DESARROLLO DE DISPOSITIVOS DE APOYO PARA EL PROYECTO CULTIVO ROBÓTICO REMOTO

AUTOR:
FRANCISCO JAVIER GARCIA GIL

DIRECTOR:
Dr. CESAR AUGUSTO PEÑA CORTES

UNIVERSIDAD DE PAMPLONA
FACULTAD DE INGENIERÍAS Y ARQUITECTURA
DEPARTAMENTO DE INGENIERIA ELÉCTRICA, ELECTRÓNICA, SISTEMAS Y TELECOMUNICACIONES
PROGRAMA DE INGENIERIA ELECTRÓNICA

UNIVERSIDAD DE PAMPLONA
PAMPLONA N. DE S. – COLOMBIA
15 DE DICIEMBRE DE 2015

DESARROLLO DE DISPOSITIVOS DE APOYO PARA EL PROYECTO CULTIVO ROBÓTICO REMOTO

AUTOR:

FRANCISCO JAVIER GARCIA GIL

TRABAJO DE GRADO PARA OPTAR POR EL TÍTULO DE INGENIERO ELECTRÓNICO

DIRECTOR:

Dr. CESAR AUGUSTO PEÑA CORTES

UNIVERSIDAD DE PAMPLONA

FACULTAD DE INGENIERÍAS Y ARQUITECTURA

DEPARTAMENTO DE INGENIERIA ELÉCTRICA, ELECTRÓNICA, SISTEMAS Y TELECOMUNICACIONES

PROGRAMA DE INGENIERIA ELECTRÓNICA

PAMPLONA N. DE S. – COLOMBIA

15 DE DICIEMBRE DE 2015

UNIVERSIDAD DE PAMPLONA

FACULTAD DE INGENIERÍAS Y ARQUITECTURA

DEPARTAMENTO DE INGENIERIA ELÉCTRICA, ELECTRÓNICA, SISTEMAS Y TELECOMUNICACIONES

PROGRAMA DE INGENIERIA ELECTRÓNICA

AUTORIZACIÓN PARA SUSTENTAR
TRABAJO DE GRADO PARA OPTAR POR EL TÍTULO DE INGENIERO ELECTRÓNICO

DESARROLLO DE DISPOSITIVOS DE APOYO PARA EL PROYECTO CULTIVO ROBÓTICO REMOTO

FECHA DE INICIO DEL TRABAJO: JUNIO DE 2015

FECHA DE TERMINACIÓN DEL TRABAJO: NOVIEMBRE DE 2015

NOMBRES Y FIRMAS DE AUTORIZACIÓN PARA SUSTENTAR

AUTOR: FRANCISCO JAVIER GARCIA GIL _____

DIRECTOR: Dr. CÉSAR AUGUSTO PEÑA _____

DIR. DE PROGRAMA: M.Sc. (c) JUDITH CRISTANCHO _____

JURADO CALIFICADOR:

JURADO 1: Dr. CRISTHIAN MANUEL DURÁN _____

JURADO 2: Dr. OSCAR EDUARDO GUALDRÓN _____

JURADO 3: Dr. CÉSAR AUGUSTO PEÑA _____

PAMPLONA, COLOMBIA

15 DE DICIEMBRE DE 201

DESARROLLO DE DISPOSITIVOS DE APOYO PARA EL PROYECTO CULTIVO ROBÓTICO REMOTO

DEDICATORIA:

A Dios y La Virgen del Carmen, que son mi guía.

A mis papás Martha Lucía y Javier Antonio, que son mi gran orgullo y el mejor ejemplo.

A mi hermana Paula Marcela, quien ha sido mi apoyo incondicional y mi gran amiga.

A mi regalito del cielo Juan Alejandro y al Cuñao.

A mis segundas mamás: Tía Mella, Tía Chechy, Tía Nubia, Tía Paulita y a mi Tío Armando.

A mis hermanos: Armando José, Pipe, Julio y a mi sobrino Juan Felipe.

A mis ángeles de la guarda, mis abuelos desde el cielo.

A toda mi familia.

Y a mis parceros, mis grandes amigos.

AGRADECIMIENTOS:

A Dios y La Virgen del Carmen.

A los promotores de este triunfo: mis padres y mi hermanita, quienes con su ejemplo y dedicación me impulsan a salir adelante con mucho amor, honestidad, confianza y entrega.

A mi familia, por estar siempre pendientes, por apoyarme en todos mis proyectos y por mantenerme en sus oraciones.

A mis parceros, mi parche, mis grandes amigos que aportaron mucho a mi desarrollo profesional y personal.

A mi gran amiga Zaida Liliana y a todo mi grupo Ritmos de mi Tierra, por hacer de ésta una estupenda experiencia.

A mis maestros por sus enseñanzas, en especial a todos aquellos que en los momentos difíciles me brindaron una voz de aliento para continuar.

Al Doctor Cesar Peña por su orientación y dedicación.

A mis compañeros por brindarme su apoyo.

A Pamplona por abrirme sus puertas y aportar mucho a mi vida estudiantil y personal.

"En tiempos de cambio, quienes estén abiertos al aprendizaje se adueñaran del futuro, mientras que aquellos que creen saberlo todo estarán bien equipados para un mundo que ya no existe".

ERIC HOFFER

"Los analfabetos del siglo XXI no serán aquellos que no sepan leer y escribir, sino aquellos que no puedan aprender, desaprender y reaprender".

ALVIN TOFFLER

"La primera tarea de la educación es agitar la vida, pera dejándola libre para que se desarrolle".

MARIA MONTESSORI

RESUMEN

Este proyecto pretende desarrollar una serie de dispositivos de apoyo para un robot agrícola que se va a encargar de sembrar y regar un cultivo; de igual manera se obtienen los valores correspondientes a diferentes variables físicas como lo son la temperatura, la humedad, la luminosidad y el nivel de dos tanques.

Para poder regar las diferentes plantas presentes en el cultivo, se desarrolla un sistema de desplazamiento lineal el cual se detiene en la posición deseada.

Teniendo en cuenta que el encargado de recibir los datos de las variables físicas es Arduino, se utiliza el protocolo UDP con su respectiva comunicación para enviar los datos obtenidos a Matlab y hacer su visualización y el accionamiento del sistema de riego en la interfaz gráfica.

Posteriormente se realizan las pruebas de los dispositivos desarrollados.

Palabras claves: Arduino, UDP, Matlab, Sensores, Dispositivo transportador lineal.

ABSTRACT

This project aims to develop a series of devices support an agricultural robot that will take care of planting and watering a crop; likewise the values corresponding to different physical variables such as temperature, humidity, light and the level of two tanks are obtained.

Water to different plants present in the culture, a system of linear movement which stops at the desired position takes place.

Given that the charge of receiving the data of physical variables is Arduino, the UDP protocol is used with its respective communication to send the data to Matlab and make your viewing and operation of the irrigation system in the GUI.

Subsequently developed tests are performed devices.

Keywords: Arduino, UDP, Matlab, sensors, linear conveyor device.

TABLA DE CONTENIDO

INTRODUCCIÓN 15
CAPÍTULO 1 16
1. PLANTEAMIENTO DEL PROBLEMA 16
1.1 JUSTIFICACIÓN 16
1.2 OBJETIVOS: 17
1.2.1 OBJETIVO GENERAL 17
1.2.2 OBJETIVOS ESPECÍFICOS 17
1.3 ACOTACIONES 18

CAPÍTULO 2 19
2. MARCO TEÓRICO Y ESTADO DEL ARTE 19
2.1 MARCO TEÓRICO: 19
2.1.1 PROTOCOLO UDP: 19
2.1.1.2 PUERTOS UDP 20
2.1.2 ARDUINO MEGA 2560 22
2.1.3 ARDUINO ETHERNET SHIELD 24
2.1.4 AX12W DYNAMIXEL: 26
2.1.4.1 Protocolo de comunicaciones [7]: 28
2.1.5 USB2Dynamixel [8]: 30
2.1.6 BOMBAS LAVAPARABRISAS [9] 33
2.1.7 SENSOR DHT11 [10]: 33
2.1.8 SENSOR DE LUMINOSIDAD: FOTORESISTOR [11]: 36
2.2 ESTADO DEL ARTE: 38
CAPÍTULO 3 42
3 DESARROLLO DE DISPOSITIVOS DE APOYO 42
3.1 DESARROLLO DEL SISTEMA DE DESPLAZAMIENTO LINEAL: 44
3.2 DESARROLLO DEL SISTEMA DE CAPTURAS DE SEÑALES: 50
3.3 CONTROL DE LOS ACTUADORES: 52
3.4 DESARROLLO DEL CÓDIGO EN ARDUINO IDE: 53
3.5 DESARROLLO DE LA INTERFÁZ GRÁFICA 58
CAPÍTULO 4 74
2 RESULTADOS: 74
a. PRUEBAS INICIALES: 74
b. PRUEBAS CON LA INTERFÁZ GRÁFICA: 83
5. CONCLUSIONES: 93
BIBLIOGRAFÍA 94

LISTA DE FIGURAS

Figura 1. ENCABEZADO DEL SEGMENTO [2] 20
Figura 2. ARDUINO MEGA 2560 [4] 22
Figura 3. INTERFAZ ARDUINO IDE [5] 23
Figura 4. ARDUINO ETHERNET SHIELD[7] 24
Figura 5. CABLE ETHERNET[7] 25
Figura 6. AX-12W Dynamixel [7] 28
Figura 7. ASIGNACIÓN DE PINES [7] 28
Figura 8. ENVÍO Y RECEPCION DE PAQUETE [7] 29
FIGURA 9. TABLA DE CONTROL 30
FIGURA 10. USOS DEL USB2DYNAMIXEL 31
FIGURA 11. ESTRUCTURA USB2DYNAMIXEL [8] 32
FIGURA 12. BOMBA LAVAPARABRISAS[9] 33
Figura 13. DHT11 34
FIGURA 14. CARACTERÍSTICAS DHT11 [10] 35
FIGURA 15. LDR 36
FIGURA 16. SÍMBOLO ELECTRÓNICO LDR 36
FIGURA 17. ROBOT TODOTERRENO PARA CULTIVOS[13] 40
FIGURA 18. PROYECTO DE AGRICULTURA URBANA[14] 41
FIGURA 19. DIAGRAMA GENERAL DE SENSORES Y ACTUADORES 43
FIGURA 20. DIAGRAMA DE COMUNICACIONES 44
FIGURA 21. BASE ROBOT 45
FIGURA 22. ESTRUCTURA ROBOT 46
FIGURA 23. RODAMIENTO 46
FIGURA 24. POLEAS 47
FIGURA 25. PRIMER ACOPLAMIENTO 47
FIGURA 26. SOPORTE VARILLA 48
FIGURA 27. SOPORTE Y VARILLA 48
FIGURA 28. PRENSA CORREAS 49
FIGURA 29. DISPOSITIVO DE DESPLAZAMIENTO LINEAL 49
FIGURA 30. DIAGRAMA DE COMUNICACIONES 51
FIGURA 31. DIAGRAMA DE CONEXIONES 52
FIGURA 32. DIAGRAMA DE FLUJO PROGRAMA SERVIDOR 53
FIGURA 33. INTERFÁZ DE PRESENTACION 58
FIGURA 34. INTERFÁZ DE CONTROL MANUAL 59
FIGURA 35. DIAGRAMA DE FLUJO INTERFAZ GRAFICA 61
FIGURA 36. PRUEBA SENSORES DE LUMINOSIDAD Y DHT11 74

FIGURA 37. PRUEBA SENSORES DE LUMINOSIDAD Y DHT11 CON PERTURBACIONES 75
FIGURA 38. RESULTADO PRUEBA SENSORES DE LUMINOSIDAD Y DHT11 CON PERTURBACIONES..... 75
FIGURA 39. SENSORES DE NIVEL SIN PERTURBACION 76
FIGURA 40. PRUEBA SENSORES DE NIVEL..... 76
FIGURA 41. SENSORES DE NIVEL CON PERTURBACION 77
FIGURA 42. FINAL DE CARRERA ÓPTICO PERTURBACION..... 77
FIGURA 43. PRUEBA PERTURBACION FINAL DE CARRERA ÓPTICO 78
FIGURA 44. PRUEBA PERTURBACION FINAL DE CARRERA ÓPTICO 78
FIGURA 45. SENSOR DE HUMEDAD EN TIERRA SIN PERTURBACIÓN..... 79
FIGURA 46. SENSOR DE HUMEDAD EN TIERRA CON PERTURBACIÓN 79
FIGURA 47. PRUEBA SENSOR DE HUMEDAD EN TIERRA..... 80
FIGURA 48. ACTIVACIÓN DE VÁLVULA 81
FIGURA 49. DESPLAZAMIENTO LINEAL 81
FIGURA 50. DESPLAZAMIENTO LINEAL CON EL ROBOT 82
FIGURA 51. DESPLAZAMIENTO LINEAL CON EL ROBOT YA DESPLAZADO..... 82
FIGURA 52. RED A LA QUE ME DEBO CONECTAR..... 83
FIGURA 53. LUMINOSIDAD Vs TIEMPO..... 84
FIGURA 54. HUMEDAD Y TEMPERATURA Vs TIEMPO 85
FIGURA 55. HUMEDAD Y TEMPERATURA Vs TIEMPO 85
FIGURA 56. LUMINOSIDAD Vs TIEMPO..... 86
FIGURA 57. SENSORES DE NIVEL DEL TANQUE 86
FIGURA 58. ACTIVACIÓN DE LA CÁMARA. 86
FIGURA 59. DIAGRAMA ELECTRÓNICO. 87
FIGURA 60. PCB TARJETA DE DESARROLLO..... 88
FIGURA 61. TARJETA DE DESARROLLO REAL 89
FIGURA 62. PRUEBA DE MOVIMIENTO SOBRE EL RIEL..... 90
FIGURA 63. PRUEBA DE MOVIMIENTO SOBRE EL RIEL CON EL ROBOT 90
FIGURA 64. ROBOT AGRICOLA..... 91
FIGURA 65. RIEGO ROBOT AGRÍCOLA..... 92

LISTA DE TABLAS

Tabla 1. LISTA PARCIAL DE PUERTOS [3] 22
Tabla 2. CARACTERÍSTICAS DE ARDUINO ETHERNET SHIELD[6] 26
Tabla 3. FUNCIONES Y DESCRIPCIÓN 60

INTRODUCCIÓN

Este proyecto se enfoca hacia el control de procesos y busca activamente encontrar la manera de aplicar cierta variedad de conceptos y desarrollarlos en el campo de la agricultura.

"La agricultura ha sufrido varias transformaciones desde que los humanos nos asentamos, dejando atrás la caza y recolecta. Pero está a punto de sufrir la mayor de todas**:** los humanos dejaremos las granjas a los robots y el software."[1]

En el campo de la robótica, se evidencia un constante interés en el desarrollo de dispositivos enfocados hacia el área de la agricultura, es por esto que en la Universidad de Pamplona se está desarrollando el proyecto Cultivo Robótico Remoto el cual requería la realización de algunos dispositivos de apoyo dando como resultado la intención de realizar un sistema transportador lineal que sirva de apoyo para este proyecto.

Para los cultivos, es inminentemente necesario calcular los valores correspondientes a las variables físicas que intervienen en ellos; de esta manera, se determinó extraer los valores de variables como luminosidad, temperatura y humedad a través de una tarjeta Arduino. Esta misma tarjeta me permite enviar a través del protocolo UDP enviar y recibir los datos solicitados, además de activar el sistema de riego según se requiera.

[1] ALEX BARRIENTOS. Automatización en la granja: bienvenido a la década de la agricultura robot [en línea]. < http://hipertextual.com/2015/11/agricultura-robot > [citado el 30 de Noviembre de 2015]

CAPÍTULO 1

1. PLANTEAMIENTO DEL PROBLEMA

En el campo de la agricultura, hay distintos procesos que ya sea por una concepción cultural o por la falta de recursos pueden volverse tediosos. Es constante la necesidad de implementar diferentes procesos electrónicos que permitan de forma automática, identificar necesidades y de esta manera buscar la solución a estas falencias. Para el proyecto Cultivo Robótico Remoto ha sido un problema representativo la falta de algunos dispositivos de apoyo que permitan:

- Adquirir los valores de las variables físicas como humedad, temperatura y luminosidad.
- Obtener un dispositivo que permita a un robot moverse linealmente en una longitud determinada.
- Controlar un dispositivo de riego de sustancias liquidas.

1.1 JUSTIFICACIÓN

Por lo expresado en el ítem anterior, busco con mi trabajo desarrollar los diferentes dispositivos de apoyo tales como:

- Diferentes sensores que me permitan obtener los valores correspondientes a variables físicas como temperatura, humedad y luminosidad.
- Un transportador lineal que me permita ubicar un robot antropomórfico de cinco grados de libertad en diferentes posiciones, teniendo en cuenta los requerimientos del

sistema.

- Un dispositivo que controle el riego de diferentes sustancias líquidas.

Estos dispositivos se realizan con el fin de dar solución a la problemática expresada anteriormente y de apoyar al desarrollo del proyecto Cultivo Robótico Remoto.

1.2 OBJETIVOS:

1.2.1 OBJETIVO GENERAL

Desarrollar diferentes dispositivos de apoyo para el proyecto Cultivo Robótico Remoto con el fin de monitorizar variables físicas como temperatura, humedad y luminosidad, además de controlar una articulación prismática que sirva de plataforma móvil para un robot.

1.2.2 OBJETIVOS ESPECÍFICOS

- Implementar un sistema transportador lineal para un robot antropomórfico de cinco grados de libertad.
- Adquirir los datos de las variables físicas tales como humedad, temperatura y luminosidad para el proyecto Cultivo Robótico Remoto.
- Realizar el sistema de comunicación que permita operar el sistema transportador del robot y monitorizar las variables físicas solicitadas.
- Desarrollar el sistema de medida del nivel de agua contenido en un tanque incluyendo la etapa de comunicación.

