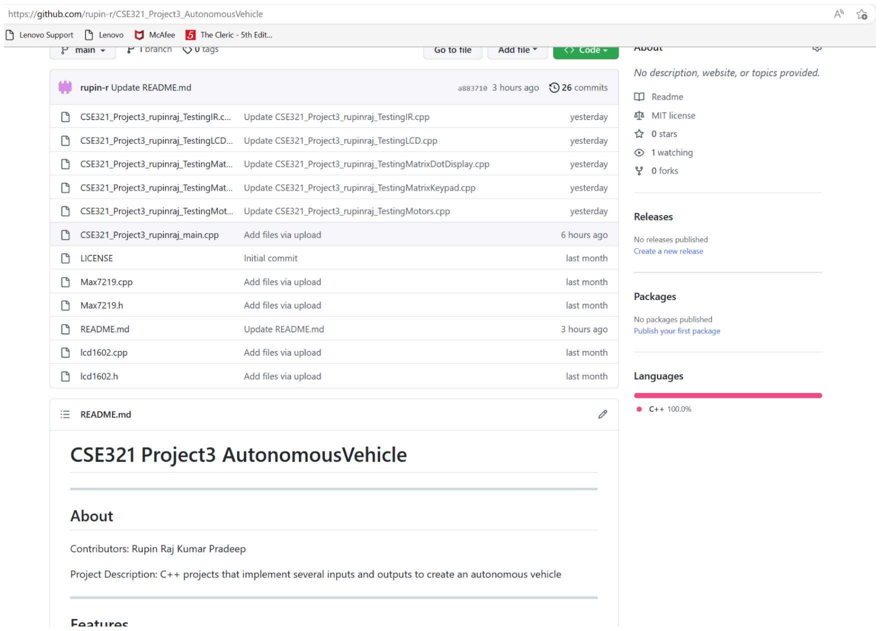
Project 3 - Autonomous Vehicle

Rupin Raj Kumar Pradeep

Table of Contents

Code Repository Screenshot


https://github.com/rupin-r/CSE321_Project3_AutonomousVehicle

rupin-r Update README.md a883710 3 hours ago 26 commits

CSE321_Project3_rupinraj_TestingIR.c...	Update CSE321_Project3_rupinraj_TestingIR.cpp	yesterday
CSE321_Project3_rupinraj_TestingLCD...	Update CSE321_Project3_rupinraj_TestingLCD.cpp	yesterday
CSE321_Project3_rupinraj_TestingMat...	Update CSE321_Project3_rupinraj_TestingMatrixDotDisplay.cpp	yesterday
CSE321_Project3_rupinraj_TestingMat...	Update CSE321_Project3_rupinraj_TestingMatrixKeypad.cpp	yesterday
CSE321_Project3_rupinraj_TestingMot...	Update CSE321_Project3_rupinraj_TestingMotors.cpp	yesterday
CSE321_Project3_rupinraj_main.cpp	Add files via upload	6 hours ago
LICENSE	Initial commit	last month
Max7219.cpp	Add files via upload	last month
Max7219.h	Add files via upload	last month
README.md	Update README.md	3 hours ago
lcd1602.cpp	Add files via upload	last month
lcd1602.h	Add files via upload	last month

README.md

CSE321 Project3 AutonomousVehicle

About

Contributors: Rupin Raj Kumar Pradeep

Project Description: C++ projects that implement several inputs and outputs to create an autonomous vehicle

Features

About

No description, website, or topics provided.

Readme
MIT license
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

● C++ 100.0%

Revision History:

11/11/22: Created the document repository and code repository. Completed the problem statement questions in a separate document and added the link in this document. Added a simple BOM to the document and a link to the excel sheet with more information. Added an index with keywords and definitions. Several of the information links point to lecture slides as they are relevant to the project in terms of what is expected. Added an initial plan for the solution to the problem in the form of a flowchart.

11/11/22: Added to the index and also copied it into a separate file for submission

11/18/22: Completed testing of the matrix keypad, LCD display, and IR sensor. Started on the matrix dot display, but didn't finish. Wrote a memo detailing the current state of the project. Also edited the ReadMe to show updates to the current code progress.

11/20/22: Completed the matrix dot display API and tested it.

11/21/22: Wrote comments into the main file, rewrote the readme, and rewrote the memo.

