ECSE211 Final Presentation

10 minutes

Topics

1. Introduction: 1:45
   * General intro: 0:30 **(Bryan)**
   * Objective: 0:15 **(Bryan)**
   * Specifications: 0:30 **(Patrick)**
   * Requirements: 0:30 **(Patrick)**

**Cumulative: 1:45**

1. Varied: 0:45/1:00
   * Design process: 0:30/0:45 **(Luka)**
   * Tools used: 0:15 **(Luka)**

**Cumulative: 2:30/2:45**

1. Hardware: 1:30/2:00
   * Intro to hardware: 0:30 **(Enan)**
   * Landing Gear Feature: 0:30 **(Enan)**
   * Other features: 0:45/1:00 **(Enan)**

**Cumulative: 4:15/4:45**

1. Software: 1:45/2:00
   * Requirements: 0:30 **(Patrick)**
   * Use of code from previous lab: 0:15 **(Volen)**
   * Software logic: 1:00/1:15 **(Volen)**

**Cumulative: 6:00/6:45**

1. Testing: 1:15/1:30
   * Testing phases: 0:45/1:00 **(Tianyi)**
   * Test results: 0:30 **(Tianyi)**

**Cumulative: 7:15/8:15**

1. Conclusion: 1:15/1:45
   * Challenges faced: 0:45/1:00 **(Bryan)**
   * What we learnt: 0:30/0:45 **(Luka)**

**Cumulative: 8:30/10:00**

Introduction

1. General intro

* Team 11
* DPM Winter 2018
* Designed and built a robot to play Capture the Flag

1. Objective

* Design robot with following aspects:
* Fully autonomous
* Can navigate and localize itself
* Can avoid obstacles
* Identify flags (i.e. colored blocks)
* Return to the starting area after identifying the target flag

1. Specifications

* Playing field of 12\*12 feet, with origin in LL corner
* Part of the grid is a river and can’t be crossed by the robot
* Use tunnel or bridge to cross river
* Game parameters received through a Wi-Fi server (they include starting corner, localization of search area, river, etc).

1. Requirements

* Sequence of events:
* Receive Wi-Fi data
* Localize within the corner in under 30 seconds
* Navigate to tunnel/bridge and cross
* Search for opponent flag and indicate capture
* Navigate back to bridge/tunnel and cross
* Go back to starting corner

Varied

1. Design process

* Four major steps:
* Research
* Building process
* Testing and implementation
* Optimization and implementation

1. Tools used

* Variety of tools, covering our needs for every field:
* Hardware: Photoshop/InDesign for design, Lego kits for building
* Software: GitHub for code management, eclipse for development
* Documentation: Microsoft Office, Gantt software, Google Drive

Hardware

1. Intro to Hardware

* Started with a Lego Mindstorms Kit, which contains:
  + Lego pieces to build
  + Programmable brick to run the software
  + Motors to move
  + Several sensors to detect and analyze the robot’s environment
* Several designs were built before finalizing this one:
  + One with a single light sensor for odometry correction
  + One with the block-detecting light sensor in the front instead of above
  + One with four wheels, etc.

1. Landing Gear Feature

* Main hardware challenge: build robot that can cross both tunnel and bridge. Difficulty: wheelbase that can traverse the tunnel is too large for the bridge.
* Idea: landing gear that deploys when necessary

1. Other features

* Fenders in front of each wheel to realign ourselves if we hit a wall in the tunnel
* US sensor at the front, attached to small motor so it can rotate
* One light sensor behind each wheel (not in front so that the US sensor can rotate)
* One light sensor above (at fixed height) to detect the color of a block

Software

1. Requirements

* Write software in Java
* Use the leJOS firmware
* Keep the code as portable as possible due to low processing power
* No offloading processes outside of the brick

1. Use of code from previous lab

* Labs before project allowed us to develop software for each key operation:
  + Obstacle detection
  + Odometry
  + Navigation
  + Localization
  + Search and Localize

1. Software logic

* Controller. Main class. Creates objects and dictates the overall robot behavior.
* WiFiData. Receives important data (relative to the grid layout, the team numbers…) and transmits them to the Controller.
* Robot. Contains robot constants (track, forward speed…) and methods for angle/distance conversion to motor rotations.
* USLocalizer. Allows the robot to determine its initial orientation, but not its position.
* LightLocalizer. Allows the robot to determine its position and orientation, using the black grid lines on the board.
* Odometer. Assuming the robot already knows its position, allows it to know where it is when on the move.
* Navigation. All movement-related behavior. Go forwards, backwards, travel a certain distance, etc.
* SearchAndLocalize. Circle around the search area and attempt to find the target block.
* ColourCalibration. Identify the colour that the light sensor is detecting, and check whether it is the target.

Testing

1. Testing phases

* Hardware components
  + US and Light sensors – check that they correctly detect what we want
  + Wheel and Track optimization
  + Landing gear – check that it correctly deploys and closes
* Software components
  + Localization – check that <30 seconds, and accuracy.
  + Navigation – check that it goes straight and doesn’t shift.
  + Search – check that the robot can find a block
  + Odometer – check that the distance measurements are accurate.
* System integration
  + WiFi integration – check that the WiFiData class properly receives the data
  + Complete system component integration – checking the state of the overall system.

1. Test results

* By testing the sensors, we were able to determine their accuracy levels and the appropriate distance for data reading, placing them on the robot accordingly.
* We had some difficulty with Navigation, the robot would often slightly shift left or right when going straight. This issue is fixed however by performing LightLocalization more often.
* The Odometer exhibits an excellent accuracy.
* The Landing Gear allows for traversal stability and superior weight distribution.

Conclusion

1. Challenges faced
2. What we learnt