**Design and Development of Obstacle Avoiding Robot using Camera and Laser Line Generator**



**Project Report**

***Submitted in partial fulfilment of the requirements***

***Of the degree of***

**Bachelor of Engineering**

**(Mechatronics Engineering) By**

## Aishwarya Chavan ID No.: TU6F1516001 Konika Khatri ID No.: TU6F1516021 Yadnesh Kasbe ID No.: TU6F1516013

**Under the guidance of Prof. Rajkumar Patil-Tekale**

**Department of Mechatronics Engineering**

**TERNA ENGINEERING COLLEGE**

**Plot No.12, Sector-22, Phase II, Nerul (W), Navi Mumbai-400706**

**UNIVERSITY OF MUMBAI**

## 2018-2019

**CERTIFICATE**



This is to certify that the project entitled **“Design and Development of Obstacle Avoiding Robot using Camera and Laser Line Generator”** is a bonafide work of **“Ms. Aishwarya Chavan” (TU6f1516001)**, **“Ms. Konika Khatri” (TU6F1516023)** and **“Mr. Yadnesh Kasbe” (TU6f1516013)** submitted to the University of Mumbai in partial fulfilment of the requirement for the award of the degree of **“BE”** in **“Mechatronics Engineering”.**

\_

( Mr. Rajkumar Patil-Tekale ) ( Prof. Vikram Vyawahare ) Guide Head Of Department

( Dr. L K Ragha ) Principal

## PROJECT REPORT APPROVAL FOR B.E.



This project report entitled ***Design and Development of Obstacle Avoiding Robot using Camera and Laser Line Generator.***

***Aishwarya Chavan TU6F1516001 Konika Khatri TU6F1516021***

***Yadnesh Kasbe TU6F1516013***

Is approved for the degree of ***Bachelor in Engineering*** in ***Mechatronics Engineering***.

Examiners

1.

2.

Co-ordinator

\_

Date: Place:

## DECLARATION



We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Aishwarya Chavan TU6F1516001

Konika Khatri TU6F1516021 \_

Yadnesh Kasbe TU6F1516013

## ACKNOWLEDGEMENT



The project has attained completion through the efforts of many people. We would like to thank our internal guide **Mr. Rajkumar Patil-Tekale** for providing guidelines and giving timely suggestions and inputs to improve the project. He has been a driving force in this project. He has taken immense efforts to guide and explain the concepts behind this project. A special mention should be made about Terna Engineering College and the staff of Mechatronics Engineering department.

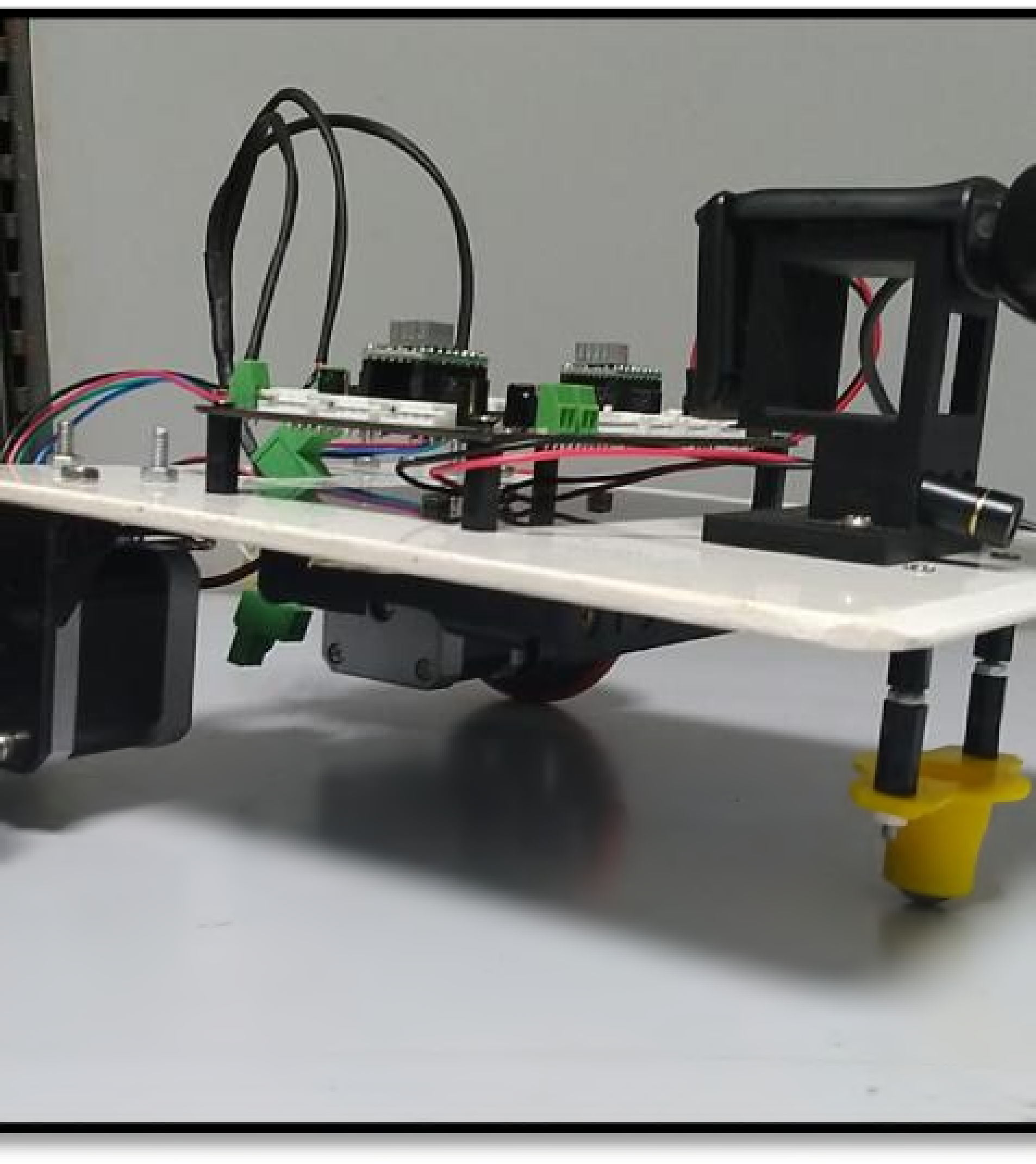
We also extend our humble appreciation towards our project co-ordinator, Prof. Dattatray Shinde and our H.O.D., Prof. Vikram Vyawahare for helping us with the best of their knowledge and for their unconditional and tireless support.

