CG3002 Embedded System Design Project

Semester 1 2014/2015

"Show My Way"

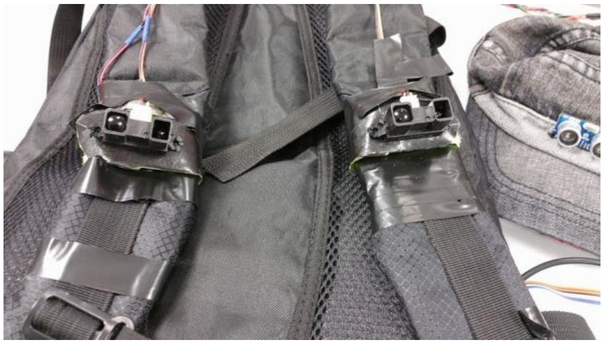
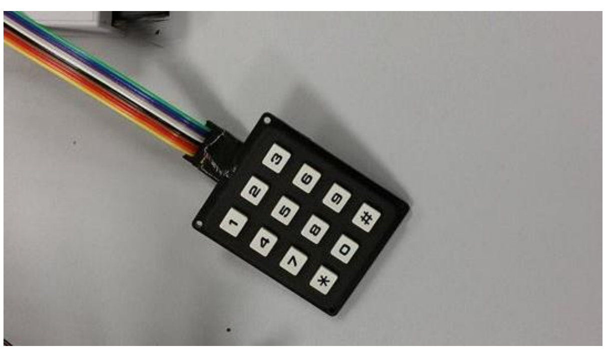
Final Report

Group No: **11**	Name	Student Number
Group Member #1	Sebastian Wong	A0101856J
Group Member #2	Andre Lim	A0098021L
Group Member #3	Bob Wong	A0094718U
Group Member #4	Angela Heng	A0101695H
Group Member #5	Chew Yixiu	A0101742W
Group Member #6	Zhang Yanqing	A0100985E

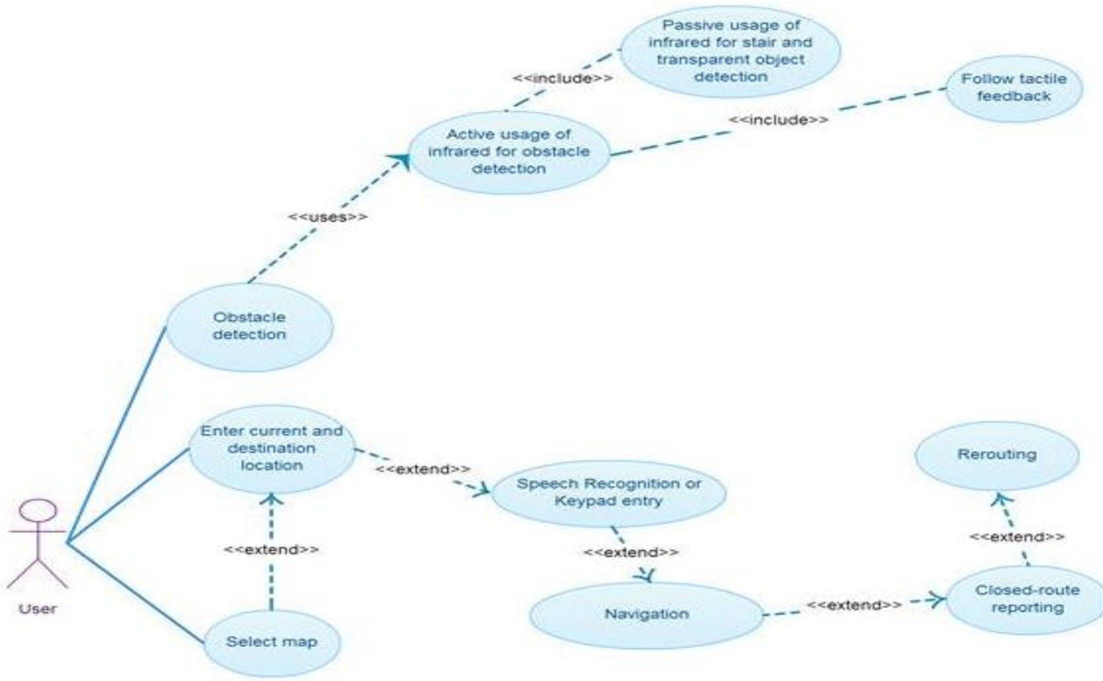
Section 1 System Architecture

Photos of device:

