**PROJECT REQUIRMENTS**

**4.1 HARTWARE DESCRIPTION.**

## HARDWARE SPECIFICATION

4.1.1. ARDUINO UNO

4.1.2 INFRARED SENSOR

4.1.3 ROBOTIC ARM

4.1.4 SERVO MOTOR

4.1.5 DC MOTOR

4.1.6 L293D Motor Driver

**4.1.1 ARDUINO UNO**

**ARDUINO UNO**

The Arduino Uno is an open-source microcontroller board based on the Microchip ATmega328P microcontroller. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board has 14 digital I/O pins (six capable of PWM output), 6 analog I/O pins, and is programmable with the Arduino IDE (Integrated Development Environment), via a type B USB cable.

The Arduino Uno board can be powered via a USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack.

**SPECIFICATION**

Microcontroller: ATmega328P

Operating Voltage: 5V

Input Voltage (recommended): 7-12V

In out Voltage (limit): 6-20V

Digital I/O Pins: 14 (of which 6 provide PWM output)

PWM Digital I/O Pins: 6

Analog Input Pins: 6

DC Current per I/O Pin: 20 mA

DC current for 3.3V Pin: 50 mA

Flash Memory: 32 KB (ATmega328P) of which 0.5 KB used by bootloader

SRAM: 2 KB (ATmega328P)

EEPROM: 1 KB (ATmega328P)

Clock Speed: 16 MHz

LED\_BUILTIN: 13

**ARDUINO UNO USED IN AGRI-FIELD**

This small computer is used as the brain of the robot. It can be programmed to control the way buttons, motors, switches, lights, and other electronic parts work together. Arduino is hands-on, which is one of the many reasons it’s appealing to kids. The Arduino Edge Control is made for farming with precision. Improve plant health, reduce human error, automate tasks, adjust to weather, and share real-time insights on crop conditions are all benefits of optimizing water, fertilizer, and pesticide use. The planting process's essential values, such as temperature, humidity, soil moisture, and light intensity, are measured by sensor.



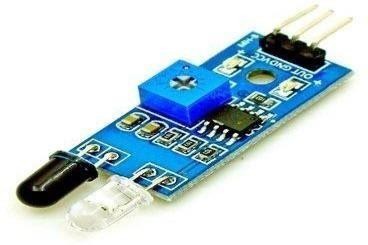
**Arduino UNO**

**4.1.2 IR SENSOR**

**IR SENSOR**

An infrared (IR) sensor is an electronic device that measures and detects infrared radiation in its surrounding environment. Infrared radiation was accidentally discovered by an astronomer named William Herschel in 1800. While measuring the temperature of each color of light (separated by a prism), he noticed that the temperature just beyond the red light was highest. IR is invisible to the human eye, as its wavelength is longer than that of visible light (though it is still on the same electromagnetic spectrum). Anything that emits heat (everything that has a temperature [above around five](https://www.livescience.com/50260-infrared-radiation.html) [degrees Kelvin](https://www.livescience.com/50260-infrared-radiation.html)) gives off infrared radiation.

We are using three IR detect sensor in our project for use of infrared thermometer technology in crop-based agriculture enables a more precise estimation of the plant's water status by measuring the temperature of the canopy over time. The sensor elements detect the heat radiation (infrared radiation) that changes over time and space as a result of people's movement within a predetermined angle range. IR sensors utilize Infrared light to decide if an item is available. These sensors are great for discovery between 100cm-500cm (1-5 meters/3-15 feet).

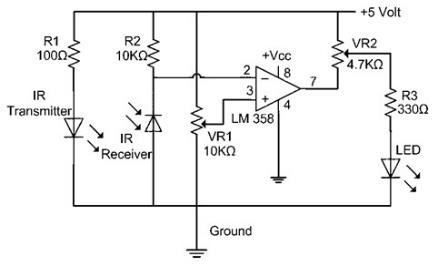


**IR SENSOR**

## IR SENSOR CIRCUIT DIAGRAM

An infrared sensor circuit is one of the basic and popular sensor modules in an electronic device. This sensor is analogous to human’s visionary senses, which can be used to detect obstacles and it is one of the common applications in real-time. This circuit comprises the following components

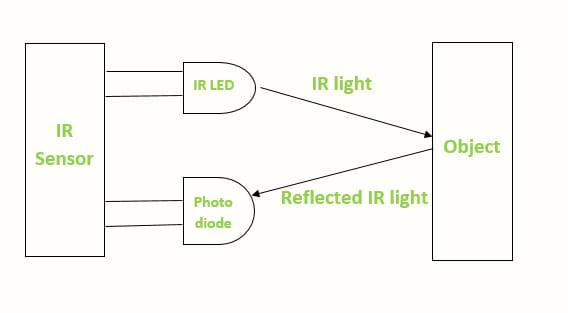
* LM358 IC 2 IR transmitter and receiver pair
* Resistors of the range of kilo-ohms.
* Variable resistors.
* LED (Light Emitting Diode).



