EDP AVS

Generated by Doxygen 1.8.14

Contents

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 Navigator Class Reference

`#include <Navigator.h>`

Public Member Functions

- Navigator ()
- void addObstacle (int xPos, int yPos)
- void testMap ()
- void printMap ()
- void testObstacleData ()

Public Attributes

- int grid_ [HEIGHT+2][WIDTH+2][DATA]

Private Member Functions

- void createMap ()
- int convertToArray (int coord)

3.1.1 Constructor & Destructor Documentation

3.1.1.1 Navigator()

`Navigator::Navigator ( )`

3.1.2 Member Function Documentation

3.1.2.1 addObstacle()

```
void Navigator::addObstacle (
            int xPos,
            int yPos )
```

Adds an obstacle to the map

Parameters

xPos	Cartesian x-coordinate of the obstacle grid square
yPos	Cartesian y-coordinate of the obstacle grid square

3.1.2.2 convertToArray()

```
int Navigator::convertToArray (
            int coord )  [private]
```

Converts y grid coordinate to y array coordinate

Parameters

coord	Grid coordinate

Returns

Array coordinate

3.1.2.3 createMap()

```
void Navigator::createMap ( )  [private]
```

Creates a map that with a given length, height and a boundary wall of obstacles assigning an initial weighting of -1

Parameters

height	Height of the grid to be traversed
length	Length of the grid to be traversed

3.1.2.4 printMap()

```
void Navigator::printMap ( )
```

Prints the grid with the associated priority value for each grid square

3.1.2.5 testMap()

```
void Navigator::testMap ( )
```

Tests the createMap() function with a length of 10 and height of 10. Tests the prioritiseMap() function with a target location at 9, 9

3.1.2.6 testObstacleData()

```
void Navigator::testObstacleData ( )
```

Adds obstacles from hardcoded obstacle position data

3.1.3 Member Data Documentation

3.1.3.1 grid_

```
int Navigator::grid_[HEIGHT+2][WIDTH+2][DATA]
```

The documentation for this class was generated from the following files:

- D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/bot↩ Main/Navigator.h
- D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/bot↩ Main/Navigator.cpp

3.2 NewPing Class Reference

```
#include <NewPing.h>
```

Public Member Functions

- NewPing (uint8_t trigger_pin, uint8_t echo_pin, unsigned int max_cm_distance=MAX_SENSOR_DISTANCE)
- unsigned int ping (unsigned int max_cm_distance=0)
- unsigned long ping_cm (unsigned int max_cm_distance=0)
- unsigned long ping_in (unsigned int max_cm_distance=0)
- unsigned long ping_median (uint8_t it=5, unsigned int max_cm_distance=0)
- void ping_timer (void(∗userFunc)(void), unsigned int max_cm_distance=0)
- boolean check_timer ()

Static Public Member Functions

- static unsigned int convert_cm (unsigned int echoTime)
- static unsigned int convert_in (unsigned int echoTime)
- static void timer_us (unsigned int frequency, void(∗userFunc)(void))
- static void timer_ms (unsigned long frequency, void(∗userFunc)(void))
- static void timer_stop ()

Public Attributes

- unsigned long ping_result

Private Member Functions

- boolean ping_trigger ()
- void set_max_distance (unsigned int max_cm_distance)
- boolean ping_trigger_timer (unsigned int trigger_delay)
- boolean ping_wait_timer ()

Static Private Member Functions

- static void timer_setup ()
- static void timer_ms_cntdwn ()

Private Attributes

- uint8_t _triggerBit
- uint8_t _echoBit
- volatile uint8_t * _triggerOutput
- volatile uint8_t * _echoInput
- volatile uint8_t * _triggerMode
- unsigned int _maxEchoTime
- unsigned long _max_time

3.2.1 Constructor & Destructor Documentation

3.2.1.1 NewPing()

```
NewPing::NewPing (
            uint8_t trigger_pin,
            uint8_t echo_pin,
            unsigned int max_cm_distance = MAX_SENSOR_DISTANCE )
```

3.2.2 Member Function Documentation

3.2.2.1 check_timer()

```
boolean NewPing::check_timer ( )
```

3.2.2.2 convert_cm()

```
unsigned int NewPing::convert_cm (
            unsigned int echoTime )  [static]
```

3.2.2.3 convert_in()

```
unsigned int NewPing::convert_in (
            unsigned int echoTime )  [static]
```

