Robotics project report

Tasks overview

The main goal of the project is to program a motor vehicle robot that could sense its surroundings and successfully follow a track to reach a predetermined destination.

Task 1 objective is to fine tune processing of robot's sensor readings to enable accurate navigation in a given environment. This calibration will have a major impact on later tasks.

Task 2 objective is the development of the robot's navigation and obstacle detection capabilities through the LDR and IR sensors respectively.

Task 3 objective is further enhancement to navigation capabilities by means of receiving rudimentary instructions and decoding them.

Task 4 objective is using enhanced navigational capabilities and acting upon decoded instructions

Methodology

Calibration

Calibration of motors and sensors is done every time the arduino is initialised within setup() function. LDRs are first a

A special setupInternal function will be leveraged to encapsulate all calibration and basic component checks functionality. This function will be triggered by default in Arduino's setup function as well as invoked on demand by pressing the right button during main loop sequence

LDRs are calibrated by continuous analogRead of dark and light track sources. 25 readings are performed over a period of 250ms to capture an average value on the range of dark values and later light values. In order to give a more flexible detection, the offset is calculated as the 25% difference between light and dark values.

IR is calibrated by continuous analogRead of a receiver detecting an obstacle. 25 readings are performed over a period of 250ms to capture an average value for when an obstacle is detected. Additional offset of 12 is added for more flexibility.

Motor calibration was done separately to determine stop values for the motors. To ensure that motors operate in sync, 10 readings of a basic “move forward, rotate right, rotate left, move backward” are performed to detect any possible deviations.

Implementation

Foundational tenet of software implementation for this project was to follow object-oriented programming principles and model each component as an object through a class definition encapsulating their state and exposing behaviour through functions. More complex units were grouped into control classes that expose a more complex behaviour through mediating individual component objects.

Direction detection navigational capabilities represented by three LDRs were grouped to LDR control which allowed calibration through configure function and getDirectionOfMovement function which provides orientation decision and detection of barcode reading. At its core orientation decisions are made through binary dark / light LDR reading comparison. For example “left light, middle dark, right light” yields “forward”. Another example “all three LDRs dark” yields “start barcode reading”.

Blocking detection capability represented by IR is encapsulated by control object which allows calibration and isBlocked functions. When block is detected, main loop sequence would issue motors halt instruction and wait until obstacle is removed./.

In order to receive feedback for decision making by navigational controls, LEDcontrol is used which encapsulates three LED components and signals with appropriate colour codes during calibration and main sequence.

Motor functionality is represented by two motor components grouped into DrivetrainControl object. DrivetrainControl exposes basic motor functionalities such as moving, turning, rotating and following direction of movement determined by LDRcontrol. Various movement functions are relying on basic setSpeed function that accepts as input delta power applied to left and right motors as two parameters. For example “moving forward” actioned by “10, 10” and “rotate left” actioned by “-10, 10”

Enhanced navigation through rudimentary instructions is modelled by a logical construct represented by BarcodeControl object. Its responsibility is to use LDRcontrol direction of movement output and use it to determine complex barcode instructions to provide additional navigation hints. The algorithm assumes 16 navigation readings after “start barcode reading” input is provided by LDRcontrol. Three groups of inputs are assumed to form a barcode instruction. Group 1 is first dark bar, group 2 is straight line after group 1 and group 3 is optional second dark bar. For example “stop barcode pattern” would be represented with “group 1 = 3 reads, group 2 = 10 reads” and “turn right barcode pattern” would be represented by “group 1 = 3 reads, group 2 = 6 reads, group 3 = 3 reads”. BarcodeControl exposes getDirectionOfMovement akin to LDRcontrol which provides direction of movement dependent

on a complete barcode reading. This can be passed to the DrivetrainControl in the same fashion thus overriding decision made by LDRs.

High level main sequence flow diagram can be seen in figure below.

