Prototyping

Generated by Doxygen 1.9.4

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

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

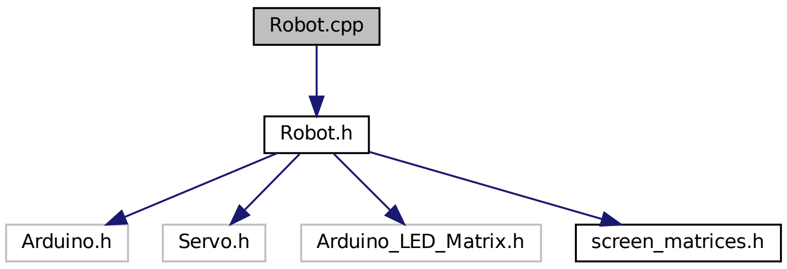
2 File Documentation

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

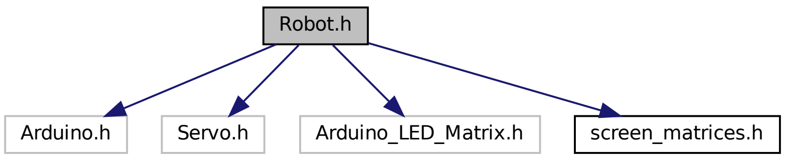
2025

2.2 Robot.h File Reference

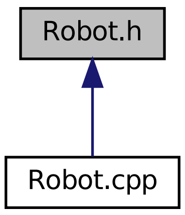
Robot class definition for line-following robot with obstacle detection.

```
#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

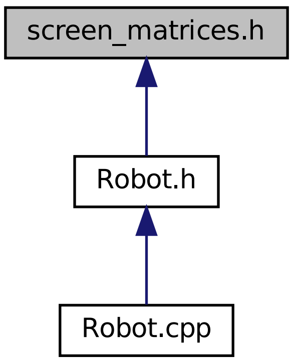
Go to the documentation of this file.

```

#ifndef ROBOT_H
#define ROBOT_H

#include <Arduino.h>
#include <Servo.h>
#include "Arduino_LED_Matrix.h"
#include "screen_matrices.h"

enum RobotState {
  FOLLOW_LINE,
  INSPECT_OBSTACLE,
  AVOID_OBSTACLE
};

enum Colors {
  RED,
  GREEN,
  BLUE
};

class Robot {
private:
  Servo myservo;
  ArduinoLEDMatrix matrix;
  RobotState state;
  uint8_t ENA, ENB, IN1, IN2, IN3, IN4;
  uint8_t IR_LEFT, IR_RIGHT, SERVO;
  uint8_t TRIGGER_PIN, ECHO_PIN;
  uint8_t S0, S1, S2, S3, sensorOut;
  uint32_t timerError;
  uint8_t k;
  uint8_t distance;
  void followLine();

  void avoidObstacle();

  void inspectObstacle();

  void motorLeft(short speed);

  void motorRight(short speed);

  bool checkDistance();

  Colors checkColors();
public:
  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);

  void init();

  void run();
};

#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.

```

const uint32_t stopSign[] = {
    0xf010820,
    0x42f42f42,
    0x41080f0
};

const uint32_t forwardSign[] = {
    0x600f01f,
    0x83fc7fe0,
    0xf00f00f0
};

const uint32_t rightSign[] = {
    0x2003003,
    0x83fc3fc0,
    0x38030020
};

const uint32_t leftSign[] = {
    0x400c01c,
    0x3fc3fc1,
    0xc00c0040
};
```

Index