**CHAPTER 1: INTRODUCTION**

* 1. **PURPOSE**

The primary purpose of the project is to enhance the use of personal computers by adding functionalities like gesture-based control. These gestures can be used to perform various tasks a computer performs but in an efficient and easy manner. Many tasks in a computer require extensive key combinations to perform and it is easier to remember gestures rather than key combinations. Gestures including simple swipe of one’s hand or holding up a hand can trigger various actions in a computer using this project. Rather than searching for the keys or reaching out to the keyboard again and again for performing actions, we can simply perform the same tasks using gestures and thereby increasing the ease of use for a user. Usually, the computers that provide this functionality can cost up to 1 lakh rupees which is not feasible for everyone to purchase. Most laptops that are purchased lie within the range of Rs.30000- Rs.60000 and with the help of this project the same functionality can be provided to the users without having to pay a premium. So, to extend the functionality of mid-range laptops, we are using a combination of Arduino and a couple of Ultrasonic sensors to give the functionality of gesture-based control on any laptop.

* 1. **OBJECTIVE**

The main objective of this project is to control various operations and tasks in a computer using gestures. This project is going to enhance the ease of use of any personal computer by allowing us to use gestures in place of different key combinations to perform tasks in any application. The world is moving away from buttons and towards touch screen, gesture controls and even voice recognition to make it easy for users to operate the products. With evolving times, the user has started to prefer having everything on their fingertips rather than having to fumble around to do tasks and this project helps to achieve exactly that by allowing users to use hand-gestures and perform tasks which would have taken longer if they were to search for the key combinations. We are using this project to incorporate the functionality of high end laptops in mid-range laptops so that in can be available to as many users as possible. This project basically helps to control applications and the actions performed in it using simple hand movements and, in this project, we aim to cover the functionality of VLC media player and any web browser like fast forwarding or altering volume levels in the former and swiping web pages up-down or changing tabs in the latter. For example, moving our hand towards the sensors will increase the volume and reverse of this gesture will trigger lowering of the volume. Basic functionality can also be performed with the help of this project like Play/Pause and some advanced functions like switching between multiple applications. All these features are incorporated in the project which will help us to move in the correct direction of a touchless future at a bare minimum cost.

**1.3 MOTIVATION**

The primary motivation behind building the project is to provide high-end functionality in budget-oriented computers without spending a fortune. Nowadays high-end laptops come with in-built features like gesture-based control and touch screens. Even though touchscreen can’t be introduced to non-touch displays, we can add the functionality of gesture control in any laptop or computer using Arduino and a couple of ultrasonic sensors. Indian consumers are primarily focused on budget-oriented market and therefore are not willing to spend a lot of money for personal computers but still wish to have as much similar functions as a high-end computer has to offer. In order to meet these expectations of the consumers, we have come up with a simple replacement to the A.I. powered gestures, that is using an Arduino board and ultrasonic sensors to read hand gestures and give similar results on a computer when any hand movement is detected in front of the sensors. HP released a laptop featuring similar gesture-based controls but failed to keep the pricing appropriate as it costed around Rs.100000 and had poor battery life due to this added functionality. So, to improve on this technology, we are using this project as a medium which will allow users to experience the same features of controlling one’s computer with hand movements without having to pay a lot of money.

**1.4 DEFINITON AND OVERVIEW**

Leap Motion technology focuses on providing gesture-based controls on any laptop and helps in allowing users to operate their computers with ease. Without reaching out for the keyboard to perform tasks on a computer, we can simply perform the same tasks by just moving our hands in front of the sensors. With the help of Arduino and a couple of Ultrasonic sensors we are able to achieve gesture controls on a computer and still not requiring to pay a lot for having such functionality which is offered only by premium computers.

The concept behind the project is very simple. We will place two Ultrasonic (US) sensors on top of our monitor which will read the distance between the monitor and our hand using Arduino, based on this value of distance we will perform certain actions. To perform actions on our computer we use Pythonpyautogui library. Thecommands from Arduino are sent to the computer through serial port (USB). This data will be then read by python which is running on the computer and based on the read data an action will be performed. The important part of this project is to write a program for​ Arduino such that it converts the distances measured by both the sensors into the appropriate commands for controlling certain actions.

We use Python programming language to implement our next part of our code. We use python for establishing a serial communication with Arduino through the correct baud rate and then perform some basic keyboard actions. The first step with python would be to install the pyautogui module.

