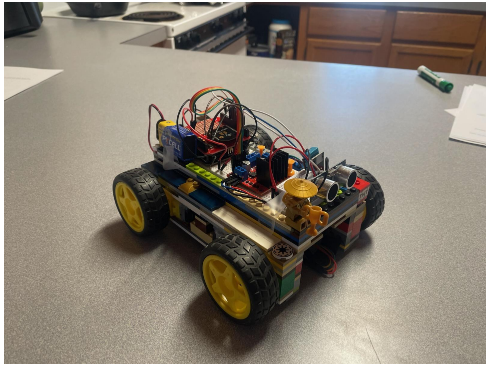
Finding the Coldest spot in Rady

Final Project for CS380 - Internet of Things
Jared Borowitz, Nathan Bell



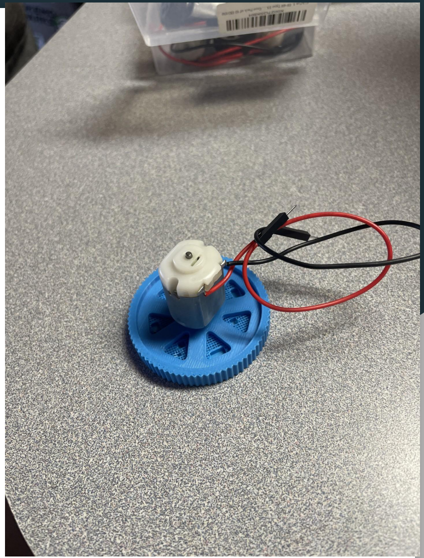
Problem

The Rady Building on the Western Colorado University campus is notorious for having super-inconsistent temperatures. Even though it is a brand new building, their fancy thermo heating and cooling have failed

Solution

We decided to build an autonomous robot that can navigate around the Rady building and monitor temperatures throughout it. With this tracking program, we can ideally determine the spot on the robot's map with the lowest temperature.

Initial Ideas

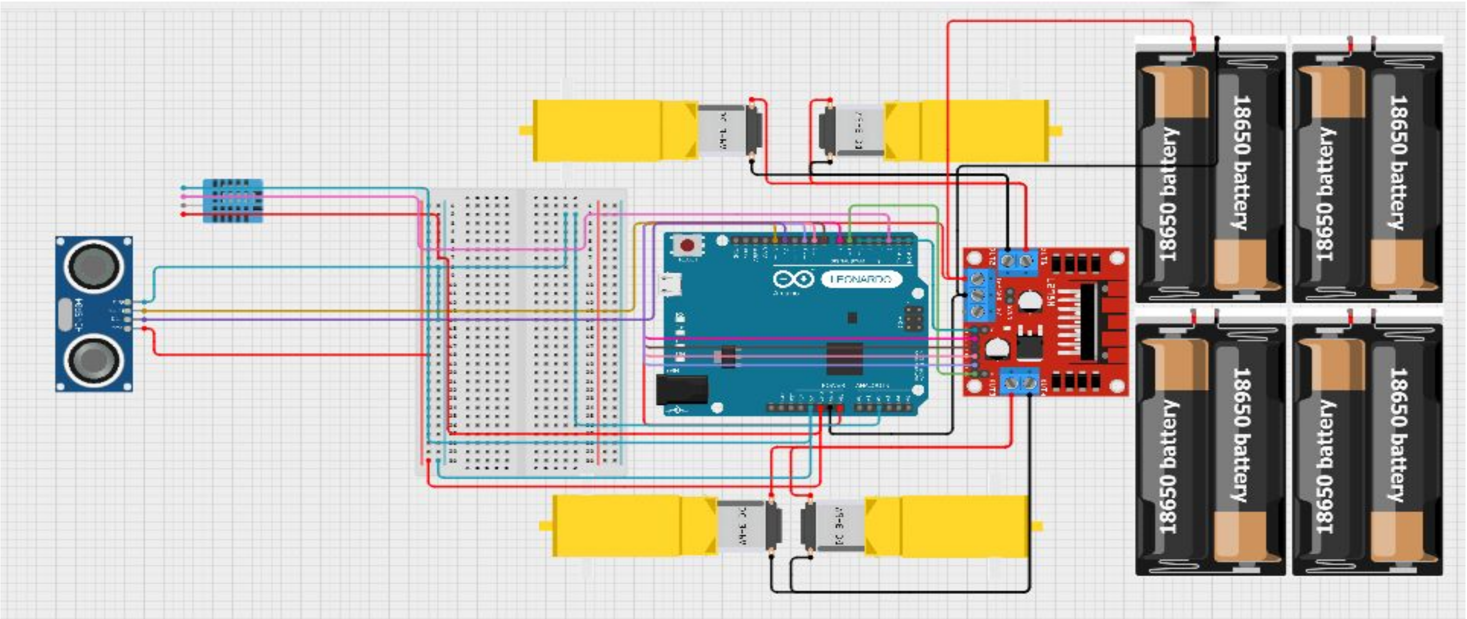


- Use DC Motors and 3D printed wheels
- Use a gyroscope to measure our turning angles
- Map any given space in the Rady building using TKinter
- Power with a single 4xAA battery pack ~6V

