###### A MAJOR PROJECT REPORT ON

**“SMART UNMANNED BORDER SECURITY (SUBS)**

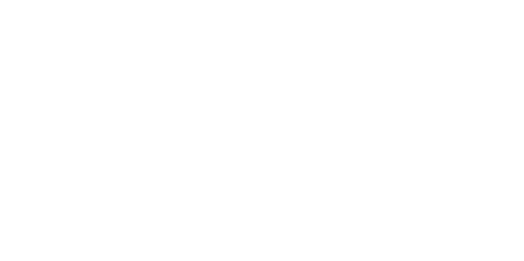
**SYSTEM”**

***Submitted in partial fulfillment of th*e *requirements for the award of the degree of***

# BACHELOR OF TECHNOLOGY

***in***

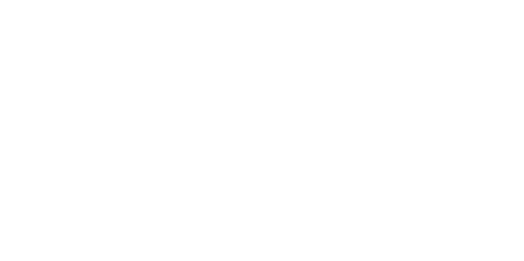
**ELECTRONICS AND COMMUNICATION ENGINEERING**



**SUBMITTED BY:**

**Ravi Chandra Jha(0631150281515) Shivang Pandey(0411502815) VaibhavGoel(08611502815)**

**VivekChauhan(09411502815)**



**GUIDED BY:**

**Mr. SreenivasuluDakala**

**Assistant Professor**



**BHARATI VIDYAPEETH’S COLLEGE OF ENGINEERING**

**A-4, PASCHIM VIHAR, ROHTAK ROAD, NEW DELHI - 110063 AFFILIATED TO**

**GURU GOBIND SINGH INDRAPRASTHA UNIVERSITY, DELHI 2015-2019**

**CANDIDATE’S DECLARATION**



It is here by certified that the work which is being presented in the B.Tech Major Project Report entitled **“SMART UNMANNED BORDER SECURITY (SUBS) SYSTEM”** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** and submitted in the **Department of Electronics and Communication Engineering** of **BHARATI VIDYAPEETH’S COLLEGE OF ENGINEERING, NewDelhi (Affiliated to Guru Gobind Singh Indraprastha University, Delhi)** It is an authentic record of our own work carried out during a period from **January 2019 to April 2019** under the guidance of **Mr. Sreenivasulu Dakala, Asst. Professor.**

The matter presented in the B. Tech Major Project Report has not been submitted by me for the award of any other degree of this or any other Institute.

|  |  |  |  |
| --- | --- | --- | --- |
| **Ravi Chandra Jha** | **Shivang Pandey** | **VaibhavGoel** | **Vivek Chauhan** |
| **06311502815** | **07411502815** | **08611502815** | **09411502815** |
|  |  |  |  |

#### CERTIFICATE

This is to certify that the above statement made by the candidate is correct to the best of my knowledge. They are permitted to appear in the External Major Project Examination.

###### Mr. Sreenivasulu Dakala Dr. Anuradha Basu

**Asst. Professor HOD, ECE**

The B. Tech Major Project Viva-Voce Examination of **Ravi Chandra Jha(06311502815),**

**Shivang Pandey(07411502815), VaibhavGoel(08611502815), Vivek Chauhan(09411502815)**

has been held on **15th April,2019.**

**Project Coordinator Project Coordinator (Signature of External Examiner)**

#### ABSTRACT

The ordinary border patrol system suffers from intensive human involvement. This unmanned border security system can be collaborated with high-tech devices, like unmanned aerial vehicles, unattended ground sensors, and surveillance towers equipped with wireless camera. However, any single technique encounters inextricable problems, such as high false alarm rate and line of sight constrains. There requires a coherent system that co-ordinates various technologies to improve the system accuracy. In this project general idea of border security system, wireless sensor network architecture for border patrol system, is introduced. Border security robot utilize an ultrasonic sensor for intruder detection, a metal detector can be used to detect the presence of explosives and a wireless camera for monitoring the scenario continuously at the remote station. Mechanical control of robotic vehicle along with robotic arm can be done from the remote station. This is initiated with a WIFI module.

This project is intended to establish our border as well as remote locations more secure. The Smart Unmanned Border Security (SUBS) System aims at reducing the burden on forces and providing an additional layer of security with high precision and accuracy. It acts as a force multiplier for the current existing system. The system is activated when the laser fencing is breached and discharges the gun when an intruder is detected. This report discusses the power efficiency with its feasibility and applicability in remote locations. The overall challenges faced in the implementation of a wireless network, secure network connection and power consumption. It highlights the security concerns in the network and the encryption techniques used. The project finds multiple areas of applications other than border surveillance such as securing other perimeters secure like airport, inventories, other secure facilities and in implementing smart doors in smart homes.

#### ACKNOWLEDGEMENT



We express our deep gratitude to **Mr. Sreenivasulu Dakala**, Asst. Professor, Department of Electronics and Communication Engineering for his valuable guidance and suggestion throughout our project work. We are thankful to **Dr. Manoj Sharma,** for her valuable guidance.

We are also thankful to **Dr. Dharmender Saini, Principal** for providing us the facilities to carry out our project work.

|  |  |  |  |
| --- | --- | --- | --- |
| **Ravi Chandra Jha** | **Shivang Pandey** | **VaibhavGoel** | **Vivek Chauhan** |
| **06311502815** | **07411502815** | **08611502815** | **09411502815** |
|  |  |  |  |

|  |  |  |
| --- | --- | --- |
|  | **TABLE OF CONTENTS** |  |
|  |  |  |
| **Candidate Declaration Certificate**  **Abstract Acknowledgement**  **Table of Contents** |  | **ii iii iv v vi** |
| **List of Figures** |  | **viii** |
| **List of Tables** |  | **X** |
| [**List of Acronyms**](#_bookmark0) |  | [**xi**](#_bookmark0) |
| **1.Introduction** |  | **1** |

* 1. **1.1** [Motivation 2](#_bookmark1)
  2. **1.2** Objectives 3
  3. **1.3** [Scope and Application](#_bookmark2) 3
  4. **1.4** Challenges 4

1. Literature Survey 5
   1. **2.1** Introduction 5
   2. **2.2** Literature Review 5
   3. **2.3** Inferences Drawn 6
2. Description of Project 7
   1. **3.1** SUBS System 7
   2. **3.2** Components Used 8

**3.2.1** Hardware Implementation 8

**3.2.2**Software Implementation 27

* + 1. **3.3**Architecture 37
  1. **3.3.1** Block Diagram 38
  2. **3.3.2** Process Flowchart 40

4. Resultand Discussion 41

* 1. Result 41
  2. Conclusion 43

1. Future Scope 44

[References 4](#_bookmark11)5

[Appendix](#_TOC_250000) 48

# List of Figures

Figure 1.1: Indian Borderline with neighboring countries

Figure 1.2: Target areas

Figure 3.1: Existing system for patrolling

Figure 3.2: Real view of Arduino

Figure 3.3: Real view of NodeMCU

Figure 3.4: Connection of Relay

Figure 3.5: Ultrasonic Sensor sensing object

Figure 3.6: 2D model of Ultrasonic Sensor

Figure 3.7: Depicting application of load in servo motor

Figure 3.8: PWM signal showing working of servo

Figure 3.9: SG90 servo motor dimensions

Figure 3.10: Connection of LDR

Figure 3.11: 2D model for laser diode

Figure 3.12: Effect of temperature on operation of laser

Figure 3.13: Plot showing laser beam divergence in parallel and perpendicular plane

Figure 3.14: Plot between forward voltage and forward current

Figure 3.15: Plot showing output power v/s forward current

Figure 3.16: Laser Diode connection

Figure 3.17: 2D model for laser diode

Figure 3.18: NodeMCUPyFlasher

Figure 3.19: ESplorer

Figure 3.20: ESP8266 operating in Station mode

Figure 3.21: ESP8266 operating in Soft Access Point mode

Figure 3.22: ESP8266 operating in Station + Soft Access Point mode

Figure 3.23: Block Diagram of SUBS System

Figure 3.24: A single slave node

Figure 3.25: Proposed master slave configuration of nodes

Figure 3.26: Process flowchart and data flow of SUBS System

Figure 4.1: Wireless switching via MQTT broker

Figure 4.2: Ultrasonic data on serial monitor

Figure 4.3: LDR data on serial monitor

Figure 4.4: Operating module via app

Figure 4.5: Operating module via Google Assistant

# List of Tables

Table 1: Relay pin description

Table 2: Ultrasonic sensor pin description

Table 3: Servo motor wire configuration

Table 4: WiFi module functions

# 

# List of Acronyms

|  |  |
| --- | --- |
| IoT | Internet of Things |
| MQTT | Message Queuing Telemetry Transport |
| WSN | Wireless Sensors Network |
| NVD | Night Vision Devices |
| LDR | Light Dependent Resistor |
| LED | Light Emitting Diode |
| MCU | MicroController Unit |
| PWM | Pulse Width Modulation |
| SUBS | Smart Unmanned Border Security |
| IDE | Integrated Development Environment |
| WIFI | Wireless Fidelity |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Chapter 1: Introduction**

Border management is a security function that calls for coordination and concerted action by various government agencies within our country. The aim is to secure our frontiers and safeguard our nation from the risks involved in the movement of goods and people from India to other countries and vice versa. Border management itself is a multifaceted term and may include, but is not limited to, the regulation of legal and illegal immigration, ensuring safe and secure movement of authorized people and goods, and prevention of smuggling, human trafficking, and infiltration. The rapid growth that India is currently experiencing presents an array of opportunities and underlines the need for effective border management. In an increasingly globalized and service-oriented economy like ours, we rely heavily on the movement of goods and people. A competent border management system calls for the tight coupling of technology and infrastructure that is capable of handling the geopolitical, social and economic challenges we face in India owing to our vast border fronts. India has a very large and complex border, covering around 15,106.7 km, which it shares with Bangladesh, China, Pakistan, Nepal, Myanmar and Bhutan, as well as a small portion with Afghanistan. What further increase the complexity and criticality are the varied terrain, climatic conditions and relationship with some of the neighboring countries. The various agencies involved in border management, such as customs, immigration, armed forces, border security and intelligence agencies, are characterized by a need to cooperate with an expanding range of external information sources (technology driven and intelligence based) in order to develop a common operating picture of the movements within and beyond our borders.

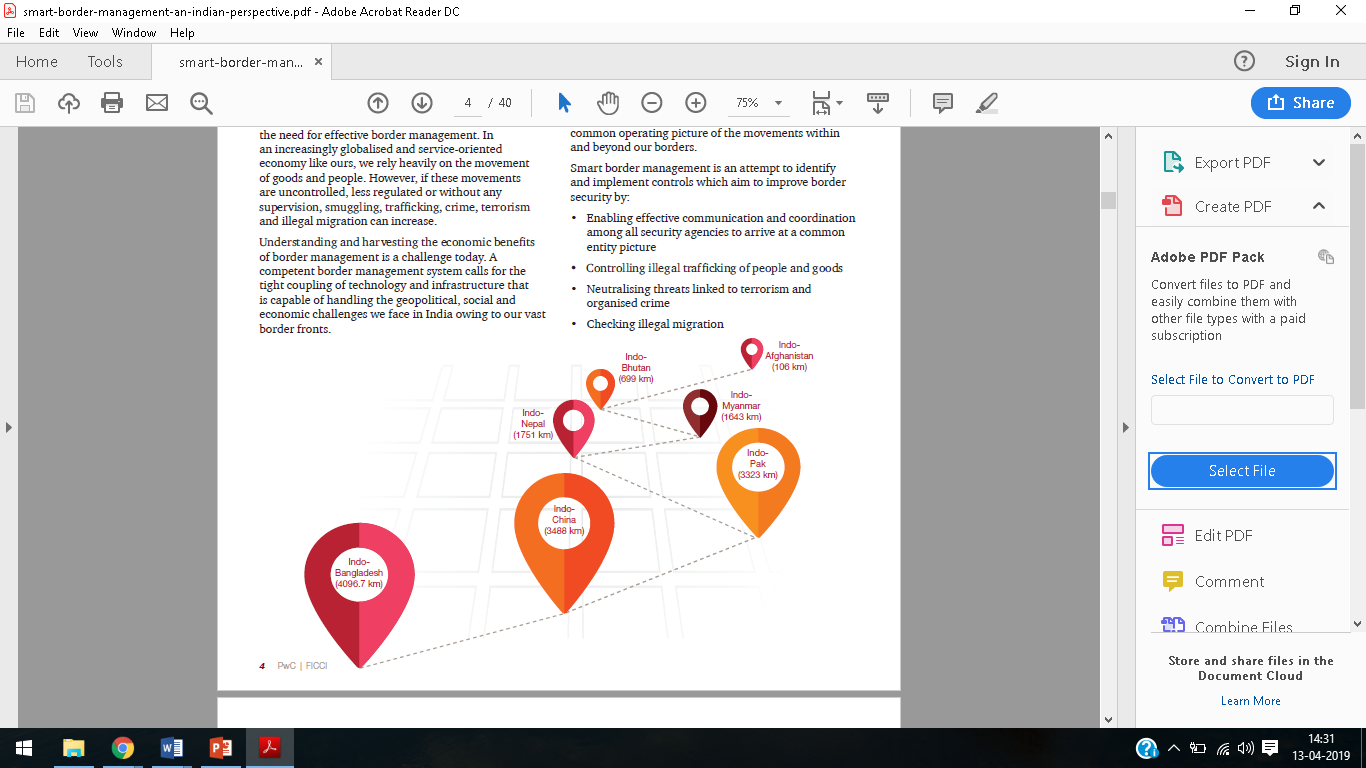
Smart border management is an attempt to identify and implement controls which aim to improve border security by:

• Enabling effective communication and coordination among all security agencies to arrive at a common entity picture.

• Neutralizing threats linked to terrorism and organizedcrime.

• Checking illegal migration.

We share our borders with seven different countries (Bangladesh, China, Pakistan, Nepal, Myanmar, Bhutan and Afghanistan) with all possible terrains—namely deserts, fertile lands, swampy marshes, snow-covered peaks and tropical evergreen jungles. This kind of vast terrain makes us vulnerable to insurgency, illegal migration and smuggling.



##### 

Figure 1.1 Indian Borderline with neighboring countries

##### 1.1 Motivation

Our project is inspired by the severities faced by the armed forces personnel securing our countries border which was depicted in a famous story “The Rope” by Captain Raghuraman and famous which ignited Kargilwar.. On 15 May 1999 Lt SaurabhKalia and five other soldiers – [Sepoys](https://en.wikipedia.org/wiki/Sepoy) Arjun Ram, Bhanwar Lal Bagaria, Bhika Ram, Moola Ram and Naresh Singh. of the [4 Jat Regiment](https://en.wikipedia.org/wiki/Jat_Regiment) had gone for a routine patrol of the Bajrang Post in the Kaksar sector in the rugged, treeless [Ladakh](https://en.wikipedia.org/wiki/Ladakh) mountains. After a continuous cross fire with Pakistan armed forces from across the LOC, he and his troops ran out of ammunition. They were finally encircled by a platoon of Pakistani rangers and captured alive before Indian reinforcements could reach them. No trace of the patrol was left, meanwhile Radio [Skardu](https://en.wikipedia.org/wiki/Skardu) of Pakistan announced that Captain SaurabhKalia had been captured by Pakistani troops. It was after this that India discovered hundreds of guerrillas had established fortified positions on the peaks of the hills deep inside the Indian side of the Line of control, with sophisticated equipment and supply lines back to [Pakistan-controlled Kashmir](https://en.wikipedia.org/wiki/Kashmir).