Now we will map buttons to various gestures that are detected by the Ultrasonic sensors. This mapping of buttons will help us control various actions to be performed in the computer. The various actions to be performed are:

* Switch to next tab in a browser window.
* Scroll Up/Down on a webpage.
* Quick switch between two applications.
* Play/Pause a video.
* Increase/Decrease the volume in a media player.

**CHAPTER 2: OVERALL DESCRIPTION**

**2.1 PROJECT PERSPECTIVE**

Human Machine Interface or HMI is a system comprising of hardware and software that helps in communication and exchange of information between the user (human operator) and the machine.

We normally use LED Indicators, Switches, Touch Screens and LCD Displays as a part of HMI devices. Another way to communicate with machines like Robots or Computers is with the help of Hand Gestures.

Instead of using a keyboard, mouse or joystick, we can use our hand gestures to control certain functions of a computer like play/pause a video, move left/right in a photo slide show, scroll up/down in a web page and many more.

In this project, we have implemented a simple Arduino based hand gesture control where you can control few functions of your web browser like switching between tabs, scrolling up and down in web pages, shift between tasks (applications), play or pause a video and increase or decrease the volume (in VLC Player) with the help of hand gestures.

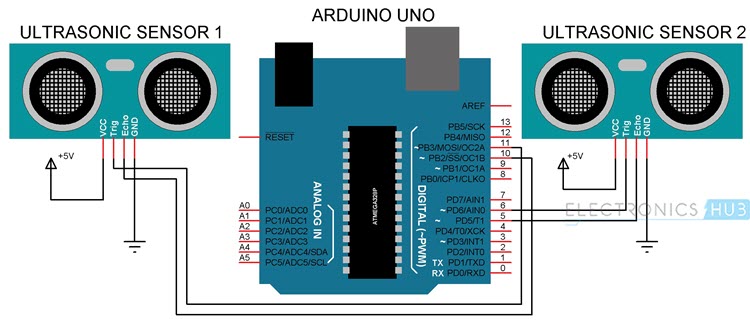
**2.1.1 PRINCIPLE BEHIND THE PROJECT**

The principle behind the Arduino based Hand Gesture Control of Computer is actually very simple. All you have to do is use two Ultrasonic Sensors with Arduino, place your hand in front of the Ultrasonic Sensor and calculate the distance between the hand and the sensor. Using this information, relevant actions in the computer can be performed.

The position of the Ultrasonic Sensors is very important. Place the two Ultrasonic Sensors on the top of a laptop screen at either end. The distance information from Arduino is collected by a Python Program and a special library called PyAutoGUI will convert the data into keyboard click actions.

**2.1.2 CIRCUIT DIAGRAM:**

The circuit diagram of Arduino part of the project is shown in the following image. It consists of an Arduino UNO board and two Ultrasonic Sensors and you can power up all these components from the laptop’s USB Port.



**Fig 1: Arduino Pin Diagram**

**2.2 PROJECT FUNCTIONS**

**2.2.1 WORKING:**

The important part of this project is to write a program for Arduino such that it converts the distances measured by both the sensors into the appropriate commands for controlling certain actions.

We have already seen a project called [**PORTABLE ULTRASONIC RANGE METER**](https://www.electronicshub.org/portable-ultrasonic-range-meter/), where you can measure the distance of an object placed in front of an Ultrasonic Sensor with the help of Arduino.

A similar concept is used here to measure the distance of your hand in front of both the Ultrasonic Sensors in this project. The fun part starts after calculating the distance.

The hand gestures in front of the Ultrasonic sensors can be calibrated so that they can perform five different tasks on your computer.

* Switch to Next Tab in a Web Browser
* Switch to Next Tab in a Web Browser
* Scroll Down in a Web Page
* Scroll Up in a Web Page
* Switch between two Tasks (Chrome and VLC Player)
* Play/Pause Video in VLC Player
* Increase Volume
* Decrease Volume

The following are the 5 different hand gestures or actions that I’ve programmed for demonstration purpose.

**Gesture 1:** Place your hand in front of the Right Ultrasonic Sensor at a distance (between 15CM to 35CM) for a small duration and move your hand away from the sensor. This gesture will Scroll Down the Web Page or Decrease the Volume.

**Gesture 2:** Place your hand in front of the Right Ultrasonic Sensor at a distance (between 15CM to 35CM) for a small duration and move your hand towards the sensor. This gesture will Scroll up the Web Page or Increase the Volume.