- Obtener el Control de un dispositivo de riego de agua, fertilizantes u otras sustancias líquidas.
- Realizar la puesta a punto y las pruebas del sistema del transportador lineal y el dispositivo de riego.

1.3 ACOTACIONES

- ✓ El informe a entregar será una monografía.
- ✓ El robot mencionado en los objetivos no será desarrollado en este trabajo.
- ✓ Los dispositivos desarrollados en este trabajo se entregarán en la sustentación final.
- ✓ Los elementos utilizados en este trabajo serán de bajo costo y en muchos casos se utilizarán elementos presentes en el laboratorio o pertenecientes a los desarrolladores del proyecto de agricultura.
- ✓ Los elementos mencionados anteriormente podrían ser modificados buscando mejorar el comportamiento del sistema.
- ✓ El sistema se desarrollará para un ambiente libre de perturbaciones tales como interferencias electromagnéticas, ruido, humedad, etc.
- ✓ En este trabajo no se desarrollará un diseño mecánico exhaustivo del transportador lineal, por el contrario se profundizará más en el control eléctrico y electrónico del mismo.

CAPÍTULO 2

2. MARCO TEÓRICO Y ESTADO DEL ARTE

2.1 MARCO TEÓRICO:

2.1.1 PROTOCOLO UDP:

"El protocolo UDP (*Protocolo de datagrama de usuario*) es un protocolo no orientado a conexión de la capa de transporte del modelo TCP/IP. Este protocolo es muy simple ya que no proporciona detección de errores."[2] "Cuando una maquina A envía paquetes a una maquina B, el flujo es unidireccional. La transferencia de datos es realizada sin haber realizado previamente una conexión con la máquina de destino (maquina B), y el destinatario recibirá los datos sin enviar una confirmación al emisor (la maquina A). Esto es debido a que la encapsulación de datos enviada por el protocolo UDP no permite transmitir la información relacionada al emisor. Por ello el destinatario no conocerá al emisor de los datos excepto su IP."[3] Los mensajes UDP están encapsulados y se envían en datagramas IP, como se muestra en la siguiente ilustración.[2]

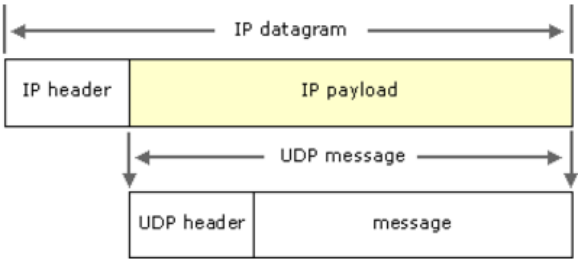

Figura 1. ENCABEZADO DEL SEGMENTO [2]

[2] Protocolo UDP [en línea]. <http://es.ccm.net/contents/284-protocolo-udp>. Noviembre de 2015.
[3] Diferencias entre los protocolos UDP y TPC [en línea]. <http://es.ccm.net/faq/1559-diferencias-entre-los-protocolos-tcp-y-udp>. Noviembre de 2015.

"Significado de los diferentes campos:

- Puerto de origen: es el número de puerto relacionado con la aplicación del remitente del segmento UDP. Este campo representa una dirección de respuesta para el destinatario. Por lo tanto, este campo es opcional. Esto significa que si el puerto de origen no está especificado, los 16 bits de este campo se pondrán en cero. En este caso, el destinatario no podrá responder (lo cual no es estrictamente necesario, en particular para mensajes unidireccionales).
- Puerto de destino: este campo contiene el puerto correspondiente a la aplicación del equipo receptor al que se envía.
- Longitud: este campo especifica la longitud total del segmento, con el encabezado incluido. Sin embargo, el encabezado tiene una longitud de 4 x 16 bits (que es 8 x 8 bits), por lo tanto la longitud del campo es necesariamente superior o igual a 8 bytes.[2]
- Suma de comprobación: es una suma de comprobación realizada de manera tal que permita controlar la integridad del segmento."[2]

2.1.1.2 PUERTOS UDP

"Los puertos UDP proporcionan una ubicación para enviar y recibir mensajes UDP. Un puerto UDP funciona como una única cola de mensajes que recibe todos los datagramas destinados al programa especificado mediante cada número de puerto del protocolo. Es decir, los programas basados en UDP pueden recibir varios mensajes a la vez. El lado de servidor de cada programa que utiliza UDP atiende los mensajes que llegan a su número de puerto conocido"[3].

[2] Protocolo UDP [en línea]. <http://es.ccm.net/contents/284-protocolo-udp>. Noviembre de 2015.
[3] <https://msdn.microsoft.com/es-es/library/cc785220(v=ws.10).aspx>. Noviembre de 2015.

"Todos los números de puerto de servidor UDP inferiores a 1.024 (y algunos números superiores) están reservados y registrados por la Autoridad de números asignados de Internet.

Cada puerto de servidor UDP se identifica mediante un número de puerto conocido o reservado. En la siguiente tabla se muestra algunos ejemplos de números de puertos de servidor UDP conocidos que utilizan programas basados en UDP estándar."[3]

Número de puerto UDP	Descripción
53	Consultas de nombres DNS
69	Protocolo trivial de transferencia de archivos (TFTP)
137	Servicio de nombres NetBIOS
138	Servicio de datagramas NetBIOS
161	Protocolo simple de administración de redes (SNMP)

520	Protocolo de información de enrutamiento (RIP, Routing Information Protocol)

Tabla 1. LISTA PARCIAL DE PUERTOS [3]

3 <https://msdn.microsoft.com/es-es/library/cc785220(v=ws.10).aspx>. Noviembre de 2015.

2.1.2 ARDUINO MEGA 2560

"El Arduino Mega 2560 es una placa electrónica que cuenta con 54 pines digitales de entrada/salida, 16 entradas analógicas, 4 UARTs, un oscilador de cristal de 16 MHz, una conexión USB, un conector de alimentación, un header ICSP, y un botón de reinicio.

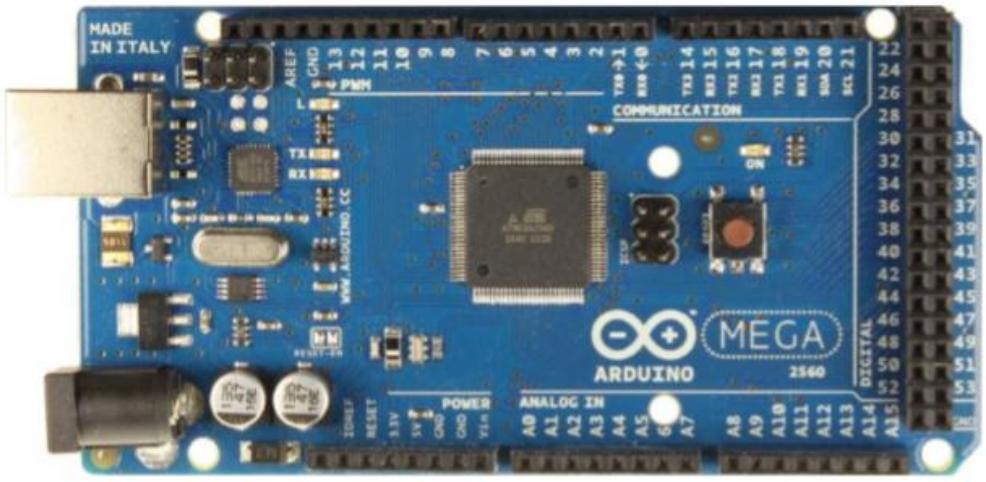

Figura 2. ARDUINO MEGA 2560 [4]

Arduino Mega posee: Microcontrolador ATmega2560, voltaje operativo de 5V, voltaje de Entrada de 7-12V, voltaje de entrada (límites) de 6-20V, corriente DC por cada Pin Entrada/Salida de 40 mA, corriente DC entregada en el Pin 3.3V de 50 mA, memoria flash: 256 KB (8KB usados por el bootloader), SRAM de 8KB, EEPROM de 4KB y una velocidad de reloj de 16 MHz."[4]

[4] <https://www.arduino.cc/en/Main/arduinoBoardMega2560#>. Noviembre de 2015

Arduino puede ser programado de una manera muy fácil utilizando el lenguaje propio de Arduino junto con la interfaz Arduino IDE.[5]

Figura 3. INTERFAZ ARDUINO IDE [5]

2.1.3 ARDUINO ETHERNET SHIELD

"La Arduino Ethernet Shield permite a una placa Arduino conectarse a Internet, sólo se tiene que conectar este módulo a una placa Arduino, y a su red con un cable RJ45. Esta placa requiere una placa Arduino, tiene 5V de tensión de funcionamiento, posee un controlador ethernet W5100 con buffer interno 16K, tiene una velocidad de conexión: 10 / 100Mb y una conexión con Arduino en el puerto SPI."[6]

[5] <https://www.arduino.cc/en/Guide/MacOSX>. Noviembre de 2015
[6] <https://www.arduino.cc/en/Main/ArduinoEthernetShield>. Noviembre de 2015

"El Wiznet W5100 ofrece una red (IP) capaz de apilar TCP y UDP. Soporta hasta cuatro conexiones de socket simultáneas. Utiliza la biblioteca de Ethernet para escribir bocetos que se conectan a Internet a través del escudo. El escudo de Ethernet se conecta a una placa Arduino usando largas cabeceras wire-wrap que se extienden a través del escudo. Esto mantiene la disposición de las clavijas intacto y permite que otro escudo sea apilad en la parte superior."[6]

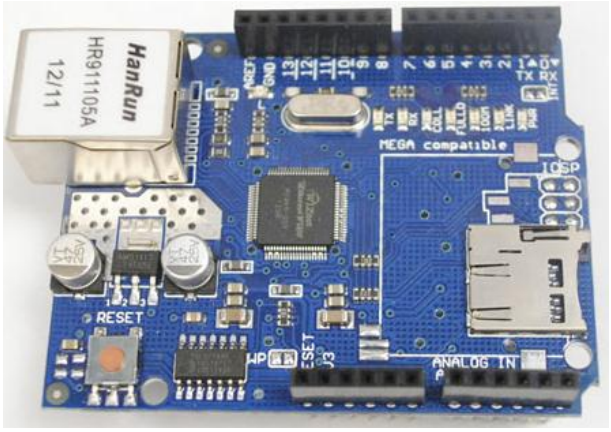

Figura 4. ARDUINO ETHERNET SHIELD[7]

Figura 5. CABLE ETHERNET[7]

[6] <https://www.arduino.cc/en/Main/ArduinoEthernetShield>. Noviembre de 2015.
[7] <http://www.tuelectronica.es/tutoriales/arduino/arduino-ethernet-shield.html>. Noviembre de 2015.

CARACTERÍSTICAS:	
Compatible con IEEE802.3af	
Ondulación baja producción y el ruido (100mVpp)	
Entrada rango de voltaje de 36V a 57V	
Sobrecarga y corto circuito de protección	
9V de salida	
Alta eficiencia convertidor DC / DC: typ 75% @ 50% de carga	
Aislamiento 1500V (entrada a la salida)	
NB: la alimentación a través del módulo Ethernet es hardware no es hecho por Arduino, es de un tercer acceso.	
El protector provee un conector Ethernet estándar RJ45.	
El botón de reinicio en el escudo restablece tanto el W5100 y la placa Arduino.	
	PWR: indica que la placa y el escudo son potencia LINK: indica la presencia de un enlace de red y parpadea cuando el escudo transmite o recibe datos FULLD: Indica que la conexión de red es full dúplex.

El escudo contiene una serie de leds informativos:	100M: Indica la presencia de un Mb / s 100 conexión de red (en contraposición a 10 Mb / s). RX: Parpadea cuando el escudo recibe datos. TX: Parpadea cuando el escudo envía datos. COLL: Parpadea cuando se detectan colisiones de red. El puente de soldadura marcada "INT" puede conectarse a permitir que la placa Arduino para recibir la notificación de interrupción impulsada por los acontecimientos de la W5100, pero esto no es apoyado por la biblioteca de Ethernet. El puente conecta el pin INT del W5100 al pin digital 2 de la Arduino.

Tabla 2. CARACTERÍSTICAS DE ARDUINO ETHERNET SHIELD[6]

6 <https://www.arduino.cc/en/Main/ArduinoEthernetShield>. Noviembre de 2015

2.1.4 AX12W DYNAMIXEL:

El servo AX-12W es una versión de rotación continua del AX-12A especial para trabajar como una rueda en aplicaciones de robótica. Básicamente este servo inteligente tiene una reducción menor en comparación con un AX-12A y por lo tanto logra velocidades mayores.
El algoritmo de control utilizado para mantener la posición del eje del actuador AX-12W se puede ajustar de forma individual para cada servo, lo que permite controlar la velocidad y la fuerza de la respuesta del motor. Todo el control de la gestión y de la posición del sensor se maneja mediante una función de microcontrolador del servo.

CARACTERÍSTICAS:

- Voltaje de trabajo recomendado: 11.1V, también opera entre 9V a 12V
- Torque máximo 15,3 kg · cm , 212 oz · In

- Velocidad sin carga: 430RPM - Peso : 52.9 g - Dimensiones: 32 x 50 x 40 mm
- Resolución: 0.29 ° - Relación de reducción : 1/32
- Funcionamiento ángulo: 360 ° o giro continuo
- Voltaje de funcionamiento 9 -12V (recomendado Voltaje 11.1V)
- Corriente máxima 1.3ª - Corriente en reposo 50 mA
- Temperatura de trabajo: -5 ° C ~ 85 ° C
- Protocolo TTL Half Duplex asíncrono de serie
- Limite Módulo 254 direcciones válidas
- Material Plástico Engranajes y Cuerpo -Motor Tubular

[7]< http://www.manualslib.com/manual/13679/Axis-Dynamixel-Ax-12.html>. Noviembre de 2015.

Figura 6. AX-12W Dynamixel [7]

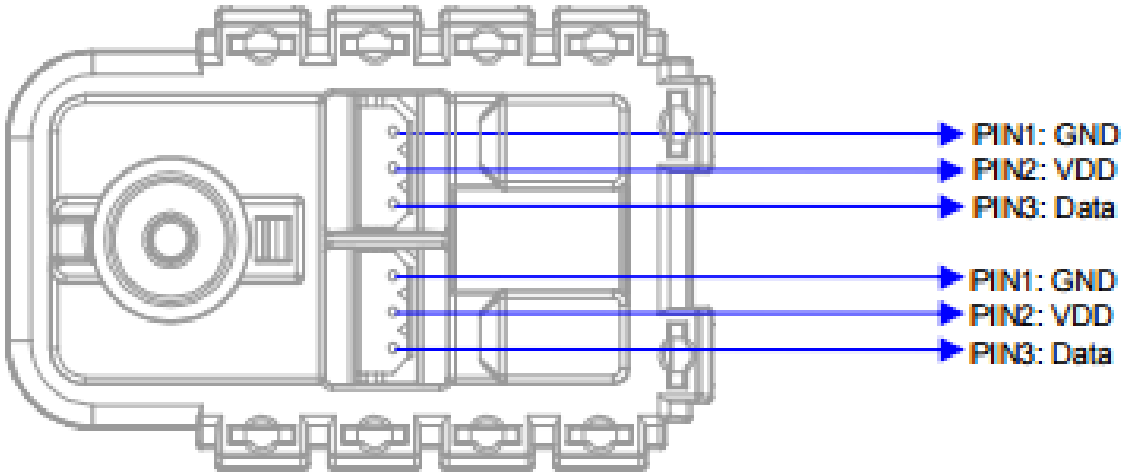

Figura 7. ASIGNACIÓN DE PINES [7]

[7] < http://www.manualslib.com/manual/13679/Axis-Dynamixel-Ax-12.html>. Noviembre de 2015.

2.1.4.1 Protocolo de comunicaciones [7]:

Información general:

- Paquete: El controlador principal se comunica con las unidades Dynamixel enviando y recibiendo paquetes de datos. Hay dos tipos de paquetes; la "Serie de instrucciones" (enviado desde el controlador principal a los actuadores Dynamixel) y el "paquete de estado" (enviado desde los actuadores Dynamixel al controlador principal.)

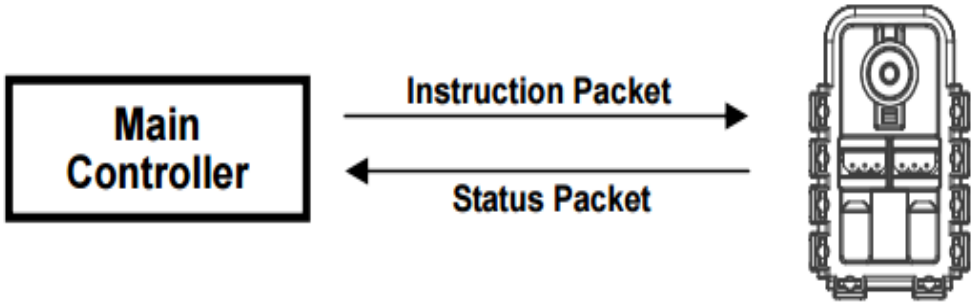

Figura 8. ENVÍO Y RECEPCION DE PAQUETE [7]

- Comunicación: para la conexión del sistema a continuación, si el controlador principal envía un paquete de instrucciones con el conjunto de ID para N, sólo la unidad Dynamixel con este valor de ID regresará su respectivo paquete de estado y realizar la instrucción requerida.

- Paquete de instrucciones: La Serie de instrucciones es el paquete enviado por el controlador principal para las unidades Dynamixel para enviar comandos.

[7]< http://www.manualslib.com/manual/13679/Axis-Dynamixel-Ax-12.html>. Noviembre de 2015.