12/7/22: Started writing code into the main file and implementing BFS using custom made data structures. However, after testing, it was found that most likely the pointers were not being stored properly because malloc was not used and data was disappearing during the execution of BFS. As a result, the path became empty and the vehicle would not run.

12/8/22: After seeing this issue, I got rid of all the custom made data structures as well as the matrix dot display because there was not enough time to implement an extra peripheral on top of the four existing ones. Instead, I implemented BFS using exclusively arrays and had to shorten the range of the movement to a 5x5 square placed NE of the vehicle. This can be made bigger simply using a bigger array and changing loop indices.

12/9/22: The array implementation was successful, but stopping the eventqueue posed a problem where it killed execution of the entire program as well and would not stop the motors. Took a while to figure out I should just stop the motors beforehand. With this fix, all the peripherals worked smoothly and as expected for the demo.

12/15/22: Added files to the github repository and started editing the google doc repository to update the status of the project.

Problem Statement:

Objective: Create an autonomous device that will be able to find a path through obstacles to a final destination.

Solution: Create a microcontroller controlled vehicle using sensory peripherals to traverse through obstacles in a 2D path.

Specific Implementation:
Coordinates entered into the microcontroller through a matrix keypad will be set relative to the size and location of the vehicle. IR sensors above the vehicle will detect obstacles on the path to the end destination and record them into a graph. Pathfinding through a 2D graph, each space in the graph the size of the vehicle, will be used to create a path between the vehicle location and end location. While following the path, distance covered will be controlled by stepper motors following a time based interrupt system. Any input received by the IR sensor detecting an obstacle will trigger an interrupt to adjust the path if necessary.

For more information including constraints to the project:
https://docs.google.com/document/d/1d84lQx9ppzSkupiB4gvuJ4GgIhl5HucIsDBKiIIQB9I/edit?usp=sharing

Index:

Keyword	Definition	Purpose	Link	Page	Relevance
Interrupt	A hardware triggered action raised by a flag.	Allow tasks of higher priority to run before other tasks complete.	Lecture 16 and past code	26, 28, 35	An interrupt is required for any IR obstacle detection.
Priority	Importance of a task relative to others in regards to timing.	Controls the order in which interrupts routines are called.	Lecture 28	Last slide	Priority will be given to IR obstacle detection over driving the vehicle
Thread	A piece of code meant to be run simultaneously rather than sequentially.	Have multiple tasks operate at the same time/	Lecture 28	7, 8, 9	Threads will be run to edit the path while the vehicle is driving

Critical Section	When two threads attempt to access the same memory state and one of them tries to change it.	This section must be protected for normal project behavior.	Lecture 28	7, 8	There will exist a critical section when one thread is editing the path the vehicle is driving on
EventQueue	A queue to hold threads to run sequentially.	After one thread finishes, the next begins. When used with threads that access the same memory state, it protects the critical section.	Lecture 25	31 to end	An EventQueue will be used to hold the path to drive on. This queue will be edited by the interrupt routine.
Spin Polling	Constantly checking for inputs until one is received. Then continue checking.	Check for input that will be received more than once	Lecture 15 and past code	29	Spin polling will be used for initial user input into the matrix keypad
Periodic Interrupts	Creating an interrupt after a certain amount of time has elapsed	Control the behavior of threads and put in delays	Lecture 31	6, 7, 8	Periodic interrupts will be necessary to stop each driving thread so that only a certain distance will be covered. This will have the highest priority.
Watchdog timer	A timer used in applications that are typically self-reliant.	This timer ensures that tasks are being completed in a timely manner and can alter the state if this time constraint is not met.	Watchdog Timer	All	Since this system will be self-reliant, properly setting the watchdog timer will be a requirement for the system to fix itself.
GPIO	Stands for general purpose input and output.	The GPIO is necessary for controlling peripherals connected to the nucleo.	Lecture 8 and past code	30 to 43	GPIO configuration will be necessary for all the connected components in the system.