**Gesture 3:** Swipe your hand in front of the Right Ultrasonic Sensor. This gesture will move to the Next Tab.

**Gesture 4:** Swipe your hand in front of the Left Ultrasonic Sensor. This gesture will move to the Previous Tab or Play/Pause the Video.

**Gesture 5:** Swipe your hand across both the sensors (Left Sensor first). This action will switch between Tasks.

**2.2.2 PROGRAMMING CONCEPTS**

**ARDUINO CODE CONCEPT**

In the Arduino Code, the gesture mentioned above have been converted into 5 Commands that are sent to the Serial Port. Using these 5 commands, you can write a Python Program to control certain Keyboard Functions in order to achieve the required task.

**PYTHON PROGRAMMING**

Writing Python Program for Arduino based Hand Gesture Control is very simple. You just need to read the Serial data from Arduino and invoke certain keyboard key presses. In order to achieve this, you have to install a special Python Module called PyAutoGUI.

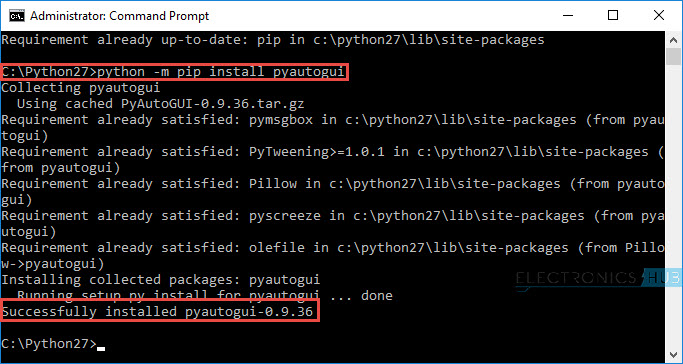
**INSTALLING PyAutoGUI**

The following steps will guide you through the installation of PyAutoGUI on Windows Computers. The module PyAutoGUI will help you to programmatically control the mouse and keyboard.

With the help of PyAutoGUI, we can write a Python Program to mimic the actions of mouse like left click, right click, scroll, etc. and keyboard like keypress, enter text, multiple key press, etc. without physically doing them. Let us install PyAutoGUI.

If you remember in the previous project, where we controlled an LED on Arduino using Python, we have installed Python in the directory “C:\Python27”.

Open Command Prompt with Administrator privileges and change to the directory where you have installed Python (in my case, it is C:\Python27).



**Fig 2: Snapshot of PyAuto GUI Installation**

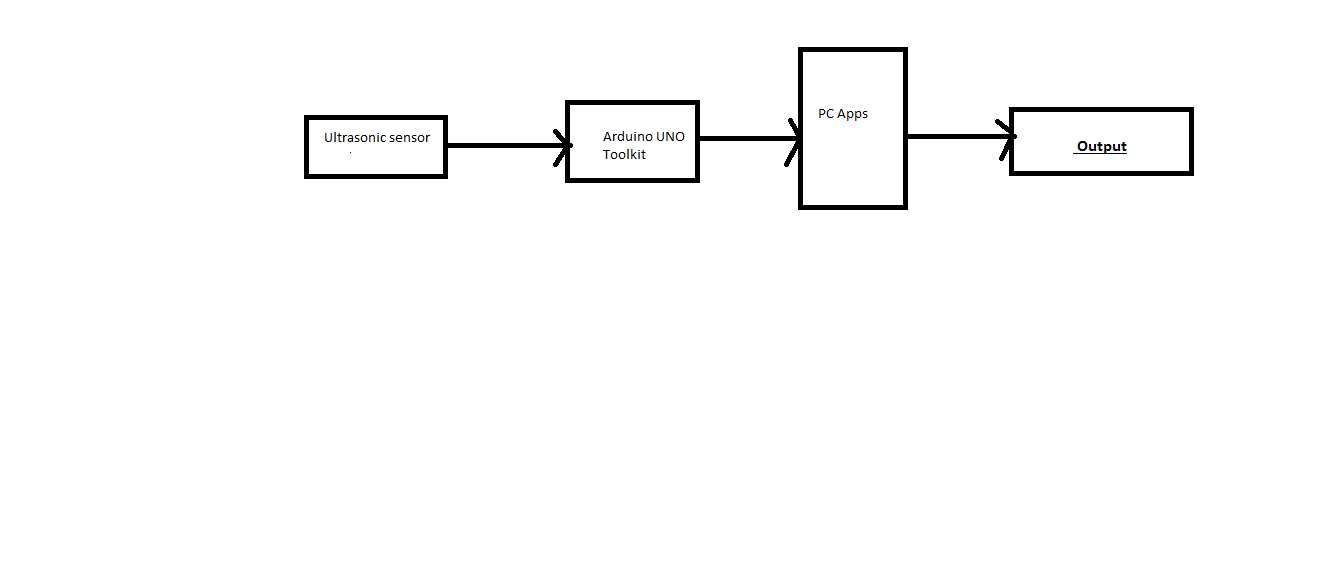
In the Arduino Code, the Arduino sends out five different texts or commands through Serial Port upon detecting appropriate hand gestures. These commands are