- Tabla de Control: Consiste en datos relativos a la situación actual y el funcionamiento, lo que existe dentro de Dynamixel. El usuario puede controlar cambiando los datos de la Tabla de Control a través de Serie de instrucciones.

Control Table

	Address	Item	Access	Initial Value
EEPROM Area	0(0X00)	Model Number(L)	RD	12(0x0C)
	1(0X01)	Model Number(H)	RD	0(0x00)
	2(0X02)	Version of Firmware	RD	?
	3(0X03)	ID	RD,WR	1(0x01)
	4(0X04)	Baud Rate	RD,WR	1(0x01)
	5(0X05)	Return Delay Time	RD,WR	250(0xFA)
	6(0X06)	CW Angle Limit(L)	RD,WR	0(0x00)
	7(0X07)	CW Angle Limit(H)	RD,WR	0(0x00)
	8(0X08)	CCW Angle Limit(L)	RD,WR	255(0xFF)
	9(0X09)	CCW Angle Limit(H)	RD,WR	3(0x03)
	10(0x0A)	(Reserved)	-	0(0x00)
	11(0X0B)	the Highest Limit Temperature	RD,WR	85(0x55)
	12(0X0C)	the Lowest Limit Voltage	RD,WR	60(0X3C)
	13(0X0D)	the Highest Limit Voltage	RD,WR	190(0xBE)
	14(0X0E)	Max Torque(L)	RD,WR	255(0XFF)
	15(0X0F)	Max Torque(H)	RD,WR	3(0x03)
	16(0X10)	Status Return Level	RD,WR	2(0x02)
	17(0X11)	Alarm LED	RD,WR	4(0x04)
	18(0X12)	Alarm Shutdown	RD,WR	4(0x04)
	19(0X13)	(Reserved)	RD,WR	0(0x00)
	20(0X14)	Down Calibration(L)	RD	?
	21(0X15)	Down Calibration(H)	RD	?
	22(0X16)	Up Calibration(L)	RD	?
	23(0X17)	Up Calibration(H)	RD	?
RAM Area	24(0X18)	Torque Enable	RD,WR	0(0x00)
	25(0X19)	LED	RD,WR	0(0x00)
	26(0X1A)	CW Compliance Margin	RD,WR	0(0x00)
	27(0X1B)	CCW Compliance Margin	RD,WR	0(0x00)
	28(0X1C)	CW Compliance Slope	RD,WR	32(0x20)
	29(0X1D)	CCW Compliance Slope	RD,WR	32(0x20)
	30(0X1E)	Goal Position(L)	RD,WR	[Addr36]value
	31(0X1F)	Goal Position(H)	RD,WR	[Addr37]value
	32(0X20)	Moving Speed(L)	RD,WR	0
	33(0X21)	Moving Speed(H)	RD,WR	0
	34(0X22)	Torque Limit(L)	RD,WR	[Addr14] value
	35(0X23)	Torque Limit(H)	RD,WR	[Addr15] value
	36(0X24)	Present Position(L)	RD	?
	37(0X25)	Present Position(H)	RD	?
	38(0X26)	Present Speed(L)	RD	?
	39(0X27)	Present Speed(H)	RD	?
	40(0X28)	Present Load(L)	RD	?
	41(0X29)	Present Load(H)	RD	?
	42(0X2A)	Present Voltage	RD	?
	43(0X2B)	Present Temperature	RD	?
	44(0X2C)	Registered Instruction	RD,WR	0(0x00)
	45(0X2D)	(Reserved)	-	0(0x00)
	46[0x2E)	Moving	RD	0(0x00)
	47[0x2F)	Lock	RD,WR	0(0x00)
	48[0x30)	Punch(L)	RD,WR	32(0x20)
	49[0x31)	Punch(H)	RD,WR	0(0x00)

FIGURA 9. TABLA DE CONTROL

[7]< http://www.manualslib.com/manual/13679/Axis-Dynamixel-Ax-12.html>. Noviembre de 2015.

2.1.5 USB2Dynamixel [8]:

El USB2Dynamixel es un dispositivo utilizado para controlar los actuadores de Dynamixel en un PC. El USB2Dynamixel se utiliza mediante la conexión a un puerto USB de un PC, y está equipado con conectores 3p 4p y se conecten con una variedad de dynamixels. Además, se puede utilizar para transformar un puerto USB en un puerto serie para un PC. Esta función se puede utilizar eficazmente con algunos controladores exclusivos de Dynamixel tales como la CM-2, CM-2 + y CM-5 están conectados a un puerto UBS o cuando el ZIG2Serial está conectado a un puerto USB para controlar un robot por radio.

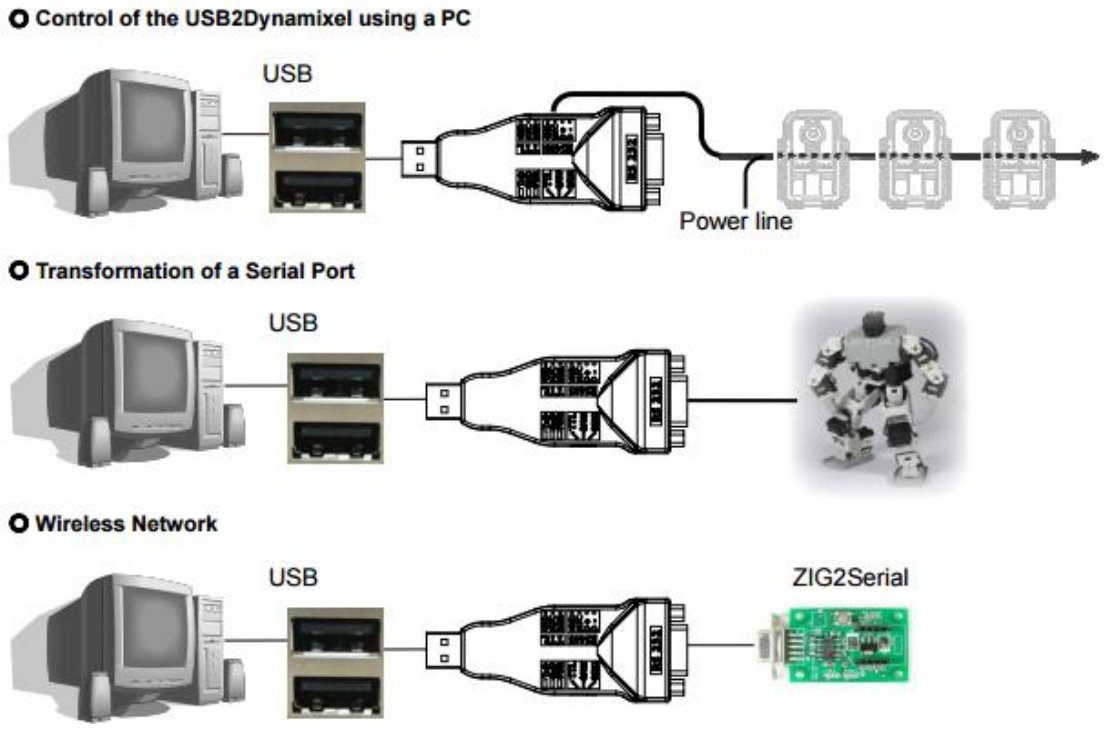

FIGURA 10. USOS DEL USB2DYNAMIXEL

[8]<http://support.robotis.com/en/product/auxdevice/interface/usb2dxl_manual.htm>.Novoembre de 2015.

Requisitos Del Sistema [8]:

- PC: PC compatible con IBM y USB 1.1 o superior

- SO: Windows 2000, Windows XP
- CPU: Intel Pentium III a 1 GHz, AMD Athlon XP de 1 GHz o superior
- RAM: 256 MB o superior

Estructura:

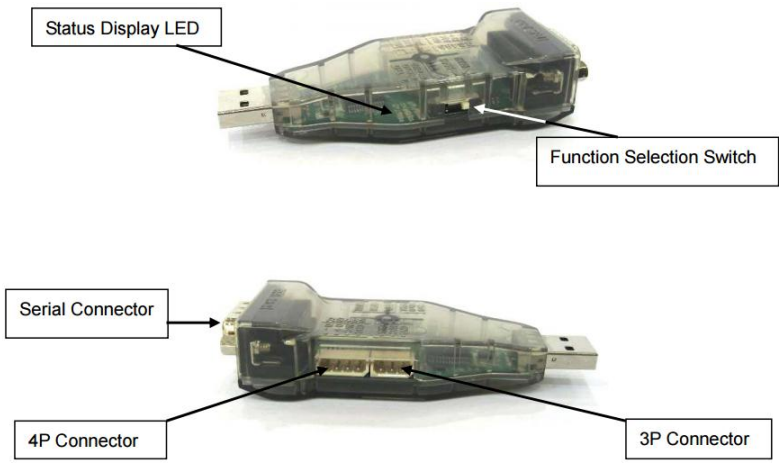

FIGURA 11. ESTRUCTURA USB2DYNAMIXEL [8]

Pantalla de estado LED: Muestra el estado de la fuente de alimentación, TxD (escribir datos) y RxD (datos de lectura).

Función de interruptor Selección: Selecciona cómo comunicación TTL, RS-485 y RS-232

3P Conector: Se conecta a AX Dynamixel serie utilizando la red TTL

4P conector: Se conecta a DX o una serie Dynamixel RX utilizando RS-485

Conector serie: Transforma un puerto USB en un puerto serie utilizando la red RS-232

---

[8]<http://support.robotis.com/en/product/auxdevice/interface/usb2dxl_manual.htm>.Novoembre de 2015.

Conexión de Dynamixel serie AX [8]:

1. Ajuste el interruptor de selección de función en el modo TTL.

2. Conectar el cable al conector 3P de USB2Dynamixel y el conector 3P de la Dynamixel (Hay dos conectores en la Dynamixel, en cualquiera se pueden conectar)
3. Dynamixels se pueden conectar en serie mediante un cable 3P.
4. Conecte la línea de alimentación a la última Dynamixel.

2.1.6 BOMBAS LAVAPARABRISAS [9]

Los eyectores de agua nos ayudan a limpiar el parabrisas gracias al líquido que expulsan. Los eyectores de agua se encuentran situados normalmente en el capó del coche. El agua llega hasta ahí por medio de una tubería que sale del propio depósito de lavaparabrisas.
El agua es impulsada hasta los eyectores por medio de la bomba de agua que se encuentra situada en el depósito del lavaparabrisas.

FIGURA 12. BOMBA LAVAPARABRISAS[9]

Teniendo en cuenta los parámetros y el modo de trabajo de estas válvulas, estos elementos me sirven para implementar el sistema de riego de algunos cultivos.

[9]<http://www.actualidadmotor.com/eyectores-de-agua-del-lavaparabrisas>. Noviembre de 2015.

2.1.7 SENSOR DHT11 [10]:

El DHT11 es un sensor que proporciona una salida de datos digital. Entre sus ventajas se puede mencionar su bajo costo y el despliegue de datos digitales

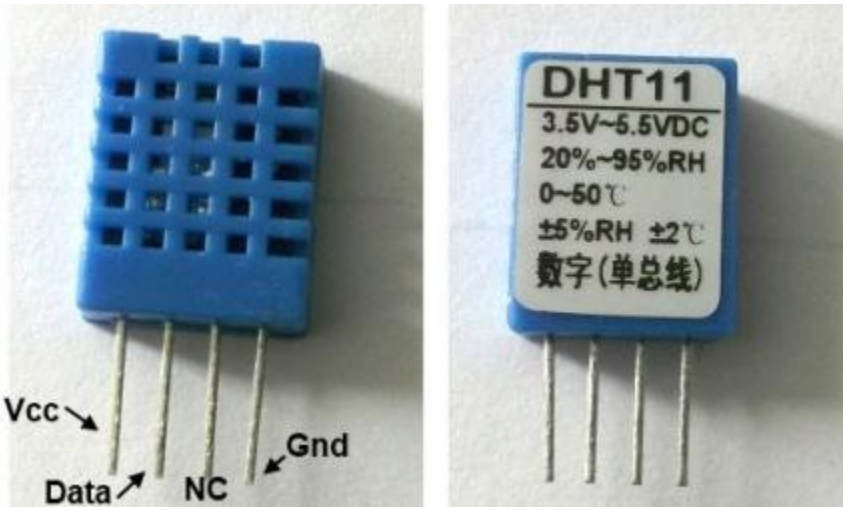

Figura 13. DHT11

Este sensor se caracteriza por tener la señal digital calibrada por lo que asegura una alta calidad y una fiabilidad a lo largo del tiempo, ya que contiene un microcontrolador de 8 bits integrado. Está constituido por dos sensores resistivos (NTC y humedad). Tiene una excelente calidad y una respuesta rápida en las medidas. Puede medir la humedad entre el rango 20-95% y la temperatura entre 0-50°C.

El protocolo de comunicación es a través de un único hilo. Presenta un tamaño reducido, un bajo consumo y la capacidad de transmitir la señal hasta 20 metros de distancia.

[10]<http://panamahitek.com/dht11-sensor-de-humedadtemperatura-para-arduino/>. Noviembre de 2015.

Model	DHT11
Power supply	3-5.5V DC
Output signal	digital signal via single-bus
Sensing element	Polymer resistor
Measuring range	humidity 20-90%RH; temperature 0-50 Celsius
Accuracy	humidity +-4%RH (Max +-5%RH); temperature +-2.0Celsius
Resolution or sensitivity	humidity 1%RH; temperature 0.1Celsius
Repeatability	humidity +-1%RH; temperature +-1Celsius
Humidity hysteresis	+-1%RH
Long-term Stability	+-0.5%RH/year
Sensing period	Average: 2s
Interchangeability	fully interchangeable
Dimensions	size 12*15.5*5.5mm

FIGURA 14. CARACTERÍSTICAS DHT11 [10]

Para poder leer datos desde este sensor de una forma sencilla se necesita descargar una librería que ha sido escrita para este propósito.

[10]<http://panamahitek.com/dht11-sensor-de-humedadtemperatura-para-arduino/>. Noviembre de 2015

2.1.8 SENSOR DE LUMINOSIDAD: FOTORESISTOR [11]:

El **LDR** (*Light Dependent Resistor*) o resistencia dependiente de la luz o también fotocélula, es una resistencia que varía su resistencia en función de la luz que incide sobre su superficie. Cuanto mayor sea la intensidad de la luz que incide en la superficie del LDR menor será su resistencia y cuanto menos luz incida mayor será su resistencia.

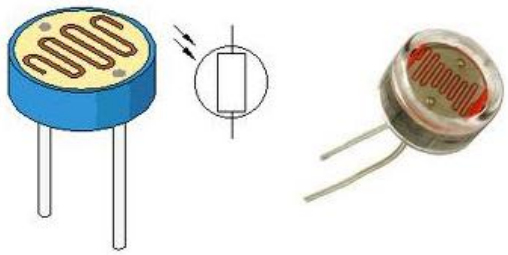

FIGURA 15. LDR

Símbolo electrónico:

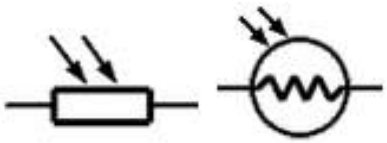

FIGURA 16. SÍMBOLO ELECTRÓNICO LDR

Material de fabricación: Los materiales fotosensibles más utilizados para la fabricación de las resistencias LDR son, el sulfuro de talio, el sulfuro de cadmio, el sulfuro de plomo, y el seleniuro de cadmio.

[11] <http://electronica-electronics.com/info/LDR-fotoresistencia.html>. Noviembre de 2015.

Funcionamiento: Cuando la LDR no está expuesta a radiaciones luminosas los electrones están firmemente unidos en los átomos que la conforman pero cuando sobre ella inciden radiaciones luminosas esta energía libera electrones con lo cual el material se hace más conductor, y de esta manera disminuye su resistencia. Las resistencias LDR solamente reducen su resistencia con una radiación luminosa situada dentro de una determinada banda de longitudes de onda. Las construidas con sulfuro de cadmio son sensibles a todas las radiaciones luminosas visibles, las construidas con sulfuro de plomo solamente son sensibles a las radiaciones infrarrojas[11].

Valor Óhmico: Si se mide entre sus extremos se encuentra que pueden llegar a medir en la oscuridad valores cercanos al Mega Ohm (1MΩ) y expuestas a la luz se medirán valores en el entorno de los 100 Ω. [11]

Tiempo de Respuesta: El tiempo de respuesta típico de un LDR está en el orden de la décima de segundo. [11]

Aplicaciones: Se emplean en iluminación, apagado y encendido de alumbrado (interruptores crepusculares), en alarmas, en cámaras fotográficas, en medidores de luz. Las de la gama infrarroja en control de máquinas y procesos de contaje y detección de objetos[11].

[11]<http://electronica-electronics.com/info/LDR-fotoresistencia.html>. Noviembre de 2015.

[12]<http://contextoganadero.com/ganaderia-sostenible/la-agro-robotica-un-concepto-que-va-camino-la-consolidacion>. Noviembre de 2015.

2.2 ESTADO DEL ARTE:

En Oceanía, el investigador australiano Salah Sukkarieh arrancó la fase de exploración de su proyecto, el cual busca que las labores del campo también sean desempeñadas por robots, la cual hace dos años despertó la desconfianza y el recelo de los dueños de las fincas en Australia. La idea principal era detectar indicadores agrícolas importantes como el rendimiento, la salud de los árboles y la intensidad de la floración y los frutos[12].

Esta tecnología ha mostrado una amplia gama de beneficios entre los que se cuentan resolver el problema de la escasez de mano de obra en Australia a través de la recolección automatizada. Los agricultores empezaron a descubrir las ventajas y, según los investigadores de la Universidad de Sidney, están entusiasmados[12].