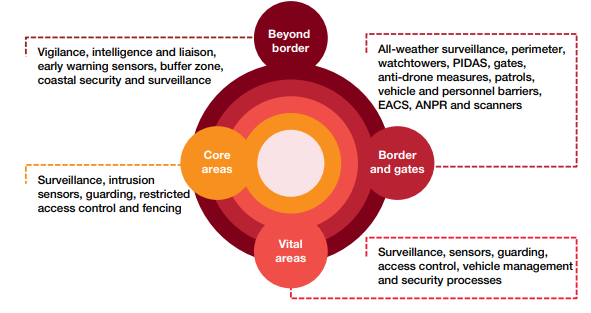
Lt SaurabhKalia and his men were in captivity from 15 May 1999 – 7 June 1999 (over twenty-two days), and subjected to torture as evident from injuries to their bodies when they were handed over by the Pakistani Army on 9 June 1999. [Post-mortem](https://en.wikipedia.org/wiki/Post-mortem) examinations revealed that the Pakistanis had tortured their prisoners by: burning their bodies with cigarettes, piercing the ear-drums with hot rods, puncturing eyes before removing them, breaking most of their teeth and bones, fracturing their skulls, cutting the lips, chipping of nose, chopping off limbs and private organs of the soldiers, and finally shooting them dead, as evidenced by bullet wounds to the temple. The post-mortem also confirmed the injuries were inflicted ante-mortem (before death).

On 9 June 1999, N. K. Kalia received the body of his son, Lt SaurabhKalia, handed over by the Pakistani army to Indian army commanders at the Kargil sector, with evidence of torture.

##### 1.2 Objectives

* Laser Fencing as an alternate to Iron Fencing on Border Areas.
* Ultrasonic Sensor for surveillance after the Border Fencing Region.
* Breaking of Border Fencing and Activation of ultrasonic radar-based gun.
* Radar moves clockwise and anti-clockwise and search for any intruders.
* If any intruder found, it will be shot using our prototype laser.
* Collect data of shooting and illegal border crossing and process data further on cloud.
* Low power consumption for automated gun.
* Low cost of product and optimum use to implement on harsh environmental areas.
* Enhancing security by communicating on a public broker on a specific topic using private key.

##### 1.3 Scope and Application

It is a multi-facet perimeter securing system capable of modification for multiple applications in various fields and areas. It has a broad domain for future specific needs.

##### 1.4 Challenges

Figure 1.2 Target Areas

False detection and false trigger. Power consumption in wireless sensor network. Security is an essential pillar of the Internet and one that ISOC perceives to be equally essential and ‘the’ most significant challenge for the IoT. The IoT creates unique challenges to privacy, many that go beyond the data privacy issues that currently exist. Much of this stems from integrating devices into our environments without us consciously using them. Without standards to guide manufacturers, developers sometimes design products that operate in disruptive ways on the Internet without much regard to their impact. If poorly designed and configured, such devices can have negative consequences for the networking resources they connect to and the broader Internet. Like privacy, there are a wide range of regulatory and legal questions surrounding the IoT, which need thoughtful consideration. The broad scope of IoT challenges will not be unique to industrialized countries. In fact, the IoT holds significant promise for delivering social and economic benefits to emerging and developing economies. Data capturing majorly occurs through various sensors, PLCs, etc., which are connected to IoT gateways to collect & transmit.

**Chapter 2: Literature Survey**

##### 2.1 Introduction

Wireless sensor network (WSN) refers to a group of spatially dispersed and dedicated sensors for monitoring and recording the physical conditions of the environment and organizing the collected data at a central location. WSNs measure environmental conditions like temperature, sound, pollution levels, humidity, wind, and so on.The Internet of Things (IoT) is a potential solution to this problem. By integrating systems of sensors, actuators, and control systems into existing military infrastructures, the military can become more efficient and effective. Below we have outlined three separate areas for IoT military applications. Logistics, The effect of IoT on logistics and supply chain management has been huge; industries such as agriculture and healthcare have recently benefited. In terms of military application, the connected sensors and digital analytics that IoT technology offers can be used to track supplies and equipment from their source to where they are needed on the battlefield. Smart Bases, Incorporating IoT devices and sensors into military bases can have several positive effects. Automated security screening, for example, increases safety while decreasing manpower, and a network of security cameras connected to their environment via sensors and to a central network via the Internet will also minimize security risks. Smart management of resources – electricity and water for example – will increase the capacity and output of military bases while ensuring that the wellbeing of all individuals inside the base is protected. Data warfare, by collecting data from a wide range of military platforms – including aircraft, weapon systems, ground vehicles, and troops themselves – the military can increase the effectiveness of their intelligence, surveillance, and reconnaissance systems. This wealth of information will allow the armed forces to identify key threats faster and with more accuracy.

##### Literature review

Issues of illegal border crossing or infiltration lie almost at every international border. Beside Indian borders, the US-Mexico border is yet another example of a very sensitive one. Other borders that stand in same queue are China-North Korea border, Israel-Palestine border, Syria-Turkey border and many others. Reports from Federation for American Immigration Reforms (FAIR) on border security measures reveal the fact that around 20000 illegal border crossings were recorded from the US-Mexico border side in 2012. Another report from European Union states nearly the same facts on surveillance. These reports also talk about preventive measures as employment of patrolling, border checks, and supervisory duties by special task forces. In the infrastructural resources provided are regular armaments with night vision devices (NVD’s), floodlighting at border side roads and use of drones for patrolling in air. But neither of the report talks of the automated usage of this infrastructure for surveillance and patrol. A report from Department of Border Management, Ministry of Home Affairs reveals the initiatives taken for border surveillance and patrolling at Indian borders. Border side roads, floodlighting, border out posts (BOP’s), air support and budget towards communication equipment, surveillance equipment, weapons, bullet proofs, vehicles, etc. accounts to those initiatives. These add to the strength of BSF but a need of automated surveillance and combat system still remains. Human detection has been achieved by a number of techniques and background subtraction technique is one of those. Frame differencing technique for human detection is derived from background subtraction technique. These two techniques are more prone to noise; shadow and ambient light illumination. The problem of shadow removal for accurate detection is covered in. Some of other techniques use face features for human detection. Viola-Jones technique is one of them and it uses texture information of the face for face detection and consequently the human detection. Texture information is coded into features called haar-like features. Cascaded classifier uses these features to classify the face. Viola-Jones fails for tilted and rotated faces. SVM is used with HoG features in for pedestrian’s detection. Skin colour is also used as unique feature to distinguish human being from other class of objects and background. A number of approaches for skin colour modelling have been proposed in literature. The approaches using skin colour information for human detection fails if skin portion of human body covered. Intruders are trained enough towards the use of these camouflage techniques. To avoid demerits faced by above discussed techniques, the optical flow information is used in proposed technique to detect human being.

##### 2.3 Inferences Drawn

The unmanned surveillance system poses substantial challenges in the detection of intruder and differentiating enemy from friendly intrusion and hence multiple false detections and warning systems are triggered. The power consumption is another aspect which has to be taken into taken consideration for applicability in remote areas, networking and security constraints are another aspect for encryption and providing a secure platform for various sensors in a network. Unauthorized access and centralized control of all nodes has to be granted to only verified user.

**Chapter 3: Description of Project**

This project is intended to establish our border as well as remote locations more secure. The Smart Unmanned Border Security (SUBS) System aims at reducing the burden on forces and providing an additional layer of security with high precision and accuracy. This paper discusses the power efficiency with its feasibility and applicability in remote locations. It highlights the security concerns in the network and the data can be plotted against various specifications.

**3.1 SUBS**

Stronger the protection of borders, safer is the nation. With increasing intrusions and breaching of iron fences along the borders, there has been a growing section of people advocating for unmanned automatic gun control systems to be implemented at the borders. There are virtually no places left where intrusions have not been there, be it the most inhabitable Siachen Glaciers, damp and swampy. So there is a need to safeguard these places with least or no human supervision. The Smart Unmanned Border Security (SUBS) System aims at reducing the burden on forces and providing an additional layer of security with high precision and accuracy. It aims at eliminating enemies with utmost accuracy and reducing human-made errors. SUBS attend the internet of things (IoT) enabled the compact, flexible, light-weight and robust system for providing border security electronically and automatically. It is a network of wireless sensors to sense sense (detect) intrusions. This electro-prototyping project is based on Master-Slave Node Communication model and provides intercommunication between slave nodes possible as well as centralized control on the robust network via Master nodes and works fine even in case of one or more slave nodes failure. This project uses a public broker for MQTT communications using Pub-Sub model for secure information communication. The complete system can be manually switched ON and OFF according to the requirement wirelessly. Another aspect of this project is that, by using laser instead of iron fencing, the SUBS can be operated in idle mode and its activation can be made dependent on the breaching of laser fencing. Hence, the power consumption is markedly reduced. The project aims at determining a substitute for human surveillance at the borders.****

Figure 3.1 Existing system for patrolling

**3.2Components Used**

Figure 3.1Deployment area

The project implementation is sub-divided into two sections namely hardware implementation and software implementation which touches upon the description of components and technology stack used.

**3.2.1 Hardware Components**

**Arduino Uno**

It is one of the most commonly used open source platform available on the market. Arduino UNO board uses Atmel microcontroller and has a frequency of 16MHz. It has the sufficient number of I/O pins for analog, digital as well as PWM (pulse width modulation) data. It has a power jack for external dc supply as well as Serial USB adapter for communication with the computer. The devices connected to it can be easily programmed/configured using Arduino IDE (Integrated Development Environment). It is used because of its simplicity, flexibility, and fault tolerant capabilities. Additionally, it is inexpensive with power-requirements.

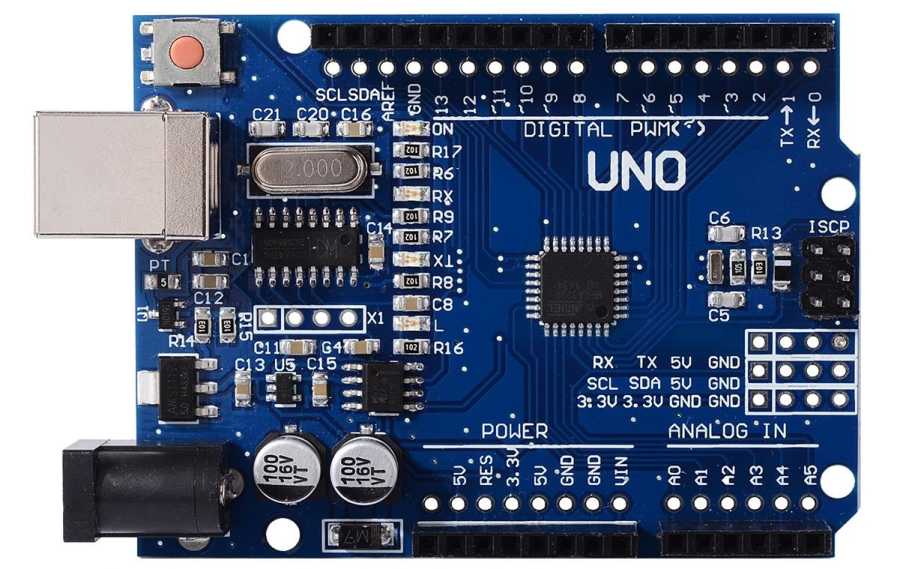
****

Fig 3.2: Real view of Arduino Uno

## Technical specifications

* Microcontroller: Microchip ATmega328P
* Operating Voltage: 5 Volts
* Input Voltage: 7 to 20 Volts
* Digital I/O Pins: 14 (of which 6 provide PWM output)
* Analog Input Pins: 6
* DC Current per I/O Pin: 20 mA
* DC Current for 3.3V Pin: 50 mA
* Flash Memory: 32 KB of which 0.5 KB used by bootloader
* SRAM: 2 KB
* EEPROM: 1 KB
* Clock Speed: 16 MHz
* Length: 68.6 mm
* Width: 53.4 mm
* Weight: 25 g

### General Pin functions

* **LED**: There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
* **VIN**: The input voltage to the Arduino/Genuino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
* **5V**: This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 20V), the USB connector (5V), or the VIN pin of the board (7-20V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board.
* **3V3**: A 3.3-volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
* **GND**: Ground pins.
* **IOREF**: This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.
* **Reset**: Typically used to add a reset button to shields which block the one on the board.

### Special Pin Functions

Each of the 14 digital pins and 6 Analog pins on the Uno can be used as an input or output, using pin Mode (),digital Write (), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller. The Uno has 6 analog inputs, labelled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function.

In addition, some pins have specialized functions:

* **Serial** / UART: pins 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
* **External Interrupts**: pins 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
* **PWM** (**P**ulse **W**idth **M**odulation): 3, 5, 6, 9, 10, and 11 Can provide 8-bit PWM output with the analogWrite() function.
* **SPI** (**S**erial **P**eripheral **I**nterface): 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
* **TWI** (**T**wo **W**ire **I**nterface) / [I²C](https://en.wikipedia.org/wiki/I%C2%B2C): A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.
* **AREF** (**A**nalog **R**eference): Reference voltage for the analog inputs.

## Communication

The Arduino/Genuino Uno has a number of facilities for communicating with a computer, another Arduino/Genuino board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A Software Serial library allows serial communication on any of the Uno's digital pins.

### Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino/Genuino Uno board is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip.

This setup has other implications. When the Uno is connected to a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened.

**NodeMCU**

It is a module which consists of esp8266 Wi-Fi chip for network communications. It has 64 kb of instruction memory and 96 kb of data memory. It occupies eight ports out of which two are the general purpose I/O ports. It considers general purpose I/O ports and runs at a frequency of 80MHz. This board is addressable over SPI and UART protocols. Simultaneously this Wi-Fi chip can act as a station as well as an access point for the connection with the Internet. It works on 3.3V logic and does not attach to an inbuilt level shifter.