* Next
* Previous
* Down
* Up
* Change

Using these commands along with few functions in PyAutoGUI (like hotkey, scroll, key Down, press and key Up), you can write a simple Python Code that will execute the following tasks of keyboard and mouse.

* Data = “next” – – > Action = Ctrl+PgDn
* Data = “previous” – – > Action = Ctrl+PgUp
* Data = “down” – – > Action = Down Arrow
* Data = “up” – – > Action = Up Arrow
* Data = “change” – – > Action = Alt+Tab

**2.3 Block Diagram**

****

**Fig 3: Block Diagram**

The above is a block diagram for showing the steps of execution of the hand gesture control system using Arduino toolkit and a couple of ultrasonic sensors.

**2.4.1 PROJECT ASSUMPTIONS**

An assumption is a belief of what you assume to be true in the future. You make assumptions based on your knowledge, experience or the information available on hand. These are anticipated events or circumstances that are expected to occur during your project’s life cycle.

Assumptions are supposed to be true but do not necessarily end up being true. Sometimes they may turn out to be false, which can affect your project significantly. They add risks to the project because they may or may not be true.

The assumptions made by us in our project is that we cover a lot of actions by mapping buttons to actions within a limited distance which is equal to 60 cm. This in turn sometimes leads to overlapping of actions giving it a slight accuracy issue.

**2.4.2 CONSTRAINTS**

Constraints are limitations imposed on the project, such as the limitation of cost, schedule, or resources, and you have to work within the boundaries restricted by these constraints. All projects have constraints, which are defined and identified at the beginning of the project.

Constraints are mainly of two types:

* **Business constraints**
* **Technical constraints**

In our project currently, there are no business constraints since our project is not working on a global scale, so we have to deal with only technical constraints.

Our main technical constraints were:

* No previous knowledge of Arduino
* No past project experience with Arduino
* Ultrasonic sensors not working with full accuracy
* Project funding sources are limited, with no contingency.

**CHAPTER 3: SYSTEM REQUIREMENTS**

**3.1 EXTERNAL INTERFACE REQUIREMENTS:**

**Arduino** is an open source computer hardware and software company, project, and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices and interactive objects that can sense and control objects in the physical and digital world. The project's products are distributed as open-source hardware and software, which are licensed under the GNU Lesser General Public License (LGPL), permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially in preassembled form, or as do-it yourself (DIY) kits.

Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards or Breadboards (*shields*) and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs from personal computers. The microcontrollers are typically programmed using a dialect of features from the programming languages C and C++. In addition to using traditional compiler toolchains, the Arduino project provides an Integrated Development Environment (IDE) based on the Processing language project.

The Arduino project started in 2003 as a program for students at the Interaction Design Institute Ivrea in Ivrea, Italy, aiming to provide a low-cost and easy way for novices and professionals to create devices that interact with their environment using sensors and actuators. Common examples of such devices intended for beginner hobbyists include simple robots, thermostats, and motion detectors.

The name *Arduino* comes from a bar in Ivrea, Italy, where some of the founders of the project used to meet. The bar was named after Arduin of Ivrea who was the margrave of the March of Ivrea and King of Italy from 1002 to 1014.

The Arduino project started at the [Interaction Design Institute Ivrea](https://en.wikipedia.org/wiki/Interaction_Design_Institute_Ivrea) (IDII) in [Ivrea](https://en.wikipedia.org/wiki/Ivrea), Italy.[[2]](https://en.wikipedia.org/wiki/Arduino#cite_note-kushner-2) At that time, the students used a [BASIC Stamp](https://en.wikipedia.org/wiki/BASIC_Stamp) microcontroller at a cost of $100, a considerable expense for many students. In 2003 Hernando Barragán created the development platform [*Wiring*](https://en.wikipedia.org/wiki/Wiring_(development_platform)) as a Master's thesis project at IDII, under the supervision of [Massimo Banzi](https://en.wikipedia.org/wiki/Massimo_Banzi) and [Casey Reas](https://en.wikipedia.org/wiki/Casey_Reas), who are known for work on the [Processing](https://en.wikipedia.org/wiki/Processing_(programming_language)) language. The project goal was to create simple, low cost tools for creating digital projects by non-engineers.