La comisión europea plantea que los cambios meteorológicos estacionales abruptos como las riadas, las heladas prematuras y las tempestades de nieve generan múltiples quebraderos de cabeza en forma de pérdida de cultivos o ganado a aquellos que viven del campo. Aunque muchos de estos problemas son naturales y hasta cierto punto inevitables, la industria agropecuaria no deja de adaptarse a este tipo de retos. No obstante, puede que una de las principales invenciones del siglo XX, el ordenador, dé paso a una nueva revolución[12].

La práctica agropecuaria ha afrontado sus retos con innovaciones en cada ámbito, como por ejemplo sistemas inteligentes que regulan los motores y ahorran combustible. Las tecnologías han ganado terreno en este sector impulsadas por la necesidad de ampliar la temporada de cultivo y la producción.

[12]<http://contextoganadero.com/ganaderia-sostenible/la-agro-robotica-un-concepto-que-va-camino-la-consolidacion>. Septiembre de 2015.

Una de ellas, la referente a satélites y sensores, permite que la maquinaria agrícola realice el trabajo en los terrenos de cultivo de manera autónoma, aumentando así la eficiencia en la distribución de semillas, fertilizantes y plaguicidas. No obstante, este tipo de mejoras está alcanzando un límite de desarrollo[12].

A partir de ahora se pretende incorporar cada uno de estos sistemas en los de producción «ciberfísica», que permiten registrar el proceso completo de manera electrónica, desde el ordenador de la explotación hasta la propia cosecha. Estos sistemas podrían mejorar notablemente la eficiencia y la calidad. Investigadores del Instituto Fraunhofer de Ingeniería de Software Experimental (IESE) de Kaiserslautern (Alemania) desarrollaron un programa informático que demostrará los beneficios de los sistemas en red para la agricultura y la ganadería europeas[12].

En Argentina crean robot todo terreno para cultivos intensivos, Se trata de una plataforma con inteligencia artificial que puede desplazarse, hacer monitoreo 3D, fertilizar e incluso podar. El robot cuenta con gran adaptabilidad para hacer mapeo de imágenes 3D, aspersión de fertilizantes, poda y otras labores que se integran con equipos electrónicos[13].

[12]<http://contextoganadero.com/ganaderia-sostenible/la-agro-robotica-un-concepto-que-va-camino-la-consolidacion>. Septiembre de 2015.

[13]<http://www.contextoganadero.com/internacional/en-argentina-crean-robot-todo-terreno-para-cultivos-intensivos>. Septiembre de 2015.

FIGURA 17. ROBOT TODOTERRENO PARA CULTIVOS[13]

El robot también puede ir una y otra vez al lugar asignado a realizar distintas funciones y recordar posteriormente lo que ya hizo, pues bajos los principios de la inteligencia artificial integrados al proyecto, el robot podrá realizar acciones de acuerdo al panorama que se le presente[13].

La Escuela Colombiana de Ingeniería Julio Garavito, busca hacer de la tecnología un aliado al sector agroindustrial. Desde hace un año trabaja en un proyecto de agricultura urbana que hará más fácil el cuidado de cultivos con posibilidades de censar, analizar y controla las diferentes variables biológicas que puedan afectar las también llamadas “huertas caseras”[14].

[13]<http://www.contextoganadero.com/internacional/en-argentina-crean-robot-todo-terreno-para-cultivos-intensivos>. Septiembre de 2015.
[14]<http://www.larepublica.co/el-tema-de-tecnolog%C3%ADa-agricultura-urbana-empieza-abrir-campo_155796>. Septiembre de 2015.

FIGURA 18. PROYECTO DE AGRICULTURA URBANA[14]

En este proyecto los estudiantes de ingeniería electrónica juegan un papel muy importante ya que continuamente están trabajando en prototipos que puedan automatizar los procesos de cosecha y terminar con la necesidad de estar constantemente en la supervisión del cultivo[14].

En Colombia, eventos como Bogotá Robótica o Expociencia y Expotecnología presentaron novedades en materia de robótica para el agro, lo cual indica que el país hace parte de esta tendencia que ya tiene destacados ejemplos en otros lugares del mundo. Robots que monitorean indicadores agrícolas

En la Universidad de Pamplona el programa de Ingeniería Mecatrónica está iniciando un estudio en esta área.

[14]<http://www.larepublica.co/el-tema-de-tecnolog%C3%ADa-agricultura-urbana-empieza-abrir-campo_155796>. Septiembre de 2015.

CAPÍTULO 3

3 DESARROLLO DE DISPOSITIVOS DE APOYO

Para obtener un adecuado desarrollo de los diferentes dispositivos que servirán de apoyo para el proyecto Cultivo Robótico Remoto es importante hacer las respectivas conexiones entre los sensores, la placa Arduino, la Ethernet Shield, el AX12W de Dynamixel, los actuadores y el computador. A continuación se ilustrará a través de una gráfica los diferentes factores que intervienen para un adecuado funcionamiento de los dispositivos.

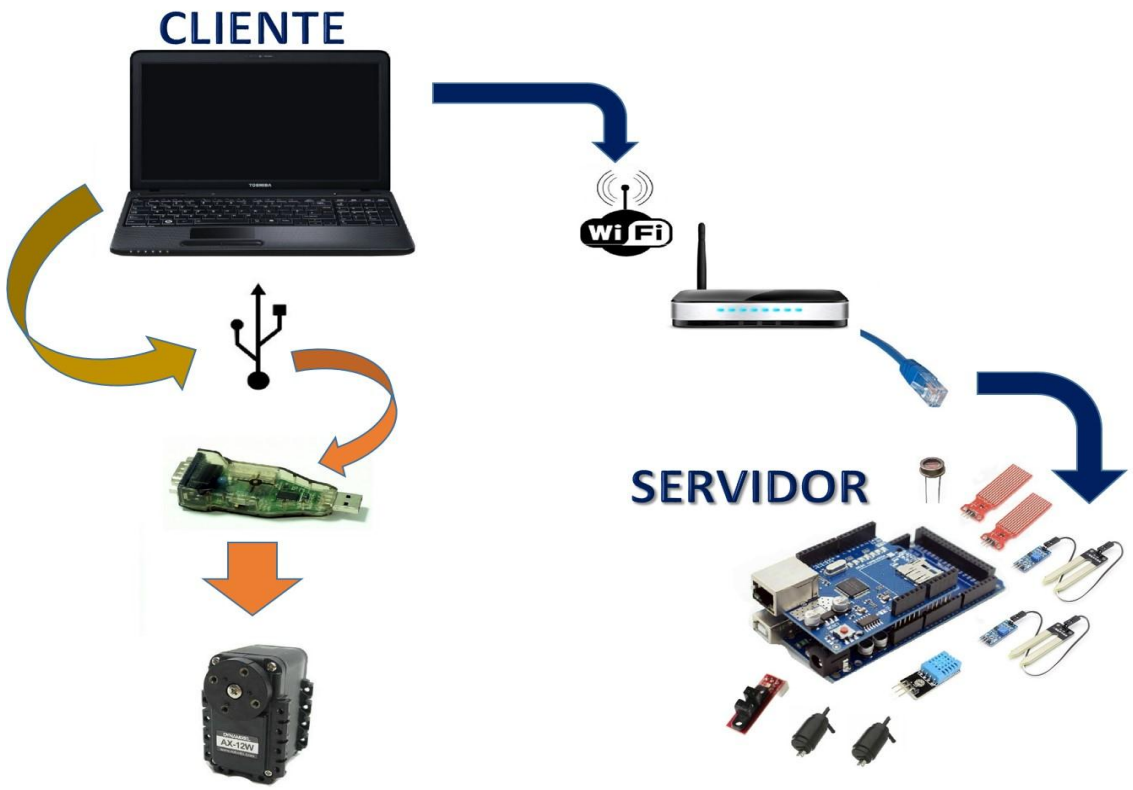

FIGURA 19. DIAGRAMA GENERAL DE SENSORES Y ACTUADORES

Como lo ilustra el diagrama, para la comunicación entre el Arduino y el pc se utilizará el protocolo UDP. También se puede observar que el AX12W de Dynamixel, se comunica con el pc a través del USB2Dynamixel.

En el desarrollo de este proyecto, se utiliza la salida análoga A0 del Arduino para adquirir los valores correspondientes al sensor de luminosidad. La obtención de los valores de temperatura y humedad se hacen a través de la salida digital 5 de esta misma tarjeta. La captura de los datos correspondientes al nivel de los tanques, se realizan por las salidas análogas A1 y A2. Para el accionamiento de las válvulas, se determinan las entradas digitales D6 y D7.

Se decide implementar un final de carrera óptico conectado a la entrada D8 el cual me garantiza el frenado del robot, en las diferentes partes del circuito.

3.1 DESARROLLO DEL SISTEMA DE DESPLAZAMIENTO LINEAL:

Para el desarrollo del sistema de desplazamiento lineal, la comunicación se presenta de la siguiente manera:

FIGURA 20. DIAGRAMA DE COMUNICACIONES

Así mismo, teniendo en cuenta los parámetros necesarios para un adecuado movimiento por parte del Robot, se decide solicitar una asesoría sobre la construcción de estos dispositivos, dando como resultado lo siguiente:

- Inicialmente se solicitan los parámetros dimensionales del robot. Teniendo en cuenta esto, se obtiene la placa que se muestra en la siguiente figura.

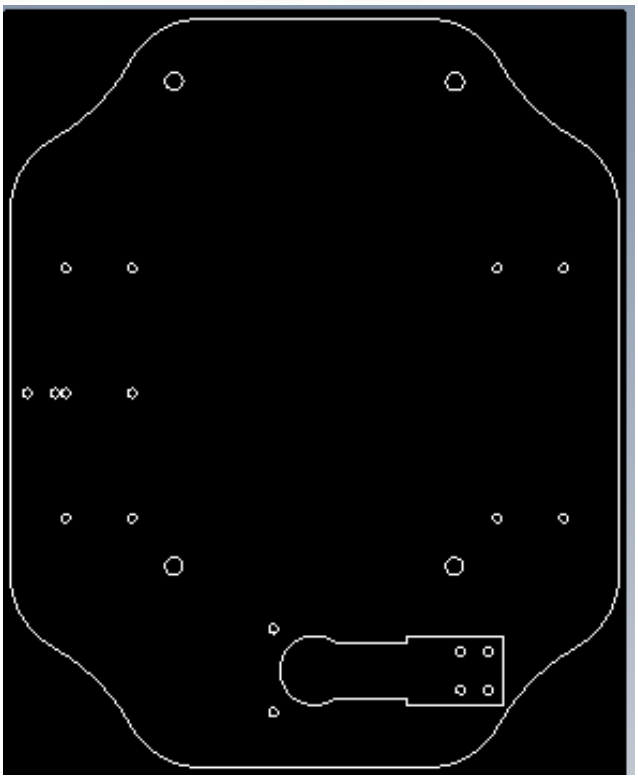

FIGURA 21. BASE ROBOT

- Para desarrollar la estructura base, se tiene en cuenta una superficie metálica de 105 centímetros de largo por 30 centímetros de ancho.

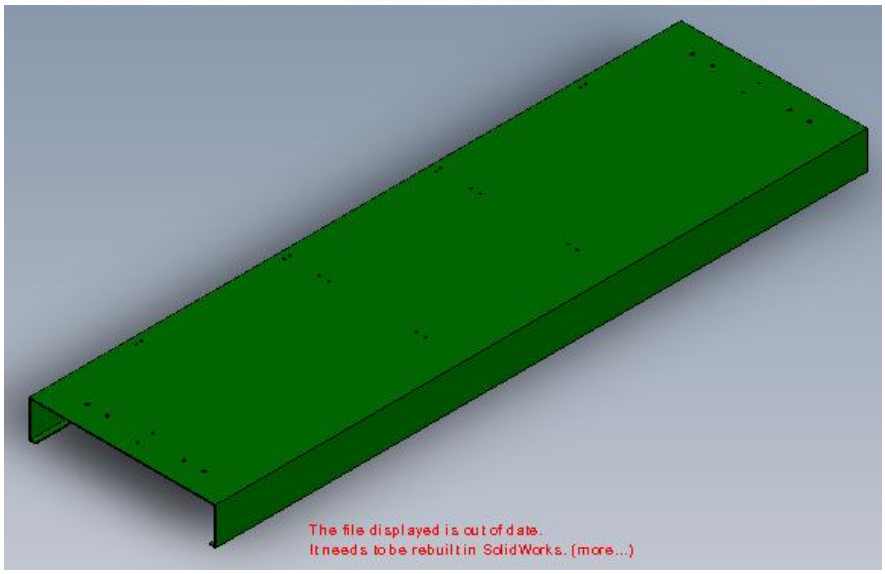

FIGURA 22. ESTRUCTURA ROBOT

- Posteriormente se decide que para obtener un adecuado funcionamiento de este dispositivo, se deben utilizar en la construcción de la base del robot los elementos que se ven a continuación:

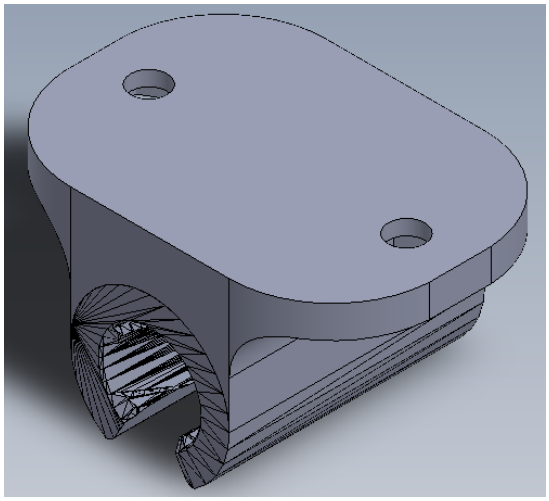

FIGURA 23. RODAMIENTO

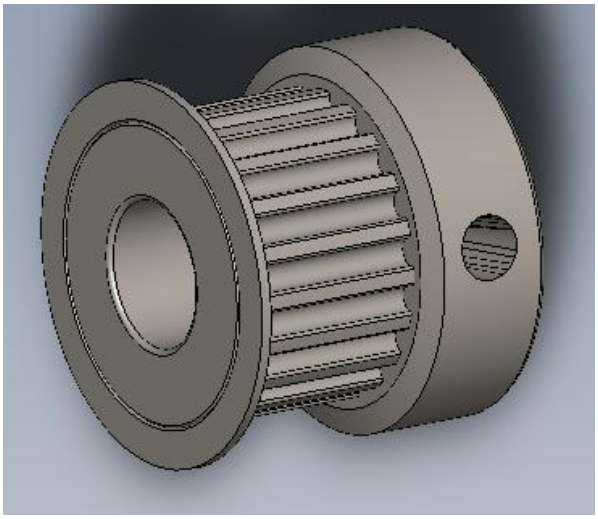

FIGURA 24. POLEAS

- El ensamblaje de la base del robot es el siguiente:

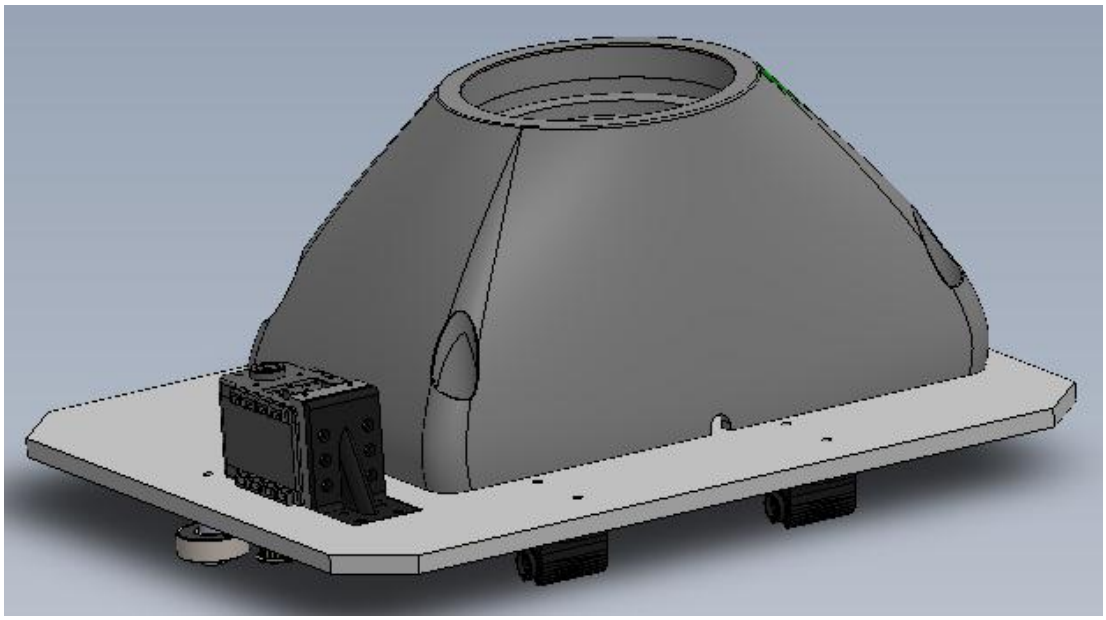

FIGURA 25. PRIMER ACOPLAMIENTO

- Para obtener el movimiento sobre la superficie metálica, se tienen en cuenta los siguientes elementos.

