Automated Watrer Planter
Microcontroller Based System Design

- Fairuz Nawar, Ahmed Rad Sakib , Anonnya Ghosh

Objective

Since Bangladesh is agricultural, there is a constant demand to use technology for the betterment of agricultural production. Our goal is to make farming easier by enabling farmers to water the crops or plants automatically so plants get their optimum water intake and to get a suitable environment by ensuring the moisture in soil is also ideal for proper growth. This automated water planter makes the task of watering plants much more efficient and easy. With the help of our system, Moisture sensitive plants that require measured water intake can be cultivated and taken care of with greater ease. Our system also enables precise irrigation. Our system will eradicate **Moisture Stress** on plants which hamper's their growth and well-being.

Social Values

Watering plants is a very essential and somewhat troublesome part of farming, gardening and in taking care of plants. Our **Automated Water Planter** ensures plants get their proper moisture which gives in to having an optimal environment for their growth which ultimately leads to better production and economic growth. Moisture Stress is a serious issue while taking care of plants and it can be solved easily with our system. Before, Moisture sensitive plants were risky to cultivate as these plants have a lower rate of survival and production, if not taken care of properly but our **Automated Water Planter** reduces that risk and ensures a safe environment for the plants. Aside from farming, Gardeners and Plant Nurseries can also use this system for improvement and comfort.

Required Components

These following parts and tools are required for building this project

- **1x Arduino MEGA:** Among all other microcontrollers, Arduino Mega provides the flexibility of working with more memory space

and processing power that allows one to work with a number of sensors at once. This board has 54 pins and 16 analog pins with a good memory to store the code.
In our Automated Water Planter System, Arduino MEGA is the main microcontroller that controls and co-ordinates all other components used.

- **1x Soil Moisture Sensor:** Soil Moisture Sensors measure the volumetric water content indirectly by using some other property of the soil, such as electrical resistance, dielectric constant, or interaction with neutrons, as a proxy for the moisture content and outputs a voltage which is further analysed according to use.
We are using this sensor to determine the soil moisture each time before opening the gate of the water tank.

- **1x Ultrasonic Sensor HC-SR04:** HC-SR04 is an Ultrasonic Sensor mainly used to determine the distance of the target object. It measures accurate distance using a non-contact technology - A technology that involves no physical contact between sensor and object. In our project this sensor is used to calculate the the height of the water level which is further used to calculate water volume in the water tank to inform users when there is not enough water in the tank by sounding a Buzzer.

- **1x 16*2 LCD:** A 16x2 liquid-crystal display (LCD) is very basic module and is very commonly used in various devices and circuits which can display 16 characters per line and there are 2 such lines. It is a flat-panel display that uses the light-modulating properties of liquid crystals combined with polarizers.
We are using this LCD to display the required information to the users.

- **1x Phone Keypad:** A 3*4 Phone Keypad allows an individual to quickly input numeric values for further use.
In our system, the keypad is used to take input from users as required.

- **1x Basic Motor:** A Basic Motor is a device used to convert electricity into mechanical energy by using principles of electromagnetism. Motors produce a motion which is circular and can move objects in any direction
This motor is used to open and close the gate of the water tank in the Automated Water Planter System.

- **1x Piezo Buzzer:** A Piezo buzzer is a tiny speaker that can be connected directly to an Arduino. From the Arduino, sounds can be made with it by using tone according to desired frequency for amount of time the user wants.
  We are using this buzzer to notify the user when there is not enough water in the tank when it is time for watering.

- **1x Inductor:** Inductors are used as the energy storage device in many switched-mode power supplies to produce DC current. The inductor supplies energy to the circuit to keep current flowing during the "off" switching periods and enables topographies where the output voltage is higher than the input voltage.
  In our project the inductor is used to make and LC filter circuit which is used with the Soil Moisture Sensor to generate proper voltage signals.

- **1x Capacitor:** A capacitor stores energy in the electric field between its plates, depending on the voltage across it. The voltage across the capacitor will drive a current through the inductor.
  Capacitor and Inductor together form the LC filter required to be used with Soil Moisture Sensor for proper signal generation.

- **4x Variable Resistor:** Variable Resistors are used when working with electrical circuitry because they help to control voltage and/or currents. They specifically work with voltage and currents that are a part of the circuit.
  These Variable POT are used in Sensors' test pin for changing the values is Simulation to get different test results.

