

Prototyping

Generated by Doxygen 1.9.4

1 Class Documentation	1
1.1 Robot Class Reference	1
1.1.1 Detailed Description	1
1.1.2 Constructor & Destructor Documentation	2
1.1.3 Member Function Documentation	3
2 File Documentation	3
2.1 Robot.cpp File Reference	3
2.1.1 Detailed Description	4
2.2 Robot.h File Reference	4
2.2.1 Detailed Description	5
2.2.2 Enumeration Type Documentation	5
2.3 Robot.h	6
2.4 screen_matrices.h File Reference	6
2.4.1 Detailed Description	7
2.4.2 Variable Documentation	7
2.5 screen_matrices.h	9
Index	11

1 Class Documentation

1.1 Robot Class Reference

Main robot control class that handles line following, obstacle detection and avoidance.

```
#include <Robot.h>
```

Public Member Functions

- [Robot](#) (uint8_t ENA, uint8_t ENB, uint8_t IN1, uint8_t IN2, uint8_t IN3, uint8_t IN4, uint8_t IR_LEFT, uint8_t IR_RIGHT, uint8_t SERVO, uint8_t TRIGGER_PIN, uint8_t ECHO_PIN, uint8_t S0, uint8_t S1, uint8_t S2, uint8_t S3, uint8_t sensorOut, [RobotState](#) initState, uint8_t k, uint8_t distance)

Constructor for [Robot](#) class.

- void [init](#) ()

Initialize robot hardware and pins.

- void [run](#) ()

Main robot operation function, called repeatedly in loop.

1.1.1 Detailed Description

Main robot control class that handles line following, obstacle detection and avoidance.

1.1.2 Constructor & Destructor Documentation

1.1.2.1 Robot() `Robot::Robot (`

```

    uint8_t ENA,
    uint8_t ENB,
    uint8_t IN1,
    uint8_t IN2,
    uint8_t IN3,
    uint8_t IN4,
    uint8_t IR_LEFT,
    uint8_t IR_RIGHT,
    uint8_t SERVO,
    uint8_t TRIGGER_PIN,
    uint8_t ECHO_PIN,
    uint8_t S0,
    uint8_t S1,
    uint8_t S2,
    uint8_t S3,
    uint8_t sensorOut,
    RobotState initState,
    uint8_t k,
    uint8_t distance )
```

Constructor for `Robot` class.

Constructor implementation for `Robot` class.

Parameters

<i>ENA</i>	Enable pin for left motor
<i>ENB</i>	Enable pin for right motor
<i>IN1</i>	Direction control pin 1 for left motor
<i>IN2</i>	Direction control pin 2 for left motor
<i>IN3</i>	Direction control pin 1 for right motor
<i>IN4</i>	Direction control pin 2 for right motor
<i>IR_LEFT</i>	Left infrared sensor pin
<i>IR_RIGHT</i>	Right infrared sensor pin
<i>SERVO</i>	Servo motor control pin
<i>TRIGGER_PIN</i>	Ultrasonic sensor trigger pin
<i>ECHO_PIN</i>	Ultrasonic sensor echo pin
<i>S0</i>	Color sensor frequency scaling selection pin S0
<i>S1</i>	Color sensor frequency scaling selection pin S1
<i>S2</i>	Color sensor photodiode selection pin S2
<i>S3</i>	Color sensor photodiode selection pin S3
<i>sensorOut</i>	Color sensor output pin
<i>initState</i>	Initial state of the robot
<i>k</i>	Proportional control constant
<i>distance</i>	Threshold distance for obstacle detection in cm

Initializes all pins and parameters for the robot

1.1.3 Member Function Documentation

1.1.3.1 `init()` `void Robot::init ()`

Initialize robot hardware and pins.

Initialize all pins and components.

Sets up pin modes for motors, sensors, and initializes servo and LED matrix

1.1.3.2 `run()` `void Robot::run ()`

Main robot operation function, called repeatedly in loop.

Main robot operation function.