3.2.2.4 ping()

```
unsigned int NewPing::ping (
            unsigned int max_cm_distance = 0 )
```

3.2.2.5 ping_cm()

```
unsigned long NewPing::ping_cm (
            unsigned int max_cm_distance = 0 )
```

3.2.2.6 ping_in()

```
unsigned long NewPing::ping_in (
            unsigned int max_cm_distance = 0 )
```

3.2.2.7 ping_median()

```
unsigned long NewPing::ping_median (
            uint8_t it = 5,
            unsigned int max_cm_distance = 0 )
```

3.2.2.8 ping_timer()

```
void NewPing::ping_timer (
            void(*)(void) userFunc,
            unsigned int max_cm_distance = 0 )
```

3.2.2.9 ping_trigger()

```
boolean NewPing::ping_trigger ( )  [private]
```

3.2.2.10 ping_trigger_timer()

```
boolean NewPing::ping_trigger_timer (
            unsigned int trigger_delay )  [private]
```

3.2.2.11 ping_wait_timer()

```
boolean NewPing::ping_wait_timer ( )  [private]
```

3.2.2.12 set_max_distance()

```
void NewPing::set_max_distance (
            unsigned int max_cm_distance )  [private]
```

3.2.2.13 timer_ms()

```
void NewPing::timer_ms (
            unsigned long frequency,
            void(*)(void) userFunc )  [static]
```

3.2.2.14 timer_ms_cntdwn()

```
void NewPing::timer_ms_cntdwn ( )  [static], [private]
```

3.2.2.15 timer_setup()

```
void NewPing::timer_setup ( )  [static], [private]
```

3.2.2.16 timer_stop()

```
void NewPing::timer_stop ( )  [static]
```

3.2.2.17 timer_us()

```
void NewPing::timer_us (
            unsigned int frequency,
            void(*)(void) userFunc )  [static]
```

3.2.3 Member Data Documentation

3.2.3.1 _echoBit

```
uint8_t NewPing::_echoBit  [private]
```

3.2.3.2 _echoInput

```
volatile uint8_t* NewPing::_echoInput  [private]
```

3.2.3.3 _max_time

```
unsigned long NewPing::_max_time  [private]
```

3.2.3.4 _maxEchoTime

```
unsigned int NewPing::_maxEchoTime  [private]
```

3.2.3.5 _triggerBit

```
uint8_t NewPing::_triggerBit  [private]
```

3.2.3.6 _triggerMode

```
volatile uint8_t* NewPing::_triggerMode  [private]
```

3.2.3.7 _triggerOutput

```
volatile uint8_t* NewPing::_triggerOutput  [private]
```

3.2.3.8 ping_result

```
unsigned long NewPing::ping_result
```

The documentation for this class was generated from the following files:

- D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/bot↩ Main/NewPing.h
- D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/bot↩ Main/NewPing.cpp

3.3 ObstacleDetection Class Reference

```
#include <ObstacleDetection.h>
```

Public Member Functions

- ObstacleDetection ()

  Default constructor - not used currently.
- ObstacleDetection (ObstacleSensor *frontSensorPtr, ObstacleSensor *leftSensorPtr, ObstacleSensor *rightSensorPtr, Navigator *navPtr)
- void detectAllSensors ()
- void odsToNavTestObstacles ()

  Filled with dummy obstacles to be outputted to navigation system for testing of OD-NM interface.

Private Member Functions

- uint8_t detectLeftSensor ()
- uint8_t detectRightSensor ()
- uint8_t detectFrontSensor ()

Private Attributes

- ObstacleSensor * frontSensorPtr_

  Pointer to front sensor.
- ObstacleSensor * leftSensorPtr_

  Pointer to left sensor.
- ObstacleSensor * rightSensorPtr_

  Pointer to right sensor.
- Navigator * navPtr_

  Pointer to the navigation system object.

Static Private Attributes

- static const uint8_t iterations = 5

  Number of times the sensors are activated to average distance of detection.

3.3.1 Detailed Description

ObstacleDetection object controls the activation patterns of all ObstacleSensor objects and outputs detected obstacle grid references to the navigation system.

INPUTS: Requires information detailing ObstacleSensor objects

OUTPUTS TO: Grid reference to navigation system

3.3.2 Constructor & Destructor Documentation

3.3.2.1 ObstacleDetection() [1/2]

```
ObstacleDetection::ObstacleDetection ( )
```

Default constructor - not used currently.