Aishwarya Chavan TU6F1516001 Konika Khatri TU6F1516021

Yadnesh Kasbe TU6F1516013

## `

**PHOTOGRAPH OF THE PROJECT**



|  |  |
| --- | --- |
| **INDEX**  **TITLE** | **PAGE NO.** |
| Abstract | 1 |
| Chapter 1: Introduction | 2 |
| 1.1 Background | 5 |
| 1.2 Scope Of The Project | 5 |
| 1.3 Problem Statement | 5 |
| Chapter 2: Literature Survey | 6 |
| 2.1 Technical/ Research Papers | 7 |
| Chapter 3: Component | 10 |
| 3.1 Mechanical Components | 11 |
| 3.1.1 Chassis | 11 |
| 3.1.2 Camera And Laser Stand | 12 |
| 3.1.3 Wheel | 13 |
| 3.1.4 L Bracket Clamps | 14 |
| 3.1.5 Caster Wheel | 15 |
| 3.1.6 Battery Case | 15 |
| 3.2 Electronic And Electrical Components | 16 |
| 3.2.1 Hellobot Base Board | 16 |
| 3.2.2 Stepper Motor | 22 |
| 3.2.3 USB to TTL Connector | 24 |
| 3.2.4 Battery | 25 |
| 3.2.5 Camera | 26 |
| 3.2.6 Laser Line Generator | 27 |
| 3.3 Block Diagram | 28 |
| Chapter 4: Implementation | 29 |
| 4.1 Image Processing | 30 |
| Chapter 5: Conclusion | 42 |

|  |  |  |
| --- | --- | --- |
|  | **TITLE** | **PAGE NO.** |
| Applications |  | 43 |
| Future Scope |  | 42 |
| References |  | 44 |

**ABSTRACT**

The project title is “**Obstacle Avoidance Robot Using Camera and Laser Line Generator**”. As the technology advances, the need for automated device increases. So, to fulfil the industries requirement it has become mandatory to build automatic robots, which can move from one work station to another by avoiding obstacles coming in the path. Thus to overcome the issue, “**Obstacle Avoidance Robot”** were build**.**

The basic components used in this project, are LASER, camera, Microcontroller and motors. LASER is the acronym for "Light Amplification by Stimulated Emission of Radiation". It was first built in 1960 by Theodre H. Maiman at Hughes Research Laboratories. The LASER is used for sensing the surrounding environment. Fine objects like needle can also be detected by laser, but ultrasonic sensor cannot detect such fine object. LASER is used as its accuracy of sensing environment is better than other sensors. With the help of microcontroller and camera, the moment of robot is controlled by actuating the motor in appropriate direction.

The main objective of this robot is to change its direction automatically whenever any obstacle comes in its way. This technology is used everywhere from industries and home appliances.

# CHAPTER 1 INTRODUCTION

### CHAPTER 1: INTRODUCTION

“Mechatronics” is an interdisciplinary branch of the engineering and science that includes mechanical engineering, electronic engineering, computer science, information engineering, and others. Robotics deals with the design, construction, operation, and use of robots, as well as computer systems for their control, sensory feedback and information processing.

In robotics, obstacle avoidance is the task of satisfying some control objective subject to non-intersection or non-collision position constraints. Normally obstacle avoidance is considered to be distinct from path planning in that one is usually implemented as a reactive control law while the other involves the pre-computation of an obstacle-free path which a controller will then guide a robot along. With recent advanced in the autonomous vehicles sector, a good and dependable obstacle avoidance feature of a driverless platform is also required to have a robust obstacle detection module.

These technologies are used to develop machines that can substitute for humans and replicate human actions. Robots can be used in many situations and for lots of purposes, but today many are used in dangerous environments (including bomb detection and deactivation), manufacturing processes, or where humans cannot survive (e.g. in space). Robots can take on any form but some are made to resemble humans in appearance. This is said to help in the acceptance of a robot in certain replicative behaviours usually performed by people. Such robots attempt to replicate walking, lifting, speech, cognition, and basically anything a human can do. Many of today's robots are inspired by nature, contributing to the field of bio-inspired robotics.

The concept of creating machines that can operate autonomously dates back to classical times, but research into the functionality and potential uses of robots did not grow substantially until the 20th century. Throughout history, it has been frequently assumed by various scholars, inventors, engineers, and technicians that robots will one day be able to mimic human behaviour and manage tasks in a human-like fashion. Today, robotics is a rapidly growing field, as technological advances continue; researching, designing, and building new robots serve various practical purposes whether domestically commercially

or militarily. Many robots are built to do jobs that are hazardous to people such as defusing bombs, finding survivors in unstable ruins, and exploring mines and shipwrecks. Robotics is also used in STEM (science, technology, engineering, and mathematics) as a teaching aid. The advent of nano-robots, microscopic robots that can be injected into the human body, could revolutionize medicine and human health.

**Obstacle Avoiding Robot and Types:** Obstacle Avoiding Robot using IR Sensor Obstacle Avoiding Robot using Raspberry-PI

Obstacle Avoiding Robot using 8051 Microcontroller Obstacle Avoiding Robot using Webcam

#### BACKGROUND

The project is basically a combination of Mechanical, Electronics and IT domain. The design of titled project aims at knowledge integration and combined application of Mechanical, Electronics and IT domain. As there is great demand of automation and rapid processing the titles project is best example for the same.

#### SCOPE OF THE PROJECT

This project aims to create a webcam based obstacle avoidance robot which will act as an intelligent car.

1. Software control of the line type ( dark or light ) for automatic detection possible
2. Distance sensing and position logging and transmission ( mapping )
3. With change in programming we can use it for weight lifting and auto parking assistance
4. It can also be in driver less cars
5. With few changes this system can also be implemented in drones
6. It can go in hazardous environment where humans cannot go
7. It can be used in hospitals and industries for specific applications
8. It can also be used in home appliances
9. By using 2 cameras we can create a 3D image of the environment
10. It can also be used in military applications example spying
11. By using different cameras we can change the applications example night vision, thermal, etc
12. Assertive navigation system for wheel chair and blind people

#### PROBLEM STATEMENT

To design a system which will avoid the obstacles in its path, to perform particular indoor applications.

# CHAPTER 2 LITERATURE SURVEY

### CHAPTER 2: LITERATURE SURVEY

**2.1 TECHNICAL/ RESEARCH PAPERS:**

|  |  |
| --- | --- |
| Technical Paper 01 | |
| Title: A Moving Object Tracked by Mobile Robot with Real-Time Obstacles Avoidance | |
| Author | Chung-Hao Chen  (University of Tennessee, Knoxville, TN, 37996, USA) Chang Cheng  D. Page  A. Koschan  M. Abidi |
| Published Date | 24 Aug. 2006 |
| Published On | 18th International Conference on Pattern Recognition (ICPR'06) |
| Extract | This paper describes a robotic application that tracks a moving object by utilizing a mobile robot with multiple sensors. The robotic platform uses a visual camera to sense the movement of the desired object and a range sensor to help the robot detect and then avoid obstacles in real time while continuing to track and follow the desired object. In terms of real-time obstacle avoidance capacity, this paper also presents a modified potential field algorithm called dynamic goal potential field algorithm (DGPF) for this robotic application specifically.  Experimental results show that the robotic and intelligent system can fulfil the requirements of tracking an object and avoiding obstacles  simultaneously when the object is moving |
| Link | https://ieeexplore.ieee.org/abstract/document/1699715 |