State machine that controls robot behavior based on current state

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

- [Robot.h](#)
- [Robot.cpp](#)

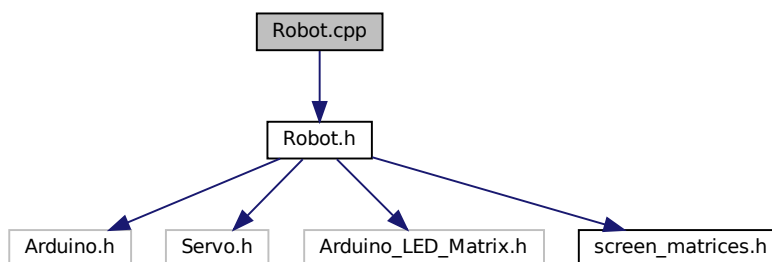
2 File Documentation

2.1 Robot.cpp File Reference

Implementation of the [Robot](#) class methods.

```
#include "Robot.h"
```

Include dependency graph for Robot.cpp:



2.1.1 Detailed Description

Implementation of the [Robot](#) class methods.

Author

Group C4

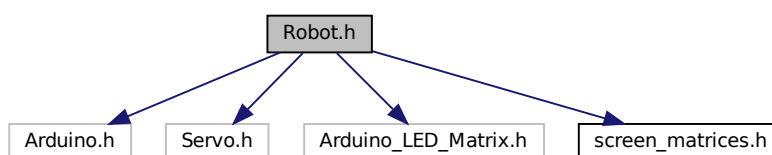
Date

2025

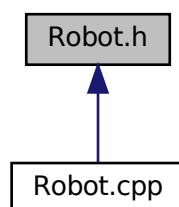
2.2 Robot.h File Reference

[Robot](#) class definition for line-following robot with obstacle detection.

```
#include <Arduino.h>
#include <Servo.h>
#include "Arduino_LED_Matrix.h"
#include "screen_matrices.h"
Include dependency graph for Robot.h:
```



This graph shows which files directly or indirectly include this file:



Classes

- class [Robot](#)

Main robot control class that handles line following, obstacle detection and avoidance.

Enumerations

- enum [RobotState](#) { [FOLLOW_LINE](#) , [INSPECT_OBSTACLE](#) , [AVOID_OBSTACLE](#) }
Defines the possible states of the robot.
- enum [Colors](#) { [RED](#) , [GREEN](#) , [BLUE](#) }
Defines the colors that can be detected by the color sensor.

2.2.1 Detailed Description

[Robot](#) class definition for line-following robot with obstacle detection.

Author

Group C4

Date

2025

2.2.2 Enumeration Type Documentation

2.2.2.1 Colors enum [Colors](#)

Defines the colors that can be detected by the color sensor.

Enumerator

RED	Red color
GREEN	Green color
BLUE	Blue color

2.2.2.2 RobotState enum [RobotState](#)

Defines the possible states of the robot.

Enumerator

FOLLOW_LINE	Robot is following a line
INSPECT_OBSTACLE	Robot is inspecting an obstacle
AVOID_OBSTACLE	Robot is avoiding an obstacle

2.3 Robot.h

[Go to the documentation of this file.](#)

```

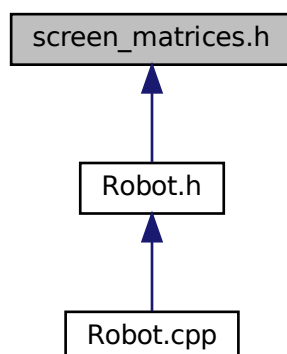
1
2
3
4
5
6
7
8 #ifndef ROBOT_H
9 #define ROBOT_H
10
11 #include <Arduino.h>
12 #include <Servo.h>
13 #include "Arduino_LED_Matrix.h"
14 #include "screen_matrices.h"
15
16
17
18
19
20 enum RobotState {
21     FOLLOW_LINE,
22     INSPECT_OBSTACLE,
23     AVOID_OBSTACLE
24 };
25
26
27
28
29
30 enum Colors {
31     RED,
32     GREEN,
33     BLUE
34 };
35
36
37
38
39
40 class Robot {
41 private:
42     Servo myservo;
43     ArduinoLEDMatrix matrix;
44     RobotState state;
45     uint8_t ENA, ENB, IN1, IN2, IN3, IN4;
46     uint8_t IR_LEFT, IR_RIGHT, SERVO;
47     uint8_t TRIGGER_PIN, ECHO_PIN;
48     uint8_t S0, S1, S2, S3, sensorOut;
49     uint32_t timerError;
50     uint8_t k;
51     uint8_t distance;
52     void followLine();
53
54
55
56
57
58
59
60
61     void avoidObstacle();
62
63
64
65
66     void inspectObstacle();
67
68
69
70
71
72     void motorLeft(short speed);
73
74
75
76
77
78     void motorRight(short speed);
79
80
81
82
83
84     bool checkDistance();
85
86
87
88
89
90     Colors checkColors();
91 public:
92     Robot(uint8_t ENA,
93           uint8_t ENB,
94           uint8_t IN1,
95           uint8_t IN2,
96           uint8_t IN3,
97           uint8_t IN4,
98           uint8_t IR_LEFT,
99           uint8_t IR_RIGHT,
100          uint8_t SERVO,
101          uint8_t TRIGGER_PIN,
102          uint8_t ECHO_PIN,
103          uint8_t S0,
104          uint8_t S1,
105          uint8_t S2,
106          uint8_t S3,
107          uint8_t sensorOut,
108          RobotState initState,
109          uint8_t k,
110          uint8_t distance);
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137     void init();
138
139
140
141
142     void run();
143 };
144
145 #endif //ROBOT_H