Design

The circuit diagram is given below.

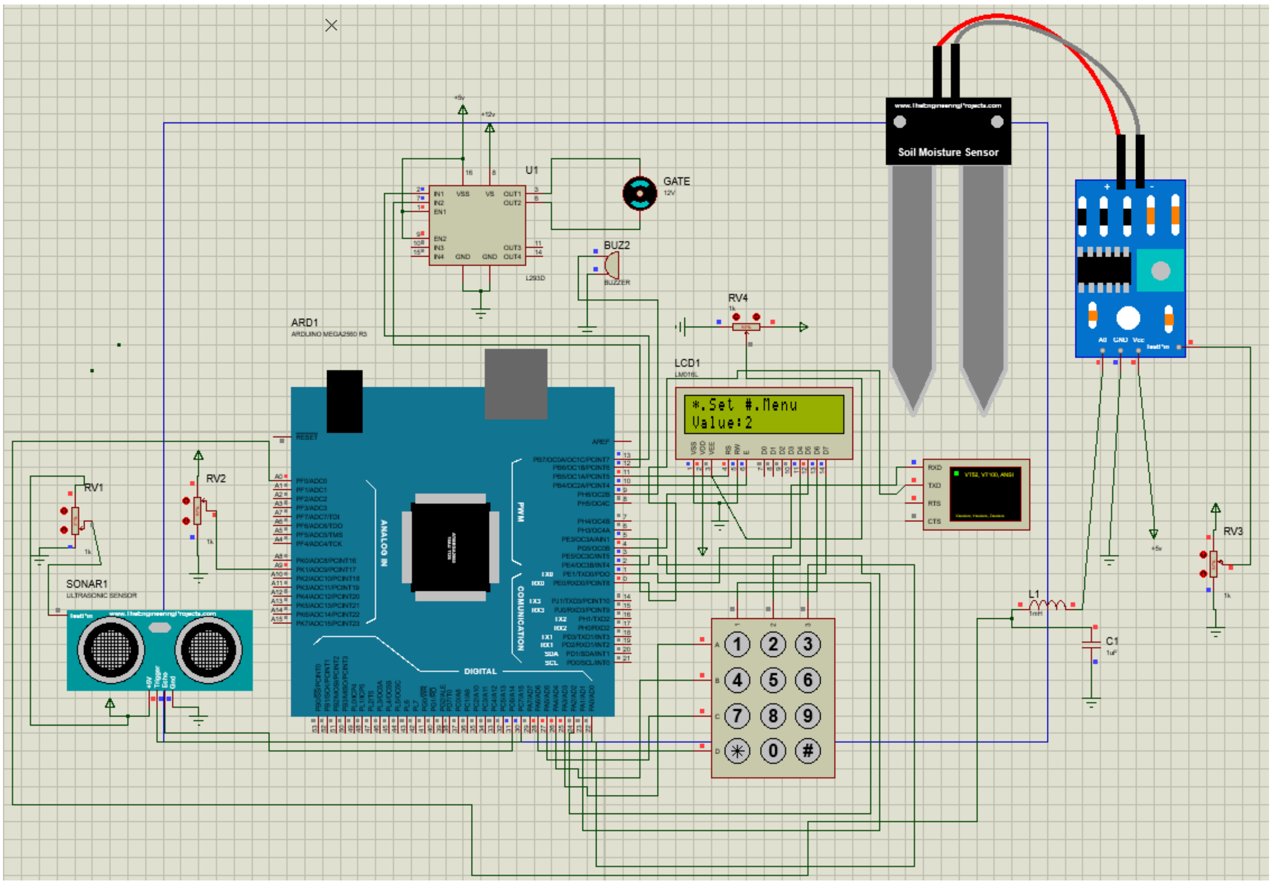


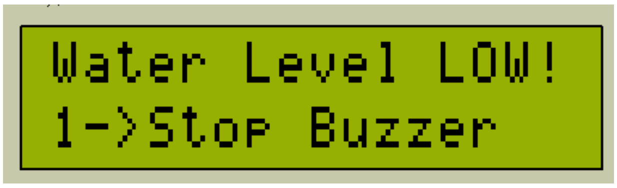
Figure 1: Simulation of Automated Water Planter

Working Procedure

Following are the step by step working procedure of the Automated Water Planter.