|  |  |
| --- | --- |
| Technical Paper 02 | |
| Title:Robot path obstacle locator using webcam and laser emitter | |
| Author | Shamed Shojaeipour Sallehuddin Mohamed Haris Ali Shojaeipour Rassoul Keshvari Shirvan,  Muhammad Khalid Zachariah |
| Published Date | 25 August 2010. |
| Published On | Science Direct |
| Extract | This paper provides a framework for a measurement system with which a mobile robot could measure its distance from obstacles using a single robot mounted webcam and a laser emitter. Using images captured by the webcam, the presence of obstacles are first identified. Then, using the rangefinder, the distance to the obstacle is calculated. Future work would be to use the rangefinder measurements as input commands to an actual robot. The system would also be further developed for use in real  outdoor settings with more variability in ambient light conditions. |
| Link | https://[www.sciencedirect.com/science/article/pii/S1875389210005626](http://www.sciencedirect.com/science/article/pii/S1875389210005626) |

|  |  |
| --- | --- |
| Technical Paper 03 | |
| Title: Approaching Camera-based Real-World Navigation Using Object Recognition | |
| Author | Zejia Zheng Xie He  Juyang Weng |
| Published Date | 2015 |
| Extract | In this paper we design an on-line learning agent for self-navigation based on object recognition. The agent is purely vision-based with inexpensive webcams in comparison with the laser based approaches with costly scanners suffering from a series of failures such as inability to recognize wet road surfaces, dark surfaces and objects at large distances. The agent is able to attend to important objects in the current visual input and take corresponding actions according to the recognition result. The system learns online and performs in real-time, minimizing cost of data collection and manual labelling. Our agent demonstrates robust performance in validation and generalization testing scenarios.  The next step in our research is to apply our system to outdoor environment navigation instead of structured indoor environments. Challenges include increased appearance variances compared to this current experiment, and ballooning network size, which may slow  down the real-time learning |

# CHAPTER 3 COMPONENTS

### CHAPTER 3: COMPONENTS

#### MECHANICAL COMPONENTS

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | **Quantity** | **Component Name** | **Quantity** |
| Chassis | 1 No. | Camera and Laser Stand | 1 No. |
| Castor wheel | 1 No. | Wheel | 2 Nos. |
| Brackets | 2 Nos. | Link Connector | 4 Nos. |
| Battery case | 1 No. | Allen Bolts M4 x 6 | 16 Nos. |
| Allen Bolts M4 x 10 | 4 Nos. | Allen Bolts M3 x 6 | 10 Nos. |
| Allen Bolts M3 x 10 | 6 Nos. | CSK M3 x 10 | 4 Nos. |
| Round Head BoltM2.5x16 | 8 Nos. | Round Head Bolt M2 x 16 | 2 Nos. |
| Round Head Bolt M2x10 | 12 Nos. | Hex Nut M2 | 14 Nos. |
| Hex Nut M2.5 | 8 Nos. | Hex Nut M3 | 16 Nos. |
| Hex Nut M4 | 8 Nos. | Hex Nut M8 | 2 Nos. |
| Length Spacer 10mm | 1 No. | Length spacer 3mm | 1 No. |
| Allen Key( 3x2.5x1.5)mm | 1 No. | Spanner | 1 No. |

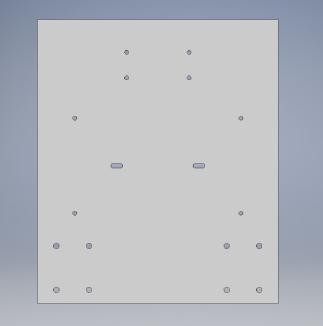
* + 1. **Chassis**

It is the base frame of the robot on which the mechanical as well as electronic components are mounted.

The chassis is made of acrylic sheet. It is also referred as PMMA sheet. PMMA (Polymethyl Methacrylate) or Acrylic is also known as Acrylic glass or Plexiglas.

The chassis used here is designed in a rectangular shape with dimensions as Length= 220 mm, Breadth= 170 mm and Thickness of sheet is 3 mm.

Dimensions: 220mm \* 170mm \* 3 mm

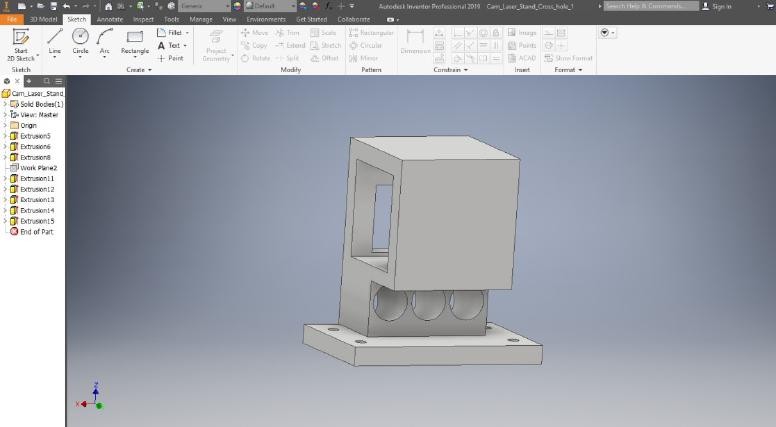


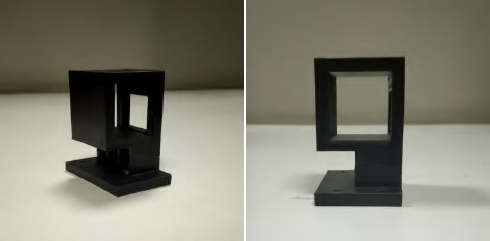
#### Camera and Laser Stand

As mentioned in the title ‘camera’ and ‘laser line generator’ are the main components of the Obstacle Avoiding Robot. To place the camera and laser line generator on the chassis the following stand is designed as per the size of camera and laser.

The camera used is Logitech C310 HD.

The stand makes it easy to rest the camera in such a way that it can record the surrounding in a proper way and also to insert the laser line generator in a hole such that it can remain steady and we can observe the perfectly dark and clear laser line from it.







Camera and Laser Line Generator Mounted on Stand

#### Wheel

No. Of wheels used: 2

These2 wheels are placed at the rear portion of Chassis. These wheels are mounted on the shaft of stepper motors on the either sides.



#### L Bracket Clamps

It enables the easy rolling movement of the robot. They are essentially housings that include a wheel or ball and a mounting to install the caster to objects.

It is attached to chassis at the front end.



#### Caster wheel

Bracket clamps are versatile tools that serve to temporarily hold work securely in the place. In the titled project it is used in the mechanical assembly to hold the stepper motors.



#### Battery Case



* 1. **ELECTRONIC AND ELECTRICAL COMPONENTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Component Name** | **Quantity** | **Component Name** | **Quantity** |
| Hellobot Base Board | 1 No. | UART to TTL Connector | 1 No. |
| Stepper Motor | 2 Nos. | Battery | 4 Nos. |
| Mini USB Cable | 1 No. | Power Cable | 1 No. |
| Web Camera | 1 No. | Laser Line Generator | 1 No. |

#### Hellobot Base Board

Hello Bot Base Board is an ARM Cortex-M0 based development board powered by STM32F072RBT6 microcontroller. It comes with Prime Framework (a suite of software libraries) and is suitable for beginners, hobbyists as well as advance embedded developers.

Breadboard’s modular design makes it fit for any educational embedded and robotics applications. It can be used in projects that require high computing power and need to acquire sensor data quickly & accurately such as drones, wheeled robots. It can also be used to learn the ARM processor in an affordable and simple way.