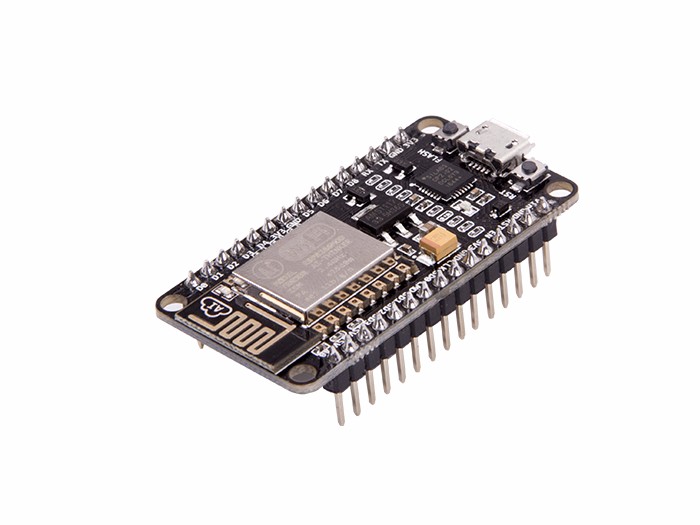


Fig 3.3: Real view of NodeMCU

### Features of 5-Pin 5V Relay

* Trigger Voltage (Voltage across coil) : 5V DC
* Trigger Current (Nominal current) : 70mA
* Maximum AC load current: 10A @ 250/125V AC
* Maximum DC load current: 10A @ 30/28V DC
* Compact 5-pin configuration with plastic moulding
* Operating time: 10msec Release time: 5msec
* Maximum switching: 300 operating/minute (mechanically)

### Relay Pin ****Configuration****

|  |  |  |
| --- | --- | --- |
| **Pin Number** | **Pin Name** | **Description** |
| 1 | Coil End 1 | Used to trigger(On/Off) the Relay, Normally one end is connected to 5V and the other end to ground |
| 2 | Coil End 2 | Used to trigger(On/Off) the Relay, Normally one end is connected to 5V and the other end to ground |
| 3 | Common (COM) | Common is connected to one End of the Load that is to be controlled |
| 4 | Normally Close (NC) | The other end of the load is either connected to NO or NC. If connected to NC the load remains connected before trigger |
| 5 | Normally Open (NO) | The other end of the load is either connected to NO or NC. If connected to NO the load remains disconnected before trigger |

Table 3.1 Relay pin description

### Equivalent Relays

3V Relay, [12V Relay](https://components101.com/switches/12v-relay), 1-channel Relay module, 4-channel Relay Module.

### How to use a Relay

Relays are most commonly used switching device in electronics. Before we proceed with the circuit to drive the relay we have to consider two important parameter of the relay. Once is the Trigger Voltage, this is the voltage required to turn on the relay that is to change the contact from Common->NC to Common->NO. Our relay here has 5V trigger voltage, but you can also find relays of values 3V, 6V and even 12V so select one based on the available voltage in your project. The other parameter is your Load Voltage & Current, this is the amount of voltage or current that the NC,NO or Common terminal of the relay could withstand, in our case for DC it is maximum of 30V and 10A. Make sure the load you are using falls into this range.

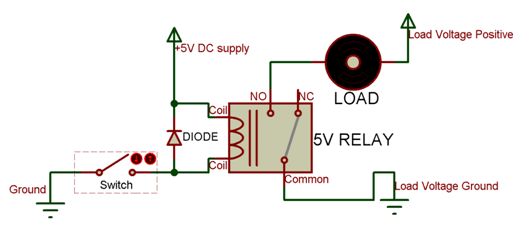


Figure 3.4 Connection of relay

The above circuit shows a bare-minimum concept for a relay to operate. Since the relay has 5V trigger voltage we have used a +5V DC supply to one end of the coil and the other end to ground through a switch. This switch can be anything from a small transistor to a microcontroller or a microprocessor which can perform switching operating. You can also notice a diode connected across the coil of the relay, this diode is called the Fly backDiode. The purpose of the diode is to protect the switch from high voltage spike that can produced by the relay coil. As shown one end of the load can be connected to the Common pin and the other end is either connected to NO or NC. If connected to NO the load remains disconnected before trigger and if connected to NC the load remains connected before trigger.

### Applications of Relay

* Commonly used in switching circuits.
* For Home Automation projects to switch AC loads
* To Control (On/Off) Heavy loads at a pre-determined time/condition
* Used in safety circuits to disconnect the load from supply in event of failure
* Used in Automobiles electronics for controlling indicators glass motors etc.

### Ultrasonic Sensor

### We have used a mid range HC-SR04 ultrasonic sensor in our project.

### Ultrasonic Sensor Pin ****Configuration****

|  |  |  |
| --- | --- | --- |
| **Pin Number** | **Pin Name** | **Description** |
| 1 | VCC | The VCC pin powers the sensor, typically with +5V |
| 2 | Trigger | Trigger pin is an Input pin. This pin has to be kept high for 10us to initialize measurement by sending US wave. |
| 3 | Echo | Echo pin is an Output pin. This pin goes high for a period of time which will be equal to the time taken for the US wave to return back to the sensor. |
| 4 | Ground | This pin is connected to the Ground of the system. |

Figure 3.2 Ultrasonic sensor pin description

### HC-SR04 Sensor

### Features

* Operating voltage: +5V
* Theoretical  Measuring Distance: 2cm to 450cm
* Practical Measuring Distance: 2cm to 80cm
* Accuracy: 3mm
* Measuring angle covered: <15°
* Operating Current: <15mA
* Operating Frequency: 40Hz

**Equivalent distance measuring Sensors**

Transmitter Receiver pair, IR sensor module, IR sensor pair, IR Analog distance sensor

**HC-SR04 Ultrasonic Sensor - Working**

As shown above the **HC-SR04 Ultrasonic sensor** is a 4 pin module, whose pin names are Vcc, Trigger, Echo and Ground respectively. This sensor is a very popular sensor used in many applications where measuring distance or sensing objects are required. The module has two eyes like projects in the front which forms the Ultrasonic transmitter and Receiver. The sensor works with the simple high school formula that

**Distance = Speed × Time**

The Ultrasonic transmitter transmits an ultrasonic wave, this wave travels in air and when it gets objected by any material it gets reflected back toward the sensor this reflected wave is observed by the Ultrasonic receiver module as shown in the picture below



Figure 3.5 Ultrasonic sensor sensing object

Now, to calculate the distance using the above formulae, we should know the Speed and time. Since we are using the Ultrasonic wave we know the universal speed of US wave at room conditions which is 330m/s. The circuitry inbuilt on the module will calculate the time taken for the US wave to come back and turns on the echo pin high for that same particular amount of time, this way we can also know the time taken. Now simply calculate the distance using a microcontroller or microprocessor.

### How to use the HC-SR04 Ultrasonic Sensor

**HC-SR04 distance sensor** is commonly used with both microcontroller and microprocessor platforms like Arduino, ARM, PIC, Raspberry Pie etc. The following guide is universally since it has to be followed irrespective of the type of computational device used.

Power the Sensor using a regulated +5V through the Vcc ad Ground pins of the sensor. The current consumed by the sensor is less than 15mA and hence can be directly powered by the on board 5V pins (If available). The Trigger and the Echo pins are both I/O pins and hence they can be connected to I/O pins of the microcontroller. To start the measurement, the trigger pin has to be made high for 10uS and then turned off. This action will trigger an ultrasonic wave at frequency of 40Hz from the transmitter and the receiver will wait for the wave to return. Once the wave is returned after it getting reflected by any object the Echo pin goes high for a particular amount of time which will be equal to the time taken for the wave to return back to the sensor.

The amount of time during which the Echo pin stays high is measured by the MCU/MPU as it gives the information about the time taken for the wave to return back to the Sensor. Using this information the distance is measured as explained in the above heading.

### Applications

* Used to avoid and detect obstacles with robots like biped robot, obstacle avoider robot, path finding robot etc.
* Used to measure the distance within a wide range of 2cm to 400cm
* Can be used to map the objects surrounding the sensor by rotating it
* Depth of certain places like wells, pits etc can be measured since the waves can penetrate through water

### 2D model of the component

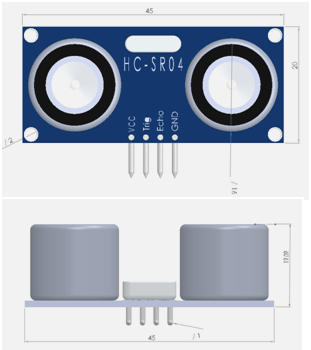
****

Figure 3.6 2D model of ultrasonic sensor

**Servo Motor**

### Wire ****Configuration****

|  |  |  |
| --- | --- | --- |
| **Wire Number** | **Wire Colour** | **Description** |
| 1 | Brown | Ground wire connected to the ground of system |
| 2 | Red | Powers the motor typically +5V is used |
| 3 | Orange | PWM signal is given in through this wire to drive the motor |

Table 3.3 Servo motor wire configuration

### TowerPro SG-90 Features

* Operating Voltage is +5V typically
* Torque: 2.5kg/cm
* Operating speed is 0.1s/60°
* Gear Type: Plastic
* Rotation : 0°-180°
* Weight of motor : 9gm
* Package includes gear horns and screws

### SG-90 Servo Motor Equivalent

MG90S Metal Gear, MG995 High Torque Metal Gear, VTS-08A Analog Servo

**Selecting our Servo Motor**

There are lots of servo motors available in the market and each one has its own speciality and applications. The following two paragraphs will help you identify the right type of servo motor for your project/system.

Most of the hobby Servo motors operates from 4.8V to 6.5V, the higher the voltage higher the torque we can achieve, but most commonly they are operated at +5V.  Almost all hobby servo motors can rotate only from 0° to 180° due to their gear arrangement so make sure you project can live with the half circle if no, you can prefer for a 0° to 360° motor or modify the motor to make a full circle. The gears in the motors are easily subjected to wear and tear, so if your application requires stronger and long running motors you can go with metal gears or just stick with normal plastic gear.

Next comes the most important parameter, which is the torque at which the motor operates. Again there are many choices here but the commonly available one is the 2.5kg/cm torque which comes with the Towerpro SG90 Motor. This 2.5kg/cm torque means that the motor can pull a weight of 2.5kg when it is suspended at a distance of 1cm. So if you suspend the load at 0.5cm then the motor can pull a load of 5kg similarly if you suspend the load at 2cm then can pull only 1.25. Based on the load which you use in the project you can select the motor with proper torque. The below picture will illustrate the same.

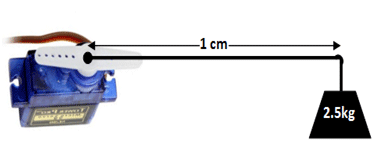


Figure 3.7 Depicting application of load in servo motor

### How to use a Servo Motor

After selecting the right Servo motor for the project, comes the question how to use it. As we know there are three wires coming out of this motor. The description of the same is given on top of this page. To make this motor rotate, we have to power the motor with +5V using the Red and Brown wire and send PWM signals to the Orange colour wire. Hence we need something that could generate PWM signals to make this motor work, this something could be anything like a 555 Timer or other Microcontroller platforms like Arduino, PIC, ARM or even a microprocessor like Raspberry Pie.

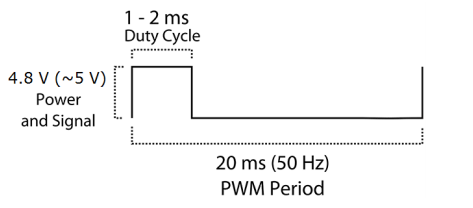


Figure 3.8 PWM signal showing working of servo

From the picture we can understand that the PWM signal produced should have a frequency of 50Hz that is the PWM period should be 20ms. Out of which the On-Time can vary from 1ms to 2ms. So when the on-time is 1ms the motor will be in 0° and when 1.5ms the motor will be 90°, similarly when it is 2ms it will be 180°. So, by varying the on-time from 1ms to 2ms the motor can be controlled from 0° to 180°

### Applications

* Used as actuators in many robots like Biped Robot, Hexapod, robotic arm etc..
* Commonly used for steering system in RC toys
* Robots where position control is required without feedback
* Less weight hence used in multi DOF robots like humanoid robots

### SG90 Servo Motor Dimensions

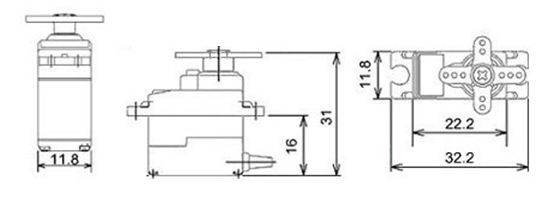
****

Figure 3.9 SG90 servo motor dimensions

### LDR

The **LightDependent Resistor (LDR)** is just another special type of Resistor and hence has no polarity. Meaning they can be connected in any direction. They are breadboard friendly and can be easily used on a perf board also. The symbol for LDR is just as similar to Resistor but adds to inward arrows as shown above. The arrows indicate the light signals.

**LDR Features**

* Can be used to sense Light
* Easy to use on Breadboard or Perf Board
* Easy to use with Microcontrollers or even with normal Digital/Analog IC
* Small, cheap and easily available
* Available in PG5 ,PG5-MP, PG12, PG12-MP, PG20 and PG20-MP series

### Other Resistor based Components

Higher Power Resistor, [Potentiometer](https://components101.com/potentiometer) (Variable Resistor), [Thermistor](https://components101.com/ntc-thermistor-10k).

**Where to use a LDR**

A**photoresistor** or **LDR** (Light Dependent Resistor), as the name suggests will change it resistance based on the light around it. That is when the resistor is placed in a dark room it will have a resistance of few Mega ohms and as we gradually impose light over the sensor its resistance will start to decrease from Mega Ohms to few Ohms.This property helps the LDR to be used as a **Light Sensor**. It can detect the amount of light falling on it and thus can predict days and nights. So if you are looking for a sensor to sense light or to distinguish between days and nights then this sensor is the cheap and modest solution for you.

**How to use a LDR sensor**

A LDR is one of the [different types of resistors](https://components101.com/articles/resistor-basics-types-and-uses), hence using it is very easy. There are many ways and different circuit in which an LDR can be used. For instance it can be used with Microcontroller Development platforms like [Arduino](https://components101.com/microcontrollers/arduino-uno), [PIC](https://components101.com/pic16f877a-pin-diagram-description-features-datasheet) or even normal Analog IC’s like Op-amps. A potential Divider is a circuit which has two resistors in series. A constant voltage will be applied across the both the resistor and the output voltage will be measured from the lower resistor. In our case, the lower resistor will be a **LDR**and the constant voltage will be +5V. The set-up is shown below

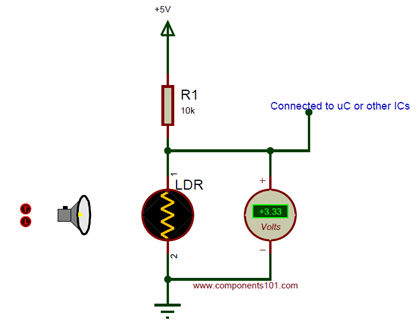


Figure 3.10 Connectionof LDR

A DC multimeter is used to monitor the voltage across the LDR. As the Lamp is moved towards the resistor the resistance value of the LDR will decrease as a result the voltage drop across it will decrease. The near you bring the Lamp the lower the voltage will get and the farther you move away your Voltage value will increase. **LDR provides a variable voltage based on Light**, we can feed this voltage either to a Microcontroller and use ADC to convert the Voltage to meaning full data or use a Op-amp as a Voltage comparator and trigger an specific output like LED or [Buzzer](https://components101.com/buzzer-pinout-working-datasheet) for a specific Voltage from the LDR.