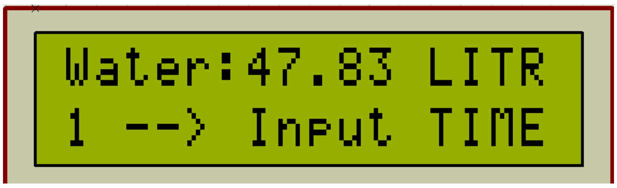
- **Measuring Water Level :** When the simulation is run, the water level in the water tank is measured using an Ultrasonic Sensor. Ultrasonic Sensor determines the height of the water which is further used to calculate the volume of water present in the tank.

- **Sounding Buzzer :** If Water is running low in the tank, a buzzer will go off to notify the user. A user can stop the buzzer by pressing "1" on the keypad.



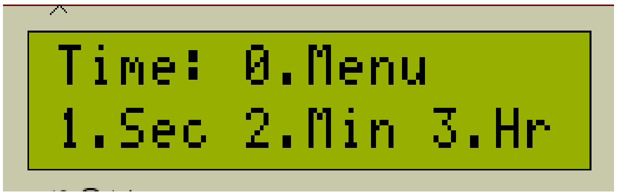
```
Water Level LOW!
1->Stop Buzzer
```

- **Taking Input From User :** If the Water level is okay then the water volume is shown. This is the Main Menu.



```
Water:47.83 LITR
1 --> Input TIME
```

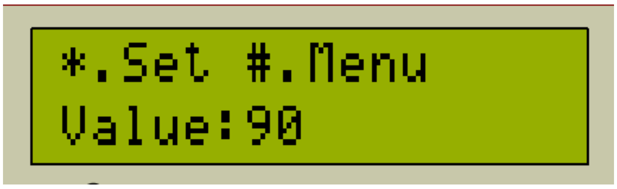
When user presses "1" the following screen comes in the LED.



```
Time: 0.Menu
1.Sec 2.Min 3.Hr
```

  - **Pressing 0 :** Will take the user back to the Main Menu.
  - **Pressing 1 :** Will allow the user to put in the time interval in seconds.
  - **Pressing 2 :** Will allow the user to put in the time interval in minutes.
  - **Pressing 3 :** Will allow the user to put in the time interval in hours.

This time interval's value will be used to calculate the next watering time for the plants. After pressing 1, 2 or 3 the following screen comes.

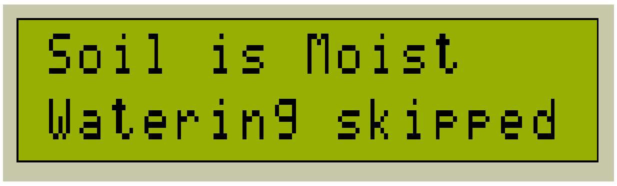


```
*.Set #.Menu
Value:90
```

  - **Pressing 1-9 in keypad :** User can give their time interval in numerical value using the phone keypad. The value will be printed on the screen. Here, Value = 90 which means after setting the time interval, the next watering time will be after 90 minutes of the current time.

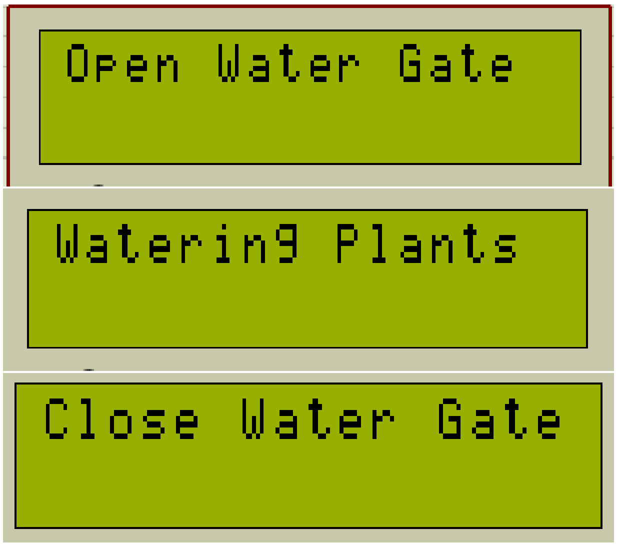
  - **Pressing '*' :** The value shown on the screen will be set as time interval.
  - **Pressing '#' :** The screen will return to the first screen which was showing Water Volume.