The Hellobot base board used for the titled project consist of:

* Microcontroller
* Motor Driver

#### Micro-Controller

The STM32F072RB microcontrollers incorporate the high-performance ARM®Cortex®-M0 32-bit RISC core operating at up to 48 MHz frequency, high- speed embedded memories (up to 128 Kbytes of Flash memory and 16 Kbytes of SRAM), and an extensive range of enhanced peripherals and I/Os. All devices offer standard communication interfaces (two I2Cs, two SPI/I2S, one HDMI CEC and four USARTs), one USB Full-speed device (crystal-less), one CAN, one 12- bit ADC, one 12-bit DAC with two channels, seven 16-bit timers, one 32-bit timer and an advanced-control PWM timer.

The STM32F072RB microcontrollers operate in the -40 to +85 °C and -40 to

+105 °C temperature ranges; from a 2.0 to 3.6 V power supply. A comprehensive set of power-saving modes allows the design of low-power applications.

The STM32F072RB microcontrollers include devices in seven different packages ranging from 48 pins to 100 pins with a die form also available upon request.

Depending on the device chosen, different sets of peripherals are included.

These features make the STM32F072RB microcontrollers suitable for a wide range of applications such as application control and user interfaces, hand-held equipment, A/V receivers and digital TV, PC peripherals, gaming and GPS platforms, industrial applications, PLCs, inverters, printers, scanners, alarm systems, video intercoms and HVACs.

#### Features

1. Core: ARM®32-bit Cortex®-M0 CPU, frequency up to 48 MHz
2. Memories
   1. 64 to 128 Kbytes of Flash memory
   2. 16 Kbytes of SRAM with HW parity
   3. CRC calculation unit
   4. Reset and power management
      1. Digital and I/O supply: VDD= 2.0 V to 3.6 V
      2. Analog supply: VDDA= VDD to 3.6 V
      3. Selected I/Os: VDDIO2= 1.65 V to 3.6 V
      4. Power-on/Power down reset (POR/PDR)
      5. Programmable voltage detector (PVD)
      6. Low power modes: Sleep, Stop, Standby
      7. VBAT supply for RTC and backup registers
   5. Clock management

3 to 32 MHz crystal oscillator

1. 32 kHz oscillator for RTC with calibration
2. Internal 8 MHz RC with x6 PLL option
3. Internal 40 kHz RC oscillator
4. Internal 48 MHz oscillator with automatic trimming based on ext. synchronization
5. Up to 87 fast I/Os
   1. All mappable on external interrupt vectors
   2. Up to 68 I/Os with 5V tolerant capability and 19 with independent supply VDDIO2
6. Seven-channel DMA controller
7. One 12-bit, 1.0 μs ADC (up to 16 channels)
   1. Conversion range: 0 to 3.6 V
   2. Separate analog supply: 2.4 V to 3.6 V
8. One 12-bit D/A converter (with 2 channels)
9. Two fast low-power analog comparators with programmable input and output
10. Up to 24 capacitive sensing channels for touch key, linear and rotary touch sensors
11. Calendar RTC with alarm and periodic wakeup from Stop/Standby
12. 12 timers
    1. One 16-bit advanced-control timer for six-channel PWM output
    2. One 32-bit and seven 16-bit timers, with up to four IC/OC, OCN, usable for IR control decoding or DAC control
    3. Independent and system watchdog timers
    4. SysTick timer
13. Communication interfaces
    1. Two I2C interfaces supporting Fast Mode Plus (1 Mbit/s) with 20 mA current sink, one supporting SMBus/PMBus and wakeup
    2. Four USARTs supporting master synchronous SPI and modem control, two with ISO7816 interface, LIN, IrDA, auto baud rate detection and wakeup featu e

r

* 1. Two SPIs (18 M bit/s) with 4 to 16 programmable bit frames, and with I2S interface multiplexed
  2. CAN interface
  3. USB 2.0 full-speed interface, able to run from internal 48 M z oscillator

H

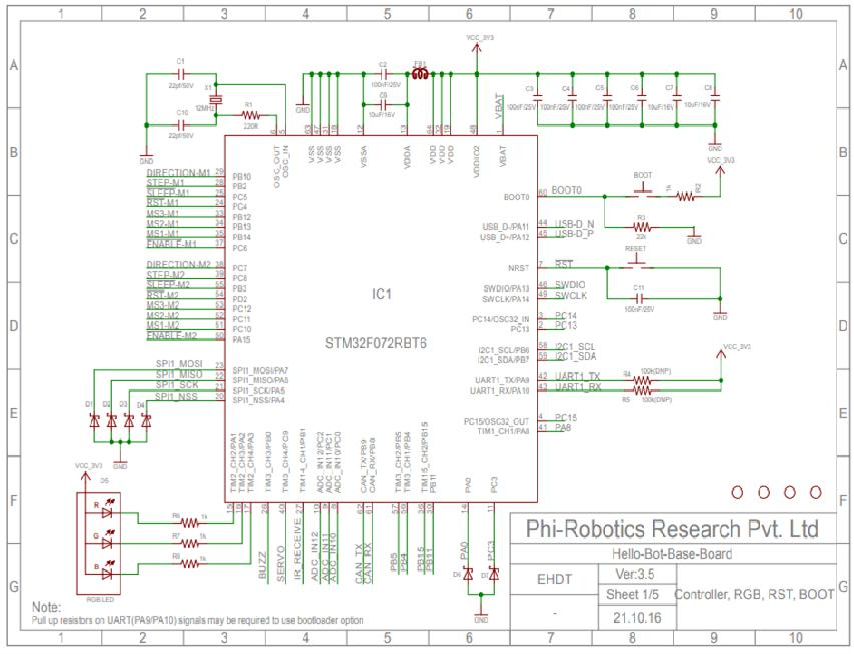
and with BC and LPM support

D

1. HDMI CEC wakeup on header reception
2. Serial wire debug (SWD)
3. 96-bit unique ID
4. All packages ECOP

A

CK®2

Pin description of Micro-controller

#### Motor Driver

The A4988 driver Stepper Motor Driver is a complete micro-stepping motor driver with built-in converter, easy to operate. It operates from 8 V to 35 V and can deliver up to approximately 1 A per phase without a heat sink or forced air flow (it is rated for 2 A per coil with sufficient additional cooling).

A4988 driver Stepper Motor Driver includes a fixed off-time current regulator; the regulator can be in slow or mixed decay mode. The converter is the key to the easy implementation of the A4988.There are no phase sequence tables, the high- frequency control interface programming etc. The application of A4988 interface is very suitable for a complex microprocessor is not available or overload. In the stepping operation, the chopping control in the A4988 automatically selects the current decay mode (slow or mixed). The mix decay current control scheme can reduce the audible motor noise, increased step accuracy, and reduced power consumption. Provide internal synchronous rectification control circuitry, in order to improve the pulse width modulation (PWM) power consumption during operation.

Internal circuit protection includes thermal shutdown with hysteresis, under- voltage lockout (UVLO) and crossover current protection. Don’t need special power-up sequencing. A4988 uses surface mount QFN package (ES), the size is 5 mm × 5 mm, nominal overall package height is 0.90 mm, with an exposed thermal pad to enhance the heat dissipation function.