**3.1.1 HARDWARE INTERFACE**

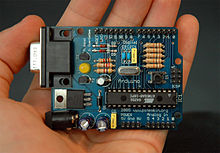
[](https://en.wikipedia.org/wiki/File:Arduino-compatible_R3_UNO_Sku066313-01.jpg)

**Fig 4: Arduino Toolkit**

Arduino-compatible R3 UNO board made in China with no Arduino logo, but with identical markings, including "*Made in Italy*" text

Arduino is [open-source hardware](https://en.wikipedia.org/wiki/Open-source_hardware). The hardware reference designs are distributed under a [Creative Commons](https://en.wikipedia.org/wiki/Creative_Commons) Attribution Share-Alike 2.5 license and are available on the Arduino website. Layout and production files for some versions of the hardware are also available.

Although the hardware and software designs are freely available under [copyleft](https://en.wikipedia.org/wiki/Copyleft) licenses, the developers have requested the name *Arduino* to be [exclusive to the official product](https://en.wikipedia.org/wiki/Generic_trademark) and not be used for derived works without permission. The official policy document on use of the Arduino name emphasizes that the project is open to incorporating work by others into the official product. Several Arduino-compatible products commercially released have avoided the project name by using various names ending in *-duino*.

[](https://en.wikipedia.org/wiki/File:Arduino316.jpg)

An early Arduino board with an [RS-232](https://en.wikipedia.org/wiki/RS-232) [serial](https://en.wikipedia.org/wiki/Serial_communication) interface (upper left) and an Atmel ATmega8 microcontroller chip (black, lower right); the 14 digital I/O pins are at the top, the 6 analog input pins at the lower right, and the power connector at the lower left.

Most Arduino boards consist of an [Atmel](https://en.wikipedia.org/wiki/Atmel) 8-bit AVR [microcontroller](https://en.wikipedia.org/wiki/Microcontroller) (ATmega8, ATmega168, [ATmega328](https://en.wikipedia.org/wiki/ATmega328), ATmega1280, ATmega2560) with varying amounts of flash memory, pins, and features. The 32-bit Arduino Due, based on the Atmel SAM3X8E was introduced in 2012.[[26]](https://en.wikipedia.org/wiki/Arduino#cite_note-26) The boards use single or double-row pins or female headers that facilitate connections for programming and incorporation into other circuits. These may connect with add-on modules termed *shields*. Multiple and possibly stacked shields may be individually addressable via an [I²C](https://en.wikipedia.org/wiki/I%C2%B2C) [serial bus](https://en.wikipedia.org/wiki/Serial_bus). Most boards include a 5 V [linear regulator](https://en.wikipedia.org/wiki/Linear_regulator) and a 16 MHz [crystal oscillator](https://en.wikipedia.org/wiki/Crystal_oscillator) or [ceramic resonator](https://en.wikipedia.org/wiki/Ceramic_resonator). Some designs, such as the LilyPad, run at 8 MHz and dispense with the onboard voltage regulator due to specific form-factor restrictions.

Arduino microcontrollers are pre-programmed with a [boot loader](https://en.wikipedia.org/wiki/Boot_loader) that simplifies uploading of programs to the on-chip [flash memory](https://en.wikipedia.org/wiki/Flash_memory). The default bootloader of the Arduino UNO is the optiboot bootloader. Boards are loaded with program code via a serial connection to another computer. Some serial Arduino boards contain a level shifter circuit to convert between [RS-232](https://en.wikipedia.org/wiki/RS-232) logic levels and [transistor–transistor logic](https://en.wikipedia.org/wiki/Transistor%E2%80%93transistor_logic) (TTL) level signals. Current Arduino boards are programmed via [Universal Serial Bus](https://en.wikipedia.org/wiki/Universal_Serial_Bus) (USB), implemented using USB-to-serial adapter chips such as the [FTDI](https://en.wikipedia.org/wiki/FTDI) FT232. Some boards, such as later-model Uno boards, substitute the [FTDI](https://en.wikipedia.org/wiki/FTDI) chip with a separate AVR chip containing USB-to-serial firmware, which is reprogrammable via its own [ICSP](https://en.wikipedia.org/wiki/In-system_programming) header. Other variants, such as the Arduino Mini and the unofficial Boarduino, use a detachable USB-to-serial adapter board or cable, [Bluetooth](https://en.wikipedia.org/wiki/Bluetooth) or other methods. When used with traditional microcontroller tools, instead of the Arduino IDE, standard AVR [in-system programming](https://en.wikipedia.org/wiki/In-system_programming) (ISP) programming is used.