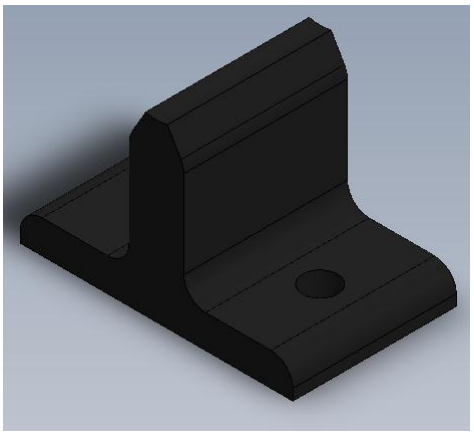

FIGURA 26. SOPORTE VARILLA

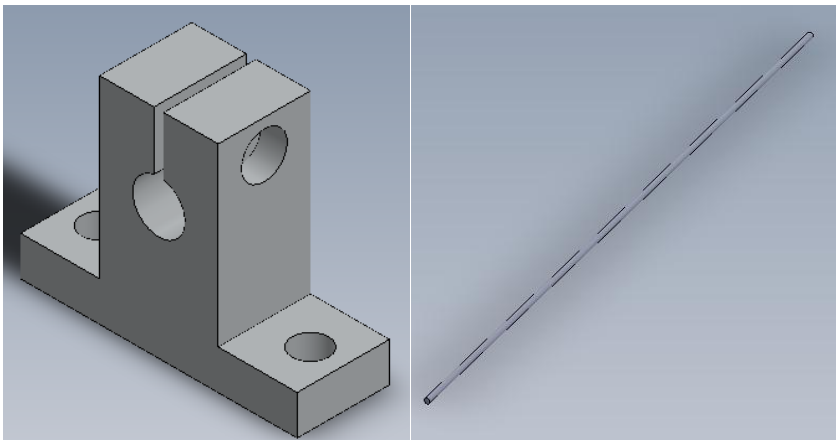

FIGURA 27. SOPORTE Y VARILLA

- Para la implementación del Dynamixel, se utiliza una polea sobre la cual está una correa que se encuentra prensada a cada extremo de la estructura metálica.

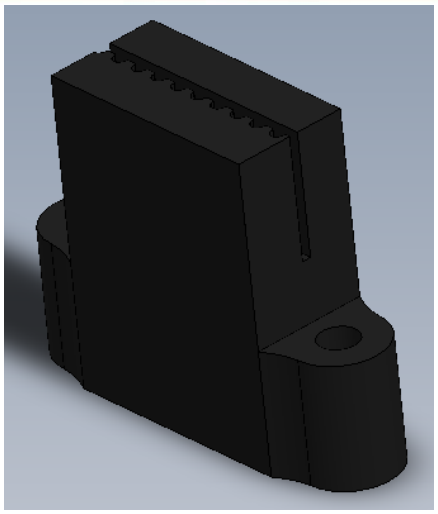

FIGURA 28. PRENSA CORREAS

- El acoplamiento final es el siguiente:

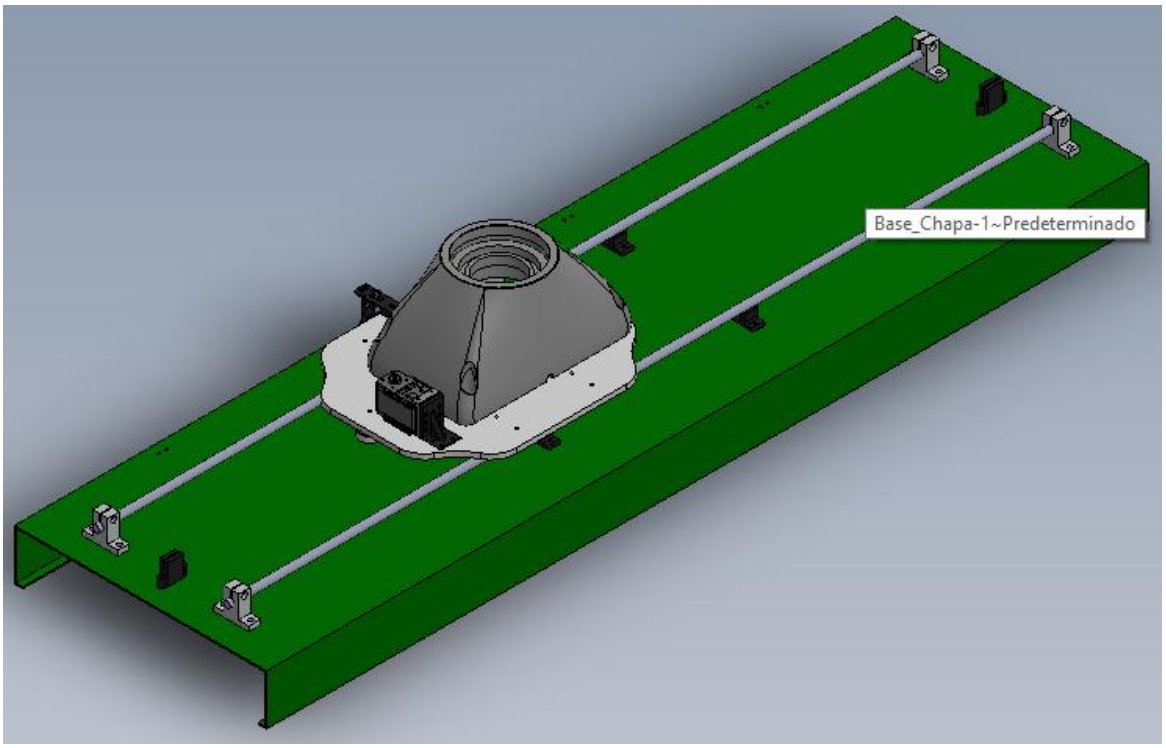

FIGURA 29. DISPOSITIVO DE DESPLAZAMIENTO LINEAL

3.2 DESARROLLO DEL SISTEMA DE CAPTURAS DE SEÑALES:

Para el desarrollo de las diferentes aplicaciones enfocadas al área de la agricultura, es de gran relevancia el conocimiento de las diferentes variables físicas que intervienen en ellas. Por lo enunciado anteriormente se determinó lo siguiente:

- Para el desarrollo de este proyecto, y teniendo en cuenta que no se necesitan demasiados recursos, se decidió trabajar con la tarjeta ARDUINO MEGA 2560, ya que esta me permite adquirir los valores de diferentes magnitudes físicas a través de sus puertos análogos y digitales, además que ofrece la posibilidad de obtener una comunicación constante utilizando el protocolo UDP y la Ethernet Shield, y de esta tarjeta se obtiene buena cantidad de información además de tener conocimientos previos sobre el manejo de la misma.

- Se determina utilizar como sensor de luminosidad a un resistor LDR ya que posee una lectura adecuada de esta variable física y su implementación es sencilla.

- Teniendo en cuenta que el sensor DHT11 me permite obtener una lectura adecuada de la humedad relativa presente en el cultivo y de la temperatura ambiente en el mismo, se determina la utilización de este sensor. Además que de los diferentes sensores de este tipo que se encuentran a la venta, es el que posee la implementación más sencilla.

- Para realizar el control de nivel de los tanques, inicialmente se pretendía utilizar finales de carrera de tal forma que me permita identificar los diferentes niveles de los tanques, pero teniendo en cuenta los problemas de implementación y poca estabilidad en los resultados, se optó trabajar con el sensor de agua para Arduino, ya que tienen una lectura óptima al tener contacto con los líquidos y es visualmente más atractivo.

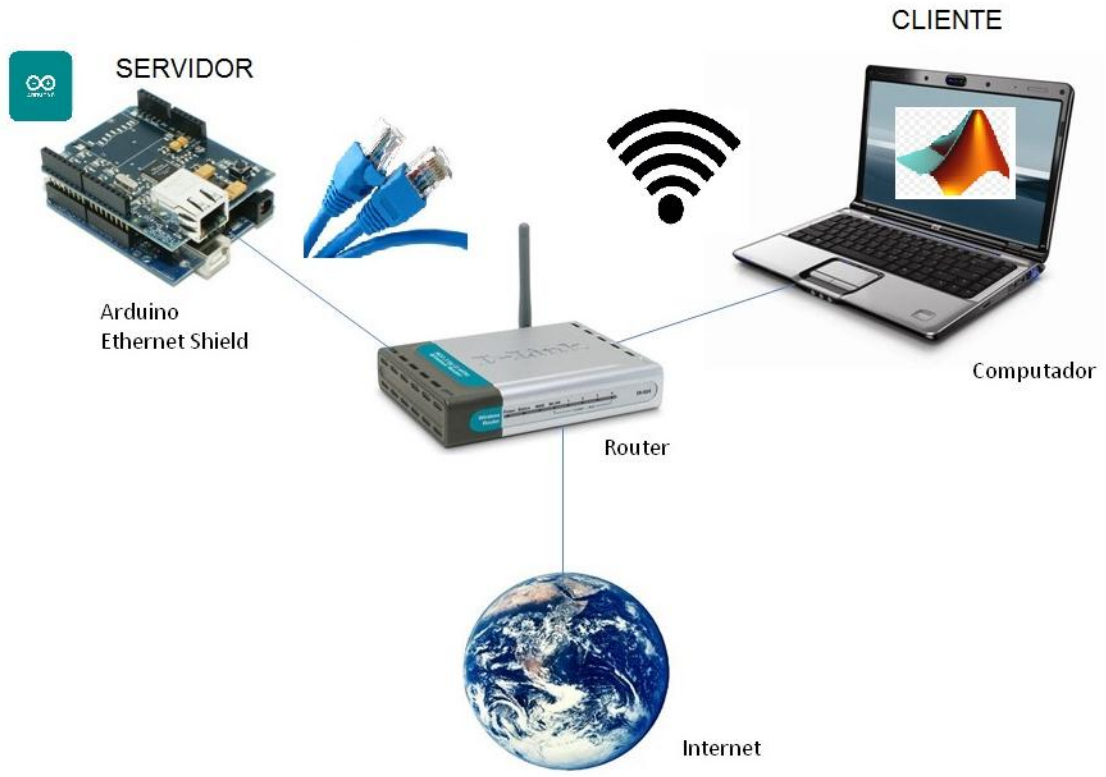

FIGURA 30. DIAGRAMA DE COMUNICACIONES

Como se puede observar en el diagrama anterior, y teniendo en cuenta que la Arduino Ethernet Shield me permite una comunicación UDP se establece lo siguiente:

- La placa Arduino Ethernet Shield se establece como el servidor.
- El computador será el cliente.
- La comunicación entre Arduino e Internet se hará at través de un cable de red hacia el router.
- La comunicación entre el pc e internet se hará a través del wifi que emite el router.
- El programa servidor es IDE Arduino.
- Una interfaz gráfica en Matlab será el cliente de esta comunicación.

De esta manera y teniendo en cuenta las conexiones como se observa en la figura 19, se realiza la captura de los datos correspondientes a los sensores de luminosidad, humedad, temperatura y nivel.

3.3 CONTROL DE LOS ACTUADORES:

Para el desarrollo del control de los actuadores, se tiene en cuenta la comunicación como se expresa en la figura 21 y se realizan las conexiones como se observan a continuación:

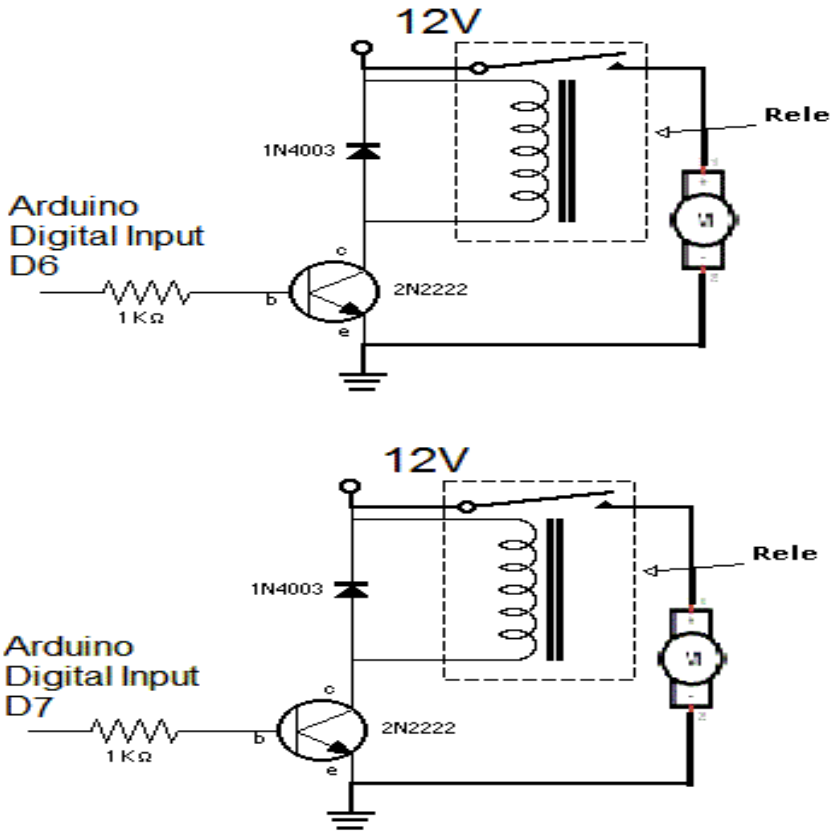

FIGURA 31. DIAGRAMA DE CONEXIONES

Como se expresa en el diagrama de conexiones, para accionar las válvulas que se encargan del sistema de riego del cultivo es necesario llevar a 1 las entradas digitales D6 para agua y D7 para fertilizante.

3.4 DESARROLLO DEL CÓDIGO EN ARDUINO IDE:

A continuación, se desarrolla el diagrama de flujo y el código correspondiente al servidor de la comunicación UDP.

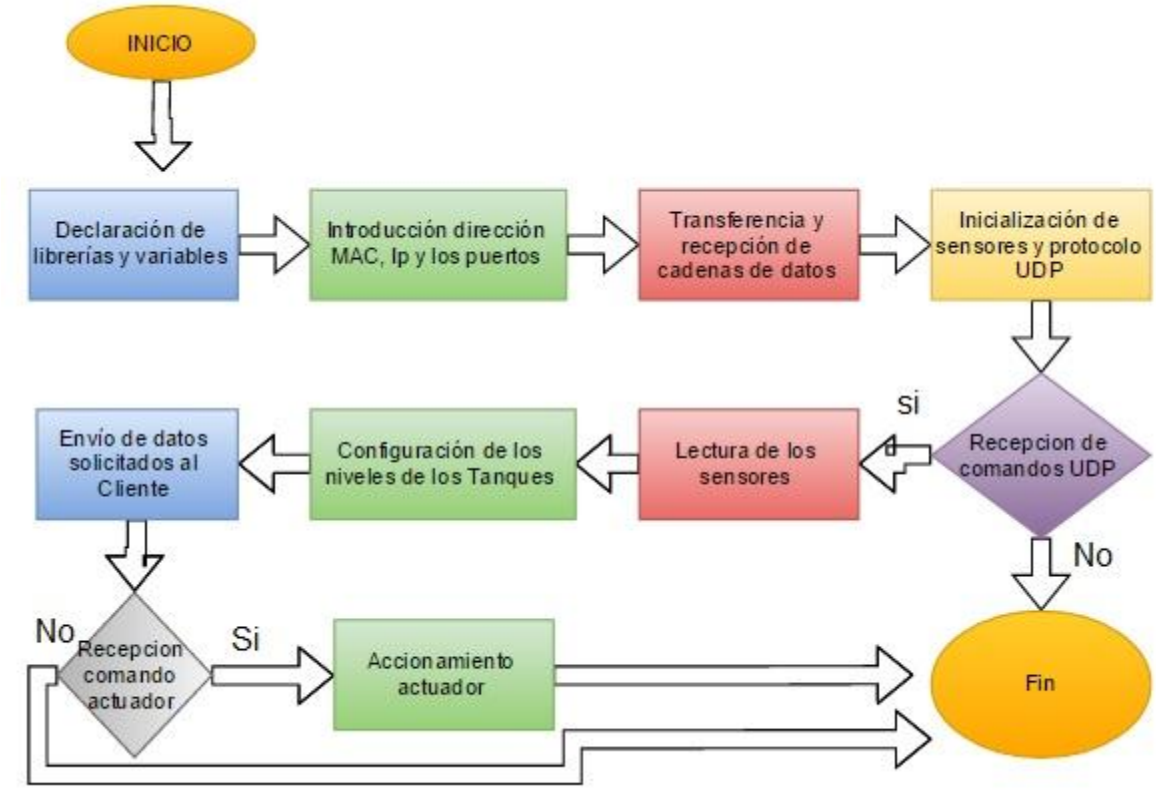

FIGURA 32. DIAGRAMA DE FLUJO PROGRAMA SERVIDOR

Descripción:

- Luego de iniciar el programa y declarar las librerías y variables como es habitual en programación, se procede a introducir los valores que me van a permitir la comunicación entre servidor y cliente.
- En la transferencia y recepción de datos, se determinan una serie de instrucciones en para que se pueda dar esta comunicación.
- La inicialización es necesaria para el testeo de sensores y la comunicación UDP.
- Si se da la recepción de comandos UDP, Lee los valores de los sensores.
- Si no, finaliza el programa.
- Luego de testear los sensores, se configuran los valores de los niveles de los tanques.
- El envío de datos, refleja los valores testeados, en la interfaz del cliente.
- Si se recibe un comando especial, acciona la válvula correspondiente.
- Si no recibe el comando especial, no acciona ninguna válvula.