Experiments

The robot was tested under a variety of conditions, mostly in relation to light level and path colour. The robot was capable of easily determining where the track was and following it in both low contrast (grey with back or grey with white) and high contrast (black and white) settings due to the very accurate calibration readings.
The IR sensor was less accurate when using analog values for obstacle detection and the conditions of the room (light level, how high the temperature of the room, reflectivity of objects in the room) influenced the value greatly. With further testing it became apparent that the digital read value was much more consistent and was affected less by erratic changes in the temperature.

The properties of the chosen obstacle also influence when they are detected. For example if an object is reflective (silver thermos) then it will be detected earlier, whereas if it is transparent (like glass) it may not even be detected at all.

Each test was performed at least 10 times, with more important tests taking up to 50 readings to ensure outliers do not greatly impact the average value calculated.

Testing

The average reading for black is 300-500, grey is 550-650 and white is 600-800. Refer to values tables for the data set.

The average reading for detecting an obstacle is about 200-300 for low light pollution and about 500-600 for high light pollution. Refer to the obstacle table in IR and to IR mean values tables.

Review

Overall, the software managed to accomplish all the set tasks and the code developed is fairly readable. The code is very efficient with a total cycle time of about 200ms per loop, which leads to very reliable data collection and in turn more accurate path following. However, large amounts of objects and global variables has led to multiple times where the code had to be refactored to free up more memory for the serial port and local variables and the memory usage still remains high at 79% usage.

If I had more time I would generalise the behaviour of the robot to make it able to navigate more effectively in more extreme environments. I would also add extra detection components (ultrasonic sensors, more LDRs, etc) so that the robot can sense where it is more effectively and exchange the Arduino Uno for an Arduino Mega to resolve the high memory usage. I would also resolve the memory leak that LDR mean and standard deviation function experiences with their values if they are not logged using serial print messages.

This project taught me a lot about objects in arduino, which will allow me to make software more effective and easier to read in future.

Appendix and references

references

How I learned to make objects in arduino
https://arduinogetstarted.com/faq/how-to-create-class-and-object-on-arduino-ide

Testing tables

Mean and SD of LDRs: Light

	Left LDR	Middle LDR	Right LDR
Mean	711	708	693
Standard deviation	16.91	1.41	1.00

Mean and SD of LDRs: Dark

	Left LDR	Middle LDR	Right LDR
Mean	459	459	440
Standard deviation	5.92	1.00	1.00

Mean and SD of IR: Blocked

	IR
Mean	187
Standard Deviation	12.33

Mean and SD of LDRs: Open

	IR
Mean	369
Standard Deviation	4.90

LDR Dark Values

reading	Left LDR	Middle LDR	Right LDR
1	265	270	262
2	383	373	346
3	434	438	409
4	373	378	363
5	439	431	385
6	487	472	485
7	480	467	487
8	497	487	492
9	468	461	457
10	531	532	542

LDR Grey Values

reading	Left LDR	Middle LDR	Right LDR

1	623	643	671
2	617	636	661
3	619	627	649
4	620	633	656
5	619	631	651
6	618	630	651
7	614	626	649
8	616	625	646
9	612	626	648
10	613	625	645

LDR Light Values

reading	Left LDR	Middle LDR	Right LDR
1	517	513	507
2	685	665	646
3	781	782	785
4	592	587	552
5	670	662	618
6	741	739	756
7	739	739	757
8	711	706	720
9	716	713	723
10	718	712	727

IR obstacle Values

reading	IR
1	415
2	368
3	151
4	414
5	435
6	301
7	408
8	363
9	337
10	331

Button functionality

reading	Button detected press?
1	yes
2	yes
3	yes
4	yes
5	yes
6	yes
7	yes
8	yes
9	yes
10	yes

led functionality

reading	Led turned on?
1	yes
2	yes
3	yes
4	yes
5	yes
6	yes
7	yes
8	yes
9	yes
10	yes