[](https://en.wikipedia.org/wiki/File:UnoConnections.jpg)

**Fig 5: Arduino Uno R2 with I/O locations**

An official Arduino Uno R2 with descriptions of the I/O locations

**3.1.2 SOFTWARE INTERFACE:**

|  |  |
| --- | --- |
| Arduino Software IDE | |
| C:\Users\user\Desktop\ARDUINO-Windows-Store.png  **Fig 6: Arduino IDE** | |
| [Screenshot of Arduino IDE showing Blink program](https://en.wikipedia.org/wiki/File:Arduino_IDE_-_Blink.png)  **Fig 7: Arduino Programming** |  |
|  |  |
| A program for Arduino hardware may be written in any programming language with compilers that produce binary machine code for the target processor. Atmel provides a development environment for their 8-bit AVR and 32-bit ARM Cortex-M based microcontrollers: AVR Studio (older) and Atmel Studio (newer). IDE The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, MacOS, Linux) that is written in the programming language Java. It originated from the IDE for the languages Processing and Wiring. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple *one-click* mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus. The source code for the IDE is released under the GNU General Public License, version 2.  The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main ()* into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program *argued* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware. |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| **3.2 FUNCTIONAL REQUIREMENTS:** |  |

### Sketch

A program written with the Arduino IDE is called a sketch. Sketches are saved on the development computer as text files with the file extension .ino. Arduino Software (IDE) pre-1.0 saved sketches with the extension .pde.

A minimal Arduino C/C++ program consist of only two functions:

* setup(): This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch.
* loop(): After setup() has been called, function loop() is executed repeatedly in the main program. It controls the board until the board is powered off or is reset.

Blink example

[](https://en.wikipedia.org/wiki/File:Power_and_Pin13_LED_on_Arduino_Compatible_Board.jpg)

**Fig 8: Power LED (red) and User LED (green) attached to pin 13 on an Arduino compatible board**

Most Arduino boards contain a [light-emitting diode](https://en.wikipedia.org/wiki/Light-emitting_diode) (LED) and a load resistor connected between pin 13 and ground, which is a convenient feature for many tests and program functions. A typical program for a beginning Arduino programmer blinks a LED repeatedly. This program uses the functions pinMode(), digitalWrite(), and delay(), which are provided by the internal libraries included in the IDE environment. This program is usually loaded into a new Arduino board by the manufacturer.

#define LED\_PIN 13

void setup() {

pinMode(LED\_PIN, OUTPUT);

}

void loop() {

digitalWrite(LED\_PIN, HIGH);

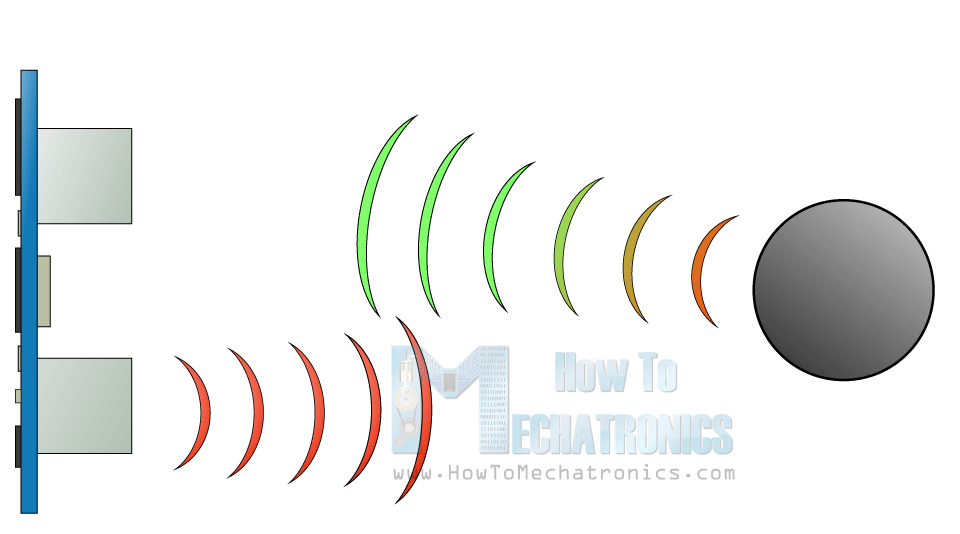
delay(1000);

digitalWrite(LED\_PIN, LOW);

delay(1000);