```

2.4 screen_matrices.h File Reference

LED matrix display patterns for robot status indication.

This graph shows which files directly or indirectly include this file:



Variables

- const uint32_t [stopSign](#) []
Stop sign pattern for LED matrix.
- const uint32_t [forwardSign](#) []
Forward arrow pattern for LED matrix.
- const uint32_t [rightSign](#) []
Right arrow pattern for LED matrix.
- const uint32_t [leftSign](#) []
Left arrow pattern for LED matrix.

2.4.1 Detailed Description

LED matrix display patterns for robot status indication.

Author

Group C4

Date

2025

2.4.2 Variable Documentation

2.4.2.1 forwardSign `const uint32_t forwardSign[]`

Initial value:

```
= {  
    0x600f01f,  
    0x83fc7fe0,  
    0xf00f00f0  
}
```

Forward arrow pattern for LED matrix.

Displays an upward arrow when robot is moving forward

2.4.2.2 leftSign `const uint32_t leftSign[]`

Initial value:

```
= {  
    0x400c01c,  
    0x3fc3fc1,  
    0xc00c0040  
}
```

Left arrow pattern for LED matrix.

Displays a left-pointing arrow when robot is turning left

2.4.2.3 rightSign `const uint32_t rightSign[]`

Initial value:

```
= {  
    0x2003003,  
    0x83fc3fc0,  
    0x38030020  
}
```

Right arrow pattern for LED matrix.

Displays a right-pointing arrow when robot is turning right

2.4.2.4 stopSign `const uint32_t stopSign[]`

Initial value:

```
= {  
    0xf010820,  
    0x42f42f42,  
    0x41080f0  
}
```

Stop sign pattern for LED matrix.

Displays an octagonal stop sign pattern when obstacle is detected

2.5 screen_matrices.h

[Go to the documentation of this file.](#)

```
1
13 const uint32_t stopSign[] = {
14     0xf010820,
15     0x42f42f42,
16     0x41080f0
17 };
18
24 const uint32_t forwardSign[] = {
25     0x600f01f,
26     0x83fc7fe0,
27     0xf00f00f0
28 };
29
35 const uint32_t rightSign[] = {
36     0x2003003,
37     0x83fc3fc0,
38     0x38030020
39 };
40
46 const uint32_t leftSign[] = {
47     0x400c01c,
48     0x3fc3fc1,
49     0xc00c0040
50 };
```


Index

AVOID_OBSTACLE

Robot.h, [5](#)

BLUE

Robot.h, [5](#)

Colors

Robot.h, [5](#)

FOLLOW_LINE

Robot.h, [5](#)

forwardSign

screen_matrices.h, [7](#)

GREEN

Robot.h, [5](#)

init

Robot, [3](#)

INSPECT_OBSTACLE

Robot.h, [5](#)

leftSign

screen_matrices.h, [8](#)

RED

Robot.h, [5](#)

rightSign

screen_matrices.h, [8](#)

Robot, [1](#)

init, [3](#)

Robot, [2](#)

run, [3](#)

Robot.cpp, [3](#)

Robot.h, [4](#), [6](#)

AVOID_OBSTACLE, [5](#)

BLUE, [5](#)

Colors, [5](#)

FOLLOW_LINE, [5](#)

GREEN, [5](#)

INSPECT_OBSTACLE, [5](#)

RED, [5](#)

RobotState, [5](#)

RobotState

Robot.h, [5](#)

run

Robot, [3](#)

screen_matrices.h, [6](#), [9](#)

forwardSign, [7](#)

leftSign, [8](#)

rightSign, [8](#)

stopSign, [8](#)

stopSign

screen_matrices.h, [8](#)