**Applications**

* Automatic Street Light
* Detect Day or Night
* Automatic Head Light Dimmer
* Position sensor
* Used along with LED as obstacle detector
* Automatic bedroom Lights
* Automatic Rear view mirror

**2D – Model of LDR (PG5)**

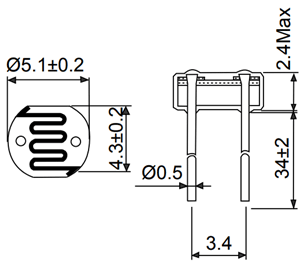
****

Figure 3.11 2D-Model of LDR

### Features

* RoHS (Restriction of Hazardous Substances) Compliant
* Quality level is high
* Cost is economical
* Wavelength from 635 nm to 660 nm
* Rise and fall time is 0.5ns
* Package available:-TO-18(dia. 5.6mm), TO-5 (dia. 9 mm)

### Technical Specification

* Operating Temperature:- -10～+40 ℃
* Storage Temperature:- -15～+85 ℃
* Output power (Po):- 5 mW
* Normal and maximum operating voltage is 2.2 and 2.7 respectively
* Threshold current in minimum, normal and maximum condition are 15, 20 and 30mA
* Operating current is 65 to 80mA
* Beam Angle deviation:-
* For both parallel and perpendicular condition its between -3 to 3 degree
* Beam divergence:-
* For parallel condition it’s between 8 to 12 degree
* For perpendicular condition it’s between 23 to 32 degree

### Characteristics Curves

#### 1. Temperature Effect on Operation of Laser Diode

This graph is between Optical output power v/s forward current. It’s clear from the graph that laser output will only be visible if obtained above the threshold value of the laser diode. Before the threshold value the output of the laser diode is zero. After the threshold value the output of laser diode increase with slightly increase in forward voltage. The **effect of temperature in the operation of Laser Diode** is shown in graph below:

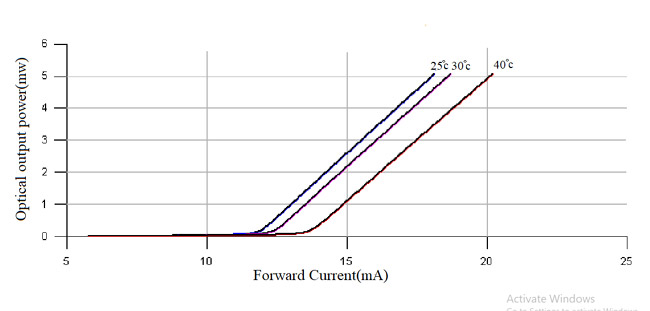


Figure 3.12 Effect of temperature on operation of laser

#### 2. Laser Beam Divergence in parallel and perpendicular plane

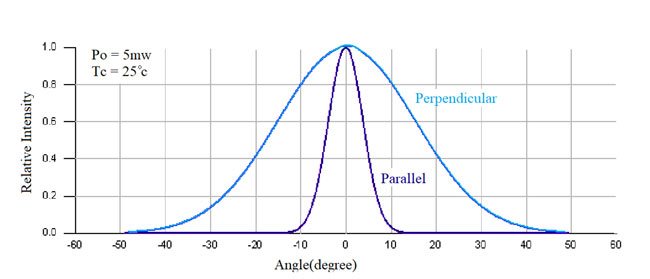


Figure 3.13 Plot showing laser beam divergence in parallel and perpendicular plane

#### 3. Forward voltage v/s Forward current

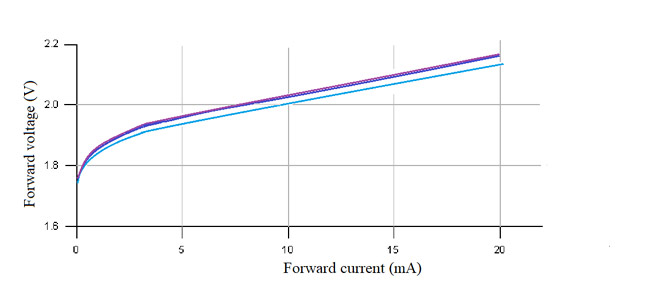


Figure 3.14 Plot between forward voltage and forward current

#### 4. Output Power v/s Forward Current

If the direction of moving current is forward and the output is continuously increasing, after the kink level the laser face a sudden breakdown which is the COD (Catastrophic Optical Damage) level. At this level due to high optical density the crystal at the face of diode melts. At the time of manufacturing of Red lasers a special care is taken to avoid surge like static electricity and increase in current, because in red laser the oscillation is occur with the low power of 2 to 3 mW even after the breakdown. As the element is damaged the laser gets damaged or not able to work.

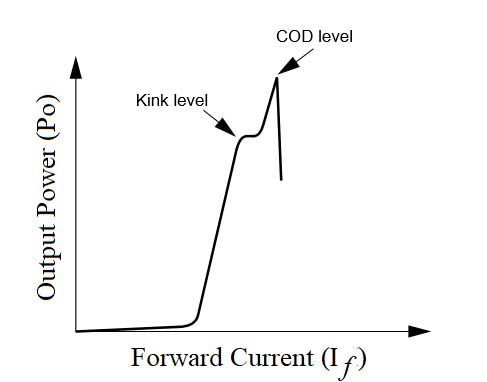


Figure 3.15 Plot showing output power v/s forward current

### 

### How to use a Laser Diode?

If we want to operate a **Laser diode** then we must have **laser diode driver circuit**. As it helps in limiting current then supply it to laser diode. A laser diode can only work properly with the help of this circuit, if we directly connect it to the supply, because of having more current it will damage and if the value of current is low then the laser diode will not operate. Laser Diode driver circuit helps in providing a correct value of current to operate the laser diode. For making a laser diode driver circuit we need a few no. of components like [resistor](https://components101.com/resistor), [capacitor](https://components101.com/electrolytic-capacitor-pinout-datasheet) and a voltage regulator IC.

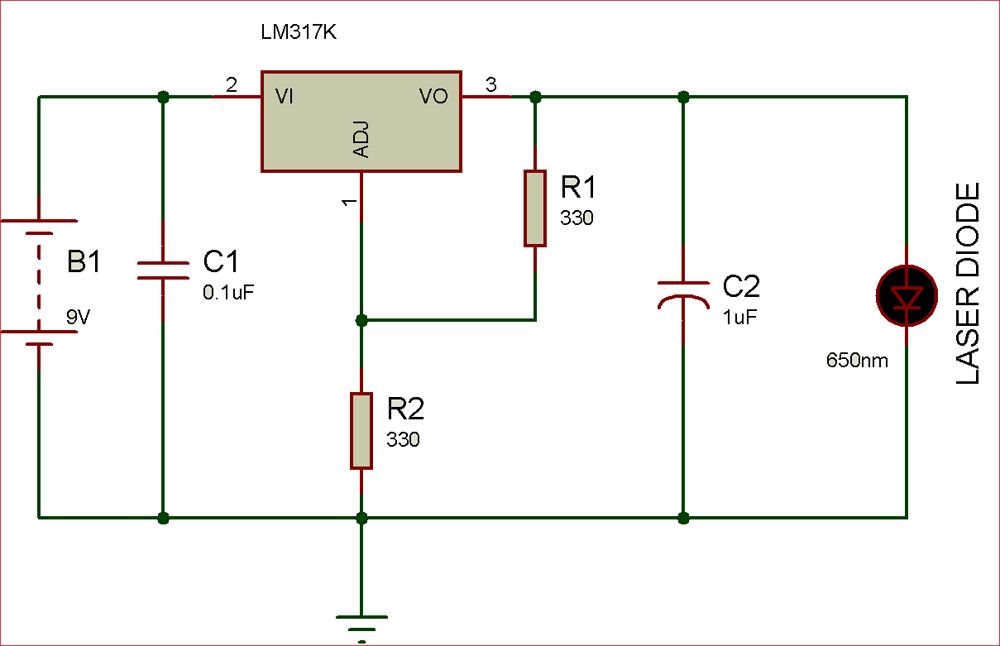


Figure 3.16 Laser Diode connection

The first capacitor in the circuit filter the High-frequency noise from the DC supply. The second capacitor works as power load balancer used to filter the fluctuating signals of output voltage. And a [voltage regulator IC](https://components101.com/7805-voltage-regulator-ic-pinout-datasheet) is used to provide a fix output voltage and we can adjust the output voltage by changing the value of resistor. You can use a [potentiometer](https://components101.com/potentiometer) instead of resistor (R2) to adjust the intensity of laser light.

### Applications

* **Industrial applications:** Engraving, cutting, scribing, drilling, welding, etc.
* **Medical applications:**to remove unwanted tissues, diagnostics of cancer cells using fluorescence, dental medication.
* Telecommunication
* Military application
* Data storage

### ****2D-Model****

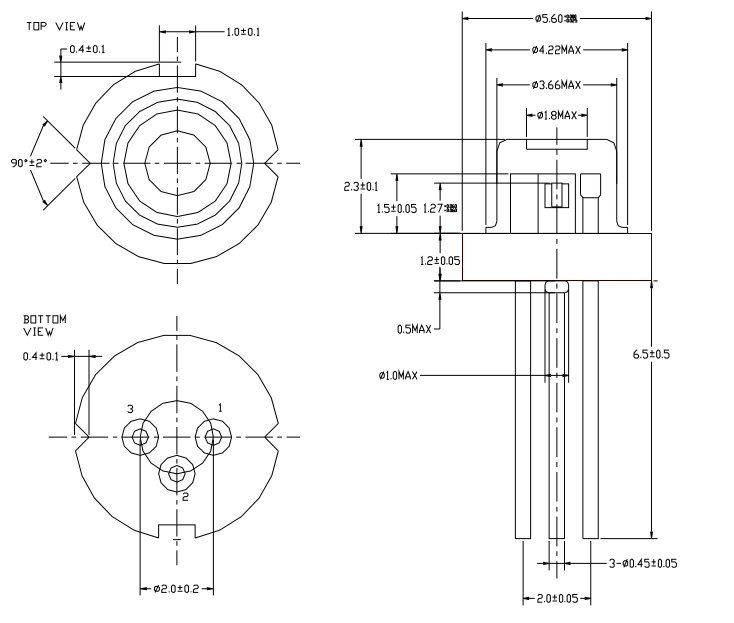


Figure 3.172D-model for laser diode

**3.2.2 Software Implementation**

The NodeMCU programming model is similar to that of [Node.js](https://en.wikipedia.org/wiki/Node.js), only in Lua. It is asynchronous and event-driven. Many functions, therefore, have parameters forcallback functions.

### Tool Overview

### esptool.py

A Python-based, open source, platform independent, utility to communicate with the ROM bootloader in Espressif ESP8266.

Supported platforms: OS X, Linux, Windows, anything that runs Python

**Running esptool.py**

Running the following command to flash an aggregatedbinary as is produced for example by the [cloud build service](https://nodemcu.readthedocs.io/en/master/build/#cloud-build-service) or the [Docker image](https://nodemcu.readthedocs.io/en/master/build/#docker-image).

esptool.py --port <serial-port-of-ESP8266>write\_flash -fm<flash-mode> 0x00000 <nodemcu-firmware>.bin

[flash-mode](https://github.com/espressif/esptool/#flash-modes) is qio for most ESP8266 ESP-01/07 (512 kByte modules) and dio for most ESP32 and ESP8266 ESP-12 (>=4 MByte modules). ESP8285 requires dout.

**Points to be kept in mind**

* To know the capacity of the flash chip on your device. It might help to double check as e.g. some ESP-01 modules come with 512kB while others are equipped with 1MB.
* esptool.py is under heavy development. It's advised you run the latest version (check with esptool.py version). Since this documentation may not have been able to keep up refer to the [esptool flash modes documentation](https://github.com/themadinventor/esptool#flash-modes) for current options and parameters.
* The firmware image file contains default settings dio for flash mode and 40m for flash frequency.
* In some uncommon cases, the [SDK init data](https://nodemcu.readthedocs.io/en/master/flash/#sdk-init-data) may be invalid and NodeMCU may fail to boot. The easiest solution is to fully erase the chip before flashing: esptool.py --port <serial-port-of-ESP8266>erase\_flash

NodeMCUPyFlasher

Self-contained [NodeMCU](https://github.com/nodemcu/nodemcu-firmware) flasher with GUI based on [esptool.py](https://github.com/espressif/esptool) and [wxPython](https://www.wxpython.org/).

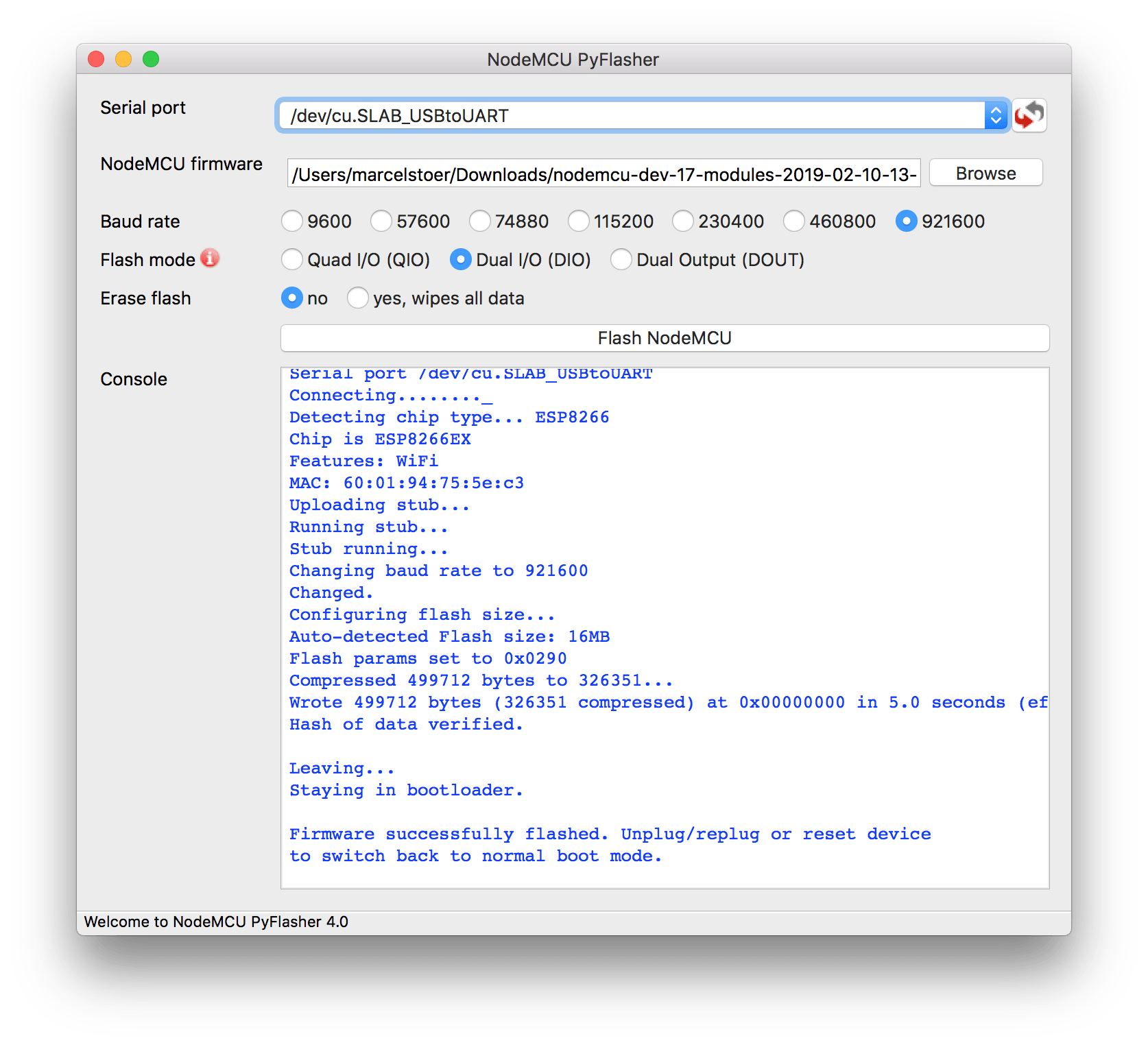


Figure 3.18 NodeMCUPyFlasher

Supported platforms: anything that runs Python, runnable .exe available for Windows and .dmgformacOS

**Putting device into flash mode**

To enable ESP8266 firmware flashing GPIO0 pin must be pulled low before the device is reset. Conversely, for a normal boot, GPIO0 must be pulled high or floating.