There are different types of infrared transmitters depending on their wavelengths, output power and response time. An IR sensor consists of an IR LED and an IR Photodiode, together they are called as Photo Coupler or Opto Coupler.

IR Transmitter or IR

Infrared Transmitter is a light emitting diode (LED) which emits infrared radiations called as IR LED’s. Even though an IR LED looks like a normal LED, the radiation emitted by it is invisible to the human eye



**IR Sensor working diagram**

**4.1.3 ROBOTIC ARM**

These are some applications of robotics in agriculture for which Robotnik robots are used: Crop condition identification and corresponding chemical application, spraying or harvesting, as required by the fruit or plant. Mobile manipulation through collaborative arms (harvesting, fruit handling). Identifying the condition of the crop and applying the appropriate chemical, spraying, or harvesting, as required by the fruit or plan. In a Farm field to pick and remove the crops. Collection and conversion of useful information for the farmer. Selective application of pesticides.



**WEEDING ROBOTIC ARM**

In most cases, a lot of factors have to be considered before the commencement of a task. Robots can be used for other horticultural tasks such as pruning, weeding, spraying and monitoring. Robots can also be used in livestock applications (livestock robotics) such as automatic milking, washing and castrating. Robots like these have many benefits for the agricultural industry, including a higher quality of fresh produce, lower production costs, and a decreased need for manual labor. They can also be used to automate manual tasks, such as weed or bracken spraying, where the use of tractors and other human-operated vehicles is too dangerous for the operators.

**SPECIFICATIONS**

* Degrees of freedom: 6 + gripper
* Length at full extension: 984 mm
* Max. reach height on robot: 1800 mm
* Mass/weight, including gripper: 8 kg
* Max. endpoint speed: 10 m/s
* Lift capacity\*: Up to 11 kg
* Continuous lift capacity at 0.5 m extension\*: 5 kg
* Drag capacity\* (on carpet): Up to 25 kg
* Operating temp: -20 C to 45 C
* Ingress protection: Water and dust resista

**4.1.4 SERVO MOTOR**

**SERVO MOTOR**

A servomotor (or servo motor) is a rotary actuator or linear actuator that allows for precise control of angular or linear position, velocity, and acceleration.[1] It consists of a suitable motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servo motors. Servomotors are not a specific class of motor, although the term servomotor is often used to refer to a motor suitable for use in a closed-loop control system. Servomotors are used in applications such as robotics, CNC machinery, and automated manufacturing.



**SERVO MOTOR**

**MECHANISM**

A servomotor is a closed-loop servomechanism that uses position feedback to control its motion and final position. The input to its control is a signal (either analogy or digital) representing the position commanded for the output shaft. The motor is paired with some type of position encoder to provide position and speed feedback. In the simplest case, only the position is measured. The measured position of the output is compared to the command position, the external input to the controller. If the output position differs from that required, an error signal is generated which then causes the motor to rotate in either direction, as needed to bring the output shaft to the appropriate position. As the positions approach, the error signal reduces to zero, and the motor stops. The very simplest servomotors use position-only sensing via a potentiometer and bang-bang control of their motor; the motor always rotates at full speed (or is stopped). This type of servomotor is not widely used in industrial motion control, but it forms the basis of the simple and cheap servos used for radio-controlled models. More sophisticated servomotors make use of an Absolute Encoder (a type of rotary encoder) to calculate the shafts position and infer the speed of the output shaft. A variable-speed drive is used to control the motor speed. Both of these enhancements, usually in combination with a PID control algorithm, allow the servomotor to be brought to its commanded position more quickly and more precisely, with less overshooting.

**PURPOSE**

They are also used for novelty items such as remote-controlled and scale-sized toy cars, toy airplanes, toy helicopters, and toy robots. Servos are especially helpful for radio-controlled airplanes to position control surfaces. But servos are mostly used for industrial purposes. Important industries such as robotics, pharmaceutics, food services and in-line manufacturing also make use of servos. Servos are also most-suited for electrically operated pieces of machinery such as elevators, rudders, walking robots, and operating grippers.