Matrix Keypad	A 4x4 collection of push buttons operated one row at a time for 4 inputs and 4 outputs.	Allows for user input in the form of numbers and some characters.	Lecture 13 and past code	26 to end	Necessary for user input of end coordinates.
LCD Display	A liquid crystal display where light is emitted through polarizers to display characters and other outputs.	Allows for a user to see a result after entering input similar to a computer monitor.	LCD library files and past code	All pages	Used to show user input from the matrix keypad as well as any errors while driving.
Matrix Dot Display	An 8x8 array of LEDs where only one column is operated at a time, switching columns at a frequency that can't be perceived by the naked eye.	Allows for output of graphical data in a simple form.	API	Sterne gugge r post	Shows the visible obstacles perceived by the vehicle while driving.
IR sensor	A sensor that emits infrared wavelengths and waits for a return signal. If the return signal is received within a certain amount of time, the output is high, otherwise it is low	Used to detect obstacles within a certain distance of the IR sensor.	Arduin o API	All of page	Spinning several IR sensors of varying adjusted distances can simulate a LIDAR that can detect and avoid obstacles over a wide range for pathfinding.
Pinout	The pin names and options corresponding to each pin found on a piece of hardware.	The connections between pins require certain inputs and outputs determined by the pinout.	Nucleo pinout	All of page	The nucleo pinout is needed to properly code each peripheral based on what pin they are connected to.

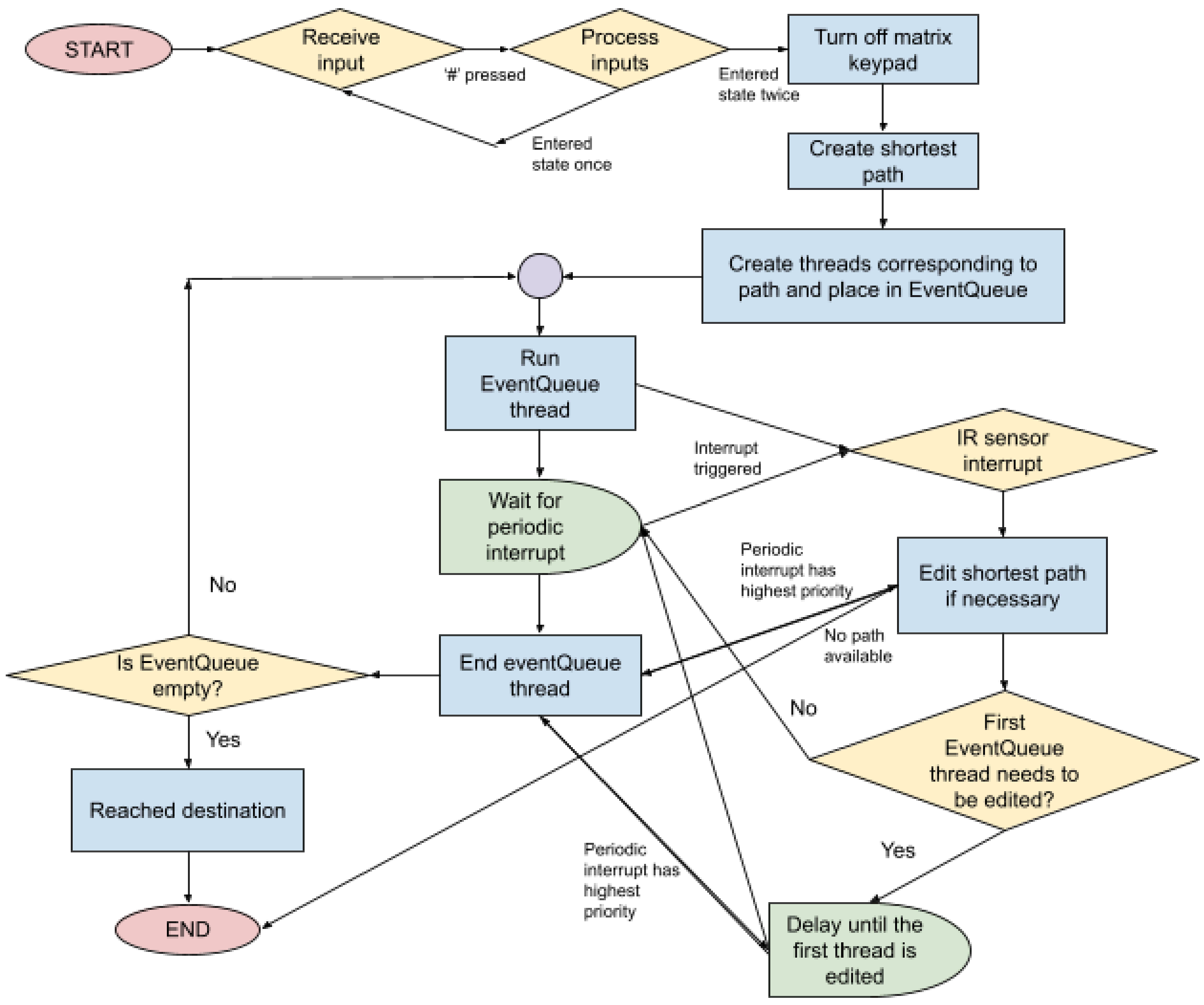
https://docs.google.com/document/d/1y8ZSst7AWTpWW7ad4y_Lp-uKkEExppT9ptRTFbXSdH8/edit?usp=sharing