If we have a [NodeMCU dev kit](https://github.com/nodemcu/nodemcu-devkit-v1.0) then we don't need to do anything, as the USB connection can pull GPIO0 low by asserting DTR and reset your board by asserting RTS.

If we have an ESP-01 or other device without built-in USB, you will need to enable flashing yourself by pulling GPIO0 low or pressing a "flash" switch, while powering up or resetting the module.

**Files to be flashed**

If we build your firmware with the [cloud builder or the Docker image](https://nodemcu.readthedocs.io/en/master/build/), or any other method that produces a combined binary, then you can flash that file directly to address 0x00000.

Otherwise, if we built your own firmware from source code:

* bin/0x00000.bin to 0x00000
* bin/0x10000.bin to 0x10000

**Upgrading Firmware**

There are three potential issues that arise from upgrading (or downgrading!) firmware from one NodeMCU version to another:

* Lua scripts written for one NodeMCU version (like 0.9.x) may not work error-free on a more recent firmware. For example, Espressif changed the socket:send operation to be asynchronous i.e. non-blocking.
* The NodeMCU flash file system may need to be reformatted, particularly if its address has changed because the new firmware is different in size from the old firmware. If it is not automatically formatted then it should be valid and have the same contents as before the flash operation. You can still run [file.format()](https://nodemcu.readthedocs.io/en/master/modules/file/#fileformat) manually to re-format your flash file system. You will know if you need to do this if your flash files exist but seem empty, or if data cannot be written to new files. However, this should be an exceptional case. Formatting a file system on a large flash device (e.g. the 16MB parts) can take some time. So, on the first boot, you shouldn't get worried if nothing appears to happen for a minute. There's a message printed to console to make you aware of this.
* The Espressif SDK Init Data may change between each NodeMCU firmware version, and may need to be erased or reflashed. Fully erasing the module before upgrading firmware will avoid this issue.

### ESPlorer

The essential multiplatforms tools for any ESP8266 developer from luatool author’s, including Lua for NodeMCU and MicroPython. Also, all AT commands are supported. Requires Java (Standard Edition - SE ver 7 and above) installed.

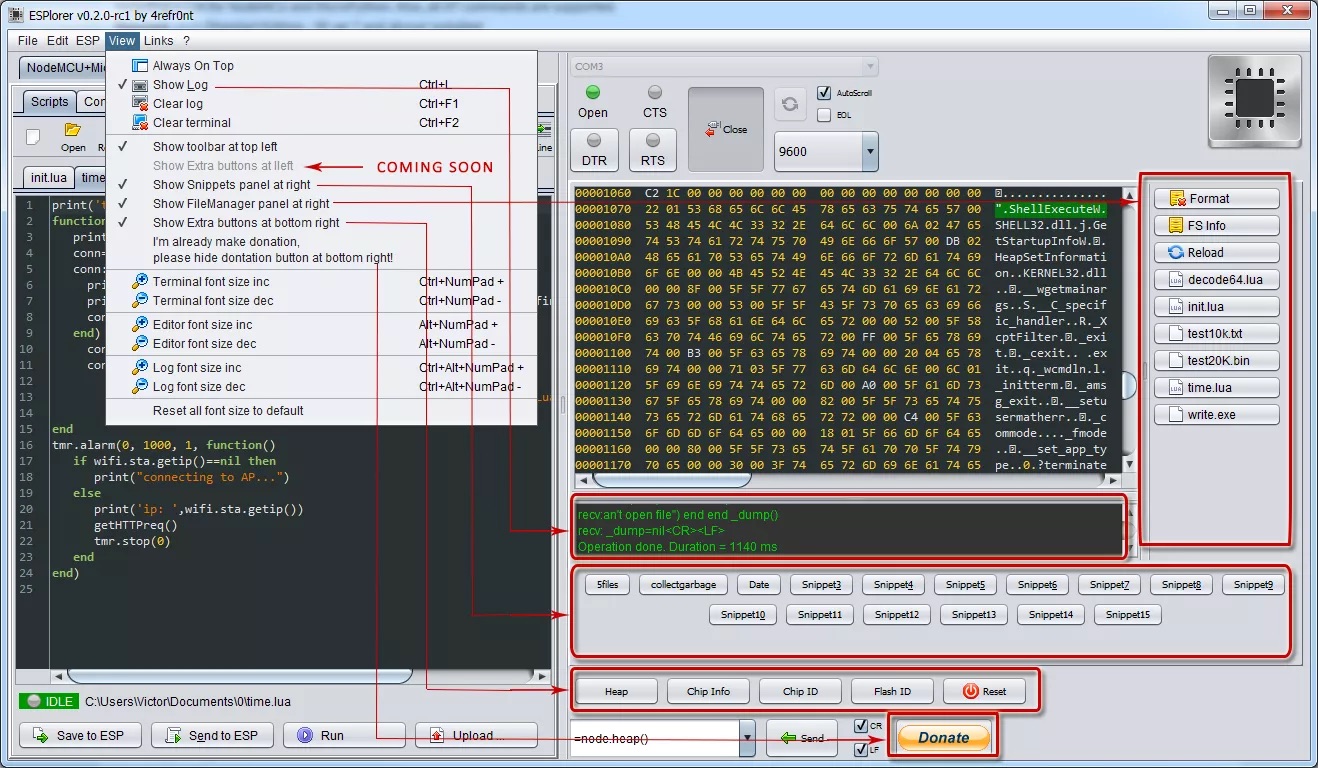


Figure 3.19 ESplorer

Supported platforms: macOS, Linux, Windows, anything that runs Java

**MQTT Module**

|  |  |
| --- | --- |
| [mqtt.Client()](https://nodemcu.readthedocs.io/en/master/modules/mqtt/#mqttclient) | Creates a MQTT client. |
| [mqtt.client:close()](https://nodemcu.readthedocs.io/en/master/modules/mqtt/#mqttclientclose) | Closes connection to the broker. |
| [mqtt.client:connect()](https://nodemcu.readthedocs.io/en/master/modules/mqtt/#mqttclientconnect) | Connects to the broker specified by the given host, port, and secure options. |
| [mqtt.client:lwt()](https://nodemcu.readthedocs.io/en/master/modules/mqtt/#mqttclientlwt) | Setup Last Will and Testament (optional). |
| [mqtt.client:on()](https://nodemcu.readthedocs.io/en/master/modules/mqtt/#mqttclienton) | Registers a callback function for an event. |
| [mqtt.client:publish()](https://nodemcu.readthedocs.io/en/master/modules/mqtt/#mqttclientpublish) | Publishes a message. |
| [mqtt.client:subscribe()](https://nodemcu.readthedocs.io/en/master/modules/mqtt/#mqttclientsubscribe) | Subscribes to one or several topics. |
| [mqtt.client:unsubscribe()](https://nodemcu.readthedocs.io/en/master/modules/mqtt/#mqttclientunsubscribe) | Unsubscribes from one or several topics. |
|  |  |

Table 3.4 WiFi Module functions

**WiFi modes**

Devices that connect to WiFi network are called stations (STA). Connection to Wi-Fi is provided by an access point (AP), that acts as a hub for one or more stations. The access point on the other end is connected to a wired network. An access point is usually integrated with a router to provide access from Wi-Fi network to the internet. Each access point is recognized by a SSID (Service Set IDentifier), that essentially is the name of network you select when connecting a device (station) to the WiFi.

Each ESP8266 module can operate as a station, so we can connect it to the WiFi network. It can also operate as a soft access point (soft-AP), to establish its own WiFi network. Therefore, we can connect other stations to such modules. Third, ESP8266 is also able to operate both in station and soft access point mode at the same time. This offers the possibility of building e.g. [mesh networks](https://en.wikipedia.org/wiki/Mesh_networking).

Station

Station (STA) mode is used to get the ESP8266 connected to a WiFi network established by an access point.

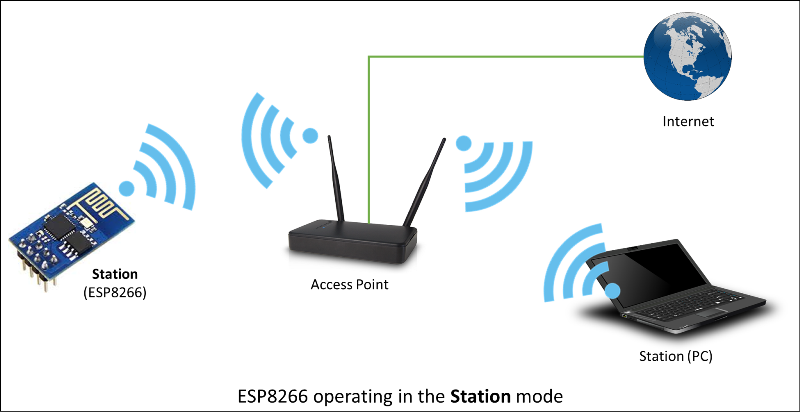


Figure 3.20 ESP8266 operating in the Station mode

**Soft Access Point**

An access point (AP) is a device that provides access to Wi-Fi network to other devices (stations) and connects them further to a wired network. ESP8266 can provide similar functionality except it does not have interface to a wired network. Such mode of operation is called soft access point (soft-AP). The maximum number of stations connected to the soft-AP is five.



Figure 3.21 ESP8266 operating in the Soft Access Point mode

The soft-AP mode is often used and an intermediate step before connecting ESP to a WiFi in a station mode. This is when SSID and password to such network is not known upfront. The module first boots in soft-AP mode, so we can connect to it using a laptop or a mobile phone. Then we are able to provide credentials to the target network. Once done ESP is switched to the station mode and can connect to the target WiFi.

Such functionality is provided by the [NodeMCU enduser setup module](https://nodemcu.readthedocs.io/en/master/modules/enduser-setup/).

**Station + Soft Access Point**

Another handy application of soft-AP mode is to set up [mesh networks](https://en.wikipedia.org/wiki/Mesh_networking). ESP can operate in both soft-AP and Station mode so it can act as a node of a mesh network.

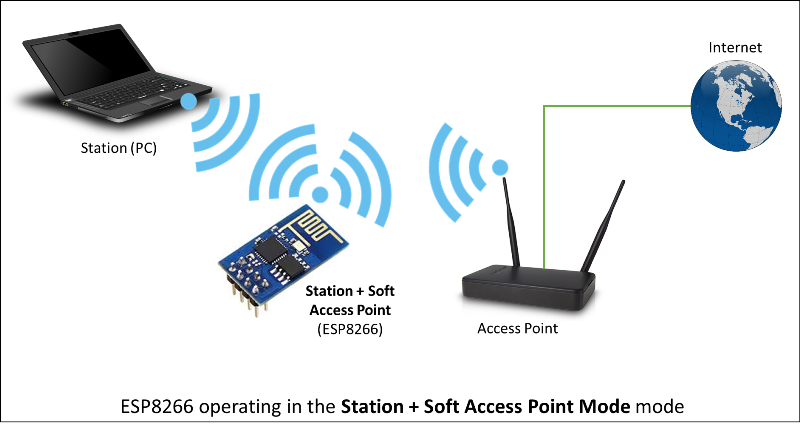


Figure 3.22 ESP8266 operating in Station + Soft Access Point mode

**Function reference**

The NodeMCUWiFi control is spread across several tables:

* [wifi](https://nodemcu.readthedocs.io/en/master/modules/wifi/#wifigetchannel) for overall WiFi configuration
* [wifi.sta](https://nodemcu.readthedocs.io/en/master/modules/wifi/#wifista-module) for station mode functions
* [wifi.ap](https://nodemcu.readthedocs.io/en/master/modules/wifi/#wifiap-module) for wireless access point (WAP or simply AP) functions

Intel XDK was an Integrated Development Environment (IDE) created by [Intel](https://en.wikipedia.org/wiki/Intel) to create native apps for mobile phones and tablets using web technologies like [HTML5](https://en.wikipedia.org/wiki/HTML5), [CSS](https://en.wikipedia.org/wiki/CSS_style_sheets) and [JavaScript](https://en.wikipedia.org/wiki/JavaScript).[[1]](https://en.wikipedia.org/wiki/Intel_XDK#cite_note-drdobbs-1) Apps were created edited and simulated using the IDE then exported to Adobe® PhoneGap or [Cordova](https://en.wikipedia.org/wiki/PhoneGap) CLI for compilation into a native code app. The IDE allowed a developer to target the same solution to different platforms, thus reducing the amount of code required to ship a cross-platform product. The XDK was first launched in October 2013.

XDK has been re-purposed by Intel for [IoT](https://en.wikipedia.org/wiki/Internet_of_things) development, and core mobile development features have been deprecated by Intel[[2]](https://en.wikipedia.org/wiki/Intel_XDK#cite_note-intel-deprecation-2)[[3]](https://en.wikipedia.org/wiki/Intel_XDK#cite_note-intel-alternatives-3)and are no longer supported. The accompanying cross-platform Cordova cloud build service has been taken offline.It is not recommended to use XDK for new phone/tablet app development projects.

**Features**

Code Hinting

Code Completion

Emulator

Device Testing via [Intel App Preview](https://software.intel.com/en-us/xdk/docs/intel-xdk-app-preview-overview) Mobile app

[Cordova](https://en.wikipedia.org/wiki/PhoneGap) & Third Party Plugins Support

**Python**

Python is an [interpreted](https://en.wikipedia.org/wiki/Interpreted_language), [high-level](https://en.wikipedia.org/wiki/High-level_programming_language), [general-purpose programming language](https://en.wikipedia.org/wiki/General-purpose_programming_language). Created by [Guido van Rossum](https://en.wikipedia.org/wiki/Guido_van_Rossum) and first released in 1991, Python has a design philosophy that emphasizes [code readability](https://en.wikipedia.org/wiki/Code_readability), notably using [significant whitespace](https://en.wikipedia.org/wiki/Significant_whitespace). It provides constructs that enable clear programming on both small and large scales. Van Rossum led the language community until stepping down as leader in July 2018.

Python features a [dynamic type](https://en.wikipedia.org/wiki/Dynamic_type) system and automatic [memory management](https://en.wikipedia.org/wiki/Memory_management). It supports multiple [programming paradigms](https://en.wikipedia.org/wiki/Programming_paradigm), including [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming), [imperative](https://en.wikipedia.org/wiki/Imperative_programming), [functional](https://en.wikipedia.org/wiki/Functional_programming) and [procedural](https://en.wikipedia.org/wiki/Procedural_programming). It also has a comprehensive [standard library](https://en.wikipedia.org/wiki/Standard_library).

Python interpreters are available for many [operating systems](https://en.wikipedia.org/wiki/Operating_system). [CPython](https://en.wikipedia.org/wiki/CPython), the [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation) of Python, is [open source](https://en.wikipedia.org/wiki/Open-source_software)software and has a community-based development model, as do nearly all of Python's other implementations. Python and CPython are managed by the non-profit [Python Software Foundation](https://en.wikipedia.org/wiki/Python_Software_Foundation).