3.3.2.2 ObstacleDetection() [2/2]

```
ObstacleDetection::ObstacleDetection (
            ObstacleSensor * frontSensorPtr,
            ObstacleSensor * leftSensorPtr,
            ObstacleSensor * rightSensorPtr,
            Navigator * navPtr )
```

Constructor which takes in three ObstacleSensor object pointers and navigator object pointer used to activate in different sequences as required

Parameters

∗*frontSensorPtr*	Pointer to front sensor Obstacle Sensor Object
∗*leftSensororPtr*	Pointer to left sensor Obstacle Sensor Object
∗*rightSensorPtr*	Pointer to right sensor Obstacle Sensor Object
∗*navPtr*	Pointer to Navigator object in order to call addObstacle function and pass obstalce details to navigation system

3.3.3 Member Function Documentation

3.3.3.1 detectAllSensors()

```
void ObstacleDetection::detectAllSensors ( )
```

Standard detection function which activates all sensors and converts any obstacles to grid references. Will be in loop function so constantly firing.

3.3.3.2 detectFrontSensor()

```
uint8_t ObstacleDetection::detectFrontSensor ( )  [private]
```

Fire front sensor only

Returns

0 or 1 for legal obstacle distance detected

3.3.3.3 detectLeftSensor()

```
uint8_t ObstacleDetection::detectLeftSensor ( )  [private]
```

Fire left sensor only

Returns

0 or 1 for legal obstacle distance detected

3.3.3.4 detectRightSensor()

```
uint8_t ObstacleDetection::detectRightSensor ( )  [private]
```

Fire right sensor only

Returns

0 or 1 for legal obstacle distance detected

3.3.3.5 odsToNavTestObstacles()

```
void ObstacleDetection::odsToNavTestObstacles ( )
```

Filled with dummy obstacles to be outputted to navigation system for testing of OD-NM interface.

3.3.4 Member Data Documentation

3.3.4.1 frontSensorPtr_

```
ObstacleSensor* ObstacleDetection::frontSensorPtr_  [private]
```

Pointer to front sensor.

3.3.4.2 iterations

```
const uint8_t ObstacleDetection::iterations = 5  [static], [private]
```

Number of times the sensors are activated to average distance of detection.

3.3.4.3 leftSensorPtr_

```
ObstacleSensor* ObstacleDetection::leftSensorPtr_  [private]
```

Pointer to left sensor.

3.3.4.4 navPtr_

```
Navigator* ObstacleDetection::navPtr_  [private]
```

Pointer to the navigation system object.

3.3.4.5 rightSensorPtr_

```
ObstacleSensor* ObstacleDetection::rightSensorPtr_  [private]
```

Pointer to right sensor.

The documentation for this class was generated from the following files:

- D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/bot↩ Main/ObstacleDetection.h
- D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/bot↩ Main/ObstacleDetection.cpp

3.4 ObstacleSensor Class Reference

```
#include <ObstacleSensor.h>
```

Public Member Functions

- ObstacleSensor ()

  Default constructor - not used.
- ObstacleSensor (uint8_t triggerPin, uint8_t echoPin, float offsetX, float offsetY, float sensorAngle)
- uint8_t activateSensor (uint8_t iterations)
- void printDistance (String sensorName)

Static Public Member Functions

- static void calculateSoundCm (uint8_t dhtPin)
- static void printSound (float temp, float hum)
- static void updateOdsData (float x, float y, float heading)

Public Attributes

- float offsetX_

  Locations and directions on vehicle relative to Posyx sensor or center or some other reference.
- float offsetY_
- float sensorAngle_
- float sensorGridAngle_

  Angle of sensor relative to the grid (i.e. sensorAngle_ + heading_)
- int8_t unitVect_ [2]

  Unit vector of vector from sensor to detected obstacle (i.e xComp + yComp)
- float distance_

  Distance from sensor to object detected.
- float objXDist_

  Distances from pozyx to object detected.
- float objYDist_
- float objX_

  Grid coordinates at location object detected.
- float objY_
- int gridX_

  Grid reference of obstacle detected converted from coordinates above.
- int gridY_
- NewPing sonar_

  NewPing object causes HC-SR04 sensor pulses and enable.

Static Public Attributes

- static int8_t bias_ = 17

  Bias for grid reference conversion.