#### Features:

* + - * 1. Automatic current decay mode detection/choice.
        2. Mixed with slow current decay mode.
        3. The low power dissipation of synchronous rectifier.
        4. Internal UVLO (ultra voltage lockout).
        5. Crossover current protection.
        6. Thermal shutdown circuit.
        7. Ground fault protection.
        8. Loading and short circuit protection

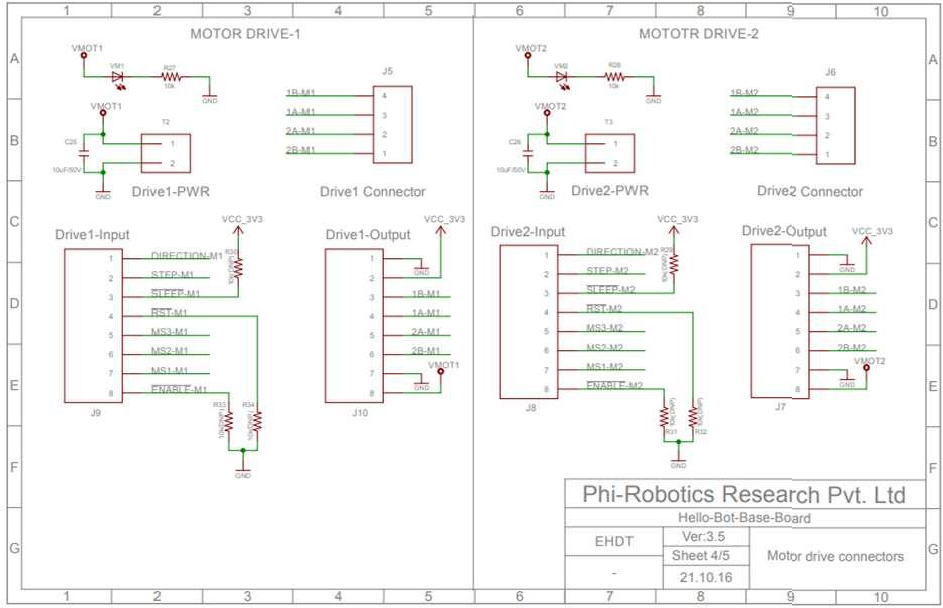
*Terna Engineering College, BE-Mechatronics* 20

Package Includes:

A4988 Stepper Motor Driver: 2 Nos.

Heat sink: 2 Nos.

|  |  |  |  |
| --- | --- | --- | --- |
| Supply Voltage (V) | 8 to 35 | Logic Input | 3-5.5V |
| Current (No Heat sink) | 1 A | Shipment Weight | 0.05 kg |
| Current (With Heat sink) | 2 A | Shipment Dimensions | 5x5x4 cm |



Pin description of motor driver

#### Stepper Motor

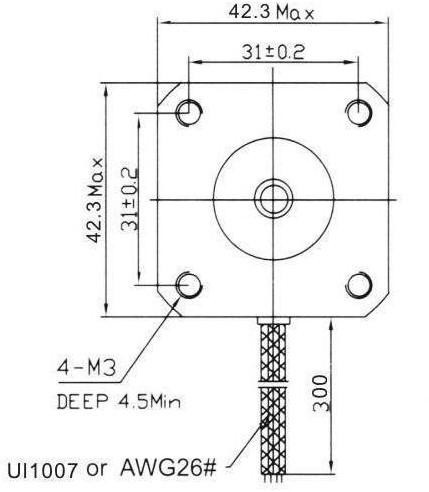
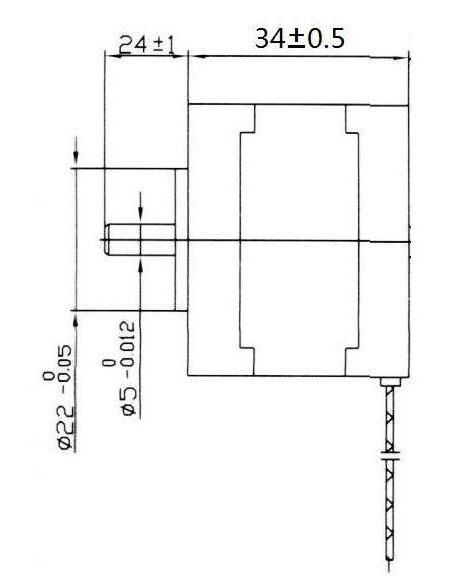
This is 4-wire bipolar stepper motor and has 1.8° per step for smooth motion and a nice holding torque. The motor is specified to have a max current of 350mA so that it could be driven easily.

Wires are put in this order: Red, Yellow, skip ground, Green, Brown (or Gray)

|  |  |
| --- | --- |
| **ITEM** | **SPECIFICATIONS** |
| Step Angle | 1.8° |
| Step Angle Accuracy | ±5% |
| Resistance Accuracy | ±10% |
| Inductance Accuracy | ±20% |
| Temperature Accuracy | 80° C Max (Rated Current, Phase On) |
| Temperature Rise | -20°C ±50°C |
| Ambient Temperature | 100M Min.,500V DC |
| Dielectric Strength | 500V AC/ for one minute |
| Shaft Radial Play | 0.02 Max (450 g load) |
| Shaft Axial Play | 0.08 Max (450 g load) |
| Maximum Radial Force | 28 N (20 mm) |
| Maximum Axial Force | 10 N |



Dimensions:



#### UART to TTL Connector

Introduction to UART communication:

In UART communic tion, two UARTs communicate directly with e ch other. The

a

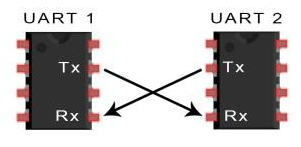
a

transmitting UART converts parallel data from a controlling device like a CPU into serial form, transmits it in serial to the receiving UART, which then converts

the serial data back i to parallel data for the receiving device. Only two wires are

n

needed to transmit data between two UARTs. Data flows from the Tx pin of the transmitting UART to the Rx pin of the receiving UART:



UARTs transmit data asynchronously, which means there is no clock signal to synchronize the output of bits from the transmitting UART to the sampling of bits by the receiving UART. Instead of a clock signal, the transmitting UART adds start and stop bits to the data packet being transferred. These bits define the beginning and end of the data packet so the receiving UART knows when to start reading the bits.

When the receiving UART detects a start bit, it starts to read the incoming bits at a specific frequency known as the baud rate. Baud rate is a measure of the speed of data transfer, expressed in bits per second (bps). Both UARTs must operate at about the same baud rate. The baud rate between the transmitting and receiving UARTs can only differ by about 10% before the timing of bits gets too far off.

Both UARTs also must be configured to transmit and receive the same data packet structure.

**USB to Serial TTL CP2102 UART Port Module**



Feature:

Implements full v2.0 USB protocol

Internal EEPROM for device ID and Product Description strings

+3.3V 100 mA output 5 volt tolerant inputs

USB Specification 2.0 compliant; full-speed (12 Mbps) All handshaking and modem interface signals

Use this device to connect your PC to a serial (TTL level) device.

Uses CP2102 chipset and has a standard 0.1 2033; pitch terminal strip to connect directly to UART or I/O pins for easy access to your MCU

USB specification 2.0 compliant with full-speed 12Mbps. Standard USB type A male and TTL 6pin connector.

