Software design description – Hardware

System overview:

\*A general overview of how the whole system should work.

The Mower is a vehicle constructed with an Arduino and a Raspberry Pi card. The Arduino controls the mower's sensors and motors, while the Raspberry Pi handles communication between the mower and the database.

It is possible to manually drive the Mower and put it into autonomous mode using an app.

The autonomous mode allows the Mower to drive within a defined area while avoiding front-end collisions.

System architecture:

\*A summary of how the arduino and raspberry pi are setup and communicating with each other. Should get a gist of how the hardware system is setup.

Detailed system design:

**MBot**

**Variables:**

**Functions:**

void \_delay(float seconds):

Input:

Output: None

Description: The program is paused for the time (in milliseconds) specified as parameter.

void move(moveDirection direction, int speed):

Input:

Output: None

Description:

void moveForward():

Input:

Output: None

Description: When this function is called, the mower will drive forward

void moveBackwards():

Input:

Output: None

Description: Call moveBackwards() to give the Mower more room to rotate.

void turnLeft():

Input:

Output: None

Description: When this function is called, the mower will turn left.

void turnRight():

Input:

Output: None

Description: When this function is called, the mower will turn right.

void stopMotors():

Input:

Output: None

Description: Call stopMotors() to prevent the Mower from crossing the line.

void collision():

Input:

Output: None

Description: when the Mower hits an obstacle, back then turn right.

void autoTurn():

Input:

Output: None

Description:

void isr\_process\_motorLeft(void):

Input:

Output: None

Description:

void isr\_process\_motorRight(void):

Input:

Output: None

Description:

int checkSensors():

Input:

Output:

Description: This function will check the state of the sensor.

String getOrientation():

Input:

Output:

Description:

int autonomousDriving(int currentState):

Input:

Output:

Description: When this function is called, the Mower will run on its own. The Mower will check the state of linesensorStateGlobal and the ultrasonic sensor in this function.

void bluetoothDriving(char nextState):

Input:

Output: None

Description: When this function is called, the mower will be controlled by the application.

void updateState(int data)

Input:

Output: None

Description: Will check the data given from Bluetooth and let the Mower be in different states by changing the state of mower StateGlobal to forward, backwards, left, right or stop(idle).

void setup():

Input:

Output: None

Description:

void \_loop():

Input:

Output: None

Description:

void loop():

Input:

Output: None

Description:

**Raspberry pi**

**Variables:**

**Functions:**

sendPositionRequest(x, y, sessionID, state, collisionFlag):

sendImageRequest(x,y):

bluetoothInit():

class CalculatePosition:

\_\_init\_\_(self):

terminate(self):

run(self, speed, newDirection):

class ReceiveBluetooth:

\_\_init\_\_(self, client):

terminate(self):

run(self):