Private Attributes

- uint8_t triggerPin_

  Pin definitions.
- uint8_t echoPin_
- const unsigned int maxDistance_ = 400

  Maximum distance of sensor (i.e. 400cm)
- unsigned long duration_

  Stores First HC-SR04 pulse duration value.
- float objGridRef_ [2]

  objGridRef_[0] = x coordinate, objGridRef_[1] = y coordinate

Static Private Attributes

- static float soundcm_ = 0.0343f

  Stores calculated speed of sound in cm/ms - defaults to speed at 20 degrees Celcius.
- static float heading_

  AVS pozyx heading data.
- static float xPos_

  AVS pozyx x position.
- static float yPos_

  AVS pozyx y position.

3.4.1 Detailed Description

ObstacleSensor object contains all angles and distances/locations of a particular sensor allowing for distance calculations from the pozyx locator to be conducted. INPUTS: Still require pozyx heading angle (interface between sensor and pozyx) Humidity and temperature details from DHT-22 for improved accuracy Interfaces with HC-SR04 sensors via NewPing class

OUTPUTS TO: Objects used by ObstacleDetection class to output grid references to nav.

3.4.2 Constructor & Destructor Documentation

3.4.2.1 ObstacleSensor() [1/2]

```
ObstacleSensor::ObstacleSensor ( )
```

Default constructor - not used.

3.4.2.2 ObstacleSensor() [2/2]

```
ObstacleSensor::ObstacleSensor (
            uint8_t triggerPin,
            uint8_t echoPin,
            float offsetX,
            float offsetY,
            float sensorAngle )
```

Constructor reads in pin details as well as relative to car location and direction details

Parameters

triggerPin	Arduino pin number of sensor trigger
echoPin	Arduino pin number of sensor echo
offsetX	Left or right offset from pozyx placement on AVS chassis
offsetY	Forward or backward offset from pozyx placement on AVS chassis
sensorAngle	Angle sensor is placed at relative to forward direction of AVS chassis

3.4.3 Member Function Documentation

3.4.3.1 activateSensor()

```
uint8_t ObstacleSensor::activateSensor (
            uint8_t iterations )
```

Calculates x and y components to obstacle found by this sensor averaged iterations times by taking into account the heading of the vehicle, the offset of the sensor relative to the pozyx tag, the angle of the sensor relative to the vehicle and the distance the sensor calculates relative to itself. Returns 0 if no obstacle detected, 1 if obstacle detected

Parameters

iterations	Number of iterations sensor measures distance to calculate average over

Returns

0 or 1 for legal obstacle distance detected

3.4.3.2 calculateSoundCm()

```
void ObstacleSensor::calculateSoundCm (
            uint8_t dhtPin )  [static]
```

Calculates the speed of sound based on humidity and temperature found by DHT22 sensor with paramater dhtPin which is the arduino pin assigned to be the data input of the dht. Will be used in setup function so only accessed once at the start. This could be changed if conditions are expected to vary throughout use time.

Parameters

dhtPin	Arduino connected pin number of DHT sensor

3.4.3.3 printDistance()

```
void ObstacleSensor::printDistance (
            String sensorName )
```

Prints on serial monitor the detected sensor distance, and converted x and y components relative to pozyx system

Parameters

sensorName	Name of sensor to be printed

3.4.3.4 printSound()

```
void ObstacleSensor::printSound (
            float temp,
            float hum )  [static]
```

Parameters

temp	Calculated temperature
hum	Calcualted humidity

3.4.3.5 updateOdsData()

```
static void ObstacleSensor::updateOdsData (
            float x,
            float y,
            float heading )  [inline], [static]
```

Updates all sensor data. Will be in loop function so constnatly firing

Parameters

x	Current X location of pozyx
y	Current Y location of pozyx
heading	Current heading of pozyx

3.4.4 Member Data Documentation

3.4.4.1 bias_

```
int8_t ObstacleSensor::bias_ = 17  [static]
```

Bias for grid reference conversion.

3.4.4.2 distance_

```
float ObstacleSensor::distance_
```

Distance from sensor to object detected.

3.4.4.3 duration_

```
unsigned long ObstacleSensor::duration_  [private]
```

Stores First HC-SR04 pulse duration value.

3.4.4.4 echoPin_

```
uint8_t ObstacleSensor::echoPin_  [private]
```

3.4.4.5 gridX_

```
int ObstacleSensor::gridX_
```

Grid reference of obstacle detected converted from coordinates above.