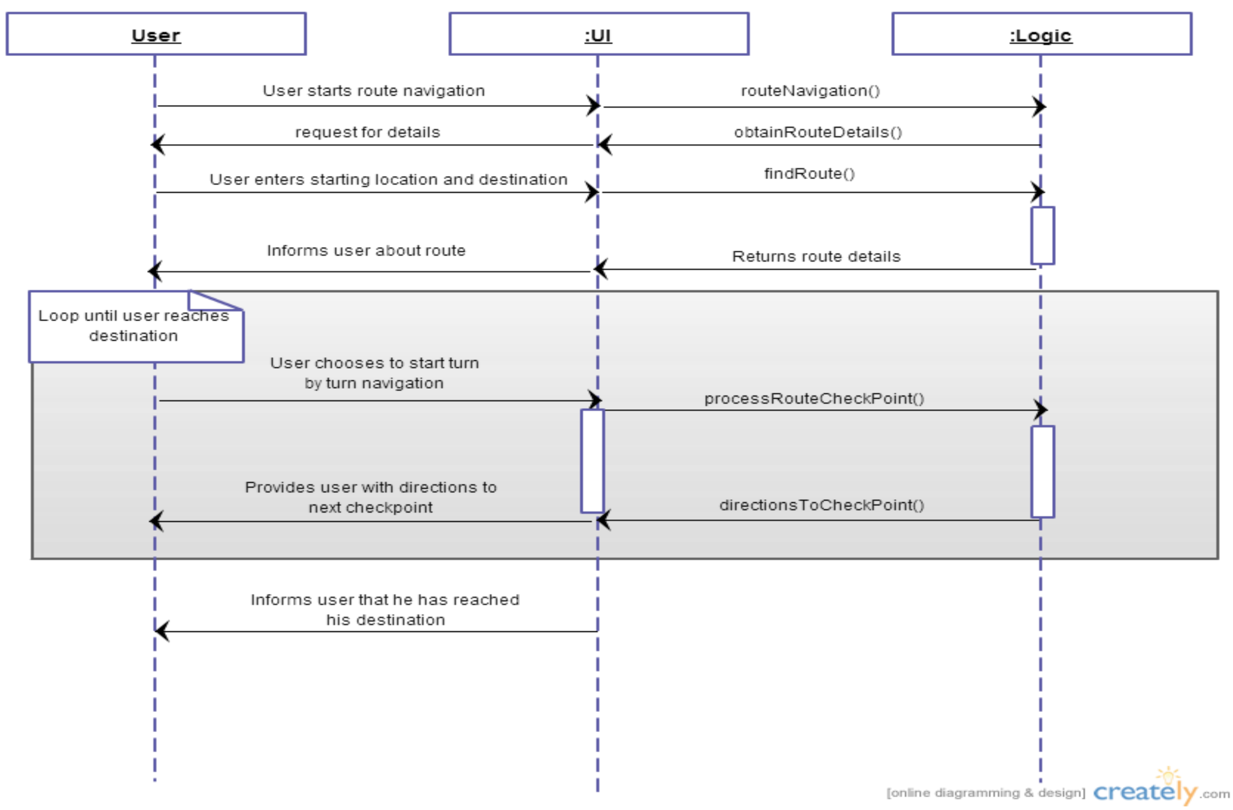
		
Overall image of device	Internal connections of device	Cap: Long range Infrared sensor and Ultrasound attached

	Bag Strap: - 2 short range Infrared sensor to detect chest level obstacles - vibration motors attached on the underside to give tactile feedback to user
	Headset with Microphone: - provides audio feedback from device to user - (optional) to receive audio input from user to device
	Keypad: - to receive user input to the device after being prompted by headset
	Belt: - Long range Infrared sensor to detect obstacles below waist level - vibration motor to give tactile feedback - switch to turn on or off the device