- **Watering the Plants :** When the time arrives for watering plants, the Soil Moisture Sensor tests the existing moisture of the soil. If the soil is already moist, then watering is skipped.



```
Soil is Moist
Watering skipped
```

If moisture in soil is low, then the motor used will go reverse for some time to open the gate of the water tank. After the gate is opened, the motor stops and lets the water to come out for some time after which the motor goes forward to close the gate.



```
Open Water Gate
```

```
Watering Plants
```

```
Close Water Gate
```

To summarize - Each time watering needs to be done, water level is checked, if it is low then buzzer will go off. Then soil moisture will be tested. If soil is moist, watering is skipped. If soil is not moist, water tank's gate opens, water goes out and the gate is closed and then system waits for the next watering time and the process is repeated.

Budget

Equipment	Quantity	Budget(Tk)
Arduino Mega	1	920
Water Bottle	1	100
Ultrasonic Sensor HC-SR04	1	120
Soil Moisture Sensor	1	100
LCD 16*2	1	200
Phone Keypad	1	100
Basic Motor	1	400
Piezo Buzzer	1	50
Inductor	1	5
Variable Resistor	4	15
Capacitor	1	5
Total		**2240**

Code

```
#include <LiquidCrystal.h>
#include <Keypad.h>
#include <Wire.h>

//soilSensorPin
int SensorPin = A0;

const int rs = 11, en = 10, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

const byte ROWS = 4; //four rows
const byte COLS = 3; //four columns
// buttons of the keypads
char hexaKeys[ROWS][COLS] = {
  {'1', '2', '3'},
  {'4', '5', '6'},
  {'7', '8', '9'},
  {'*', '0', '#'}
};
byte rowPins[ROWS] = {25, 26, 27, 28}; //connect to the row pinouts of the keypad
byte colPins[COLS] = {22, 23, 24}; //connect to the column pinouts of the keypad
//initialize an instance of class NewKeypad
Keypad cusKeypad = Keypad( makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);
int val = 0;
int angPin = A9;

int buzz = 9;
int LeftMotorForward1 = 13;
int LeftMotorReverse1 = 12;
int cursor_y = 0, cursor_x = 1;

const int echoPin = 31; // Echo Pin of Ultrasonic Sensor
const int pingPin = 30; // Trigger Pin of Ultrasonic Sensor

bool low_water = false;
bool buzzer = true;
long time_interval_in_seconds=0;
bool time_interval_set=false;
String value="";
long seconds=0;

int menu[6];//0.Main menu 1.interval menu 2.second menu 3.Minute menu 4.Hour menu 5.Water the plants
void setmenu(int j) {
  for(int i=0;i<6;i++) {
    if(i==j) menu[i]=1;
    else menu[i]=0;
  }
}
void setup() {
  setmenu(0);
  lcd.begin(16, 2);
  lcd.setCursor(0, 1);
  lcd.clear();
  pinMode(buzz, OUTPUT);
  Serial.begin(9600); // Starting Serial Communication
  pinMode(pingPin, OUTPUT); // initialising pin 30 as output
  pinMode(echoPin, INPUT); // initialising pin 31 as input

  pinMode(angPin, INPUT);
  pinMode(SensorPin, INPUT);

  pinMode(LeftMotorForward1, OUTPUT);
  pinMode(LeftMotorReverse1, OUTPUT);
}
void show_time_interval_menu() {
  char key = cusKeypad.getKey();
  lcd.setCursor(0, 0);
  lcd.print(String("Time:0.Menu"));
  lcd.setCursor(0, 1);
  lcd.print(String("1.S␣2.Min␣3.Hr"));
  if (key == ’0’) {
    lcd.clear();
    setmenu(0);
  }
  if(key==’1’) {
    lcd.clear();
    setmenu(2);
  }
  if(key==’2’) {
    lcd.clear();
    setmenu(3);
  }
  if(key==’3’) {
    lcd.clear();
    setmenu(4);
  }
}
void show_water_volume() {

  //ultrasonic

  long duration, inches, cm;

  digitalWrite(pingPin, LOW);
```

```
delayMicroseconds(2);

digitalWrite(pingPin, HIGH);
delayMicroseconds(10);

digitalWrite(pingPin, LOW);


duration = pulseIn(echoPin, HIGH); // using pulsin function to determine total time
cm = microsecondsToCentimeters(duration); // calling method

double litre_value = 3.1416 * 5 * 5 * cm; // calculating water volume by taking the height and considering the tank as a cylinder which has 5cm radious