3.4.4.6 gridY_

```
int ObstacleSensor::gridY_
```

3.4.4.7 heading_

```
float ObstacleSensor::heading_  [static], [private]
```

AVS pozyx heading data.

3.4.4.8 maxDistance_

```
const unsigned int ObstacleSensor::maxDistance_ = 400  [private]
```

Maximum distance of sensor (i.e. 400cm)

3.4.4.9 objGridRef_

```
float ObstacleSensor::objGridRef_[2]  [private]
```

objGridRef_[0] = x coordinate, objGridRef_[1] = y coordinate

3.4.4.10 objX_

```
float ObstacleSensor::objX_
```

Grid coordinates at location object detected.

3.4.4.11 objXDist_

```
float ObstacleSensor::objXDist_
```

Distances from pozyx to object detected.

3.4.4.12 objY_

```
float ObstacleSensor::objY_
```

3.4.4.13 objYDist_

```
float ObstacleSensor::objYDist_
```

3.4.4.14 offsetX_

```
float ObstacleSensor::offsetX_
```

Locations and directions on vehicle relative to Posyx sensor or center or some other reference.

3.4.4.15 offsetY_

```
float ObstacleSensor::offsetY_
```

3.4.4.16 sensorAngle_

```
float ObstacleSensor::sensorAngle_
```

3.4.4.17 sensorGridAngle_

```
float ObstacleSensor::sensorGridAngle_
```

Angle of sensor relative to the grid (i.e. sensorAngle_ + heading_)

3.4.4.18 sonar_

```
NewPing ObstacleSensor::sonar_
```

NewPing object causes HC-SR04 sensor pulses and enable.

3.4.4.19 soundcm_

```
float ObstacleSensor::soundcm_ = 0.0343f  [static], [private]
```

Stores calculated speed of sound in cm/ms - defaults to speed at 20 degrees Celcius.

3.4.4.20 triggerPin_

```
uint8_t ObstacleSensor::triggerPin_  [private]
```

Pin definitions.

3.4.4.21 unitVect_

```
int8_t ObstacleSensor::unitVect_[2]
```

Unit vector of vector from sensor to detected obstacle (i.e xComp + yComp)

3.4.4.22 xPos_

```
float ObstacleSensor::xPos_  [static], [private]
```

AVS pozyx x position.

3.4.4.23 yPos_

```
float ObstacleSensor::yPos_  [static], [private]
```

AVS pozyx y position.

The documentation for this class was generated from the following files:

- D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/bot↩ Main/ObstacleSensor.h
- D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/bot↩ Main/ObstacleSensor.cpp

Chapter 4

File Documentation

4.1 D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_↩ edpAutoCar/botMain/botMain.ino File Reference

```
#include "ObstacleSensor.h"
#include "ObstacleDetection.h"
#include "common.h"
```

Macros

- #define LEFT 270∗(PI/180);
- #define FORWARD 0∗(PI/180);
- #define RIGHT 90∗(PI/180);
- #define BACKWARD 180∗(PI/180);
- #define DIAG_FOR_RIGHT 45∗(PI/180);
- #define DIAG_FOR_LEFT 315∗(PI/180);
- #define DIAG_BACK_LEFT 225∗(PI/180);

Functions

- void setup ()
- void loop ()
- void testBlueToothGrid ()

  fake function - to be deleted - just there to see if nav.grid_ array can be accessed from here

Variables

- float avsHeading_ = 0∗(PI/180)
- float avsX_ = 0
- float avsY_ = 0
- Navigator nav
- const uint8_t frontTriggerPin = 17

  Front sensor arduino trigger pin number.
- const uint8_t frontEchoPin = frontTriggerPin

  Front sensor arduino echo pin number.
- const float frontXOffset = 0

  Front sensor x offset on chassis from pozyx location.
- const float frontYOffset = 10

  Front sensor y offset on chassis from pozyx location.
- const float frontsensorAngle = FORWARD

  Front sensor angle relative to forward of chassis.
- const uint8_t leftTriggerPin = 15

  Left sensor arduino trigger pin number.
- const uint8_t leftEchoPin = leftTriggerPin

  Left sensor arduino echo pin number.
- const float leftXOffset = -5

  Left sensor x offset on chassis from pozyx location.
- const float leftYOffset = 2

  Left sensor y offset on chassis from pozyx location.
- const float leftsensorAngle = LEFT