6 pins for 3.3V, RST, TXD, RXD, GND and amp; 5V. Baud rates: 300 bps to 1.5 Mbps.

Byte receive buffer; 640 byte transmit buffer. Hardware or X-On/X-Off handshaking supported.

Supports Windows 98SE, 2000, XP, Vista, Window7, Mac OS 9, Mac OS X and amp; Linux 2.40.

Size: 42mm X 15mm (approx) Technical Details:

|  |  |
| --- | --- |
| Item Height | 8.6 Centimetres |
| Item Width | 8 Millimetres |
| Item Weight | 22.7 g |
| Product Dimensions | 14.2 x 0.8 x 8.6 cm |
| Item model number | USB-TTL-ADPT |

#### Battery

4 batteries of 220 mV each are used to supply power to Microcontroller.



#### Camera

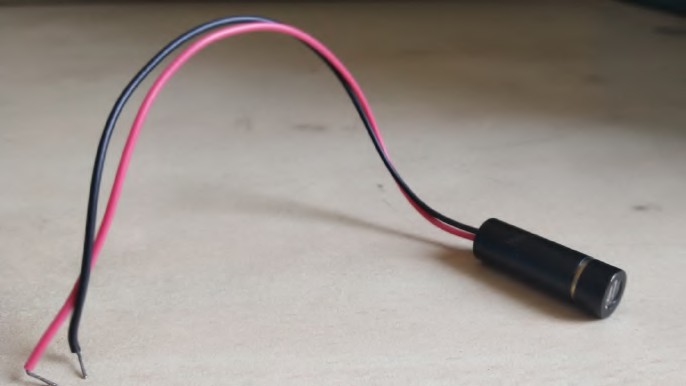
**Model: Logitech C310 HD**



**Specifications:**

|  |  |
| --- | --- |
| Connection Type | Corded USB |
| USB Type | High Speed |
| Microphone | Built in, Noise Suppression |
| Lens and Sensor Type | Plastic, CMOS |
| Focus Type | Fixed |
| Field of View | (FOV) 60° |
| Focal Length | 4.4mm |
| Optical Resolution (True) | 1280 x 960 VGA |
| Image Capture (4:3 SD) | 320x240, 640x480,1.2MP, 5.0MP |
| Image Capture (16:9 W) | 360p, 480p, 720p |
| Video Capture (4:3 SD) | 320x240, 640x480, 800x600 |
| Video Capture (16:9 W) | 360p, 480p, 720p |
| Frame Rate (max) | 30fps @ 640x480 |
| Video Effects (VFX) | Fun Filters |
| Buttons Other | NA |
| Indicator Lights (LED) | Activity/Power |
| Privacy Shade | No |
| Clip Size (max) | 0 to infinity |
| Cable Length | 5 Feet or 1.5 Meters |

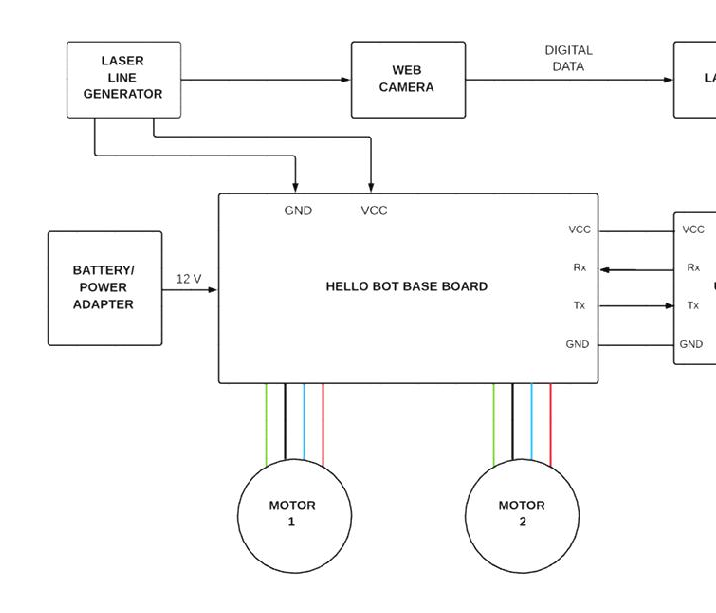
#### Laser Line Generator



Specifications:

|  |  |
| --- | --- |
| Operating Voltage | 2.8 to 5.2 V DC |
| Operating Current | 70 mA |
| Line Width | 3.08 mm(@ 2m) |
| Lasing Wavelength | 655 nm |
| Optical Power | 35 mW |
| Lasing Wavelength | 655 nm |

* 1. **BLOCK DIAGRAM**



**UART TO TTL**

**CONNECTOR**

# CHAPTER 4 IMPLEMENTATION

### CHAPTER 4: IMPLEMENTATION

#### IMAGE PROCESSING

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image. Nowadays, image processing is among rapidly growing technologies. It forms core research area within engineering and computer science disciplines too.

Image processing basically includes the following three steps:

* + - Importing the image via image acquisition tools;
    - Analysing and manipulating the image;
    - Output in which result can be altered image or report that is based on image analysis.

There are two types of methods used for image processing namely, analogue and digital image processing. Analogue image processing can be used for the hard copies like printouts and photographs. Image analysts use various fundamentals of interpretation while using these visual techniques. Digital image processing techniques help in manipulation of the digital images by using computers. The three general phases that all types of data have to undergo while using digital technique are pre-processing, enhancement, and display, information extraction.

In this lecture we will talk about a few fundamental definitions such as image, digital image, and digital image processing. Different sources of digital images will be discussed and examples for each source will be provided. The continuum from image processing to computer vision will be covered in this lecture. Finally we will talk about image acquisition and different types of image sensors.

#### Processes Involved

Obstacle Detection Obstacle Avoidance Path Follow

Code:

#include <stdio.h> #include <app\_sysinit.h> #include <pcf/ui/ui.h>

#include <pcf/graphics/graphics.h> #include <pcf/ui/mainform/mainform.h> #include <pcf/ui/control/control.h> #include <pcf/ui/button/button.h> #include <pcf/ui/imageview/imageview.h> #include <pcf/io/camera/camera.h>

#include <SSerial/log.h> //add this to use any serial function in any of your main.c keeping log .c in same folder as of main.c

#include <SSerial/serial.h> #include<stdint.h> #include<stdbool.h> #include<stdio.h> #include<stdlib.h> #include<math.h>

#define BMP\_HEADER\_SIZE 54

#define FILE\_HEADER\_SIZE 14

#define DIB\_HEADER\_SIZE 40

typedef struct { //Total 54 bytes uint16\_t type;

uint32\_t size; uint16\_t reserved1;

uint16\_t reserved2; /\* Offset to image data, bytes \*/ uint32\_t offset;

}BMPFileHeader; typedef struct {

uint32\_t dib\_header\_size; /\* Header size in bytes \*/

int32\_t width\_px; /\* Width of image \*/ int32\_t height\_px; /\* Height of image \*/

uint16\_t num\_planes; /\* Number of colour planes \*/ uint16\_t bits\_per\_pixel; /\* Bits per pixel \*/ uint32\_t compression; /\* Compression type \*/ uint32\_t image\_size\_bytes; /\* Image size in bytes \*/ int32\_t x\_resolution\_ppm; /\* Pixels per meter \*/ int32\_t y\_resolution\_ppm;

uint32\_t num\_colours; /\* Number of colours \*/ uint32\_t important\_colors; /\* Important colours \*/