**4.1.5 DC MOTOR**

**DC MOTOR**

A direct current (DC) motor is a type of electric machine that converts electrical energy into mechanical energy. DC motors take electrical power through direct current, and convert this energy into mechanical rotation. DC motors have high sustained torque. This makes them ideal for robotics applications that require a constant force. Furthermore, DC motors have a wide range of speed capabilities that make them ideal for robotics applications that require accurate speed control.

Some motors can be attached to wheels that drive a robot around. Other motors might cause joints in a robot limb to move. Yet others might move the control surfaces of a robotic airplane or submarine. A robot might have many different kinds of effectors to perform specific tasks, but many of these effectors are being moved around by motors.

What motors do is convert the electrical energy that powers the robot into mechanical energy that allows the robot to do work. There are twomeasurements of a motor that are important for understanding how much work it can do. Speed is what the maximum speed of the motor is. This is usually measured in revolutions per minute, or RPM



**DC MOTOR SUPPORING WHEELS**

DC motors are so called because they are powered by a current placed across the electrodes of the motor. The speed rating of a DC motor is the top speed it can run at. The actual speed the motor runs at is a function of how strong the current is that is applied to the motor. DC motors can be run both forwards and backwards depending on the direction of the applied current. Special circuits called "H-bridges" allow the use of switches and transistors to change the direction a motor moves without having to change any of the wires. For the most part, DC motors come with very high-speed ratings, and thus very low torque. Gears and transmissions allow us to convert the speed of a DC motor into torque that can be used for other tasks, such as moving heavy robots or lifting objects.

**4.1.6 L293D MOTOR DRIVER**

**L293D MOTOR DRIVR**

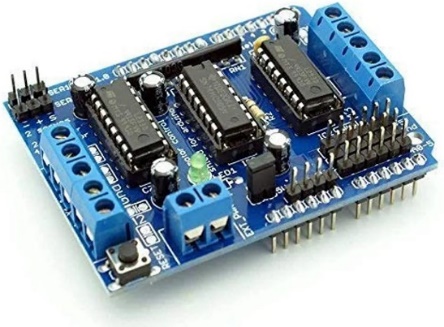
**Microcontroller**

**(Arduino UNO, Raspberry Pi, etc.**)

**Motor Driver Board**

* First the microcontroller sends signals to the motor driver. Then, the signals received by the motor driver are interpreted and stepped up with the reference voltage i.e., provider for driving the motor.
* The L293D is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. Both devices are designed to drive inductive loads such as relays, solenoids, DC and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications.
* Motor drivers acts as an interface between the motors and the control circuits. Motor requires high amount of current whereas the controller circuit works on low current signals. So, the function of motor drivers is to take a low-current control signal and then turn it into a higher-current signal that can drive a motor.
* The Motor Driver is a module for motors that allows you to control the working speed and direction of two motors simultaneously.

* This Motor Driver is designed and developed based on L293D IC. L293D is a 16 Pin Motor Driver IC. This is designed to provide bidirectional drive currents at voltages from 5 V to 36 V.
* The higher the voltage applied to the enable pin, the more power will be supplied to the motor, and the faster it will spin.

****

**L293D Motor Driver**

# **4.2 SOFTWARE DESCRIPTION**

**4.2.1. ARDUINO IDE**

**ARDUINO SOFTWARE (IDE)**

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuine hardware to upload programs and communicate with them.

**WRITING SKETCHES**

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom right-hand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

NB: Versions of the Arduino Software (IDE) prior to 1.0 saved sketches with the extension pde. It is possible to open these files with version 1.0, you will be prompted to save the sketch with the ion extension on save.

|  |  |
| --- | --- |
| https://www.arduino.cc/en/uploads/Guide/play.png | Verify  Checks your code for errors compiling it. |
| https://www.arduino.cc/en/uploads/Guide/export.png | Upload  Compiles your code and uploads it to the configured board. See [uploading](https://www.arduino.cc/en/Guide/Environment#uploading) below for details.  Note: If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer" |
| https://www.arduino.cc/en/uploads/Guide/new.png | New  Creates a new sketch. |
| https://www.arduino.cc/en/uploads/Guide/open.png | Open  Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.  Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the File | Sketch book menu instead. |
| https://www.arduino.cc/en/uploads/Guide/save.png | Save  Saves your sketch. |
| https://www.arduino.cc/en/uploads/Guide/serial_monitor.png | Serial Monitor  Opens the [serial monitor](https://www.arduino.cc/en/Guide/Environment#serialmonitor). |

Additional commands are found within the five menus: File, Edit, Sketch, Tools, Help. The menus are context sensitive, which means only those items relevant to the work currently being carried out are available.