  Left sensor angle relative to forward of chassis.
- const uint8_t rightTriggerPin = 19

  Right sensor arduino trigger pin number.
- const uint8_t rightEchoPin = rightTriggerPin

  Right sensor arduino echo pin number.
- const float rightXOffset = 5

  Right sensor x offset on chassis from pozyx location.
- const float rightYOffset = 2

  Right sensor y offset on chassis from pozyx location.
- const float rightsensorAngle = RIGHT

  Right sensor angle relative to forward of chassis.
- const uint8_t dhtPin = 14

  DHT22 Sensor arduino pin number.
- ObstacleSensor frontSensor (frontTriggerPin, frontEchoPin, frontXOffset, frontYOffset, frontsensorAngle)

  Front sensor ObstacleSensor object initialisation - used to detect obstacles at front of AVS.
- ObstacleSensor leftSensor (leftTriggerPin, leftEchoPin, leftXOffset, leftYOffset, leftsensorAngle)

  Left sensor ObstacleSensor object initialisation - used to detect obstacles at left of AVS.
- ObstacleSensor rightSensor (rightTriggerPin, rightEchoPin, rightXOffset, rightYOffset, rightsensorAngle)

  Right sensor ObstacleSensor object initialisation - used to detect obstacles at right of AVS.

4.1.1 Macro Definition Documentation

4.1.1.1 BACKWARD

```
#define BACKWARD 180*(PI/180);
```

4.1.1.2 DIAG_BACK_LEFT

```
#define DIAG_BACK_LEFT 225*(PI/180);
```

4.1.1.3 DIAG_FOR_LEFT

```
#define DIAG_FOR_LEFT 315*(PI/180);
```

4.1.1.4 DIAG_FOR_RIGHT

```
#define DIAG_FOR_RIGHT 45*(PI/180);
```

4.1.1.5 FORWARD

```
#define FORWARD 0*(PI/180);
```

4.1.1.6 LEFT

```
#define LEFT 270*(PI/180);
```

4.1.1.7 RIGHT

```
#define RIGHT 90*(PI/180);
```

4.1.2 Function Documentation

4.1.2.1 loop()

```
void loop ( )
```

4.1.2.2 setup()

```
void setup ( )
```

4.1.2.3 testBlueToothGrid()

```
void testBlueToothGrid ( )
```

fake function - to be deleted - just there to see if nav.grid_ array can be accessed from here

4.1.3 Variable Documentation

4.1.3.1 avsHeading_

```
float avsHeading_ = 0*(PI/180)
```

***** botMain.ino - Main .ino file for arduino (for now, someone else may have a better one) ***** // ***** INS↩ TRUCTIONS: ***** // ***** 1. Ensure all libraries are installed by navigating Sketch > Include Library > Manage Libraries ***** // ***** and searching for the following libraries: (i) DHT ***** // ***** (ii) Adafuit Unified Sensor ***** // ***** (iii) Adafruit AM2315 ***** // ***** 2.

4.1.3.2 avsX_

```
float avsX_ = 0
```

4.1.3.3 avsY_

```
float avsY_ = 0
```

4.1.3.4 dhtPin

```
const uint8_t dhtPin = 14
```

DHT22 Sensor arduino pin number.

4.1.3.5 frontEchoPin

```
const uint8_t frontEchoPin = frontTriggerPin
```

Front sensor arduino echo pin number.

4.1.3.6 frontSensor

```
ObstacleDetection ods & frontSensor
```

Front sensor ObstacleSensor object initialisation - used to detect obstacles at front of AVS.

Obstacle detection system object initialisation - used to activate ObstacleSensor objects as required and pass on information to the navigation system

4.1.3.7 frontsensorAngle

```
const float frontsensorAngle = FORWARD
```

Front sensor angle relative to forward of chassis.

4.1.3.8 frontTriggerPin

```
const uint8_t frontTriggerPin = 17
```

Front sensor arduino trigger pin number.

4.1.3.9 frontXOffset

```
const float frontXOffset = 0
```

Front sensor x offset on chassis from pozyx location.

4.1.3.10 frontYOffset

```
const float frontYOffset = 10
```

Front sensor y offset on chassis from pozyx location.

4.1.3.11 leftEchoPin

```
const uint8_t leftEchoPin = leftTriggerPin
```

Left sensor arduino echo pin number.