BOM:

1 Nucleo L4R5ZI
1 Breadboard
1 Matrix Keypad
1 LCD Display model 1602
1 Matrix Dot Display (not necessary)
1 IR sensor
3 servo motors with screws
6 smaller screws for holding servos in place
1 cross piece for servo
2 wheels for servo
14 male to male wires
7 female to male wires
1 roll of strong tape
A small cardboard box or spare cardboard as a base of the vehicle

BOM with more details
https://docs.google.com/spreadsheets/d/1JYWmLaQPkxI_QKjCSPqfQer58fhQvpJXz8bf3X0cGKA/edit?usp=sharing

Algorithm/Plan:


START
Receive input
'#' pressed
Process inputs
Entered state once
Entered state twice
Turn off matrix keypad
Create shortest path
Create threads corresponding to path and place in EventQueue
Run EventQueue thread
Interrupt triggered
IR sensor interrupt
Wait for periodic interrupt
Periodic interrupt has highest priority
Edit shortest path if necessary
No path available
No
Is EventQueue empty?
End eventQueue thread
No
First EventQueue thread needs to be edited?
Yes
Yes
Reached destination
Periodic interrupt has highest priority
Delay until the first thread is edited
END

Code Algorithm:

Using list format to contrast flow chart above

```
Set up input and output global variables
Set up interrupts
Set up necessary events, eventQueue, and thread
Set up intermediate variables such as flags or positional variables
Set up intermediate variables for BFS
Enter main method
Turn on matrix keypad inputs and interrupts
Turn on LCD outputs
Start running a while loop to collect two data points that will make the end point
Matrix keypad input acts as interrupt to store the two data points
Exit the polling loop for input data
Turn off matrix keypad inputs and interrupts
Turn on IR sensor inputs and interrupts
Turn on motor ouputs
Enter next polling loop for pathfinding
Create the path to follow based on the obstacle graph
Turn the path into a list of events
Add these events to an EventQueue
LOOP:
Run the eventQueue
In the situation of an IR sensor input:
        Turn off the motors
        Find a way to update the event queue and create a new path outside ISR context
        Pause the IR sensor input while
Go back to LOOP until the destination has been reached
End the program
```

Code:

Link to github repository for the project: rupin-r/CSE321_Project3_AutonomousVehicle (github.com)

Link to ReadMe backup file:
https://docs.google.com/document/d/1jBbKuM_io5CE4r9a3CqcPD9GcT0qzHiriIowRcfEKcU/edit?usp=sharing
Link to main file backup:
https://docs.google.com/document/d/1sX4OjYislHZ0JbKiV9BLRceBJiadSLEgaeASmU_BbRI/edit?usp=sharing
Link to IR sensor backup:
https://docs.google.com/document/d/1YBkw1MJKqHMCuHoyjuMRzhGXYPWQbo6IaLmhuprcfG8/edit?usp=sharing
Link to LCD display backup:
https://docs.google.com/document/d/1bZbfktRTSSb7HUpaYeGgbCO3I_P6M_fH_Mknh8QseII/edit?usp=sharing
Link to matrix dot display backup:
https://docs.google.com/document/d/1oDuxDbd9bDV1vq1qzX8Jam33OdwVXDdOH-do65QTnDY/edit?usp=sharing
Link to matrix keypad backup:
https://docs.google.com/document/d/1ZTuYkX-l1_J8wpKbbX7DlZRM3E6kvf0HHlI91J3Ol10/edit?usp=sharing
Link to motors backup:
https://docs.google.com/document/d/1vrJqBn1GDWbn3usBNk6gdWda7w_D-NVzUOq-Sd5TeiU/edit?usp=sharing
Link to lcd1602.cpp backup:
https://docs.google.com/document/d/1Xxj1h9p4J6y8y5yasIFJeiAAGL2NWiRP6cz_oI2bHyk/edit?usp=sharing
Link to lcd1602.h backup:
https://docs.google.com/document/d/1HAztsFqkjOXHuDJG_G70Jl2ChWR3tlTbGh9wavz7RoE/edit?usp=sharing
Link to max7219.cpp backup:
https://docs.google.com/document/d/1QBfMQul4JFotc-sUi3U2aTXNHTp50cTp49vgcXNFlUY/edit?usp=sharing
Link to max7219.h backup:
https://docs.google.com/document/d/1uj5x6Lz-e5GVGS6-Z7mVxLyXxaxY1WABBUgcEM6sR2M/edit?usp=sharing