litre_value = (double)(litre_value / 1000);
String message;
lcd.setCursor(0, 0);
char key = cusKeypad.getKey();
if ((int)litre_value < 20) { //if water goes down under 20 litre we show a water level low message
  low_water = true;
  if (buzzer) {
    message = String("Water␣Level␣LOW!");
    lcd.print(message);
    tone(buzz, 1000);
    lcd.setCursor(0, 1);
    lcd.print(String("1->stop␣buzzer␣␣␣"));
    if (key == '1') {
      noTone(buzz);
      buzzer = false;
    }
  }
  else {
    message = String("Water␣Level␣LOW!");
    lcd.print(message);
    lcd.setCursor(0, 1);
    lcd.print(String("1->input␣TIME␣␣␣"));
```

```

      if (key == '1') {
        lcd.clear();
        setmenu(1);
      }
    }
  }
  else {
    buzzer = true;
    low_water = false;
    message = String("Water:") + String(litre_value) + String("␣LITRE");
    lcd.print(message);
    lcd.setCursor(0, 1);
    lcd.print(String("1->input␣TIME␣␣␣"));
    noTone(buzz);
    if (key == '1') {
      lcd.clear();
      setmenu(1);
    }
  }
}
void show_seconds_menu() {
  lcd.setCursor(0,0);
  lcd.print(String("*.set␣#.Menu"));
  lcd.setCursor(0,1);
  lcd.print(String("Value:"));
  char key = cusKeypad.getKey();
  if(key=='#') {
    lcd.clear();
    value="";
    setmenu(0);
  }
  if(key!='#' and key !='*' and value.length()<7) {
    value+=String(key);
    lcd.print(value);
  }

  if(key=='*') {
    time_interval_set=true;
    char value_in_char[7];
    //Serial.println(String("value ")+String(value));
    value.toCharArray(value_in_char,value.length()+1);
    time_interval_in_seconds=atol(value_in_char);
    seconds=time_interval_in_seconds;
    //Serial.println(String(String("Got time ")+time_interval_in_seconds));
    time_interval_in_seconds+=(millis()/100);

    value="";
    lcd.clear();
    setmenu(0);
  }
}
void show_minutes_menu() {
  lcd.setCursor(0,0);
  lcd.print(String("*.set␣#.Menu"));
  lcd.setCursor(0,1);
  lcd.print(String("Value:"));
  char key = cusKeypad.getKey();
  if(key=='#') {
    lcd.clear();
    value="";
    setmenu(0);
  }
  if(key!='#' and key !='*' and value.length()<7) {
    value+=String(key);
    lcd.print(value);
  }

  if(key=='*') {
    time_interval_set=true;
    char value_in_char[7];
    //Serial.println(String("value ")+String(value));
    value.toCharArray(value_in_char,value.length()+1);
    time_interval_in_seconds=atol(value_in_char);
    time_interval_in_seconds*=60;
    seconds=time_interval_in_seconds;
    //Serial.println(String(String("Got time ")+time_interval_in_seconds));
    time_interval_in_seconds+=(millis()/100);

    value="";
    lcd.clear();
    setmenu(0);
  }
}
void show_hours_menu() {
    lcd.setCursor(0,0);
  lcd.print(String("*.set␣#.Menu"));
  lcd.setCursor(0,1);
  lcd.print(String("Value:"));
  char key = cusKeypad.getKey();
  if(key=='#') {
    lcd.clear();
    value="";
    setmenu(0);
  }
  if(key!='#' and key !='*' and value.length()<7) {
    value+=String(key);
    lcd.print(value);
  }
  if(key=='*') {
    time_interval_set=true;
    char value_in_char[7];
    //Serial.println(String("value ")+String(value));
    value.toCharArray(value_in_char,value.length()+1);
    time_interval_in_seconds=atol(value_in_char);
    time_interval_in_seconds*=60;
    time_interval_in_seconds*=60;
    seconds=time_interval_in_seconds;
    //Serial.println(String(String("Got time ")+time_interval_in_seconds));
    time_interval_in_seconds+=(millis()/100);
    value="";
    lcd.clear();
    setmenu(0);
  }
}
void  water_the_plant() {
  int SensorValue = analogRead(SensorPin);
  float SensorVolts = analogRead(SensorPin)
     *0.0048828125;
  Serial.println(String(SensorVolts));
  if((int) SensorVolts>=1) {
    lcd.setCursor(0,0);
    lcd.print(String("Open␣water␣gate"));
    digitalWrite(LeftMotorReverse1, HIGH);