**MQTT**

[MQTT](https://en.wikipedia.org/wiki/MQTT) (Message Queue Telemetry Transport) is an ISO standard (ISO/IEC PRF 20922) publish-subscribe based “light weight” messaging protocol for use on top of the TCP/IP protocol. It is designed for connections with remote locations where a “small code footprint” is required or the network bandwidth is limited.It’s very easy to use the MQTT protocol to exchange small messages between several devices.

**Basics of Publish-subscribe Model**

A message has a topic and a payload, like the subject and the content of an e-mail.

The Publisher sends a message to the network.

The Subscriber listens for messages with a particular topic.

The Broker is responsible for coordinating the communication between publishers and subscribers. It can also store messages while subscribers are offline

**Requirements**

We need a broker that is always available. Just one for the whole network. It can be a PC, a Raspberry Pi or even an EV3. If it is a Debian-based linuxsystem we can use mosquito

sudo apt-get install mosquitto

This installs and also starts the mosquitto daemon. You can check if it is working by using the systemctl command:

robot@ev3dev:~# systemctl status mosquitto

● mosquitto.service - LSB: mosquitto MQTT v3.1 message broker

Loaded: loaded (/etc/init.d/mosquitto)

Active: active (running) since Wed 2016-05-11 07:40:51 WEST; 7min ago

CGroup: /system.slice/mosquitto.service

└─685 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf

Now we are able to send and receive messages through the broker (by default mosquitto uses port 1883).

This project uses python scripts so we need to install the python library paho-mqtt. You need ‘pip3’ to install this module, so if you have not already done so, you will need to install pip3:

sudo apt-get install python3-pip

Now we can install paho-mqtt:

sudo pip3 install paho-mqtt

All scripts were tested successully on a EV3 running the latest ev3dev version and also on a Raspberry Pi 3 with a BrickPi running the same ev3dev version and a laptop running Ubuntu 16.04.

**Publisher example**

A very simple script to publish a message:

#!/usr/bin/env python3

import paho.mqtt.client as mqtt

**# This is the Publisher**

client = mqtt.Client()

client.connect("localhost",1883,60)

client.publish("topic/test", "Hello world!");

client.disconnect();

Note: if using an external broker (i.e. the mosquittodeamon is not running in the EV3 that publishes messages) replace localhost with the IP address of the device that hosts the broker.

**Subscriber example**

Any MQTT client that is connected to our broker and has subscribed for “topic/test” will receive a MQTT message with “Hello world!” as the payload. We can test it with a mobile phone (there are several free MQTT client apps available) but we can also test it on our PC or on another EV3:

#!/usr/bin/env python3

import paho.mqtt.client as mqtt

**# This is the Subscriber**

defon\_connect(client, userdata, flags, rc):

print("Connected with result code "+str(rc))

client.subscribe("topic/test")

defon\_message(client, userdata, msg):

if msg.payload.decode() == "Hello world!":

print("Yes!")

client.disconnect()

client = mqtt.Client()

client.connect("THE\_IP\_ADDRESS\_OF\_OUR\_BROKER",1883,60)

client.on\_connect = on\_connect

client.on\_message = on\_message

client.loop\_forever()

Note: the second EV3 (the “Subscriber”) just needs the “paho-mqtt” library, there is no need to install the “mosquitto” daemon.

Note: when the publisher sends a string as payload use decode() as in the example above. When the Publisher sends a number, we can use int(msg.payload)

##### 3.3 Architecture

The project can execute according to the presence differently of the laser fencing or the iron fencing. When thelatter is present, the complete system is independent of the iron fencing breaching as it is not capable of transmitting an electrical signal and solely dependent on the ultrasonic sensor ranging which has to be active all the time thus consuming power even when it is not required. The rest operation of nodes and the complete system is exactly similar. In case of laser fencing security, the laser and LDR (light detecting resistor) circuit is kept at a distance of 300cm from the nodes of SUBS. Whenever there is breaching of laser fencing that is whenever the connection between the laser light and LDR is broken, it sends a "LOW" signal to the controller and a buzzer goes 'on'. This sets the servo motor on which ultrasonic sensor is mounted to go into the motion which has been till now in OFF model.The ultrasonic sensor goes from -60 degree to +60 degrees from neutral or zero reference position. This is done to accomplish the complete -180 to +180 degree surveillances even in case of one or more node failure by expanding the range of nearby nodes. The individual node efficiency is sacrificed for the robustness and fault tolerance of the robust system. Whenever the intruder is detected by ultrasonic sensor, the relay switch sends a HIGH signal to the actuator which is laser light here starts firing continuously in the direction of the intruder and observes it continuously if it is moving. A Counter circuit is also used at the relay switch to consider the number of shots fired (switching done). A counter circuitry is used at the laser fencing to consider the number of intrusions. The distance of the intruder along with the timestamp is also transmitted through MQTT and web-socketconnection to the server (or the cloud) using gateway. Each node of SUBS System has its own identity and maintains a default list of the usernames and password to which it has to connect. In case of node failure, a trigger has generated this changes the program to vary the range of surveillance. The data from sensors is transmitted for sure transfer of the messages using MQTT due to its data time interval. The data is transmitted using Publish-Subscribe model on a public broker on the 'Topic' of one's choice. The data is transmitted using a protocol for easy understanding and retrieval of the messages. The data received can be analysed by plotting data against the number of specifications like the number of shots fired, the number of intrusions made, the frequency of intrusions in particular time, etc. for better and quick understanding. Individual node can be switched ON/OFF from headquarters by choosing the action and publishing it on a subscribed topic by the nodes.

**3.3.1 Block Diagram**

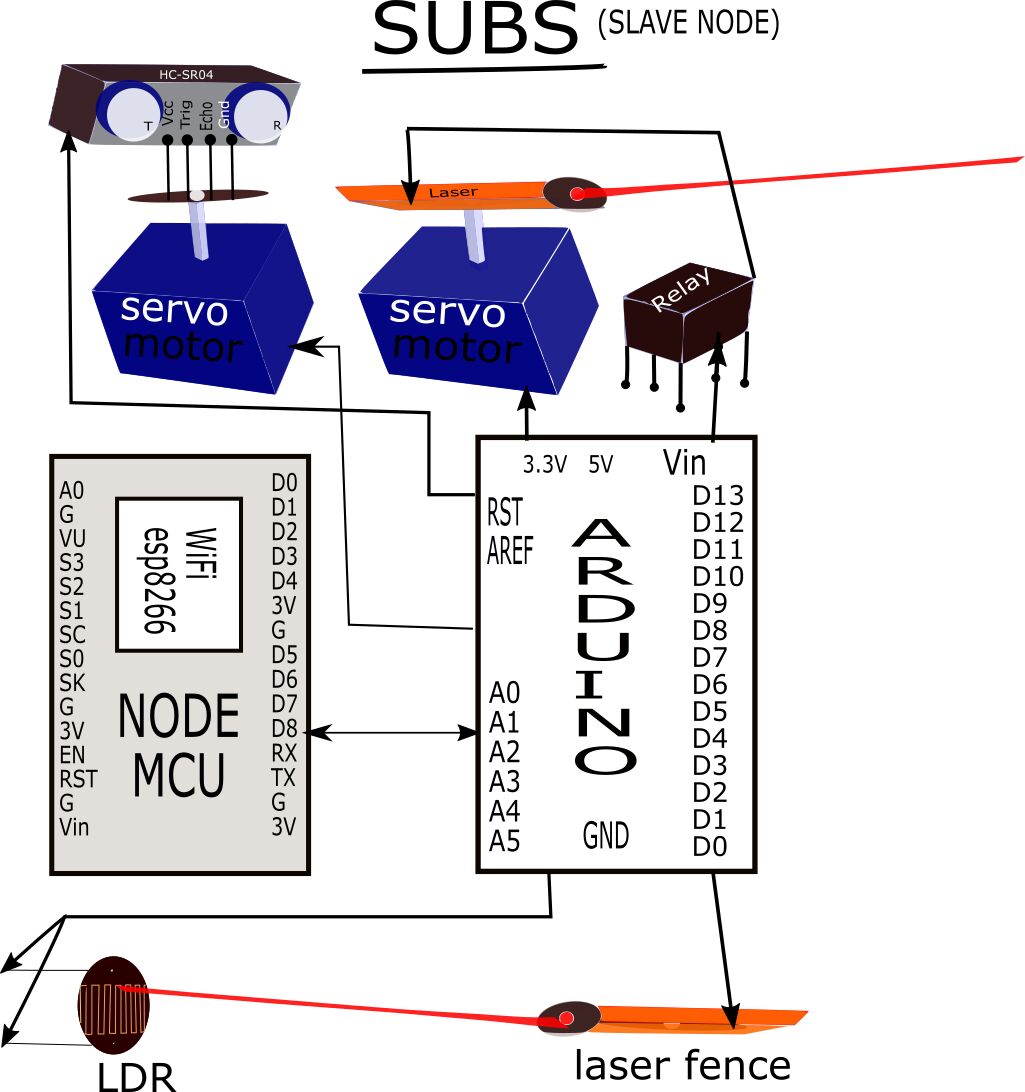


Figure 3.23 Block Diagram of SUBS System

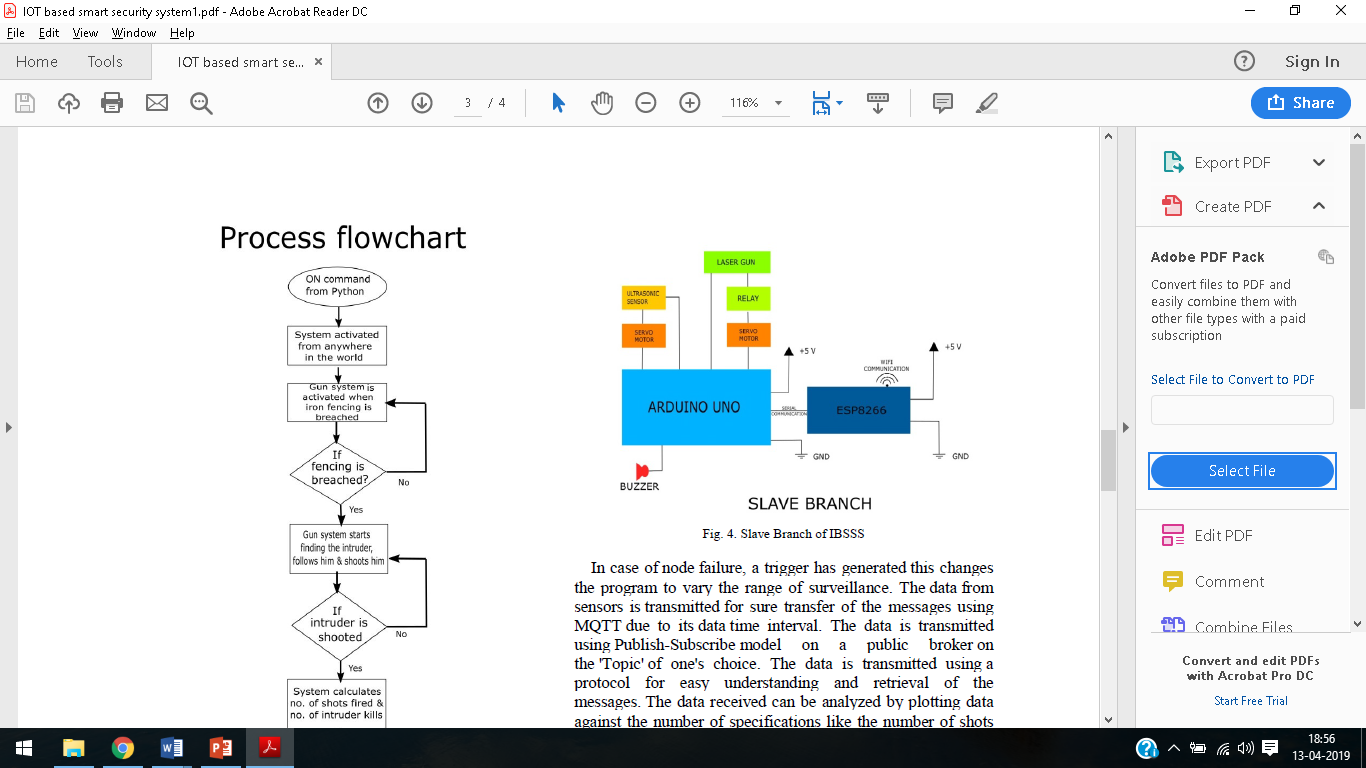


Figure 3.24 A single slave node

We have implemented a single slave node in our project whereas in the proposed project for perimeter fencing will be achieved by using multiple nodes in master and slave configuration wherein slave node will communicate with neighboring slave nodes as well as with the master node. The slave node will sense and the actuator action is also taken at slave nodes but no data aggregation takes place at the slave nodes. Data forwarding and data aggregation action would be performed by the master node.

**Proposed master-slave configuration of project**

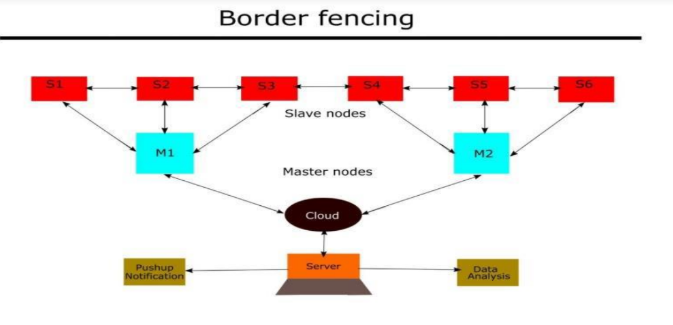
****

Figure 3.25 Proposed master slave configuration of nodes

**3.3.2 Process flowchart**

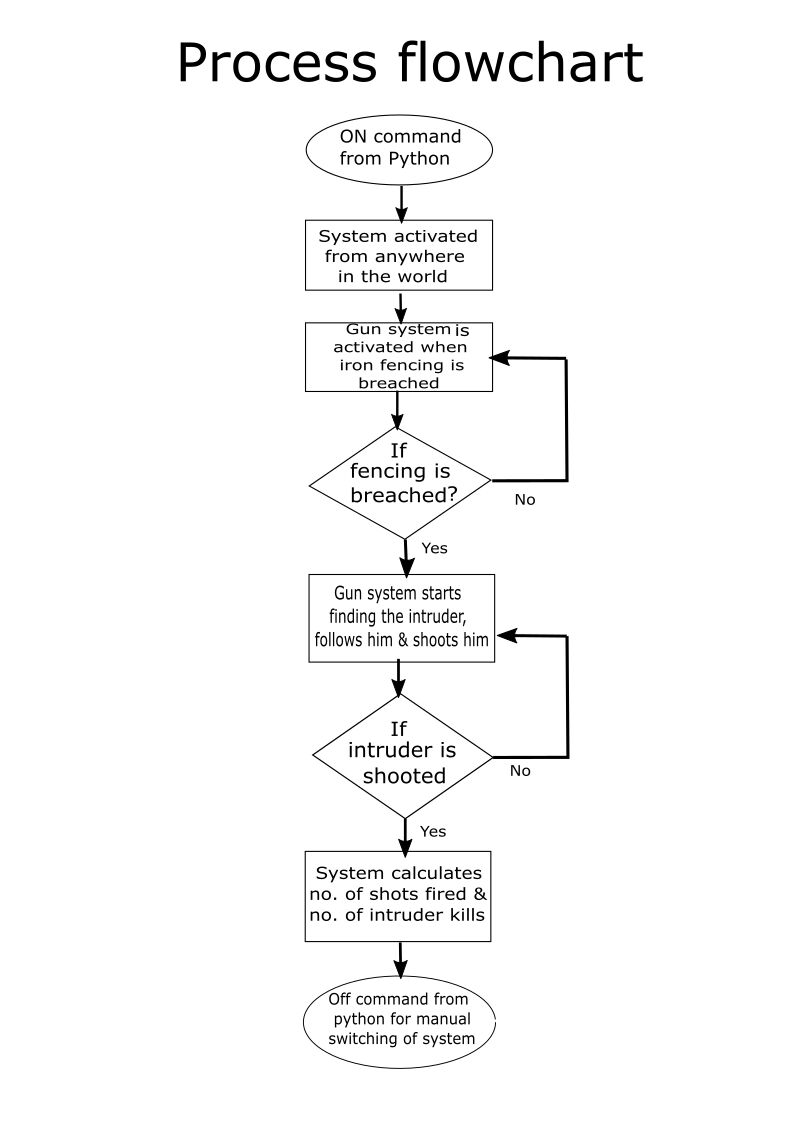


Figure 3.26 Process Flowchart and Data flow Of SUBS System

# Chapter 4: Result and Discussion

**4.1 Result**

**4.2 Conclusion**

SUBS is designed for the purpose of border security in no man region areas near the army, navy, air force bases. Itcan work in any part of the country irrespective of thegeographical conditions. This system can solely be used forsurveillance and monitoring in the no man areas. Withincreasing intrusions and breaching of iron fences alongthe borders, there has been a growing section of peopleadvocating for unmanned automatic gun control systems tobe implemented at the borders. So there is a need tosafeguard these places with least or no human supervision. This system is based on the technology of "Internetof Things".