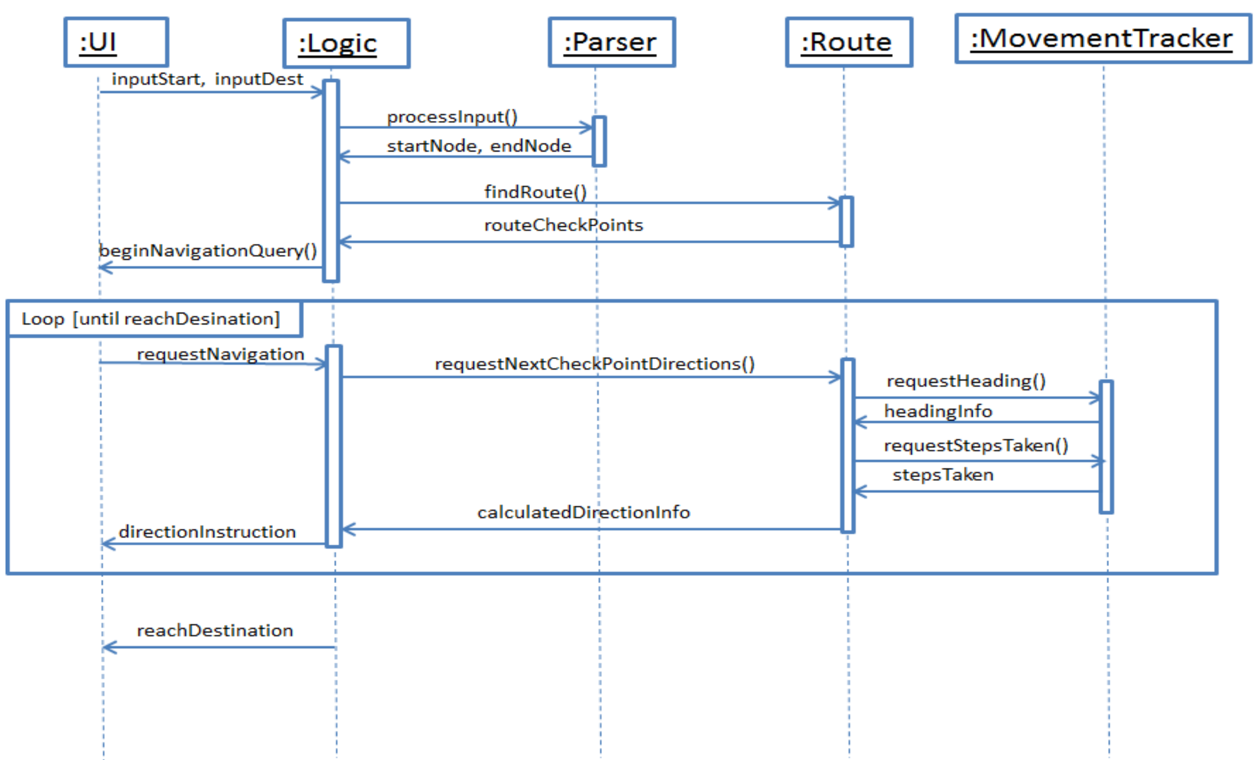
Use Case Diagram:



UML Sequence Diagram for indoor Navigation (High Level)



UML Sequence Diagram



Section 2 Firmware Design

General firmware overview:

The firmware runs on round-robin scheduling. Internal loops are completely avoided in the design so that the absence or temporary malfunction of one component will not restrict other components from functioning, and the time usage of each component is fixed. The notable, distinct parts of the system are the accelerometer-pedometer, the compass, the keypad, obstacle avoidance sensors, and finally the communications protocol used to interface the Arduino with the Raspberry Pi.