Revisions

- Replace motors and wheels with L298N driver kit and wheels
  - to achieve more dynamic driving
  - including PWM
- Add an additional inline Battery Pack (now ~12V)
- Remove the gyroscope
  - for reason of simplicity

Parts Used



- 2x 4-AA Battery pack
- 4x DC Motors
- 1x L298N DC motor drive
- 1x DHT11 Humidity and Temperature Sensor
- 1x HC-SR04 Ultrasonic Sensor
- 1x Arduino Uno
- 1x xBee Shield
- 2x xBee radio

Finite State Machine

Main States:

INIT: basic initializer

DRIVE: triggers autonomous driving

DIST: checks for an obstacle*

TEMP: reads temperature data*

STOP: manual stop

DESC: holding state while in DESC

*used for testing purposes

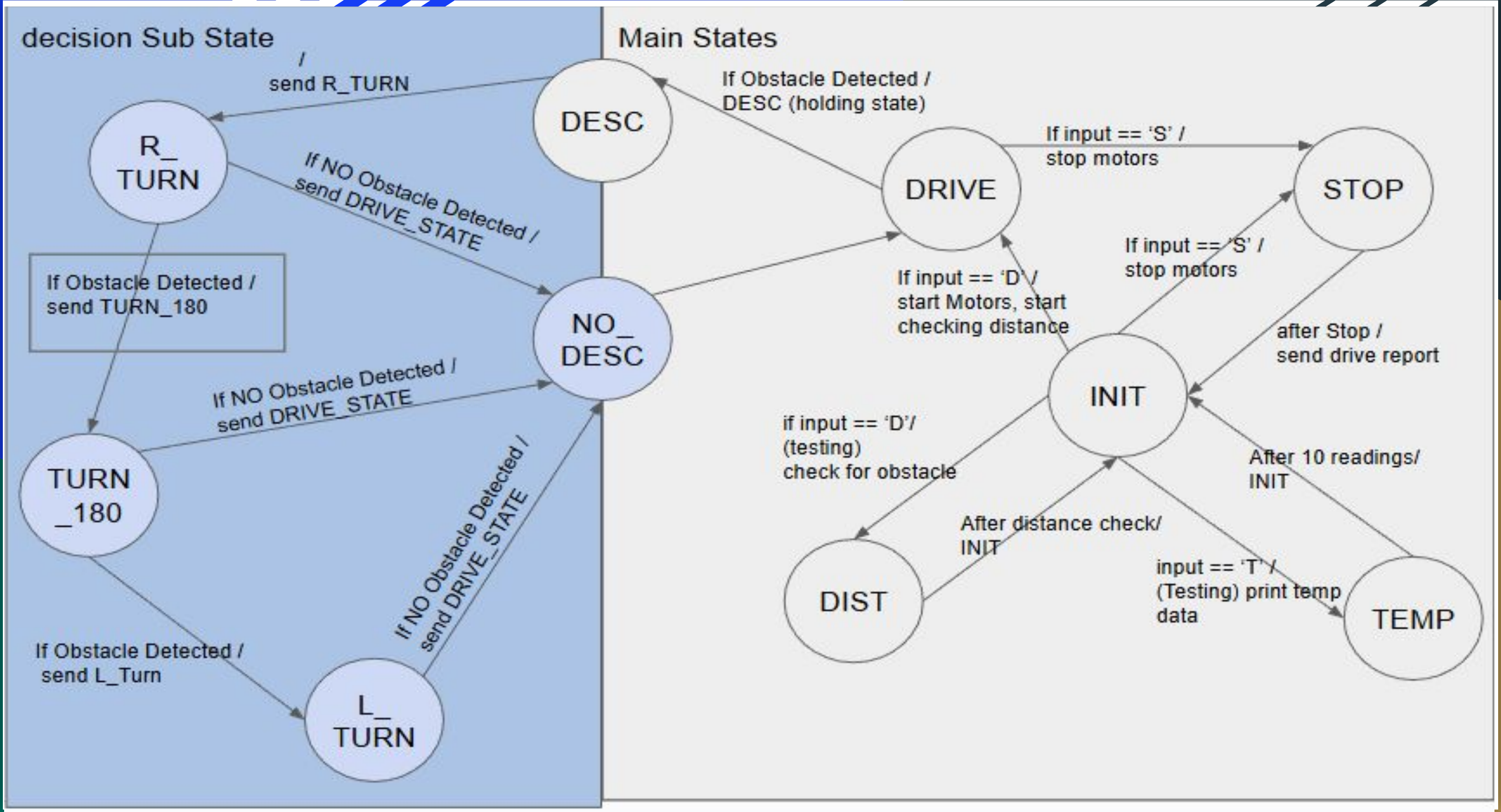
Decision States:

R_TURN: turn & check for obstacles

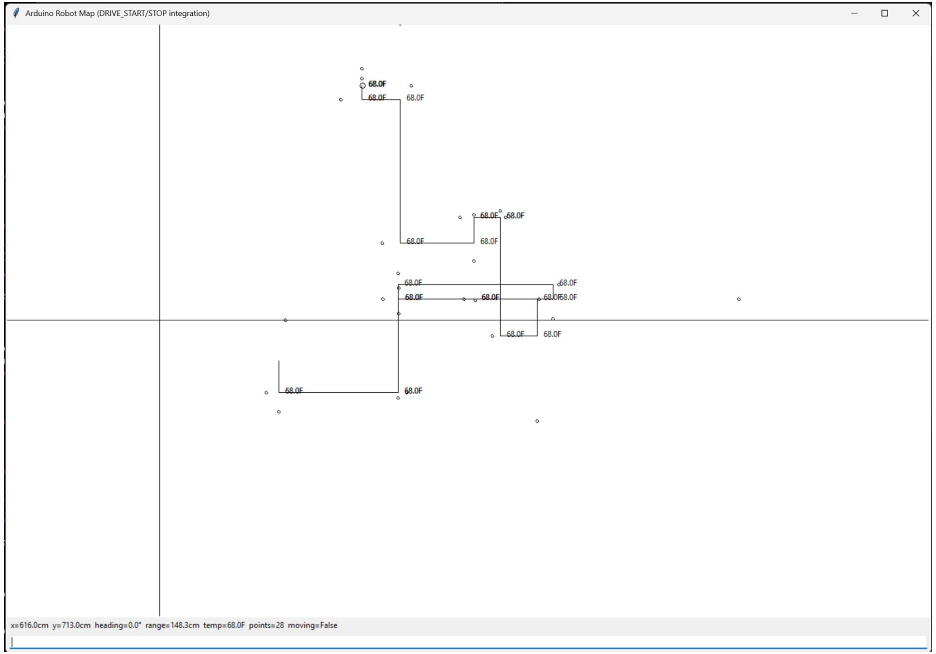
L_TURN:turn & check for obstacles

TURN_180:turn & check for obstacles

NO_DESC: return to DRIVE state


decision Sub State
Main States
/
send R_TURN
DESC
If Obstacle Detected /
DESC (holding state)
R_
TURN
If NO Obstacle Detected /
send DRIVE_STATE
DRIVE
If input == 'S' /
stop motors
STOP
If Obstacle Detected /
send TURN_180
NO_
DESC
If input == 'D' /
start Motors, start
checking distance
If input == 'S' /
stop motors
after Stop /
send drive report
If NO Obstacle Detected /
send DRIVE_STATE
INIT
TURN
_180
if input == 'D'/
(testing)
check for obstacle
After 10 readings/
INIT
If NO Obstacle Detected /
send DRIVE_STATE
After distance check/
INIT
input == 'T' /
(Testing) print temp
data
DIST
TEMP
If Obstacle Detected /
send L_Turn
L_
TURN

TKinter and Mapping



1. Our tkinter program will show the robot moving with a line behind it
2. Once the robot starts moving, it starts mapping
3. When it hits an obstacle, we mark the obstacle on the map along with our point and its temperature.
4. Based on what happens in our state machine, it will map turns in python
5. Not perfect because we are assuming that the turns are perfect 90-degree turns
6. If it gets stuck, there might be some inaccurate mapping data

End Product and Known Issues

Inconsistent Driving data

- Due to a number issues (Cheap Wheels, Current spikes, Heavy weight, limited “vision”)
- Driving data was inconsistent

Because Driving data was inconsistent

- Mapping a space became unrealistic
- The vehicle was not able to replicate anything consistently

Therefore

- A substantial number of upgrades and changes would be required
- However, the “decision making” of the vehicle proved to work according to plan

Conclusion

In order to effective achieve our goal, the vehicle would require a significant revision.

Many aspect of the project would need to be redesigned including:

- upgraded Wheels and Motors
- an additional motor driver
- a Lithium battery Pack
- additional ultrasonic sensors
- integrate a gyroscope/accelerometer
- Rescaling the generate map in real time
- Redesigning the frame to reduce weight

Demo

Includes a demonstration of:

- robot navigating its way through halls
- Switching through different states
- Mapping all movement as well as location of obstacles and temperature data