4.1.3.12 leftSensor

```
ObstacleSensor leftSensor(leftTriggerPin, leftEchoPin, leftXOffset, leftYOffset, leftsensorAngle)
```

Left sensor ObstacleSensor object initialisation - used to detect obstacles at left of AVS.

4.1.3.13 leftsensorAngle

```
const float leftsensorAngle = LEFT
```

Left sensor angle relative to forward of chassis.

4.1.3.14 leftTriggerPin

```
const uint8_t leftTriggerPin = 15
```

Left sensor arduino trigger pin number.

4.1.3.15 leftXOffset

```
const float leftXOffset = -5
```

Left sensor x offset on chassis from pozyx location.

4.1.3.16 leftYOffset

```
const float leftYOffset = 2
```

Left sensor y offset on chassis from pozyx location.

4.1.3.17 nav

```
Navigator nav
```

4.1.3.18 rightEchoPin

```
const uint8_t rightEchoPin = rightTriggerPin
```

Right sensor arduino echo pin number.

4.1.3.19 rightSensor

```
ObstacleSensor rightSensor(rightTriggerPin, rightEchoPin, rightXOffset, rightYOffset, rightsensorAngle)
```

Right sensor ObstacleSensor object initialisation - used to detect obstacles at right of AVS.

4.1.3.20 rightsensorAngle

```
const float rightsensorAngle = RIGHT
```

Right sensor angle relative to forward of chassis.

4.1.3.21 rightTriggerPin

```
const uint8_t rightTriggerPin = 19
```

Right sensor arduino trigger pin number.

4.1.3.22 rightXOffset

```
const float rightXOffset = 5
```

Right sensor x offset on chassis from pozyx location.

4.1.3.23 rightYOffset

```
const float rightYOffset = 2
```

Right sensor y offset on chassis from pozyx location.

4.2 D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/botMain/common.h File Reference

Variables

- static const int HEIGHT = 10
- static const int WIDTH = 10
- static const int DATA = 3
- const int ELEMENT_XPOS = 0
- const int ELEMENT_YPOS = 1
- const int ELEMENT_VALUE = 2

4.2.1 Variable Documentation

4.2.1.1 DATA

```
const int DATA = 3  [static]
```

4.2.1.2 ELEMENT_VALUE

```
const int ELEMENT_VALUE = 2
```

4.2.1.3 ELEMENT_XPOS

```
const int ELEMENT_XPOS = 0
```

4.2.1.4 ELEMENT_YPOS

```
const int ELEMENT_YPOS = 1
```

4.2.1.5 HEIGHT

```
const int HEIGHT = 10  [static]
```

4.2.1.6 WIDTH

```
const int WIDTH = 10  [static]
```

4.3 D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/botMain/Navigator.cpp File Reference

```
#include "Navigator.h"
```

4.4 D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/botMain/Navigator.h File Reference

```
#include <Arduino.h>
#include "common.h"
```

Classes

- class Navigator

Variables

- const int HEIGHT
- const int WIDTH
- const int DATA
- const int ELEMENT_XPOS
- const int ELEMENT_YPOS
- const int ELEMENT_VALUE

4.4.1 Variable Documentation

4.4.1.1 DATA

```
const int DATA
```

4.4.1.2 ELEMENT_VALUE

```
const int ELEMENT_VALUE
```

4.4.1.3 ELEMENT_XPOS

```
const int ELEMENT_XPOS
```

4.4.1.4 ELEMENT_YPOS

```
const int ELEMENT_YPOS
```

4.4.1.5 HEIGHT

```
const int HEIGHT
```

4.4.1.6 WIDTH

```
const int WIDTH
```

4.5 D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_↩ edpAutoCar/botMain/NewPing.cpp File Reference

```
#include "NewPing.h"
```

Functions

- ISR (TIMER2_COMPA_vect)

Variables

- void(∗ intFunc)()
- void(∗ intFunc2)()
- unsigned long _ms_cnt_reset
- volatile unsigned long _ms_cnt

4.5.1 Function Documentation

4.5.1.1 ISR()

```
ISR (
             TIMER2_COMPA_vect  )
```

4.5.2 Variable Documentation

4.5.2.1 _ms_cnt

```
volatile unsigned long _ms_cnt
```

4.5.2.2 _ms_cnt_reset

```
unsigned long _ms_cnt_reset
```

4.5.2.3 intFunc

```
void(* intFunc) ()
```

4.5.2.4 intFunc2

```
void(* intFunc2) ()
```

4.6 D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/botMain/NewPing.h File Reference

```
#include <WProgram.h>
#include <pins_arduino.h>
```