**FILE**

* New   
  Creates a new instance of the editor, with the bare minimum structure of a sketch already in place.
* Open   
  Allows to load a sketch file browsing through the computer drives and folders.
* Open    
  Provides a short list of the most recent sketches, ready to be opened.
* Sketchbook   
  Shows the current sketches within the sketchbook folder structure; clicking on any name opens the corresponding sketch in a new editor instance.
* Examples   
  Any example provided by the Arduino Software (IDE) or library shows up in this menu item. All the examples are structured in a tree that allows easy access by topic or library.

Close   
Closes the instance of the Arduino Software from which it is clicked.

Save   
Saves the sketch with the current name. If the file hasn't been named before, a name will be provided in a "Save as." window.

* Save as...   
  Allows saving the current sketch with a different name.
* Page Setup   
  It shows the Page Setup window for printing.
* Print   
  Sends the current sketch to the printer according to the settings defined in Page Setup.
* Preferences   
  Opens the Preferences window where some settings of the IDE may be customized, as the language of the IDE interface.
* Quit   
  Closes all IDE windows. The same sketches open when Quit was chosen will be automatically reopened the next time you start the IDE.

**EDIT**

* Undo/Redo   
  Goes back of one or more steps you did while editing; when you go back, you may go forward with Redo.
* Cut   
  Removes the selected text from the editor and places it into the clipboard.
* Copy   
  Duplicates the selected text in the editor and places it into the clipboard.
* Copy for Forum   
  Copies the code of your sketch to the clipboard in a form suitable for posting to the forum, complete with syntax coloring.
* Copy as HTML   
  Copies the code of your sketch to the clipboard as HTML, suitable for embedding in web pages.
* Paste   
  Puts the contents of the clipboard at the cursor position, in the editor.
* Select All   
  Selects and highlights the whole content of the editor.
* Comment/Uncomment   
  Puts or removes the // comment marker at the beginning of each selected line.
* Increase/Decrease Indent   
  Adds or subtracts a space at the beginning of each selected line, moving the text one space on the right or eliminating a space at the beginning.
* Find   
  Opens the Find and Replace window where you can specify text to search inside the current sketch according to several options.
* Find Next   
  Highlights the next occurrence - if any - of the string specified as the search item in the Find window, relative to the cursor position.
* Find Previous   
  Highlights the previous occurrence - if any - of the string specified as the search item in the Find window relative to the cursor position.

**SKETCH**

* Verify/Compile   
  Checks your sketch for errors compiling it; it will report memory usage for code and variables in the console area.
* Upload   
  Compiles and loads the binary file onto the configured board through the configured Port.
* Upload Using Programmer   
  This will overwrite the bootloader on the board; you will need to use Tools > Burn Bootloader to restore it and be able to Upload to USB serial port again. However, it allows you to use the full capacity of the Flash memory for your sketch. Please note that this command will NOT burn the fuses. To do so a Tools -> Burn Bootloader command must be executed.
* Export Compiled Binary   
  Saves a .hex file that may be kept as archive or sent to the board using other tools.
* Show Sketch Folder   
  Opens the current sketch folder.
* Include Library   
  Adds a library to your sketch by inserting #include statements at the start of your code. For more details, see [libraries](https://www.arduino.cc/en/Guide/Environment#libraries) below. Additionally, from this menu item you can access the Library Manager and import new libraries from .zip files.
* Add File...   
  Adds a source file to the sketch (it will be copied from its current location). The new file appears in a new tab in the sketch window. Files can be removed from the sketch using the tab menu accessible clicking on the small triangle icon below the serial monitor one on the right-side of the toolbar.

**TOOLS**

* Auto Format   
  This formats your code nicely: i.e., indents it so that opening and closing curly braces line up, and that the statements inside curly braces are indented more.
* Archive Sketch   
  Archives a copy of the current sketch in .zip format. The archive is placed in the same directory as the sketch.
* Fix Encoding & Reload   
  Fixes possible discrepancies between the editor char map encoding and other operating systems char maps.
* Serial Monitor   
  Opens the serial monitor window and initiates the exchange of data with any connected board on the currently selected Port. This usually resets the board, if the board supports Reset over serial port opening.
* Board   
  Select the board that you're using. See below for [descriptions of the various boards](https://www.arduino.cc/en/Guide/Environment#boards).
* Port   
  This menu contains all the serial devices (real or virtual) on your machine. It should automatically refresh every time you open the top-level tools menu.
* Programmer   
  For selecting a hardware programmer when programming a board or chip and not using the onboard USB-serial connection. Normally you won't need this, but if you're [burning a bootloader](https://www.arduino.cc/en/Tutorial/Bootloader) to a new microcontroller, you will use this.
* Burn Bootloader     
  The items in this menu allow you to burn a [bootloader](https://www.arduino.cc/en/Hacking/Bootloader) onto the microcontroller on an Arduino board. This is not required for normal use of an Arduino or Genuino board but is useful if you purchase a new AT mega microcontroller (which normally come without a bootloader). Ensure that you've selected the correct board from the Boards menu before burning the bootloader on the target board. This command also set the right fuses.