The principals used in this technology are:

A. **3A's** (Always-Anytime-Anywhere)

B. **6T's** (Track-Think-Talk-Transfer-Trigger-Tell)

More the borders are protected, stronger and safer is the nation. The IOT basedSmart Security System

aiming to reduce the burden on forces which is determiningan additional security with high accuracy. It aims ateliminating enemies with utmost accuracy and reducinghuman-made errors. Also cost of IBS is relatively less whencompared with other security devices. It is provided with anarray of actuators to eliminate the enemy in a process ofraising alarm and is connected to other succeeding nodes as well as connected to the Internet for communication.Therefore, a particular node in the system can be classified as the "thing" in the IOT ecosystem which can be uniquely addressed and identified.

# Chapter 5: Future Scope

The project can also be extended to develop a feasible web-cam-controlled device which will increase the sophistication and accuracy. It will consume less amount very of power which ensures a much-needed condition for the borders. The intruders will be detected by using a web-cam so that the gun will shoot only the humans, not the animals/birds. This system supports the manual switching of the device.

Consequently, the sent messages for the switching can also be encrypted to make it more secure. This will ensure the intruder will not have the chance to switch ON/OFF the device by the use of hacking. Since, it is very complex to change the power sources every time at the borders so the system can use the renewable sources of energy such as solar power, wind power, etc. This will increase the efficiency as well as the organized.

**Impact in future:**

➢ Reduced human supervision required and substantial loss of collateral damage to lives of

our security personnel are reduced. revolutionizing the whole security framework.

➢ Precise and accurate information on points of intrusions.

➢ Prediction and strategic development for such intrusions.

➢ Acting as a first layer of security.

➢ Modified version can act as smart doors in smart homes.

# References

1. Friedman,N.andRussell,S.1997.Imagesegmentationinvideosequences:aprobabilistic approach. In *Proc. 13th Conf. Uncertainty in Artificial Intelligence*, (Brown University, Providence, Rhode Island, USA, August 1-3, 1997). Morgan Kaufmann, San Francisco, CA, 175 -181.
2. Bodor, R., Bennett, J. and Papanikolopoulos, N. 2006. Vision-Based Human Trackingand Activity Recognition. Retrieved April 4, 2006, from Artifical Intelligence, Robotics and Vision Laboratory, Department of Computer Science and Engineering, University of Minnesota:<http://mha.cs.umn.edu/Papers/Vision_Tracking_Recognition.pdf>
3. Wei, G. and Sethi, I. K. 2000. Omni-face detection for video/image content description.In*Proceedings of the 2000 ACM Multimedia Workshops*, (Los Angeles, California, United States, 2000). ACM Press, New York, NY, USA,185-189.
4. Stauffer, C. and Grimson, W. 2000. Adaptive background mixture models for real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, (2000). IEEE Computer Society, Los Alamitos, CA,747-757.
5. Hu,W.,Tan*,*T.,Wang,L.andMaybank,S.2004.Asurveyonvisualsurveillanceofobject motion and behaviours. *IEEE transactions on systems, man, and cybernetics—part c: applications and reviews,* vol. 34, no. 3, (August 2004). IEEE Systems, Man, and Cybernetics Society,334-352.
6. Sidenbladh, H. 2004. Detecting Human Motion with Support Vector Machines. In *17th International Conference on Pattern Recognition (ICPR'04)*, 2, (2004). IEEE Computer Society, Los Alamitos, CA,188-191
7. Lipton, A. J., Fujiyoshi, H. and Patil, R. S. 1998. Moving target classification andtracking from real-time video. In *Proc. IEEE Workshop on Applications of Computer Vision*, (Princeton, NJ, USA, October 1998). IEEE Computer Society, Los Alamitos, CA,8–14.
8. Cucchiara,R.,Grana,C.,andTardini,G.2004.Track-basedandobject-basedocclusionfor people tracking refinement in indoor surveillance. In *Proceedings of the ACM 2nd internationalWorkshoponVideoSurveillanceandSensorNetworks*(NewYork,NY,USA, October 15, 2004). VSSN '04. ACM Press, New York, NY,81-87.
9. Wang,G.,Wong,T.andHeng,P.2005.Real-timesurveillancevideodisplaywithsalience. In*ProceedingsofthethirdACMinternationalworkshoponVideosurveillanceandsensor networks*, (Hilton, Singapore, 2005). ACM Press New York, NY, USA,37-44.
10. Ivanov, Y., Stauffer, C., Bobick, A. and Grimson W. L. E. 1999. Video Surveillance of Interactions.In*ProceedingsoftheSecondIEEEWorkshoponVisualSurveillance*,(1999). IEEE Computer Society, Los Alamitos, CA, USA,82.
11. Lou, J., Liu, Q., Tan, T., and Hu, W. 2002. Semantic Interpretation of Object Activities in a Surveillance System. In *Proceedings of the 16thinternational Conference on Pattern Recognition (Icpr'02) Volume 3*, vol. 3, (August 11 - 15, 2002). IEEE Computer Society, Los Alamitos, CA, USA, 30777.
12. Baker, C. L., Tenenbaum, J. B. and Saxe, R. R. (2006). Bayesian models of human action understanding. In *Advances in Neural Information Processing Systems18*.
13. Bezzi, M. and Groenevelt, R. 2006. Towards understanding and modeling individual behaviorandgroupdynamics.In*ProceedingsofthePervasive2006Workshop"Pervasive Technology Applied - Real-World Experiences with RFID and Sensor Networks"*,(Dublin, Ireland, May 7,2006).
14. Thirde, D., Borg, M., Ferryman, J., Fusier, F., Valentin, V., Bremond, F., andThonnat,

M. 2006. A Real-Time Scene Understanding System for Airport Apron Monitoring. In Proceedings *of the Fourth IEEE international Conference on Computer Vision Systems* (January 04 - 07, 2006). ICVS. IEEE Computer Society, Los Alamitos, CA, 26.

1. Wren, C. R., Azarbayejani, A., Darrell, T., and Pentland, A. P. 1997. Pfinder: Real-Time TrackingoftheHumanBody.*IEEETrans.PatternAnal.Mach.Intell.*vol.19,no.7,(July, 1997). IEEE Computer Society, Los Alamitos, CA,780-785.
2. Nascimento, J. C., Figueiredo, M. A. T. and Marques, J. S. 2005. Segmentation and Classification of Human Activities. In *Proceedings of International Workshop on Human Activity Recognition and Modelling (HAREM 2005 - in conjunction with BMVC 2005)* (Oxford, UK, September 2005), 79-86.
3. Cohen, I., Sebe, N., Cozman, F. G., and Huang, T. S. 2003. Semi-supervised learning for facial expression recognition. In *Proceedings of the 5th ACM SIGMM international Workshop on Multimedia Information Retrieval* (Berkeley, California, November 07 - 07, 2003). MIR '03. ACM Press, New York, NY,17-22.
4. Park, S. and Aggarwal, J. K. 2003. Recognition of two-person interactions using a hierarchical Bayesian network. In *First ACM SIGMM international Workshop onVideo*

*Surveillance* (Berkeley, California, November 02 - 08, 2003). IWVS '03. ACM Press, New

York, NY, 65-76.

1. Park, S. and Trivedi, M. M. 2006. Analysis and query of person-vehicle interactions in homography domain. In *Proceedings of the 4th ACM international Workshop on Video Surveillance and Sensor Networks* (Santa Barbara, California, USA, October 27 - 27, 2006). VSSN '06. ACM Press, New York, NY,101-110.
2. Jacobs, N. and Pless, R. 2006. Real-time constant memory visual summaries for surveillance.In*Proceedingsofthe4thACMinternationalWorkshoponVideoSurveillance and Sensor Networks* (Santa Barbara, California, USA, October 27 - 27, 2006). VSSN '06. ACM Press, New York, NY,155-160.
3. Chen, D., Yang, J., and Wactlar, H. D. 2004. Towards automatic analysis of social interaction patterns in a nursing home environment from video. In *Proceedings of the 6th ACM SIGMM international Workshop on Multimedia information Retrieval* (New York, NY, USA, October 15 - 16, 2004). MIR '04. ACM Press, New York, NY, 283-290.
4. Bashir, F., Porikli, F. 2006. Performance Evaluation of Object Detection and Tracking Systems. In *Proceeding of the 9thIEEE International Workshop on Performance Evaluation of Tracking and Surveillance* (New York, USA, June 18, 2006). PETS 2006. IEEE Computer Society, Los Alamitos, CA,7-14.
5. Ribeiro, P. C., and Santos-Victor, J. 2005. Human Activity Recognition from Video: modeling, feature selection and classification architecture. In *Proceedings of International Workshop on Human Activity Recognition and Modelling (HAREM 2005 - in conjunction with BMVC 2005)* (Oxford, UK, September, 2005),61-70.
6. Stauffer, C. and Grimson, W. 1999. Adaptive background mixture models for real-time tracking.In*ProceedingsoftheIEEEConf.ComputerVisionandPatternRecognition*,vol. 2,(FortCollins,CO,USA,June23-25,1999),IEEEComputerSociety,LosAlamitos,CA, 246–252.
7. Russ,J.C.1995.*TheImageProcessingHandbook(2ndEd.)*.CRCPress,Inc.BocaRaton, Fla.
8. Normal Distribution. Retrieved July 1, 2007, from Wikipedia, the free encyclopedia:<http://en.wikipedia.org/wiki/Normal_distribution>
9. Mahalanobis distance. Retrieved July 1, 2007, from Wikipedia, the free encyclopedia:<http://en.wikipedia.org/wiki/Mahalanobis_distance>
10. Keller, G. and Warrack, B. (2000). *Statistics for Management and Economics (5thedition)*

Duxbury, Pacific Grove, CA.

1. Koller, D., Weber, J., Huang, T. Malik, J. Ogasawara, G., Rao, B. and Russell, S. 1994. Toward robust automatic trafic scene analysis in real-time. In *Proceedings of the 12th Int'l Conference on Pattern Recognition (ICPR-94)*, (Jerusalem, Israel, October 9-13, 1994), 126-131.
2. Harville,M.,Gordon,G.,andWoodfill,J.2001.ForegroundSegmentationUsingAdaptive Mixture Models in Color and Depth. In *Proceedings of the IEEE Workshop on Detection and Recognition of Events in Video*, (Vancouver, BC, Canada, July 8 2001), IEEE Computer Society, Los Alamitos, CA,3-11.
3. Finlayson,G.D.,Hordley,S.D.,andDrew,M.S.2002.RemovingShadowsfromImages. In*Proceedingsofthe7thEuropeanConferenceonComputerVision-PartIV*(May28-31, 2002).A.Heyden,G.Sparr,M.Nielsen,andP.Johansen,Eds.LectureNotesInComputer Science, vol. 2353. Springer-Verlag, London,823-836.

## APPENDIX

|  |
| --- |
| Baselines.py |
| fromfuture importabsolute\_import |
| fromfuture importdivision |
| fromfuture importprint\_function |
| Importnumpyasnp |
| fromsix.movesimportxrange #pylint:disable=redefined-builtin |
| Importtensorflowastf |
| Importtranslate |
| importdata\_utils |
| importseq2seq\_model |
| #Dummyobjecttocreateparametersforalso-dummymodel |
| classObject(object): |
| Pass |
| defrunning\_average(actions\_dict,actions,k): |
| """ |
| Computetheerrorifwesimplytaketheaverageofthelastkframes. |
| Args |
| actions\_dict:Dictionarywherekeysaretheactions,andeachentryhasa |
| tupleof(enc\_in,dec\_in,dec\_out)poses. |
| actions:Listofstrings.Thekeysofactions\_dict. |
| k:Integer.Numberofframestouseforrunningaverage. |
| Returns |
| errs:adictionarywhere,foreachaction,wehavea100-longlistwiththe |
| errorateachpointintime. |
| """ |
| #Gethowmanybatcheswehave |
| enc\_in,dec\_in,dec\_out=actions\_dict[actions[0]] |
| n\_sequences=len(enc\_in) |
| seq\_length\_out=dec\_out[0].shape[0] |
| errs=dict() |
| foractioninactions: |
| #Makespacefortheerror |
| errs[action]=np.zeros((n\_sequences,seq\_length\_out)) |
| #Getthelistsforthisaction |
| enc\_in,dec\_in,dec\_out=actions\_dict[action] |
| foriinnp.arange(n\_sequences): |

|  |
| --- |
| n,d=dec\_out[i].shape |
| #Thelastframe |
| last\_frame=dec\_in[i][0,:] |
| last\_frame[0:6]=0 |
| ifk>1: |
| #Getthelastk-1frames |
| last\_k=enc\_in[i][(-k+1):,:] |
| assert(last\_k.shape[0]==(k-1)) |
| #Mergeandaveragethem |
| avg=np.mean(np.vstack((last\_k,last\_frame)),0) |
| else: |
| avg=last\_frame |
| dec\_out[i][:,0:6]=0 |
| idx\_to\_use=np.where(np.std(dec\_out[i],0)>1e-4)[0] |
| ee=np.power(dec\_out[i][:,idx\_to\_use]-avg[idx\_to\_use],2) |
| ee=np.sum(ee,1) |
| ee=np.sqrt(ee) |
| errs[action][i,:]=ee |
| errs[action]=np.mean(errs[action],0) |
| Returners |
| def denormalize\_and\_convert\_to\_euler( data, data\_mean, data\_std,  dim\_to\_ignore,actions,one\_hot): |
| """ |
| DenormalizesdataandconvertstoEulerangles |
| (alllossesarecomputedonEulerangles). |
| Args |
| data:dictionarywithhumanposes. |
| data\_mean:d-longvectorwiththemeanofthetrainingdata. |
| data\_std:d-longvectorwiththestandarddeviationofthetrainingdata. |
| dim\_to\_ignore:dimensionstoignorebecausethestdistoosmallorforother  reasons. |
| actions:listofstringswiththeactionsinthedatadictionary. |
| one\_hot:whetherthedatacomeswithone-hotencoding. |
| Returns |
| all\_denormed:alistwithnbatchentries.Eachentryisann-by-dmatrix |
| thatcorrespondstoadenormalizedsequenceinEulerangles |
| """ |
| all\_denormed=[] |