Código:

```
#include <SPI.h>
#include <Ethernet.h>
#include <EthernetUdp.h>
#include "DHT.h" //Se carga la librería DHT
#define DHTPIN 5 //Se Selecciona el pin en el que se conectará el sensor DHT
#define DHTTYPE DHT11 //Se selecciona el DHT11(hay otros DHT)
DHT dht(DHTPIN, DHTTYPE); //Se inicia una variable que será usada por Arduino para comunicarse con el sensor
//---------------------------------------------------------------------------Parte sensores
int ldr;
int sl;
int s2;
int fco;
int stl;
int st2;
int luzPin =0;  //  Se define la  entrada en  pin A0
int sensor_agua=1;  //  Se define la  entrada en  pin A1
int sensor_fertilizante=2;  // Se define la  entrada en  pin A2
int receptor;
int regar_agua=6;
int regar_fertilizante=7;
int fc_optico=8;
int senso_tierra1=3;
int senso_tierra2=4;
////-------------------------------------------------------------------------
//// Introduzca una dirección MAC y la dirección IP para el controlador de abajo.
//// La dirección IP será dependiente de la red local
////-------------------------------------------------------------------------
byte mac[] = {
```

```
  0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress ip(192, 168, 1, 177);
unsigned int localPort = 8888;      // Puerto local para escuchar en
//buffers para recibir y enviar datos
char packetBuffer[UDP_TX_PACKET_MAX_SIZE]; //para mantener paquete entrante
char  ReplyBuffer[] = "[12, 33, 444]";     // cadena a enviar de vuelta
char sensorBuffer[100];
//Una instancia EthernetUDP para dejarnos enviar y recibir paquetes a través de UDP
EthernetUDP Udp;

void setup() {
 // Iniciar el Ethernet y UDP :
  Ethernet.begin(mac,ip);
  Udp.begin(localPort);
  ////////////////////////////////
  dht.begin(); //Se inicia el sensor
  ////////////////////////////////
  pinMode (regar_agua, OUTPUT);
  pinMode (regar_fertilizante, OUTPUT);
  pinMode (fc_optico, OUTPUT);
  digitalWrite(fc_optico, HIGH); //pullup
  pinMode (senso_tierra1, OUTPUT);
  pinMode (senso_tierra2, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  // Si de allí hay datos disponibles, leer un paquete
  int packetSize = Udp.parsePacket();
  if(packetSize)
  {
    ////////////////////////////////////////////////
  int h = dht.readHumidity(); //Se lee la humedad
  int t = dht.readTemperature(); //Se lee la temperatura
  //Se imprimen las variables
  ldr = analogRead(luzPin); //Se lee la luminosidad
  s1  = analogRead(sensor_agua); //Se lee el nivel del tanque de agua
  s2  = analogRead(sensor_fertilizante); //Se lee el nivel del tanque de fertilizante
  fco = digitalRead(fc_optico); Se lee el sensor optico
  st1 = analogRead(senso_tierra1);
  st2 = analogRead(senso_tierra2);
  //------------- CONFIGURACIÓN DE LOS NIVELES S1 ------------------------------
  if (s1<=150)
  {
    s1=0;
  }
   if ((s1<=200) && (s1>150))
```

```
{
  s1=25;
}
 if ((s1<=250)&& (s1>200))
{
  s1=50;
}
 if ((s1<=300)&& (s1>250))
{
  s1=75;
}
 if (s1>=350)
{
  s1=100;
}
//-----------------------------------------------
//------------- CONFIGURACIÓN DE LOS NIVELES S2-------------------------------
if (s2<=100)
{
  s2=0;
}
 if ((s2<=150) && (s1>100))
{
  s2=25;
}
 if ((s2<=200)&& (s2>150))
{
  s2=50;
}
 if ((s2<=350)&& (s2>200))
{
  s2=75;
}
 if (s2>=300)
{
  s2=100;
}
//-----------------------------------------------

////////////////////////////////////////////////
Serial.print(s1);
////////////////////////////////////////////////
  Serial.print("Received packet of size ");
  Serial.println(packetSize);
  Serial.print("From ");
  IPAddress remote = Udp.remoteIP();
  for (int i =0; i < 4; i++)
```

```
  {
   Serial.print(remote[i], DEC);
   if (i < 3)
   {
    Serial.print(".");
   }
  }
  Serial.print(", port ");
  Serial.println(Udp.remotePort());

  // read the packet into packetBufffer
  receptor=Udp.read(packetBuffer,UDP_TX_PACKET_MAX_SIZE);
  Serial.println("packetBuffer es: ");
  Serial.print(packetBuffer);
  Serial.println("receptor es: ");

  sprintf(sensorBuffer,"[%d, %d, %d, %d, %d, %d, %d, %d]",ldr, h, t, sl, s2, fco, stl, st2);
  // send a reply, to the IP address and port that sent us the packet we received
  Udp.beginPacket(Udp.remoteIP(), Udp.remotePort());
  //Udp.write(ReplyBuffer);
  Udp.write(sensorBuffer);
  Udp.endPacket();

  for (int j=0; j < 100; j++)
  {  //----------------
   if (atoi(packetBuffer) == 1)
   {
     Serial.println("comando 1");
     digitalWrite (regar_agua, HIGH);
   }
   else if (atoi(packetBuffer) == 0)
   {
      digitalWrite (regar_agua, LOW);
      digitalWrite (regar_fertilizante, LOW);

   }
   else if (atoi(packetBuffer) == 2)
   {
    Serial.println("comando 2");
    digitalWrite (regar_fertilizante, HIGH);
   }
  }  //----------------
 }
 delay(10);
}
```

En este código se establecen todos los parámetros para la captura de los sensores de variables físicas además del accionamiento del sistema de riego y del control del motor a través de un final de carrera óptico.

3.5 DESARROLLO DE LA INTERFÁZ GRÁFICA

Con esta interfaz, se pretende dar un adecuado manejo y control de los diferentes dispositivos de apoyo que se realizaron.

A continuación se observará el desarrollo de la interfaz gráfica:

- Se inicia con una interfaz de presentación:

FIGURA 33. INTERFÁZ DE PRESENTACION

- A continuación se presenta la interfaz de desarrollo que se encargará de operar todas las funciones que se requieren.

FIGURA 34. INTERFÁZ DE CONTROL MANUAL

- 1: Se encarga de solicitar los valores de las diferentes variables físicas.
  2: Activación de la cámara.
  3: Visualización del nivel de algunos tanques.
  4: Activación de las válvulas.
  5: Controlar los movimientos del robot.
  6: finalizar el programa.

A continuación, se encontrará una tabla con las diferentes funciones que intervienen en el sistema.

Función:	Descripción
`manual_Opening`	Se encarga de la configuración inicial de la interfaz
`luminosidad`	Se encarga de extraer los valores de luminosidad, leído por el Arduino en A0.
`humedad`	Se encarga de extraer los valores de humedad, leído por el Arduino en D4.
`temperatura`	Se encarga de extraer los valores de temperatura, leído por el Arduino en D4.
`Humedad_t1`	Se encarga de extraer los valores de humedad, leído por el Arduino en A3
`Humedad_t2`	Se encarga de extraer los valores de humedad, leído por el Arduino en A4.
`porcentaje_restante`	Se encarga de la visualización del % restante de líquidos en los tanques.
`agua`	Determina el tiempo de duración del riego de agua.
`fertilizante`	Determina el tiempo de duración del riego de fertilizante.
`posicion`	Posiciona el robot dónde se desee.
`Velocidad`	Determina la velocidad del movimiento del robot.
`Cámara`	Activa el funcionamiento de la cámara
`Cerrar`	Cierra la interfaz
`automatico`	Realiza procesos especiales de forma automática.

Tabla 3. FUNCIONES Y DESCRIPCIÓN

El siguiente diagrama de flujo, explica el funcionamiento del código en la interfaz gráfica:

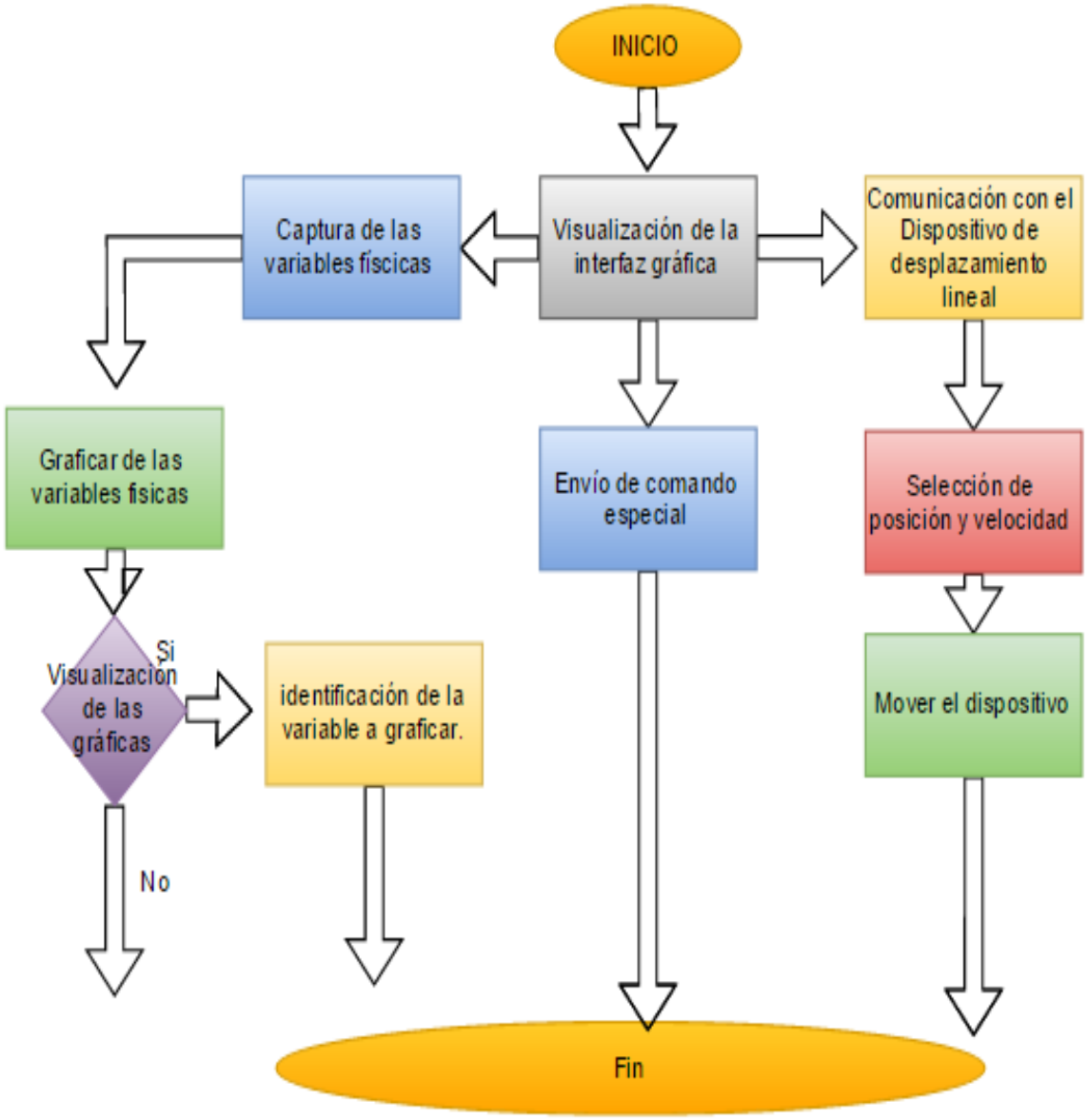

FIGURA 35. DIAGRAMA DE FLUJO INTERFAZ GRAFICA

Códigos:

```
% inicializar_comunicaciones.m

u2 = udp('192.168.1.177', 'RemotePort', 8888, 'LocalPort', 80, 'TimeOut', 5); %La
direccion ip debe ser la del servidor
fopen(u2) % Abre el puerto del cliente


      if not(isempty(instrfind))
          fclose(instrfind);
      end

      inicializar_comunicaciones;

      vec1 =[];
      vec2 =[];
      vec3 =[];

      valor_sensores_anterior = 0;
      limite_datos = 100;

      for i=1:10000
          valor_sensores=llamadosensot_rapido(u2);
          if valor_sensores == -7777
              valor_sensores = valor_sensores_anterior;
          else
              valor_sensores_anterior = valor_sensores;
          end
          vec1= [vec1 , valor_sensores(1)];
          vec2= [vec2 , valor_sensores(2)];
          vec3= [vec3 , valor_sensores(3)];

          if i >= limite_datos
              vec1(1)= [];
              vec2(1)= [];
              vec3(1)= [];
          end
      if (senso_luz==1)

              hold on
              plot(vec1,'yellow','LineWidth', 2);
              grid on
              pause(0.05);
          end

          if (senso_hume==1)
```

```
            hold on
            plot(vec2,'blue','LineWidth', 2);
            grid on
            pause(0.05);
            hold off
        end

        if (senso_temp==1)
            hold on
            plot(vec3,'red','LineWidth', 2);
            grid on
            pause(0.05);
        end

        if (senso_temp==0 && senso_luz==0 && senso_hume==0)

            axes(handles.axes2);
            cla;
            set(handles.axes2,'Visible','off');
            pause(100)

         end


    end

  fclose(u2)   % Cierra el cliente
  delete(u2)   % borra el cliente
  clear u2
```

De la manera expresada anteriormente se extraen los valores correspondientes a las variables físicas teniendo en cuenta que el vector valor sensores se llena como lo indica la siguiente figura:

```
sprintf(sensorBuffer,"[%d, %d, %d, %d, %d, %d, %d, %d]",ldr, h, t, s1, s2, fco, st1, st2);
  // send a reply, to the IP address and port that sent us the packet we received
  Udp.beginPacket(Udp.remoteIP(), Udp.remotePort());
  //Udp.write(ReplyBuffer);
  Udp.write(sensorBuffer);
  Udp.endPacket();
```

La interfaz gráfica está determinada por el siguiente código:

```
function luminosidad_Callback(hObject, eventdata, handles)
axes(handles.axes3);
lumi=get(handles.luminosidad,'Value');
hume=get(handles.humedad,'Value');
temp=get(handles.temperatura,'Value');

if (lumi==1)
    cla;
    set(handles.axes3,'Visible','on');
    senso_luz=lumi;
    senso_hume=hume;
    senso_temp=temp;
    tres_sensores

else
    cla;
    set(handles.axes3,'Visible','on');
    senso_luz=0;
    senso_hume=hume;
    senso_temp=temp;
    tres_sensores

end

function humedad_Callback(hObject, eventdata, handles)
axes(handles.axes3);
lumi=get(handles.luminosidad,'Value');
hume=get(handles.humedad,'Value');
temp=get(handles.temperatura,'Value');
if hume==1
    cla;
    senso_hume=hume;
    senso_luz=lumi;
    senso_temp=temp;
    tres_sensores
else
    cla;
    senso_hume=0;
    senso_luz=lumi;
    senso_temp=temp;
    tres_sensores
end

function temperatura_Callback(hObject, eventdata, handles)
axes(handles.axes3);
lumi=get(handles.luminosidad,'Value');
```

```
hume=get(handles.humedad,'Value');
temp=get(handles.temperatura,'Value');
if temp==1
      cla;
      senso_temp=temp;
      senso_luz=lumi;
      senso_hume=hume;
      tres_sensores
else
      cla;
      senso_temp=0;
      senso_luz=lumi;
      senso_hume=hume;
      tres_sensores

end

function porcentaje_restante_Callback(hObject, eventdata, handles)

saber=get(hObject,'Value');
if (saber ==1)
    grafica_agua=1;
    sensores_riego

else
    grafica_agua=0;
    sensores_riego
end

function agua_SelectionChangeFcn(hObject, eventdata, handles)
if hObject== handles.a1
    valor_pausa=3;
    cantidad_riegoagua
elseif hObject== handles.a2
    valor_pausa=6;
    cantidad_riegoagua
elseif hObject== handles.a3
    valor_pausa=9;
    cantidad_riegoagua
else
     valor_pausa=0;

end
function fertilizante_SelectionChangeFcn(hObject, eventdata, handles)

 if hObject== handles.f1
    valor_pausa=3;
    cantidad_riegofertilizante
```

```
elseif hObject== handles.f2
    valor_pausa=6;
    cantidad_riegofertilizante
elseif hObject== handles.f3
    valor_pausa=9;
    cantidad_riegofertilizante
else
     valor_pausa=0;

 end

 function posicion_SelectionChangeFcn(hObject, eventdata, handles)
if hObject== handles.p0
    posicion=0;
elseif hObject== handles.p1
    posicion=1;
elseif hObject== handles.p2
    posicion=2;
elseif hObject== handles.p3
    posicion=3;
else
    posicion=4;

end

function camara_SelectionChangeFcn(hObject, eventdata, handles)

if hObject== handles.encender
    testear=1;
    capturasrapidas

else
    testear=0;
    capturasrapidas
    testear=0;

end

function cerrar_Callback(hObject, eventdata, handles)
close all
clc
```

Como se podrá observar posteriormente, el vector valor_sensores va llenando o más bien actualizando cada uno de los valores correspondientes a luminosidad en la posición 1, humedad en la posición 2, temperatura en la posición 3, Sensor de nivel de agua en la posición 4, sensor de nivel de fertilizante en la posición 5, final de carrera óptico en la 6, sensor de humedad en tierra en la posición 7 y 8.