Memo:

A copy of the submitted memo

TO: Dr. Winikus
FROM: Rupin Raj Kumar Pradeep
Subject: Project 3 Autonomous Vehicle Stage 2 Progress Report
Date: 11/18/2022

PURPOSE:
The goal of the project is to implement a pathfinding system using entered coordinates from a matrix keypad and LCD display as well as BFS pathfinding with IR sensors, a matrix dot display, and motors. The report will outline the current progress, the initial setup, and planning phases as well as the plan for starting implementation.
SUMMARY:
There are currently two inputs going into the system, the matrix keypad and the IR sensors. Both the matrix keypad and IR sensor have been configured correctly only using bitwise control and without the need for API. There are several outputs to the system: motors, matrix dot display, and LCD display. The 1602 LCD, which will display the entered coordinates, has been configured using the given API. The matrix dot display required researching an API and editing it to suit the needs of the project. It is currently correctly configured for the project. The motors have not been attained yet and therefore could not be tested.
UPDATE ON PROGRESS:
A single algorithm has been created to outline the system tasks required for pathfinding. Several parts have been tested with C++ coding to ensure proper capabilities while prototyping. However, the motors have not been tested since they have not been acquired yet. Therefore, nothing has been combined into a functioning system. The code algorithm will eventually provide an outline for the expected inputs and outputs of the code when all the sensors and outputs are combined. Then the peer review should provide feedback on missing parts of the project. Also, the repositories to store both the project and the code have been set up appropriately to track future changes.
CONCERNS:
Due to an unfortunate sickness recently, I have been forced to take a week break from classes resulting in several assignments piling up. However, since break is approaching, these assignments can still be completed on time. Taking this into consideration, the project should still be complete in time, however, an extension may be necessary.
RECOMMENDATION:
During Thanksgiving week, all parts will have been tested and all algorithms will be complete. The final code and design implementation will be started with more research looking into array manipulation in mbed studio. By the end of the following week, implementation will be complete and testing will begin in order to ensure functionality before presenting.

Revisions:

There was no meaningful feedback on the opening plan so nothing changed in regards to the first submission of the opening plan.

While making the chassis for the vehicle, the first change made was that it was impossible for 4 IR sensors to actually work as expected since the range was simply too short. Instead, only 1 is used and is hooked up to a servo. A user could optionally edit the code to allow this IR sensor to rotate, but the rotation must not be continuous or the wires will tangle around the motor.

Also, 4 motors can not all be powered by the same voltage source and maxes out at 2 at a time. Therefore, the driving mechanism changed into 2 wheel drive and it was more efficient to simply get rid of the other two motors altogether.

After making the first edition of the code, the path would not start. After debugging, the path was found to be empty specifically because data from the graph was getting deleted while executing BFS.

Got rid of all abstract data structures that were previously in use in order to avoid using pointers. Instead, arrays held movement positions.

Further changes included decreasing the area of movement from 10x10 to 5x5 as well as getting rid of the matrix dot display implementation into the project. In the first stage of the project, the IR sensor was not fully implemented and would only stop the vehicle because the eventQueue refused to continue running posted events.

In the last edit of the code, the IR sensor became fully implemented by pulling all the code out of the ISR and introducing code into a polling mechanism that was indirectly controlled by the IR sensor ISR. Also, the eventQueue could be restarted just by calling dispatch_once again.