|  |
| --- |
| #expmap->rotmat->euler |
| foriinnp.arange(data.shape[0]): |
| denormed = data\_utils.unNormalizeData(data[i,:,:], data\_mean, data\_std,  dim\_to\_ignore,actions,one\_hot) |
| forjinnp.arange(denormed.shape[0]): |
| forkinnp.arange(3,97,3): |
| denormed[j,k:k+3] = data\_utils.rotmat2euler( data\_utils.expmap2rotmat(  denormed[j,k:k+3])) |
| all\_denormed.append(denormed) |
| returnall\_denormed |
| defmain(): |
| actions=["walking","eating","smoking","discussion"] |
| #TODOmakethisaruntimeoption |
| #Uncommenttheliens |
| # actions.extend(["directions", "greeting", "phoning", "posing",  "purchases", |
| # "sitting", "sittingdown", "takingphoto", "waiting", "walkingdog",  "walkingtogether"]) |
| #Parametersfordummymodel.Weonlybuildthemodeltoloadthedata. |
| one\_hot=False |
| FLAGS=Object() |
| FLAGS.data\_dir="./data/h3.6m/dataset" |
| FLAGS.architecture="tied" |
| FLAGS.seq\_length\_in=50 |
| FLAGS.seq\_length\_out=100 |
| FLAGS.num\_layers=1 |
| FLAGS.size=128 |
| FLAGS.max\_gradient\_norm=5 |
| FLAGS.batch\_size=8 |
| FLAGS.learning\_rate=0.005 |
| FLAGS.learning\_rate\_decay\_factor=1 |
| summaries\_dir="./log/" |
| FLAGS.loss\_to\_use="sampling\_based" |
| FLAGS.omit\_one\_hot=True, |
| FLAGS.residual\_velocities=False, |
| dtype=tf.float32 |
| #Baselinesareverysimple.NoneedtousetheGPU. |
| withtf.Session(config=tf.ConfigProto(device\_count={"GPU":0}))assess: |

|  |
| --- |
| model=seq2seq\_model.Seq2SeqModel( |
| FLAGS.architecture, |
| FLAGS.seq\_length\_in, |
| FLAGS.seq\_length\_out, |
| FLAGS.size,#hiddenlayersize |
| FLAGS.num\_layers, |
| FLAGS.max\_gradient\_norm, |
| FLAGS.batch\_size, |
| FLAGS.learning\_rate, |
| FLAGS.learning\_rate\_decay\_factor, |
| summaries\_dir, |
| FLAGS.loss\_to\_use, |
| len(actions), |
| notFLAGS.omit\_one\_hot, |
| FLAGS.residual\_velocities, |
| dtype=dtype) |
| #Loadthedata |
| \_, test\_set, data\_mean, data\_std, dim\_to\_ignore, dim\_to\_use =  translate.read\_all\_data( |
| actions, FLAGS.seq\_length\_in, FLAGS.seq\_length\_out, FLAGS.data\_dir, not  FLAGS.omit\_one\_hot) |
| #Getallthedata,denormalizeandconvertittoeulerangles |
| poses\_data={} |
| foractioninactions: |
| enc\_in,dec\_in,dec\_out=model.get\_batch\_srnn(test\_set,action) |
| enc\_in =denormalize\_and\_convert\_to\_euler( |
| enc\_in,data\_mean,data\_std,dim\_to\_ignore,actions,notFLAGS.omit\_one\_hot  ) |
| dec\_in =denormalize\_and\_convert\_to\_euler( |
| dec\_in,data\_mean,data\_std,dim\_to\_ignore,actions,notFLAGS.omit\_one\_hot  ) |
| dec\_out=denormalize\_and\_convert\_to\_euler( |
| dec\_out,data\_mean,data\_std,dim\_to\_ignore,actions,notFLAGS.omit\_one\_hot  ) |
| poses\_data[action]=(enc\_in,dec\_in,dec\_out) |
| #Computebaselineerrors |
| errs\_constant\_frame=running\_average(poses\_data,actions,1) |
| running\_average\_2 =running\_average(poses\_data,actions,2) |

|  |
| --- |
| running\_average\_4 =running\_average(poses\_data,actions,4) |
| print() |
| print("===Zero-velocity(runningavg.1)===") |
| print("{0:<16}|{1:4d}|{2:4d}|{3:4d}|{4:4d}".format("milliseconds",  80,160,380,400)) |
| foractioninactions: |
| print("{0:<16}|{1:.2f}|{2:.2f}|{3:.2f}|{4:.2f}".format(action, |
| errs\_constant\_frame[action][1],errs\_constant\_frame[action][3], |
| errs\_constant\_frame[action][7],errs\_constant\_frame[action][9])) |
| print() |
| print("===Runnningavg.2===") |
| print("{0:<16}|{1:4d}|{2:4d}|{3:4d}|{4:4d}".format("milliseconds",  80,160,380,400)) |
| foractioninactions: |
| print("{0:<16}|{1:.2f}|{2:.2f}|{3:.2f}|{4:.2f}".format(action, |
| running\_average\_2[action][1],running\_average\_2[action][3], |
| running\_average\_2[action][7],running\_average\_2[action][9])) |
| print() |
| print("===Runnningavg.4===") |
| print("{0:<16}|{1:4d}|{2:4d}|{3:4d}|{4:4d}".format("milliseconds",  80,160,380,400)) |
| foractioninactions: |
| print("{0:<16}|{1:.2f}|{2:.2f}|{3:.2f}|{4:.2f}".format(action, |
| running\_average\_4[action][1],running\_average\_4[action][3], |
| running\_average\_4[action][7],running\_average\_4[action][9])) |
| ifname =="main": |
| main()  kinematics.py importnumpyasnpimporth5py  importmatplotlib  importmatplotlib.pyplotasplt  importmatplotlib.animationasanimationfrommpl\_toolkits.mplot3dimportAxes3D importviz  importtime  importcopy |

|  |
| --- |
| importdata\_utils |
| deffkl(angles,parent,offset,rotInd,expmapInd): |
| """ |
| Convertjointanglesandbonelenghtsintothe3dpointsofaperson. |
| Basedonexpmap2xyz.m,availableat |
| https://github.com/asheshjain399/RNNexp/blob/7fc5a53292dc0f232867beb66c3 a9ef845d705cb/structural\_rnn/CRFProblems/H3.6m/mhmublv/Motion/exp2xyz.m |
| Args |
| angles:99-longvectorwith3dpositionand3djointanglesinexpmap  format |
| parent: 32-long vector with parent-child relationships in the  kinematictree |
| offset:96-longvectorwithbonelenghts |
| rotInd:32-longlistwithindicesintoangles |
| expmapInd:32-longlistwithindicesintoexpmapangles |
| Returns |
| xyz:32x33dpointsthatrepresentapersonin3dspace |
| """ |
| assertlen(angles)==99 |
| #Structurethatindicatesparentsforeachjoint |
| njoints =32 |
| xyzStruct=[dict()forxinrange(njoints)] |
| foriinnp.arange(njoints): |
| ifnotrotInd[i]:#Ifthelistisempty |
| xangle,yangle,zangle=0,0,0 |
| else: |
| xangle=angles[rotInd[i][0]-1] |
| yangle=angles[rotInd[i][1]-1] |
| zangle=angles[rotInd[i][2]-1] |
| r=angles[expmapInd[i]] |
| thisRotation=data\_utils.expmap2rotmat(r) |
| thisPosition=np.array([xangle,yangle,zangle]) |
| ifparent[i]==-1:#Rootnode |
| xyzStruct[i]['rotation']=thisRotation |
| xyzStruct[i]['xyz'] = np.reshape(offset[i,:], (1,3)) +  thisPosition |
| else: |

|  |
| --- |
| xyzStruct[i]['xyz']=(offset[i,:]+thisPosition).dot(xyzStruct[  parent[i]]['rotation'])+xyzStruct[parent[i]]['xyz'] |
| xyzStruct[i]['rotation']=thisRotation.dot(xyzStruct[parent[i]  ]['rotation']) |
| xyz=[xyzStruct[i]['xyz']foriinrange(njoints)] |
| xyz=np.array(xyz).squeeze() |
| xyz=xyz[:,[0,2,1]] |
| #xyz=xyz[:,[2,0,1]] |
| returnnp.reshape(xyz,[-1]) |
| defrevert\_coordinate\_space(channels,R0,T0): |
| """ |
| Bringaseriesofposestoacanonicalformsotheyarefacingthe  camerawhentheystart. |
| Adaptedfrom |
| https://github.com/asheshjain399/RNNexp/blob/7fc5a53292dc0f232867beb66c3 a9ef845d705cb/structural\_rnn/CRFProblems/H3.6m/dataParser/Utils/revertCo ordinateSpace.m |
| Args |
| channels:n-by-99matrixofposes |
| R0:3x3rotationforthefirstframe |
| T0:1x3positionforthefirstframe |
| Returns |
| channels\_rec:Thepassedposes,butthefirsthasT0andR0,and the |
| restofthesequenceismodifiedaccordingly. |
| """ |
| n,d=channels.shape |
| channels\_rec=copy.copy(channels) |
| R\_prev=R0 |
| T\_prev=T0 |
| rootRotInd=np.arange(3,6) |
| #Loopthroughthepassedposses |
| foriiinrange(n): |
| R\_diff=data\_utils.expmap2rotmat(channels[ii,rootRotInd]) |
| R=R\_diff.dot(R\_prev) |
| channels\_rec[ii,rootRotInd]=data\_utils.rotmat2expmap(R) |
| T = T\_prev + ((R\_prev.T).dot(  np.reshape(channels[ii,:3],[3,1]))).reshape(-1) |

|  |
| --- |
| channels\_rec[ii,:3]=T |
| T\_prev=T |
| R\_prev=R |
| returnchannels\_rec |
| def\_some\_variables(): |
| """ |
| Wedefinesomevariablesthatareusefultorunthekinematictree |
| Args |
| None |
| Returns |
| parent: 32-long vector with parent-child relationships in the  kinematictree |
| offset:96-longvectorwithbonelenghts |
| rotInd:32-longlistwithindicesintoangles |
| expmapInd:32-longlistwithindicesintoexpmapangles |
| """ |
| parent=np.array([0,1,2,3,4,5,1,7,8,9,10,1,12,13,14,15,13, |
| 17,18,19,20,21,20,23,13,25,26,27,28,29,28,31])-1 |
| offset = np.array([0.000000,0.000000,0.000000,- 132.948591,0.000000,0.000000,0.000000,-442.894612,0.000000,0.000000,-  454.206447,0.000000,0.000000,0.000000,162.767078,0.000000,0.000000,74.99  9437,132.948826,0.000000,0.000000,0.000000,-  442.894413,0.000000,0.000000,-  454.206590,0.000000,0.000000,0.000000,162.767426,0.000000,0.000000,74.99  9948,0.000000,0.100000,0.000000,0.000000,233.383263,0.000000,0.000000,25  7.077681,0.000000,0.000000,121.134938,0.000000,0.000000,115.002227,0.000  000,0.000000,257.077681,0.000000,0.000000,151.034226,0.000000,0.000000,2  78.882773,0.000000,0.000000,251.733451,0.000000,0.000000,0.000000,0.0000  00,0.000000,0.000000,99.999627,0.000000,100.000188,0.000000,0.000000,0.0  00000,0.000000,0.000000,257.077681,0.000000,0.000000,151.031437,0.000000  ,0.000000,278.892924,0.000000,0.000000,251.728680,0.000000,0.000000,0.00  0000,0.000000,0.000000,0.000000,99.999888,0.000000,137.499922,0.000000,0  .000000,0.000000,0.000000]) |
| offset=offset.reshape(-1,3) |
| rotInd=[[5,6,4], |
| [8,9,7], |
| [11,12,10], |
| [14,15,13], |

|  |
| --- |
| [17,18,16], |
| [], |
| [20,21,19], |
| [23,24,22], |
| [26,27,25], |
| [29,30,28], |
| [], |
| [32,33,31], |
| [35,36,34], |
| [38,39,37], |
| [41,42,40], |
| [], |
| [44,45,43], |
| [47,48,46], |
| [50,51,49], |
| [53,54,52], |
| [56,57,55], |
| [], |
| [59,60,58], |
| [], |
| [62,63,61], |
| [65,66,64], |
| [68,69,67], |
| [71,72,70], |
| [74,75,73], |
| [], |
| [77,78,76], |
| []] |
| expmapInd=np.split(np.arange(4,100)-1,32) |
| returnparent,offset,rotInd,expmapInd |
| defmain(): |
| #Loadallthedata |
| parent,offset,rotInd,expmapInd=\_some\_variables() |
| #numpyimplementation |
| withh5py.File('samples.h5','r')ash5f: |
| expmap\_gt=h5f['expmap/gt/walking\_0'][:] |
| expmap\_pred=h5f['expmap/preds/walking\_0'][:] |
| nframes\_gt,nframes\_pred=expmap\_gt.shape[0],expmap\_pred.shape[0] |

|  |
| --- |
| #Putthemtogetherandrevertthecoordinatespace |
| expmap\_all = revert\_coordinate\_space( np.vstack((expmap\_gt,  expmap\_pred)),np.eye(3),np.zeros(3)) |
| expmap\_gt =expmap\_all[:nframes\_gt,:] |
| expmap\_pred=expmap\_all[nframes\_gt:,:] |
| #Compute3dpointsforeachframe |
| xyz\_gt,xyz\_pred=np.zeros((nframes\_gt,96)),np.zeros((nframes\_pred,  96)) |
| foriinrange(nframes\_gt): |
| xyz\_gt[i,:]=fkl(expmap\_gt[i,:],parent,offset,rotInd,expmapInd  ) |
| foriinrange(nframes\_pred): |
| xyz\_pred[i,:] = fkl( expmap\_pred[i,:], parent, offset, rotInd,  expmapInd) |
| #===Plotandanimate=== |
| fig=plt.figure() |
| ax=plt.gca(projection='3d') |
| ob=viz.Ax3DPose(ax) |
| #Plottheconditioninggroundtruth |
| foriinrange(nframes\_gt): |
| ob.update(xyz\_gt[i,:]) |
| plt.show(block=False) |
| fig.canvas.draw() |
| plt.pause(0.01) |
| #Plottheprediction |
| foriinrange(nframes\_pred): |
| ob.update(xyz\_pred[i,:],lcolor="#9b59b6",rcolor="#2ecc71") |
| plt.show(block=False) |
| fig.canvas.draw() |
| plt.pause(0.01) |
| ifname =='main': |
| main() |