} BMPDibHeader;

extern pcf\_status\_t PCF\_CAMERA\_API pcf\_io\_camera\_showPropertyControl(const IoCamera\_t\* camera);

extern void sysInit();

void\* camMemAlloc(size\_t size); void camMemFree(void\* mem);

void PCF\_STDCALL CameraCaptureHandler(const IoCamera\_t\* camera, IoCameraImage\_t\* image);

void PCF\_STDCALL buttonClickHandler(UiButton\_t\* button); SerialPort\_t device;

static UiImageView\_t\* gs\_camview = NULL; static IoCamera\_t\* gs\_camera;

static UiButton\_t\* gs\_startButton = NULL; static UiButton\_t\* gs\_propertiesButton = NULL; static int32\_t mainform\_padding = 16;

static int32\_t mainform\_titlebar\_padding = 40; static int32\_t camera\_image\_width = 320; static int32\_t camera\_image\_height = 240; static int32\_t button\_height = 24;

static int32\_t button\_width = 50; static int32\_t view\_offset = 5;

int initCamera();

void addImageView(UiMainForm\_t\* mainform, int x, int y, uint32\_t width, uint32\_t height);

void addButtons(UiMainForm\_t\* mainform, int x, int y); int main() {

int failed = app\_sysinit(); if(failed) {

printf("System failed to initialize \n"); return failed;

}

//////////////////////// CAMERA //////////////////////////////// gs\_camera = NULL;

gs\_camview = NULL; printf("System Initialized\n");

// Control info for creating the main form. UiControlInfo\_t info = {0};

info.parent = NULL;

info.typeId = enControlTypeIdValue\_Mainform; info.location.X = 10;

info.location.Y = 10;

info.size.Height = camera\_image\_height + 2 \* view\_offset + mainform\_titlebar\_padding + button\_height;

info.size.Width = camera\_image\_width + 2 \* view\_offset + mainform\_padding; info.maximumSize.Height = info.size.Height;

info.maximumSize.Width = info.size.Width; info.minimumSize.Height = info.size.Height; info.minimumSize.Width = info.size.Width; info.text = "Camera View";

pcf\_status\_t status; if(pcf\_ui\_isInitialized() &&

pcf\_graphics\_isInitialized())

{

UiMainForm\_t\* mainform = pcf\_ui\_mainform\_create(&info, &status); if(mainform

{

int32\_t exitcode;

addImageView(mainform, view\_offset, view\_offset, camera\_image\_width,

camera\_image\_height);

if(gs\_camview)

{

addButtons(mainform, view\_offset, camera\_image\_height + view\_offset + 3); initCamera();

if(gs\_camera)

{

UiControl\_t\* btnControl = pcf\_ui\_button\_asControl(gs\_startButton); pcf\_ui\_control\_setEnabled(btnControl, enBoolean\_True); btnControl = pcf\_ui\_button\_asControl(gs\_propertiesButton); pcf\_ui\_control\_setEnabled(btnControl, enBoolean\_True);

}

status = pcf\_ui\_mainform\_run(mainform, &exitcode); if(gs\_camera)

{

pcf\_io\_camera\_delete(gs\_camera); //Don't forget to delete the camera.

}

}

}

}

return 0;

}

void addButtons(UiMainForm\_t\* mainform, int x, int y)

{

UiControl\_t\* parent = pcf\_ui\_mainform\_asControl(mainform);

if(parent)

{

UiControlInfo\_t info = {0}; info.parent = parent;

info.typeId = enControlTypeIdValue\_SimpleButton; info.location.X = x;

info.location.Y = y; info.size.Height = button\_height; info.size.Width = button\_width; info.text = "Start";

pcf\_status\_t status;

UiButton\_t\* button = pcf\_ui\_button\_new(&info,&status); UiControl\_t\* btnControl = NULL;

if(button != NULL)

{

gs\_startButton = button; pcf\_ui\_button\_attachClickedEventHandler(gs\_startButton, buttonClickHandler); btnControl = pcf\_ui\_button\_asControl(button); pcf\_ui\_control\_setEnabled(btnControl, enBoolean\_False);

button = NULL; btnControl = NULL;

}

int delta = button\_width + view\_offset; button = NULL;

info.location.X = x + delta; info.location.Y = y;

info.size.Width = button\_width + 20; info.text = "Properties";

button = pcf\_ui\_button\_new(&info,&status); if(button != NULL)

{

gs\_propertiesButton = button; pcf\_ui\_button\_attachClickedEventHandler(gs\_propertiesButton,

buttonClickHandler);

btnControl = pcf\_ui\_button\_asControl(button); pcf\_ui\_control\_setEnabled(btnControl, enBoolean\_False); button = NULL;

btnControl = NULL;

}

}

}

void PCF\_STDCALL buttonClickHandler(UiButton\_t\* button)

{

if(button == gs\_startButton)

{

pcf\_status\_t status = pcf\_io\_camera\_start(gs\_camera); if(PCF\_S\_IS\_SUCCESS(status))

{

UiControl\_t\* btnControl = pcf\_ui\_button\_asControl(gs\_startButton); pcf\_ui\_control\_setEnabled(btnControl, enBoolean\_False); btnControl = pcf\_ui\_button\_asControl(gs\_propertiesButton); pcf\_ui\_control\_setEnabled(btnControl, enBoolean\_False);

}

}

else if(button == gs\_propertiesButton)

{

pcf\_status\_t status = pcf\_io\_camera\_isRunning(gs\_camera); if(status != PCF\_IO\_CameraOk)

{

pcf\_io\_camera\_showPropertyControl(gs\_camera);

}

}

}

void addImageView(UiMainForm\_t\* mainform, int x, int y, uint32\_t width, uint32\_t height)

{

UiControl\_t\* parent = pcf\_ui\_mainform\_asControl(mainform); if(parent)

{

UiControlInfo\_t info = {0}; info.parent = parent;

info.typeId = enControlTypeIdValue\_ImageView; info.location.X = x;

info.location.Y = y; info.size.Height = height; info.size.Width = width; info.text = "Image"; pcf\_status\_t status;

UiImageView\_t\* imview = pcf\_ui\_imageView\_new(&info,&status); if(imview == NULL)

{

fprintf(stdout, "Failed while creating imview control, status = %d\n", status); return;

}

gs\_camview = imview;

}

}

int initCamera()

{

int camera\_count;

pcf\_status\_t status = pcf\_io\_camera\_enumerate(&camera\_count);

if(!PCF\_S\_IS\_SUCCESS(status))

{

return -1;

}

int camera\_id = camera\_count - 1; //this could be negative status = pcf\_io\_camera\_canCreateNew(camera\_id); if(!PCF\_S\_IS\_SUCCESS(status))

{

return -1;

}

IoCameraInfo\_t info;

status = pcf\_io\_camera\_getCameraInfo(camera\_id, &info); if(!PCF\_S\_IS\_SUCCESS(status))

{

return -1;

