate transit method.
8. Stop

Parameters

Game play is determined by a set of parameters, which are sent to the client (player) from a server. The following parameters are defined according to the details provided in Figure 1:

RedTeam (i=1,20) – Team starting out from red zone
GreenTeam (i=1,20) – Team starting out from green zone
RedCorner (i=1,4) – Starting corner for red team
GreenCorner (i=1,4) – Starting corner for green team
OG (i=1,5) – color of green opponent flag
OR (i=1,5) – color of red opponent flag
Red_LL (x,y) – lower left hand corner of Red Zone
Red_UR (x,y) – upper right hand corner of Red Zone
Green_LL (x,y) – lower left hand corner of Green Zone
Green_UR (x,y) – upper right hand corner of Green Zone
ZC_R (x,y) – end point corresponding to zip line in Red Zone
ZO_R (x,y) – end point together with ZC_R indicates direction of zip line
ZC_G (x,y) – end point corresponding to zip line in Green Zone
ZO_G (x,y) – end point together with ZC_G indicates direction of zip line
SH_LL (x,y) – lower left hand corner of horizontal shallow water zone
SH_UR (x,y) – upper right hand corner of horizontal shallow water zone
SV_LL (x,y) – lower left hand corner of vertical shallow water zone
SV_UR (x,y) – upper right hand corner of vertical shallow water zone
SR_LL (x,y) – lower left hand corner of search region in red player zone
SR_UR (x,y) – upper right hand corner of search region in red player zone
SG_LL (x,y) – lower left hand corner of search region in green player zone
SG_UR (x,y) – upper right hand corner of search region in green player zone

Note that the (x,y) coordinates listed correspond to the grid coordinates shown in the Figure 1. The zip line will be the same on used in Lab 5, with parameters as indicated above. The zip line can only be approached from one direction, from ZO to ZC (think of these points as the endpoints of a vector indicating the approach direction).

Game Play

Both players act almost independently, so the design can focus mainly on navigation, mobility, search and retrieval. Some collision avoidance will be necessary in the event that both players are in the same vicinity. Each team will participate in 4 rounds for which a cumulative score will be determined. The score is based on points awarded for exhibiting each of the behaviors required to play the game: localization, navigation, traversing the river using the zip line and the river crossing, searching for the flag, indicating capture, and returning to the starting corner. These points effectively validate the components of your design. On top of this we also record how long it takes for you to get across and find the flag, as well as the return trip. These figure prominently in ranking the performance of the teams with respect to the "competition".

Materials

Each team has up to 3 Lego Mindstorms kits worth of parts available. In addition, a MakerBot Replicator 2 rapid prototyping machine is available for fabricating parts for those inclined. You may also purchase additional materials, but these must receive prior approval from the

instructors. Another note – all computation must be done on board the EV3 brick(s); no offloading to an external machine is permitted.

Final Notes

Consider this to be a *preliminary* document that will evolve throughout the semester until the specifications are frozen.

Addendum

- Figure 1 is *not literal*. It is meant only to provide meaning for the parameters listed. The sizes and shapes of regions may differ considerably from what is shown in the figure.
- A robot traversing the zip line must not come into contact with the ground.
- Only materials in the Mindstorms kits can be used unless permission is obtained from the instructors.
- Once the colors have been determined for the flags, samples will be left in the lab with integer numbers specifying color. You will need to devise a method for associating each color with its defining integer code.
- Teams will be told their colors at the start of each round.
- The water path has constant width.
- Glue can be used to secure parts, but it must be of the removable variety.
- The opponent's flag cannot be moved more than a linear distance corresponding to the side of one square.
- The zip line is unidirectional (in case this is not clear from the specifications).
- External software permitted? Only with approval.

Final Considerations

You are being evaluated on your robot's ability to complete the various tasks that make up the game, so it is important to show what your machine can do. For example, if you are unable to perform the block search, skip it and proceed to the return leg. Unless your machine does nothing at all, you will be awarded points for what it can do.

dal/fpf

V1.0 October 11, 2017
V1.1 October 22, 2017
V1.2F November 23, 2017