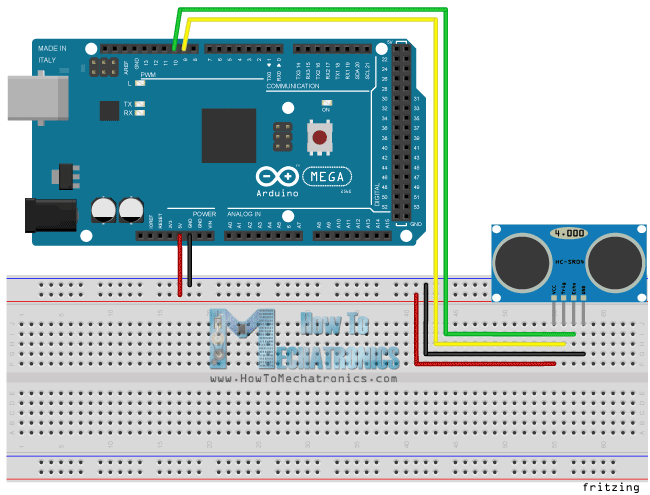
## How It Works – Ultrasonic Sensor

It emits an ultrasound at 40 000 Hz which travels through the air and if there is an object or obstacle on its path It will bounce back to the module. Considering the travel time and the speed of the sound you can calculate the distance.



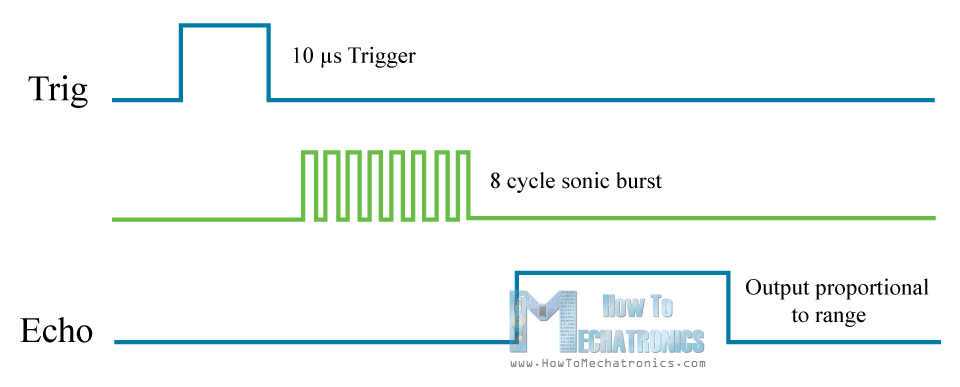
**Fig 9: Ultrasonic Sensors**

The HC-SR04 Ultrasonic Module has 4 pins, Ground, VCC, Trig and Echo. The Ground and the VCC pins of the module needs to be connected to the Ground and the 5 volts pins on the Arduino Board respectively and the trig and echo pins to any Digital I/O pin on the Arduino Board.