}

IoCamera\_t\* camera = pcf\_io\_camera\_new(&info, &status); if(camera)

{

if(PCF\_S\_IS\_SUCCESS(status))

{

IoCameraSetupArgs\_t e = {0}; e.i\_cameraId = info.id;

e.i\_captureHandler = CameraCaptureHandler; e.i\_colorModel = enIoCameraColorModel\_RGB24; e.i\_FrameInterval = 333333;

e.i\_height = 240;

e.i\_width = 320;

e.i\_memAlloc = camMemAlloc; e.i\_memFree = camMemFree;

e.i\_videoType = enIoCameraVideoType\_NTSC\_M; //Value is ignored.

status = pcf\_io\_camera\_setup(camera, &e); //The fields of setup arg get updated, so check them back after successful setup.

if(!PCF\_S\_IS\_SUCCESS(status))

{

pcf\_io\_camera\_delete(camera);

printf("Failed while setting up the camera\n"); return -1;

}

}

}

else

{

return -1;

}

gs\_camera = camera; return 0;

}

BMPFileHeader fileheader; BMPDibHeader dibheader;

void PCF\_STDCALL CameraCaptureHandler(const IoCamera\_t\* camera, IoCameraImage\_t\* image)

{

if(image != NULL && camera == gs\_camera)

{

printf("Time stamp = %f, BufferSize = %u\n", image->timeStamp, image->length); pcf\_io\_camera\_save(image, "output.bmp");

if(gs\_camview != NULL)

{

pcf\_ui\_imageView\_showRGB24(gs\_camview, image->pixels, image->width, image->height);

FILE \*fin = fopen("output.bmp", "rb"); fread(&fileheader, 1,14, fin); printf("%c\n", fileheader.type); fread(&dibheader, 1,40, fin);

printf("Width: %dpx\n", dibheader.width\_px); printf("Height: %dpx\n", abs(dibheader.height\_px)); printf("%d\n", dibheader.dib\_header\_size);

once

int width = 3\*dibheader.width\_px;

int height = abs(dibheader.height\_px);

unsigned char data[height][width];// allocate 3 bytes per pixel int size = width \* height;

fread(data, sizeof(unsigned char), size, fin); //read the rest of the data at

for (int j = 0; j < width; j+=3) { for(int i = 0; i < height; i++){

if (data[i][j]<=100 && data[i][j+1]<=100 && data[i][j+2]>=150)

{ //BGR data[i][j]=255;

data[i][j+1]=0; //stop motor,turn right check again data[i][j+2]=0;

}

}

}

FILE \*fout = fopen("out.bmp", "wb"); fwrite(&fileheader, 14, 1, fout);

fwrite(&dibheader, 40, 1, fout); fwrite(&data, size, 1, fout);

}

}

}

void\* camMemAlloc(size\_t size)

{

return malloc(size);

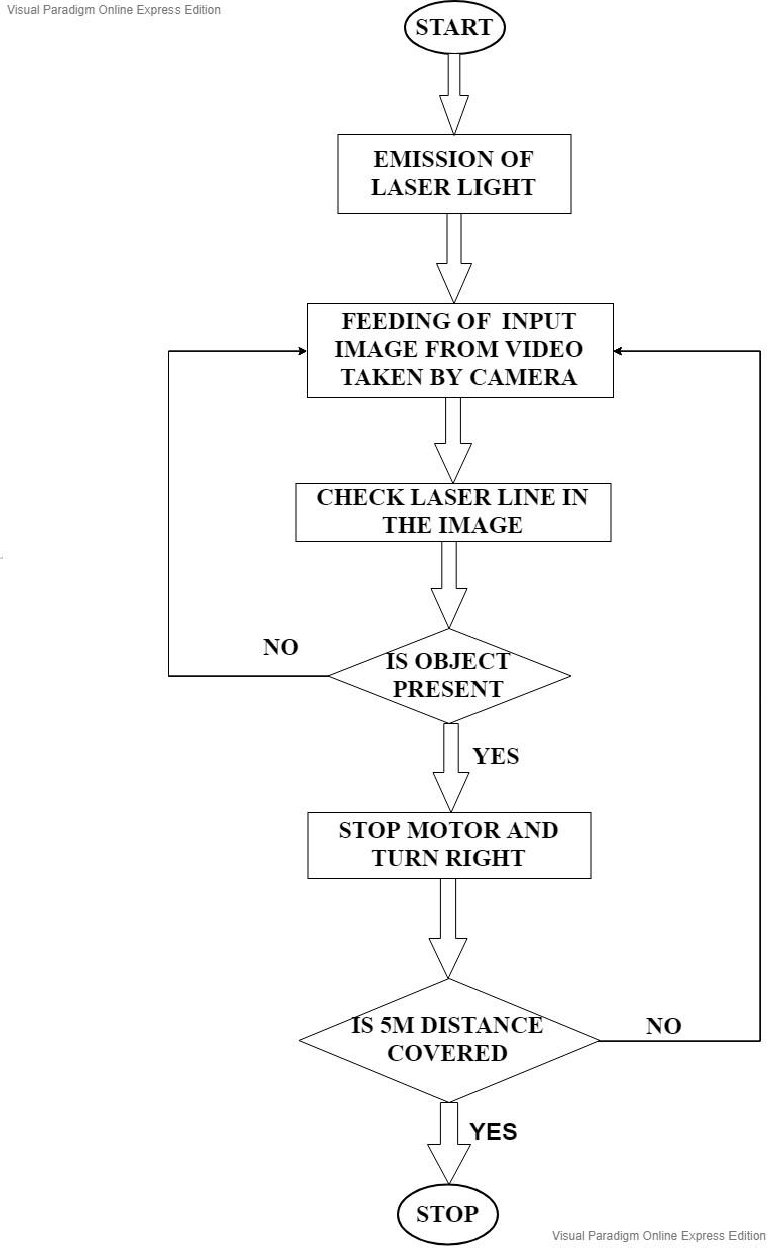
}

void camMemFree(void\* mem)

{

free(mem); return;

}



# CHAPTER 5 CONCLUSION

### CHAPTER 5: CONCLUSION

As we have seen in this project we have made a Design and Development of Obstacle Avoiding Robot Using Camera and Laser Line Generator and the result of this system is we can detect and avoid obstacles coming in the path.

#### APPLICATIONS:

1. For education purpose
2. Indoor applications
3. In Hospitals to carry medicines or other related accessories from one room to another

#### FUTURE SCOPE:

1. Home cleaning Robot
2. Robots for commercial advertisement
3. An automatic Vacuum Cleaner
4. Path Finder Robot

#### REFERENCES:

* Kumar R C, Saddam Khan M, Kumar D, Birua R, Mondal S and Parai M K 2013 International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering 2 1430-4
* Borenstein J and Koren Y 1988 Journal of Robotics and Automation 4 213-8
* Sathiyanarayanan M, Azharuddin S, Santhosh K and Khan G 2014 International Journal For Technological Research in Engineering 1 1075-7
* Huballi P, Bannikoppa S, Bhairi V and Jahagirdar R G Indian J. Sci. Res. 12 155- 8
* Bhanu B, Das S, Roberts B and Duncan D 1996 IEEE Transaction on Aerospace and Electronic Systems 32 875-97