**HELP**

Here you find easy access to a number of documents that come with the Arduino Software (IDE). You have access to Getting Started, Reference, this guide to the IDE and other documents locally, without an internet connection. The documents are a local copy of the online ones and may link back to our online website.

* Find in Reference   
  This is the only interactive function of the Help menu: it directly selects the relevant page in the local copy of the Reference for the function or command under the cursor.

**SKETCHBOOK**

The Arduino Software (IDE) uses the concept of a sketchbook: a standard place to store your programs (or sketches). The sketches in your sketchbook can be opened from the File > Sketchbook menu or from the Open button on the toolbar. The first time you run the Arduino software, it will automatically create a directory for your sketchbook. You can view or change the location of the sketchbook location from with the Preferences dialog.

Beginning with version 1.0, files are saved with a .ion file extension. Previous versions use the. pde extension. You may still open. pde named files in version 1.0 and later, the software will automatically rename the extension to. ion.

Tabs, Multiple Files, and Compilation

Allows you to manage sketches with more than one file (each of which appears in its own tab). These can be normal Arduino code files (no visible extension), C files (.c extension), C++ files (.cpp), or header files (.h).

Uploading

Before uploading your sketch, you need to select the correct items from the Tools > Board and Tools > Port menus. The [boards](https://www.arduino.cc/en/Guide/Environment#boards) are described below. On the Mac, the serial port is probably something like /dev/tty. usbmodem241 (for an Uno or Mega2560 or Leonardo) or /dev/tty.usbserial-1B1 (for a Duemilanove or earlier USB board), or/dev/tty. USA19QW1b1P1.1 (for a serial board connected with a Keyspan USB-to-Serial adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the sport section of the Windows Device Manager. On Linux, it should be /dev/ttyACMx,/dev/ttyUSBx or similar. Once you've selected the correct serial port and board, press the upload button in the toolbar or select the Upload item from the File menu. Current Arduino boards will reset automatically and begin the upload. With older boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is complete, or show an error.

When you upload a sketch, you're using the Arduino bootloader, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The bootloader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The bootloader will blink the on-board (pin 13) LED when it starts (i.e., when the board resets).

**Libraries**

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from the Sketch > Import Library menu. This will insert one or more #include statements at the top of the sketch and compile the library with your sketch. Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its #includestatements from the top of your code.

There is a [list of libraries](https://www.arduino.cc/en/Reference/Libraries) in the reference. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources or through the Library Manager. Starting with version 1.0.5 of the IDE, you do can import a library from a zip file and use it in an open sketch. See these instructions for installing a third-party library. To write your own library, see [this tutorial](https://www.arduino.cc/en/Hacking/LibraryTutorial).

Support for third-party hardware can be added to the hardware directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, bootloaders, and programmer definitions. To install, create the hardware directory, then unzip the third-party platform into its own sub-directory. (Don't use "Arduino" as the sub-directory name or you'll override the built-in Arduino platform.) To uninstall, simply delete its directory.

For details on creating packages for third-party hardware, see the [Arduino IDE 1.5 3rd party Hardware specification](https://github.com/arduino/Arduino/wiki/Arduino-IDE-1.5-3rd-party-Hardware-specification).

**Serial Monitor**

Displays serial data being sent from the Arduino or Genuino board (USB or serial board). To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down that matches the rate passed to Serial begin in your sketch. Note that on Windows, Mac or Linux, the Arduino or Genuino board will reset (rerun your sketch execution to the beginning) when you connect with the serial monitor. You can also talk to the board from Processing, Flash, Max MSP, etc. (see the [interfacing page](http://www.arduino.cc/playground/Main/Interfacing) for details).

**Preferences.**

Some preferences can be set in the preferences dialog (found under the Arduino menu on the Mac, or File on Windows and Linux). The rest can be found in the preferences file, whose location is shown in the preference dialog.