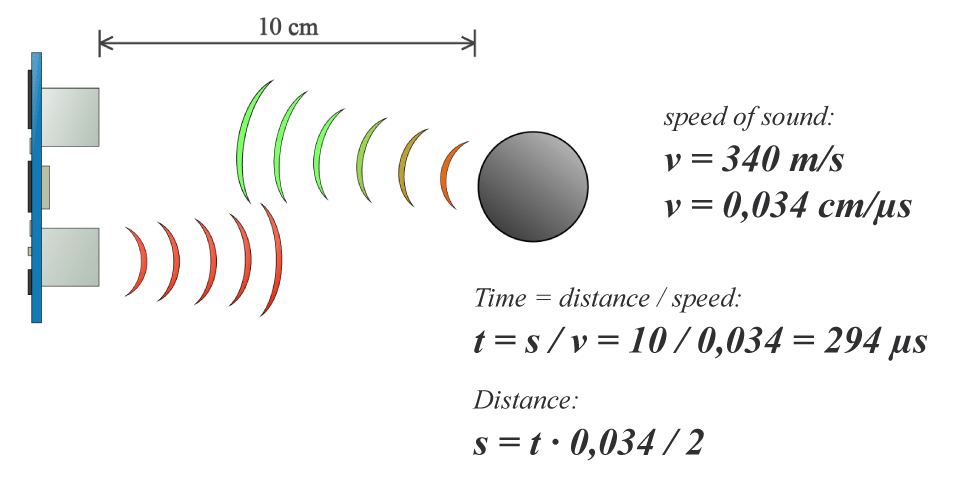


**Fig 10: Arduino with Breadboard**

In order to generate the ultrasound, you need to set the Trig on a High State for 10 µs. That will send out an 8-cycle sonic burst which will travel at the speed sound and it will be received in the Echo pin. The Echo pin will output the time in microseconds the sound wave travelled.

[](https://howtomechatronics.com/wp-content/uploads/2015/07/Ultrasonic-Sensor-Diagram.png?x57244)

For example, if the object is 10 cm away from the sensor, and the speed of the sound is 340 m/s or 0.034 cm/µs the sound wave will need to travel about 294 u seconds. But what you will get from the Echo pin will be double that number because the sound wave needs to travel forward and bounce backward.  So in order to get the distance in cm we need to multiply the received travel time value from the echo pin by 0.034 and divide it by 2.

[](https://howtomechatronics.com/wp-content/uploads/2015/07/Ultrasonic-Sensor-Equasions.png?x57244)

**Fig 11: Working of US Sensors**

**3.3 NON-FUNCTIONAL REQUIREMENTS**

**PERFORMANCE REQUIREMENTS**

Performance Requirements are required in order to ensure that the developed software/application meets the desired characteristics and performance targets. Generally, Performance Requirements are based upon following 4 factors:

1. Response Time

2. Workload

3. Scalability

4. Platform

**Response Time**

Response Time of our project is quite good, the only issue being accuracy leading to overlapping of actions.

**Workload**

Workload refers to handle multiple actions within given range which is quite well suited to our project.

**Scalability**

Scalability refers to the ability of the software to accommodate variable user requests during peak times.

**Platform**

Our project leap motion needs an Arduino IDE and python IDE to function along with Arduino toolkit and ultrasonic sensors.

**CHAPTER 4: CONCLUSION AND FUTURE WORK**

**4.1 CONCLUSION**

Leap Motion is the never used before technology that provide the users freedom of touching the system to operate it. It means, you can do everything with your system that have Multi-touch functionality without touching it. To **operate the system with the help of Leap Motion technology**, you need not any external software or driver rather you just need a device Leap Motion Controller (Small peripheral device placed on the system). The device senses the movement of the hands and fingers and operate according to that. You can move your hands and fingers in any natural gesture to operate with system. You can play/pause videos, scroll up/down in a web page and any other functions that you generally do with touches or clicks. With the gradual transition of technology from touch to touchless, various algorithms and the use of artificial intelligence has been required to correctly implement gesture-based controls. In this project, we have implemented Arduino based Hand Gesture Control of Your Computer, where few hand gestures made in front of the computer will perform certain tasks in the computer without using mouse or keyboard thereby simplifying gesture-based controls. This type of hand gesture control of computers can be used for VR (Virtual Reality), AR (Augmented Reality), 3D Design, Reading Sign Language, etc. This is a project which require minimum effort to accomplish certain tasks as today people want to accomplish a task with minimum effort.

**4.2 SCOPE OF FUTURE WORK**

In the future there is great scope of improvement in our project. Currently due to less accuracy of the sensor we are not able to map all the buttons to actions. Since the Arduino Uno toolkit we are using and the ultrasonic sensors are effective only up to a distance of 60cm so we are not able to map various actions. Also, the actions we have mapped and performed sometimes tend to overlap. So, in future these accuracy issues can be taken care of and we will try to perform other actions as well that we can perform using key presses.

Even some future work might lead towards automation of the mouse so that we can control the events that happen at the click of the mouse. In this way we will be able to control almost every action of our laptop/PC with simple hand gestures and that too cost effectively by just using and Arduino toolkit and a couple of ultrasonic sensors.

**REFERENCES**

1. www.arduino.cc/en/tutorials- Arduino basic programming techniques
2. www.c-sharpcorner.com/.../Interfacing-the-keyboard-with-arduino-in-python- How to map keys to actions
3. www.tutorialspoint.com/python- Python programming
4. Video lectures on YouTube about Arduino by Paul McWhorter