Classes

- class NewPing

Macros

- #define MAX_SENSOR_DISTANCE 500
- #define US_ROUNDTRIP_CM 57
- #define US_ROUNDTRIP_IN 146
- #define ONE_PIN_ENABLED true
- #define ROUNDING_ENABLED false
- #define URM37_ENABLED false
- #define TIMER_ENABLED true
- #define NO_ECHO 0
- #define MAX_SENSOR_DELAY 5800
- #define ECHO_TIMER_FREQ 24
- #define PING_MEDIAN_DELAY 29000
- #define PING_OVERHEAD 5
- #define PING_TIMER_OVERHEAD 13
- #define US_ROUNDTRIP_CM 50
- #define US_ROUNDTRIP_IN 127
- #define NewPingConvert(echoTime, conversionFactor) (max(((unsigned int)echoTime + conversionFactor / 2) / conversionFactor, (echoTime ? 1 : 0)))
- #define PING_OVERHEAD 1
- #define PING_TIMER_OVERHEAD 1
- #define TIMER_ENABLED false
- #define DO_BITWISE false

4.6.1 Macro Definition Documentation

4.6.1.1 DO_BITWISE

```
#define DO_BITWISE false
```

4.6.1.2 ECHO_TIMER_FREQ

```
#define ECHO_TIMER_FREQ 24
```

4.6.1.3 MAX_SENSOR_DELAY

```
#define MAX_SENSOR_DELAY 5800
```

4.6.1.4 MAX_SENSOR_DISTANCE

```
#define MAX_SENSOR_DISTANCE 500
```

4.6.1.5 NewPingConvert

```
#define NewPingConvert(
             echoTime,
             conversionFactor ) (max(((unsigned int)echoTime + conversionFactor / 2) / conversion↩
Factor, (echoTime ?  1 :  0)))
```

4.6.1.6 NO_ECHO

```
#define NO_ECHO 0
```

4.6.1.7 ONE_PIN_ENABLED

```
#define ONE_PIN_ENABLED true
```

4.6.1.8 PING_MEDIAN_DELAY

```
#define PING_MEDIAN_DELAY 29000
```

4.6.1.9 PING_OVERHEAD [1/2]

```
#define PING_OVERHEAD 5
```

4.6.1.10 PING_OVERHEAD [2/2]

```
#define PING_OVERHEAD 1
```

4.6.1.11 PING_TIMER_OVERHEAD [1/2]

```
#define PING_TIMER_OVERHEAD 13
```

4.6.1.12 PING_TIMER_OVERHEAD [2/2]

```
#define PING_TIMER_OVERHEAD 1
```

4.6.1.13 ROUNDING_ENABLED

```
#define ROUNDING_ENABLED false
```

4.6.1.14 TIMER_ENABLED [1/2]

```
#define TIMER_ENABLED true
```

4.6.1.15 TIMER_ENABLED [2/2]

```
#define TIMER_ENABLED false
```

4.6.1.16 URM37_ENABLED

```
#define URM37_ENABLED false
```

4.6.1.17 US_ROUNDTRIP_CM [1/2]

```
#define US_ROUNDTRIP_CM 57
```

4.6.1.18 US_ROUNDTRIP_CM [2/2]

```
#define US_ROUNDTRIP_CM 50
```

4.6.1.19 US_ROUNDTRIP_IN [1/2]

```
#define US_ROUNDTRIP_IN 146
```

4.6.1.20 US_ROUNDTRIP_IN [2/2]

```
#define US_ROUNDTRIP_IN 127
```

4.7 D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/botMain/ObstacleDetection.cpp File Reference

```
#include "ObstacleDetection.h"
```

4.8 D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/botMain/ObstacleDetection.h File Reference

```
#include "ObstacleSensor.h"
#include <stdint.h>
#include "Navigator.h"
```

Classes

- class ObstacleDetection

4.9 D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/botMain/ObstacleSensor.cpp File Reference

```
#include "ObstacleSensor.h"
#include "DHT.h"
```

4.10 D:/OneDrive - UNSW/Alex/ADFA/2018/Semester 1/Engineering Design Practise/git_edpAutoCar/botMain/ObstacleSensor.h File Reference

```
#include "NewPing.h"
```

Classes

- class ObstacleSensor

Index