La captura de luminosidad, temperatura y humedad se da de la siguiente manera

```
if not(isempty(instrfind))
    fclose(instrfind);
end

inicializar_comunicaciones;

vec1 =[];
vec2 =[];
vec3 =[];

valor_sensores_anterior = 0;
limite_datos = 100;

for i=1:10000
    valor_sensores=llamadosensot_rapido(u2);
    if valor_sensores == -7777
        valor_sensores = valor_sensores_anterior;
    else
        valor_sensores_anterior = valor_sensores;
    end
    vec1= [vec1 , valor_sensores(1)];
    vec2= [vec2 , valor_sensores(2)];
    vec3= [vec3 , valor_sensores(3)];

    if i >= limite_datos
        vec1(1)= [];
        vec2(1)= [];
        vec3(1)= [];
    end

    if (senso_luz==1)

        hold on
        plot(vec1,'yellow','LineWidth', 2);
        grid on
        pause(0.05);
    end
```

```
    if (senso_hume==1)
        hold on
        plot(vec2,'blue','LineWidth', 2);
        grid on
        pause(0.05);
        hold off
    end

    if (senso_temp==1)
        hold on
        plot(vec3,'red','LineWidth', 2);
        grid on
        pause(0.05);
    end

    if (senso_temp==0 && senso_luz==0 && senso_hume==0)

        axes(handles.axes2);
        cla;
        set(handles.axes2,'Visible','off');
        pause(100)

     end


end

finalizar_comunicaciones;
```

La adquisición de los valores de los sensores de nivel está determinados por el siguiente código:

```
if not(isempty(instrfind))
    fclose(instrfind);
end
inicializar_comunicaciones;
vec4 =[];
vec5 =[];
valor_sensores_anterior = 0;
limite_datos = 100;

for i=1:10000
    valor_sensores=llamadosensot_rapido(u2);
    if valor_sensores == -7777
```

```
        valor_sensores = valor_sensores_anterior;
    else
        valor_sensores_anterior = valor_sensores;
    end
    vec4= [vec4 , valor_sensores(4)];
    vec5= [vec5 , valor_sensores(5)];

    if i >= limite_datos
        vec4(1)= [];
        vec5(1)= [];
    end

    if (grafica_agua==1)
        axes(handles.axes4);
        cla;
        plot(vec4,'green','LineWidth', 2);
        grid on
        axes(handles.axes6);
        cla;
        plot(vec5,'magenta','LineWidth', 2);
        grid on
        pause(0.05);
    end


    if (grafica_agua==0)
        axes(handles.axes4);
        cla;
        set(handles.axes4,'Visible','off');
        axes(handles.axes6);
        cla;
        set(handles.axes6,'Visible','off');
        pause(100)
    end
end

finalizar_comunicaciones;
```

Es importante resaltar que los programas tres_sensores y sensores_riego tienen la estructura mencionada anteriormente, la cual pregunta y luego visualiza lo que se necesite.

Para el sistema de riego y teniendo en cuenta que el servidor recibe un 1 para agua o un 2 para fertilizante, el riego de cada sustancia se hace de la siguiente manera.

El sistema de riego se hace de la siguiente manera:

```
fprintf(u2, '0');
finalizar_comunicaciones;
```

- La captura de imágenes por una cámara se puede hacer a través de una ip o una webcam. El procesamiento para una ipcam es el siguiente:

```
if (testear==1)
    for i=1:1:100
    axes(handles.axes2);
    cam = imread('http://192.168.1.31:8080/photo.jpg');
    cam=imresize(cam,0.5);
    imshow(cam);
%     if (testear==0)
%         i=200;
%         axes(handles.axes2);
%         cla;
%         set(handles.axes2,'Visible','off');
%     end
    i=i+1;
    end

else
    axes(handles.axes2);
    cla;
    set(handles.axes2,'Visible','off');
    pause(100)

end
```

Para obtener el movimiento deseado para el riel se determina la siguiente codificación:

- El programa encargado de los movimientos del Dynamixel es el siguiente:

```
- function MOVNROB_RAPIDO( vecid, q, qp)
- nmotores = length(vecid);
- if ~((nmotores == length(q)) && (nmotores== length(qp)))
-     error('El tamano de los vectores es diferente');
- end
```

```
-   % % Load the dynamixel library
-   loadlibrary('dynamixel','dynamixel.h'); %carga la librerias
-   DEFAULT_PORTNUM = 5;  %COM5
-   DEFAULT_BAUDNUM = 1;  %1Mbps
-   response=calllib('dynamixel', 'dxl_initialize', DEFAULT_PORTNUM,
    DEFAULT_BAUDNUM);
-   if response == 1
-       disp('USB2Dynamixel! Conectado');
-
-       qm = int32(q*(1023/3000));
-       qpm = int32(qp*(1023/3000));
-
-       %Broadcast ID
-       calllib('dynamixel','dxl_set_txpacket_id', 200);
-
-       %Length is 14
-       %That handles position and speed for n dynamixels
-       %%(L + 1) * N + 4 (L: Data length for each Dynamixel actuator, N: The
-       %%number of Dynamixel actuators)
-       tamano_arreglo = (4+1)*nmotores+4;
-       calllib('dynamixel','dxl_set_txpacket_length',tamano_arreglo);
-
-       %SyncWrite instruction
-       calllib('dynamixel','dxl_set_txpacket_instruction',131);
-
-       %Starting address (goal position)
-       calllib('dynamixel','dxl_set_txpacket_parameter',0, 30);
-
-       %length of data to write to each dynamixel
-       %We're writing position and speed = 4 bytes
-       calllib('dynamixel','dxl_set_txpacket_parameter',1, 4);
-
-       %Parameters for syncwrite
-       % id | position | speed
-       %ID
-
-       for im=1:nmotores
-           %Parameters for syncwrite dynamixel id = im
```

```
-           calllib('dynamixel','dxl_set_txpacket_parameter',(im-
    1)*5+2,vecid(im) );
-           %position
-           lowByte = calllib('dynamixel','dxl_get_lowbyte', int32(qm(im)));
-           highByte = calllib('dynamixel','dxl_get_highbyte', int32(qm(im)));
-           calllib('dynamixel','dxl_set_txpacket_parameter',int32((im-1)*5+3),
    lowByte);
-           calllib('dynamixel','dxl_set_txpacket_parameter',int32((im-1)*5+4),
    highByte);
-           %Speed
-           lowByte = calllib('dynamixel','dxl_get_lowbyte', int32(qpm(im)));
-           highByte = calllib('dynamixel','dxl_get_highbyte', int32(qpm(im)));
-           calllib('dynamixel','dxl_set_txpacket_parameter',int32((im-1)*5+5),
    lowByte);
-           calllib('dynamixel','dxl_set_txpacket_parameter',int32((im-1)*5+6),
    highByte);
-       end
-
-       %transmit
-       calllib('dynamixel','dxl_tx_packet');
-
-   else
-       disp('Failed to open USB2Dynamixel!');
-   end
-   end
-
```

Esta codificación se encarga de mover es dispositivo de desplazamiento, indicándole variables como posición, id y velocidad, además de variar los valores de velocidad teniendo en cuenta que de 0 a 3000 el giro es en sentido horario y de 3001 a 6001, es en sentido contrario.

Los movimientos están determinados de la siguiente manera:

En sentido Horario:

```
qp=[2500];
q=[0];
vecid = [200] ;
MOVNROB_RAPIDO( vecid, q, qp)
revisar_posicion;
```

Donde qp es velocidad, q posición y vecid el id del Dynamixel.

En sentido contrario:

```
qp=[5501];
q=[0];
vecid = [200] ;
MOVNROB_RAPIDO( vecid, q, qp)
revisar_posicion;
```

Revisar_posicion se encarga de detener el robot por cada cambio en el sensor óptico.

```
if not(isempty(instrfind))
    fclose(instrfind);
end
inicializar_comunicaciones;
vec6 =[];
valor_sensores_anterior = 0;
limite_datos = 2;
revi=0;

for i=1:10000

    valor_sensores=llamadosensot_rapido(u2);
    if valor_sensores == -7777
        valor_sensores = valor_sensores_anterior;
    else
        valor_sensores_anterior = valor_sensores;
    end
    vec6= [vec6 , valor_sensores(6)];

    if i >= limite_datos
        vec6(1)= [];
    end
     revi=vec6(1)

     if (revi==0)

          MOV_DOWN
     else
          stop
     end

end
```

```
finalizar_comunicaciones;
```

CAPÍTULO 4

2 RESULTADOS:

a. PRUEBAS INICIALES:

Inicialmente, para poder verificar los valores correspondientes a los sensores, se hace un análisis de los sensores en Arduino IDE:

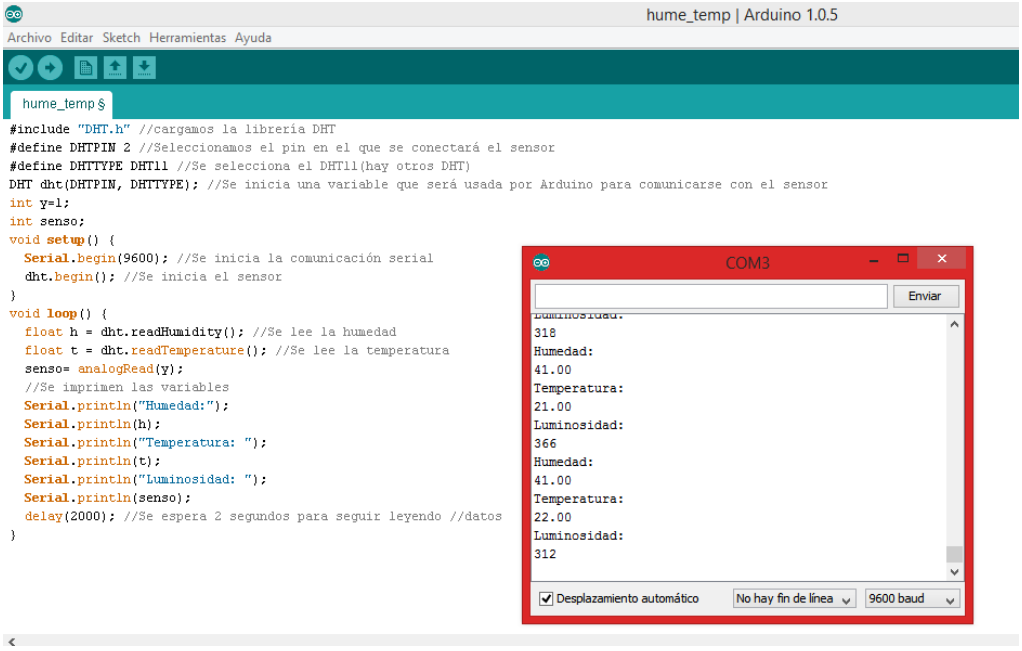

FIGURA 36. PRUEBA SENSORES DE LUMINOSIDAD Y DHT11

En el monitor serial se puede observar las variaciones en los valores de luminosidad debido a que la luz recibida no es constante.

Los Valores de humedad y temperatura se pueden ver que son constantes y al realizar la siguiente prueba, se observa como varían estos valores.

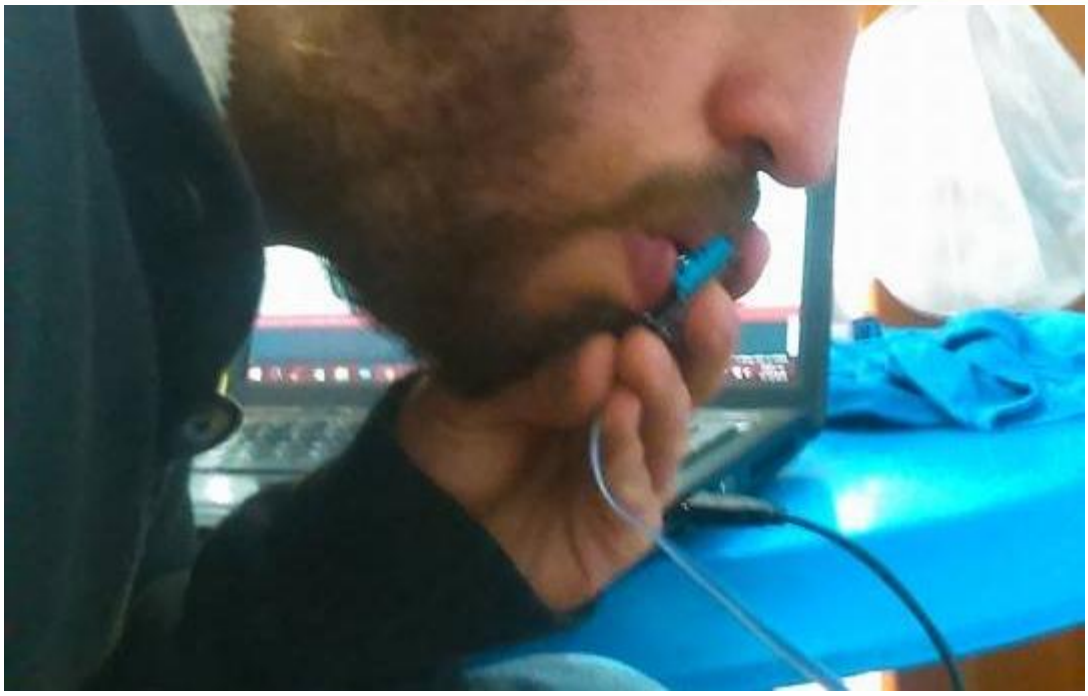

FIGURA 37. PRUEBA SENSORES DE LUMINOSIDAD Y DHT11 CON PERTURBACIONES

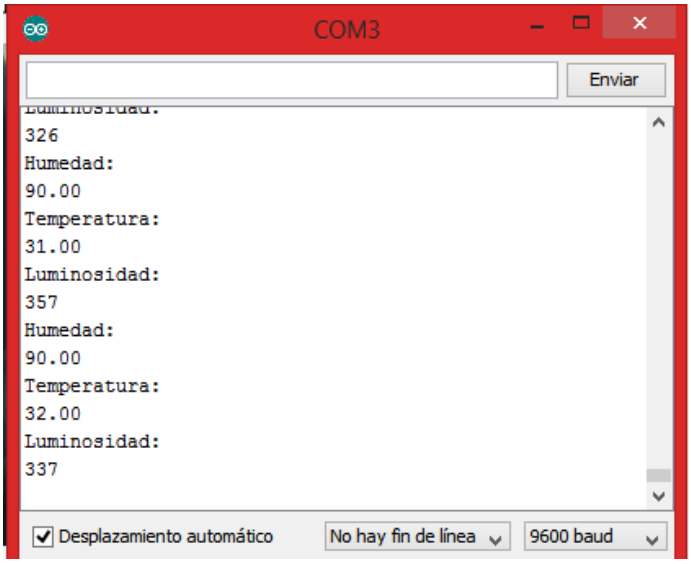

FIGURA 38. RESULTADO PRUEBA SENSORES DE LUMINOSIDAD Y DHT11 CON PERTURBACIONES

Como se observa en la gráfica, los parámetros Humedad y Temperatura varían considerablemente con la perturbación indicada en la figura 38.

Para los Sensores de Nivel, cuando los sensores no tienen líquido:

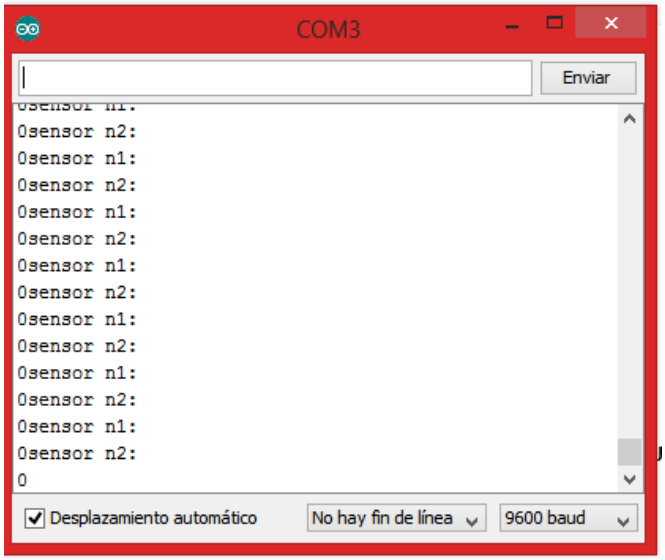

FIGURA 39. SENSORES DE NIVEL SIN PERTURBACION

En la figura se puede observar que los valores de los sensores es 0 teniendo en cuenta que no hay presencia de líquidos en los tanques

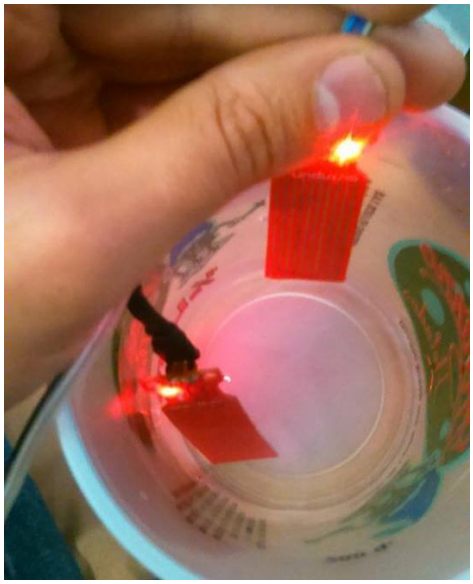

FIGURA 40. PRUEBA SENSORES DE NIVEL

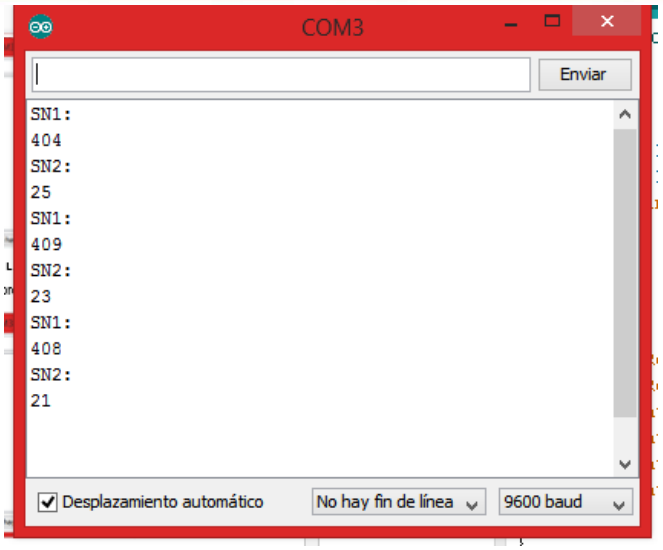

FIGURA 41. SENSORES DE NIVEL CON PERTURBACION

Como se puede explicar en el monitor serial de Arduino, las alteraciones en los sensores son evidente con la presencia de líquidos.

Ahora se procede a hacer la prueba del sensor óptico: en el que al haber ausencia de interferencia, se visualiza que su valor es 0.