Object detection protocol:

The device is equipped with two short-range IR and two long-range IR sensors. The short-range IR on the shoulder straps are used mainly for detecting chest-height obstacles, i.e. walls and people, and are tilted slightly outwards, so left and right walls can be identified without directly facing them. They have an effective range of 20 to 150 cm, and will write the distance detected into their respective variables. Another two long-range IR sensors are mounted on the cap and on the waist, with an effective range of 80 to 320cm determined experimentally, as opposed to the reported range of 100 to 550cm. Again, the distance detected will be written into their respective variables set aside for this purpose.

The motors are vibrated based on their positions and the values in the variables. The left and right motors are vibrated at full duration, 1.3s on, 1.3s off, if an object is detected at less than 70cm distance. Another motor on the shoulder is connected to the cap, and will not vibrate if the distance detected is more than 170cm. If the distance is closer, the duration of vibration gets longer inversely proportionate to the distance of the object, from 100ms on, 100ms off, to 1.3s on, 1.3s on. This also applies for the waist IR, which is tilted at a slight angle towards the floor to sense stairs and waist-height objects such as tables, chairs, and people.

Communication Protocol:

UART is used as the communication protocol between Raspberry Pi and Arduino. UART1 (tx1 and rx1) of 9600 baud rate is used in the project. We used 3-way handshake protocol to establish connection, and stop-and-wait protocol to send and receive data. All requests for data and connection are initiated by the Raspberry Pi. Arduino initiates to send data to Raspberry Pi only when the Voice flag is set or an enter is pressed, both on the keypad. A modified ISR for UART1_Rx is used for Arduino to receive data and put them into a buffer, as well as to turn on a flag to tell the main loop that a data has been received by the UART. This is to minimise unnecessary time to poll for incoming acknowledgement data from the Raspberry Pi. Raspberry Pi will request for data from the Arduino at every one second interval. Different Acknowledgement number is implemented for each sensor data, so that we can confirm if a data is received at the Raspberry Pi side.

All sensor data are stored in an array. For compass readings, since size of each cell in the array is only 8 bits (i.e. maximum value = 255), while the bearing value is more than 255, the bearing value is split into 2 8-bit values, first byte is the upper byte, second byte is the lower byte. The sensor data that will be sent to the Raspberry is as follows:

sensorData[0]	number of steps taken from the last read
sensorData[1] to sensorData[10]	compass bearings for maximum of 5 steps, each pair of bearing value corresponds to each step taken
sensorData[11] to sensorData[12]	current bearing of the user

When the Raspberry Pi first sends a READ to Arduino (after navigation in Raspberry Pi starts), the first 5 values in the data array is set to 0 to remove unnecessary data before sending them to the Raspberry Pi. Compass bearings is also attached to the 11th and 12th position in the data array.

Section 3 Hardware List and Cost Analysis

Component	Quantity	Price/unit (in SGD)	Reference (should be local)
Rapsberry Pi Model B	1	45.00	http://sg.element14.com/raspberry-pi/raspbrry-modb-512m/sbc-raspberry-pi-model-b-512mb/dp/2434669?in_merch=New%20Products
Arduino Mega	1	61.40	https://www.robot-r-us.com/arduino-stuff/arduino-mega-rev.-3-new.html
GY-87 10DOF 3-axis Gyro + 3-axis Acceleration + 3-axis Magnetic Field + Air Pressure Module	1	16.90	http://www.amazon.com/Module-Mpu6050-Hmc5883l-Bmp180-Sensor/dp/B00OUXKZ76/ref=sr_1_1?s=electronics&ie=UTF8&qid=1416198760&sr=1-1&keywords=10+dof+gy-87
Ultrasonic Sensor HY SRF-05	1	28.90	http://www.sgbotic.com/index.php?dispatch=products.view&product_id=115
Infra-Red Sensor (detection range 20 - 150cm) GP2Y0A02YK	2	21.50	Bill from SGBotic Pte Ltd
Infra-Red Sensor (detection range 100 - 550cm) GPY0A710K0F	2	30.00	Bill from element14 Pte Ltd http://sg.element14.com/sharp/gp20a710k0f/distance-measuring-sensor/dp/1618432?ref=lookahead
3 pin connector	2	2.90	Bill from SGBotic Pte Ltd