    digitalWrite(LeftMotorForward1,LOW);
    delay(1000/5);
    lcd.setCursor(0,0);
    lcd.print(String("Watering␣plants"));
    digitalWrite(LeftMotorReverse1,LOW);
    digitalWrite(LeftMotorForward1,LOW);
    delay(2000/5);
    lcd.setCursor(0,0);
    lcd.print(String("Close␣water␣gate"));
    digitalWrite(LeftMotorReverse1, LOW);
    digitalWrite(LeftMotorForward1,HIGH);
    delay(1000/5);
    digitalWrite(LeftMotorReverse1,LOW);
    digitalWrite(LeftMotorForward1,LOW);

  }
  else {
    lcd.setCursor(0,0);
    lcd.print(String("Soil␣is␣moist"));
    lcd.setCursor(1,1);
    lcd.print(String("Water␣skipped"));
    delay(2000/5);
  }
  time_interval_in_seconds=(millis()/100)+seconds;
  lcd.clear();
  setmenu(0);
}
void loop() {

  if(menu[0]==1) show_water_volume();
  if(menu[1]==1) show_time_interval_menu();
  if(menu[2]==1) show_seconds_menu();
  if(menu[3]==1) show_minutes_menu();
  if(menu[4]==1) show_hours_menu();
   Serial.println(String((millis()/100))+String("␣")+String(time_interval_in_seconds));

  if(time_interval_set==true and (millis()/100)>=time_interval_in_seconds) {
    lcd.clear();
    setmenu(5);
  }
  if(menu[5]==1) water_the_plant();
  //SoilSensorCodes
  //int SensorValue = analogRead(SensorPin);
  //float SensorVolts = analogRead(SensorPin)*0.0048828125;

  /*lcd.setCursor(0, 1);
    char key = cusKeypad.getKey();

    val = analogRead(angPin);
    val = map(val, 0, 1023, 0, 255);


    int SensorValue = analogRead(SensorPin);




    if(cm > 250)
    {
    lcd.clear();
    lcd.print("Water level low!");
    delay(500);
    tone(buzz,1000);
    delay(1000);
    noTone(buzz);
    }else
    {
      noTone(buzz);
    }
    if (key == '1')
      {
       lcd.clear();
       lcd.print("mosture value:");
       lcd.print(SensorValue);
      }else if(key == '2')
      {
       lcd.clear();
       lcd.print("Distance: ");
       lcd.print(cm);
       lcd.print("cm");
      }  */
}

long microsecondsToCentimeters(long microseconds) // method to covert microsec to centimeters
{
  return microseconds / 29 / 2;
}
```

Difficulties

Following are some of the difficulties we faced while building our system.

- As Project simulations get larger and more complex simulator warns about simulation not running in real time due to excessive CPU load. For that, functions regarding time work significantly slower and does not reflect real time and the simulator keeps flickering,and sometimes button input does not work.

- The custom sensor libraries that were used in simulation sometimes do not give proper output.

- While adding the soil sensor, we need to convert Peak to Peak voltage value to RMS value and also use and LC circuit for proper generation of voltage.

Future Work

Our system currently takes care of the moisture and watering of plants. In future we can extend this system in a larger scale for green houses and farms and also add other features to the existing one. We plan to determine the contents of soil and water to determine which ingredient should be added to the soil. We can also add an automated water drainage system. For further precision, we can use temperature and humidity sensors for determining when plants need more water or how much water they need and notify the user if plants aren't getting their required water. All of which will further help the users by making things easier and for ensuring greater production.

Conclusion

We hope that our system will bring about some change in the way irrigation is done in agriculture and it will make taking care of all kinds of plants easier for everyone. Although it has limited features we believe it can be useful in a lot of aspects. Our Automated Tree Planter takes care of irrigation and watering for plants in an efficient manner that is cheap as well as hassle free.