for GP2Y0A02YK			
5 pin connector for GPY0A710K0F	2	4.00	http://www.2dogrc.com/5-pin-connector-set-female-123.html#.VGpg5fnF_WE
vibration motors MOT105A2B	4	4.50	Bill from SUN LIGHT ELECTRONICS PTE LTD, sim lim square
Battery Charger (XIAOMI)	2	14.00	http://www.mi.com/sg/mipowerbank10400/
USB headphone	1	53.10	Bill from NUS coops
bag	1	18	Bill from Bugis Street
cap	1	15	Bill from Bugis Street
USB cable (Rpi)	1	2	Bill from Daiso
Headphone Box For internal wiring of circuits	1	-	-
Heat Shrink(2mm)	3	1	Bill from SUN LIGHT ELECTRONICS PTE LTD, sim lim square
Heat Shrink(6mm)	2	1.80	Bill from SUN LIGHT ELECTRONICS PTE LTD, sim lim square
Wires(1m)	18	0.50	Bill from SUN LIGHT ELECTRONICS PTE LTD, sim lim square
Black tape	3	3.85	Bill from NUS coops
Cable tie	15	-	-
Scissors	-	-	-

WireStriper	-	-	-
Plier	-	-	-
Duct tape	1	3	http://list.qoo10.sg/item/DUCT-TAPE-BROWN-DAIMARU/418659151
Book strap	2	1.30	Bill from NUS coops
USB cable(arduino)	1	4.50	Bill from NUS coops
Total Cost		440.35	

Additional features not demonstrated in assessment:

1) Speech Recognition

With speech recognition, our device is able to take in voice inputs such as node names (e.g. Seminar Room 1, Tutorial room 11, NUS Hacker's Room, etc.), node numbers (e.g. 15, 23, 41), and basic commands such as 'Begin', 'Start', 'Stop', or 'Cancel'.

In order to improve on the accuracy and precision of our speech recognition function, we used online tools to generate dictionaries, acoustic models, and language models for our product.

The dictionary file consists of a list of required words with a sequence of phones for the program to decode the pronunciation of our commands.

The acoustic model is used to allow the speech recognition program to adapt to our accents, recording environment, and audio transmission channels.

The language model is used to assign probabilities to different sequences of words.

2) Flexi-command string parser function
    a) to accept and process node and building names on top of its designated numbers
    b) made use of python regular expressions

   With flexi-command string parser, our device is able to take in user input from voice recognition in the form of discrete word chunks and process them into the appropriate format as seen in the JSON map file node names.
   For example, transforming the string "N U S Hackers Room" to "NUS Hacker's Room" and "to COM to hyphen to hyphen one" to "to COM2-2-1"

Section 4 Video Summary

Youtube Video Link

https://www.youtube.com/watch?v=BuOlmn9beW4

Reference

(Knupp, 2014)
(Python Software Foundation, 2014)
(Carnegie Mellon University, 2014)
(Haque, 2014)
(Texas Instruments, 2013)
(Espeak, 2007)
(http://www.seeedstudio.com/, 2009)
(TEXAS INSTRUMENTS, 2004)
(guibot, n.d.)
(http://arduinobasics.blogspot.sg/, 2012)
(http://www.adafruit.com/, 2006)
(http://luckylarry.co.uk/, 2009)
(http://robotic-controls.com/, 2013)
(http://www.sharpsme.com/, 2006)
(http://www51.honeywell.com/, 2013)
(http://hobbylogs.me, 2014)
(http://playground.arduino.cc/, 2013)
(https://github.com/, 2014)
(https://github.com, 2011)
(https://github.com, 2011)
(https://github.com, n.d.)
(https://github.com, 2011)
(https://github.com, 2011)
(http://playground.arduino.cc, n.d.)
(Brevig, n.d.)
(Stoffregen, 2011)
(Mark Stanley, n.d.)
(Haq, n.d.)
(Haq, www.github.com, n.d.)
(Love Electronics, 2011)
(Love Electronics, 2011)
(Zhao, 2010)