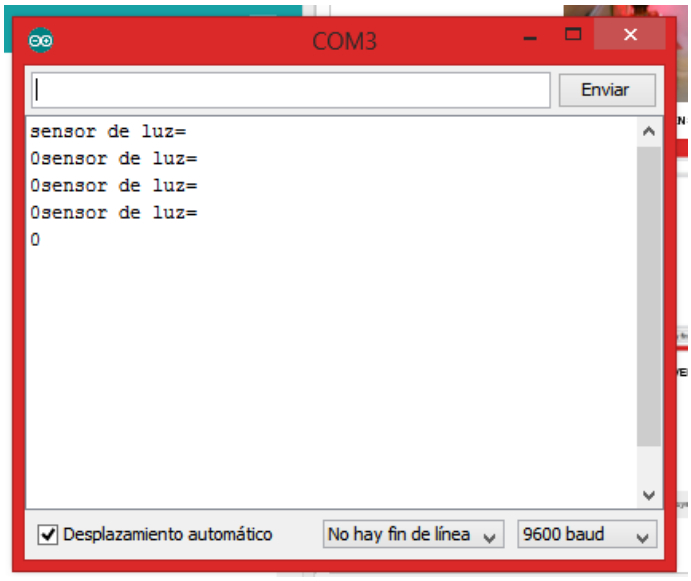

FIGURA 42. FINAL DE CARRERA ÓPTICO PERTURBACION

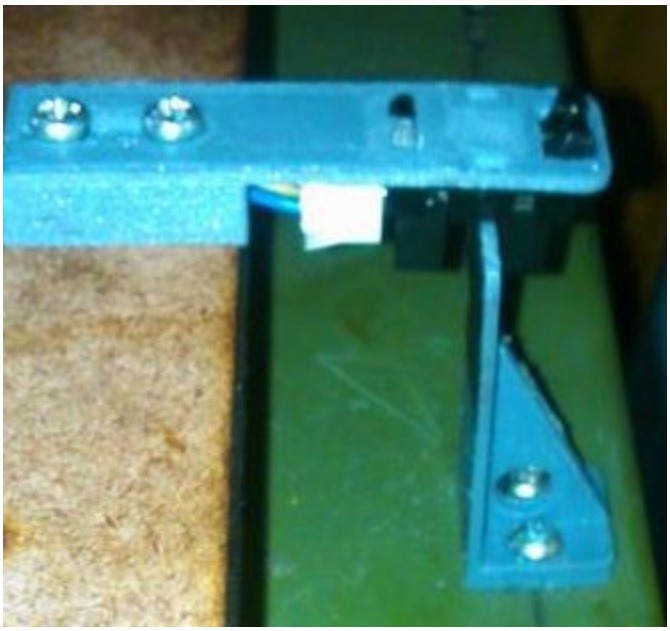

FIGURA 43. PRUEBA PERTURBACION FINAL DE CARRERA ÓPTICO

Al existir la perturbación, se puede observar que el valor es 1.

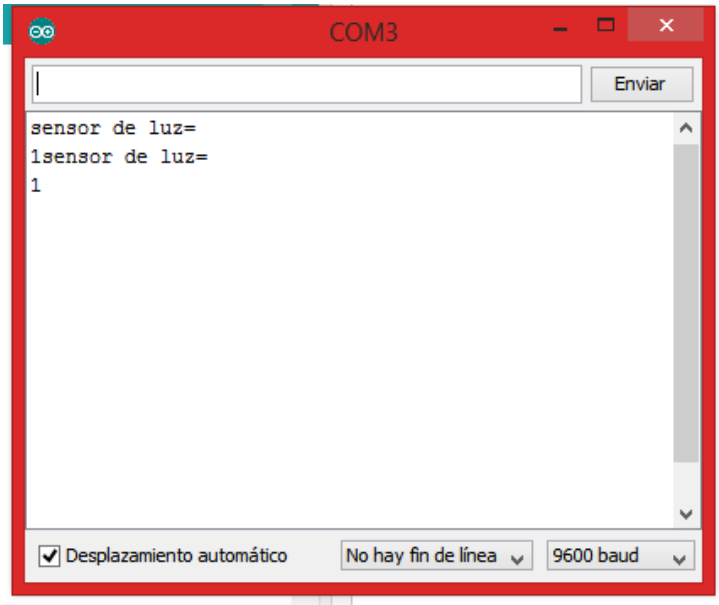

FIGURA 44. PRUEBA PERTURBACION FINAL DE CARRERA ÓPTICO

Prueba para sensor de humedad en tierra: Inicialmente se visualiza un 0, teniendo en cuenta que los sensores no se encuentran clavados en la tierra

FIGURA 45. SENSOR DE HUMEDAD EN TIERRA SIN PERTURBACIÓN

FIGURA 46. SENSOR DE HUMEDAD EN TIERRA CON PERTURBACIÓN

Al clavarse el sensor en la tierra, este extrae los valores que se ven a continuación:

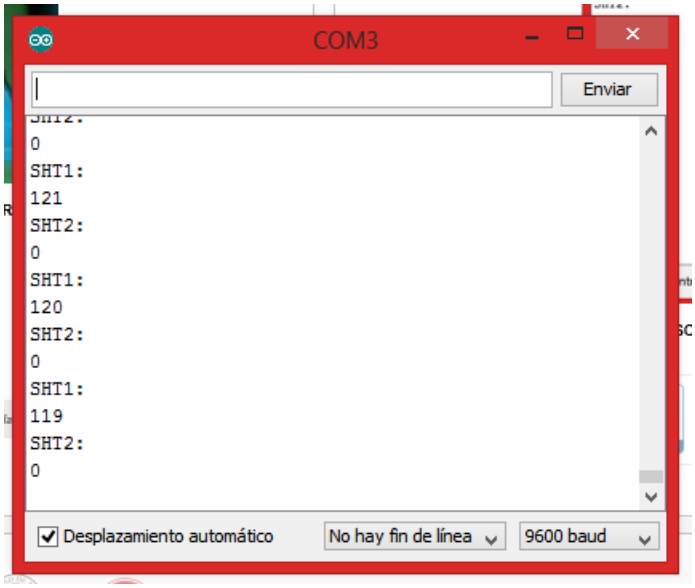

FIGURA 47. PRUEBA SENSOR DE HUMEDAD EN TIERRA

Para verificar que se pueda realizar la etapa de riego, se ejecuta el siguiente programa

El siguiente código me determina la activación del relé:

```
int pinOut = 2;
void setup() {
  // put your setup code here, to run once:
  //configuramos el pin 2 como salida
  pinMode (pinOut, OUTPUT);
}
void loop() {
  // put your main code here, to run repeatedly:
  //Le mandamos la señal de HIGH a la salida del
  //Arduino, la salida es el pin 2
  digitalWrite (pinOut, HIGH);
  //Hacemos un delay de 1 segundo
  delay (1000);digitalWrite (pinOut, LOW);
   delay (1000);
```

De esto se logra la activación de la válvula

FIGURA 48. ACTIVACIÓN DE VÁLVULA

Al obtener ya los parámetros se procede a comprobar la movilidad de la base sobre la que va el robot antropomórfico logrando resultados como los siguientes:

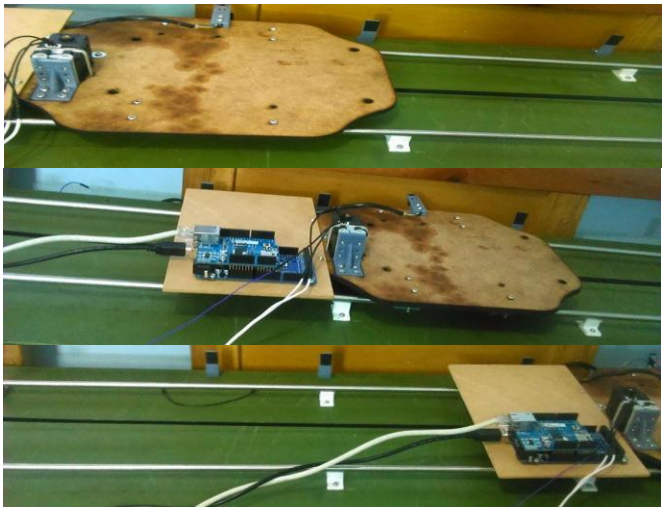

FIGURA 49. DESPLAZAMIENTO LINEAL

Ahora se procede a hacer la prueba de resistencia con peso al respectivo riel:

FIGURA 50. DESPLAZAMIENTO LINEAL CON EL ROBOT

En la figura anterior se puede observar el robot sobre la base antes del desplazamiento

FIGURA 51. DESPLAZAMIENTO LINEAL CON EL ROBOT YA DESPLAZADO

Como se puede observar, se da el desplazamiento del robot, lo que indica que el riel ha pasado la prueba de peso.

b. PRUEBAS CON LA INTERFÁZ GRÁFICA:

✓ Inicialmente hay que hacer la respectiva configuración de la red Wifi.

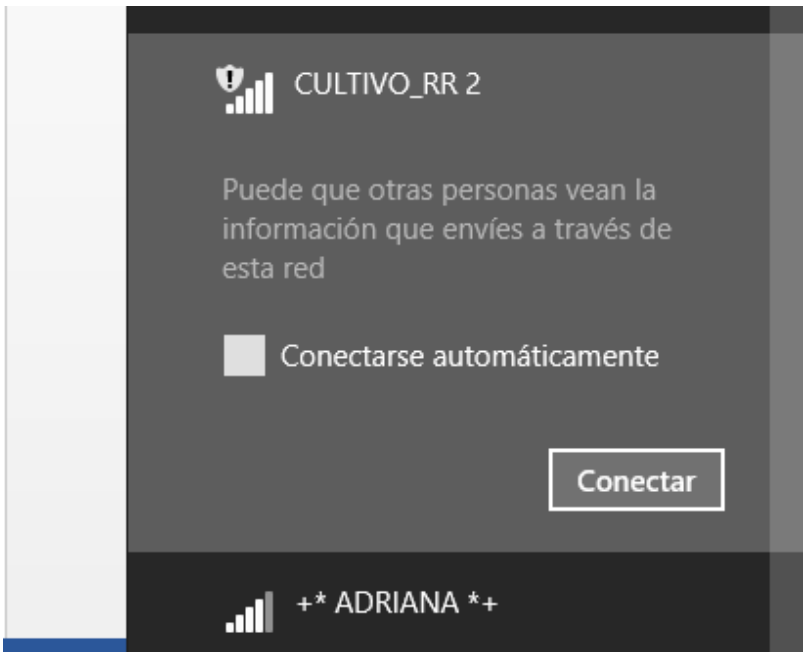

FIGURA 52. RED A LA QUE ME DEBO CONECTAR

✓ Para hacer una prueba de comunicación, se ejecuta el siguiente código en Matlab

```
if not(isempty(instrfind))
    fclose(instrfind);
end

u2 = udp('192.168.1.177', 'RemotePort', 8888, 'LocalPort', 80,
'TimeOut', 5); %La dirreccion ip debe ser la del servidor
fopen(u2) % Abre el puerto del cliente
vec1 =[];
vec2 =[];
vec3 =[];
vec4 =[];
valor_sensores_anterior = 0;
```

```
limite_datos = 100;
for i=1:10000
    valor_sensores=llamadosensot_rapido(u2);
    if valor_sensores == -7777
        valor_sensores = valor_sensores_anterior;
    else
        valor_sensores_anterior = valor_sensores;
    end
    vec1= [vec1 , valor_sensores(1)];
    vec2= [vec2 , valor_sensores(2)];
    vec3= [vec3 , valor_sensores(3)];
    vec4= [vec4 , valor_sensores(4)];
    if i >= limite_datos
        vec1(1)= [];
        vec2(1)= [];
        vec3(1)= [];
        vec4(1)= [];
    end
    f1=figure(1);clf;
    plot(vec4);
%      f2=figure(2);clf;
%      plot(vec2);
%      f3=figure(3);clf;
%      plot(vec3);
    pause(0.05);
end

fclose(u2)   % Cierra el cliente
delete(u2)   % borra el cliente
clear u2
```

A continuación se obtienen las siguientes gráficas:

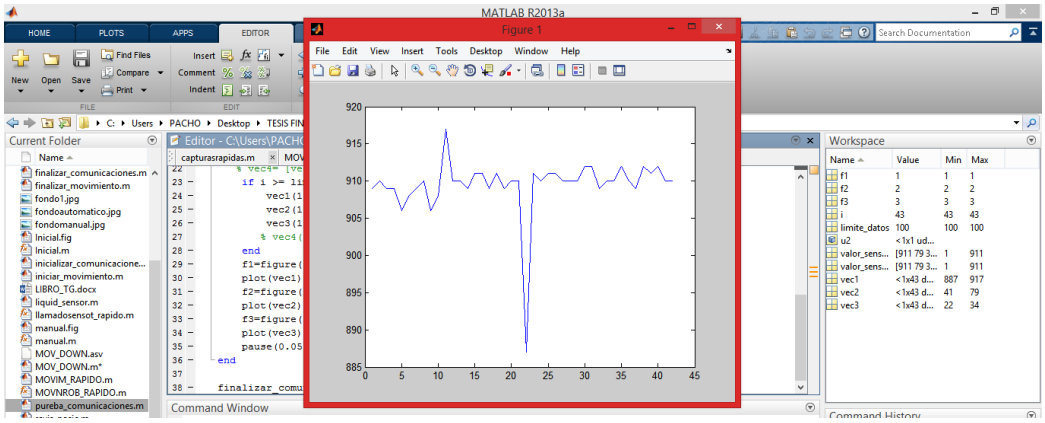

FIGURA 53. LUMINOSIDAD Vs TIEMPO

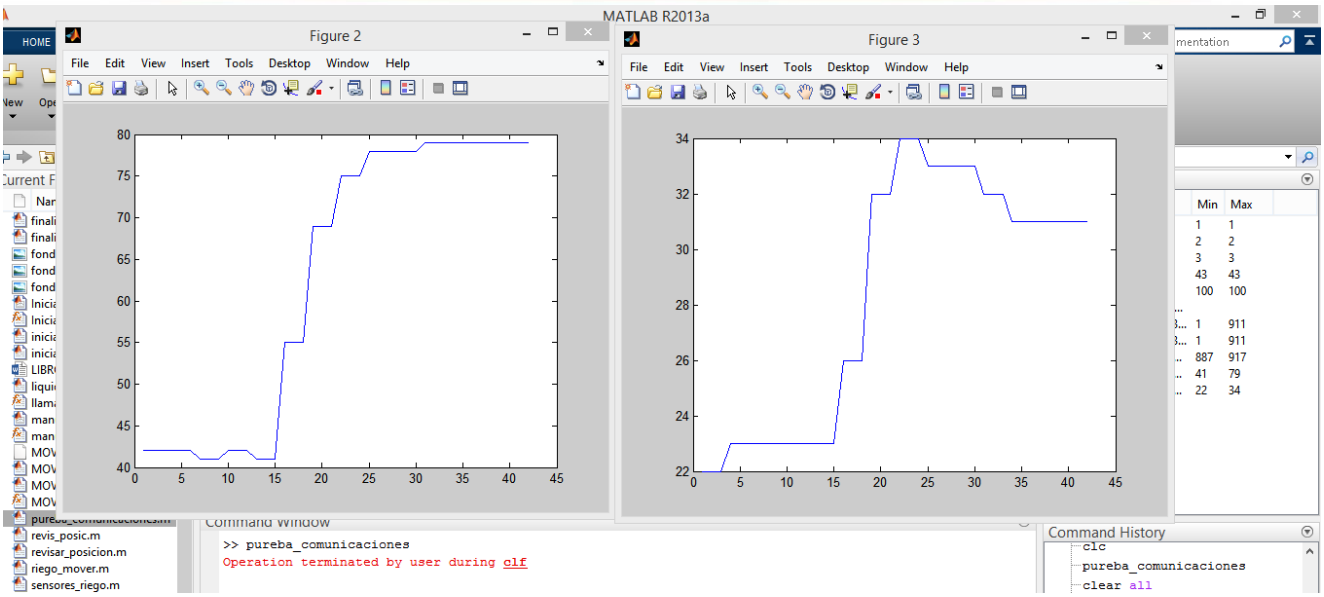

FIGURA 54. HUMEDAD Y TEMPERATURA Vs TIEMPO

Con las anteriores gráficas se puede comprobar que si existe una comunicación adecuada entre el servidor y el cliente.

Ahora se procede a realizar el control de los dispositivos a través de una interfaz gráfica, logrando los siguientes resultados:

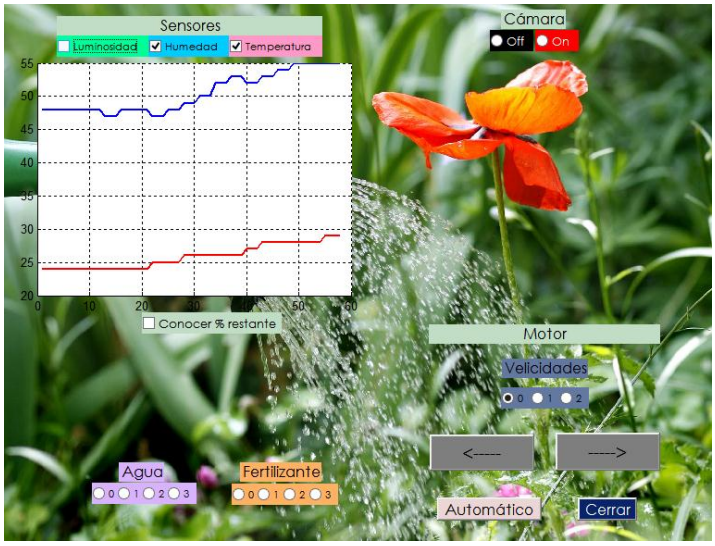

FIGURA 55. HUMEDAD Y TEMPERATURA Vs TIEMPO

En la anterior figura se puede observar las variaciones de la humedad y la temperatura en tiempo real.

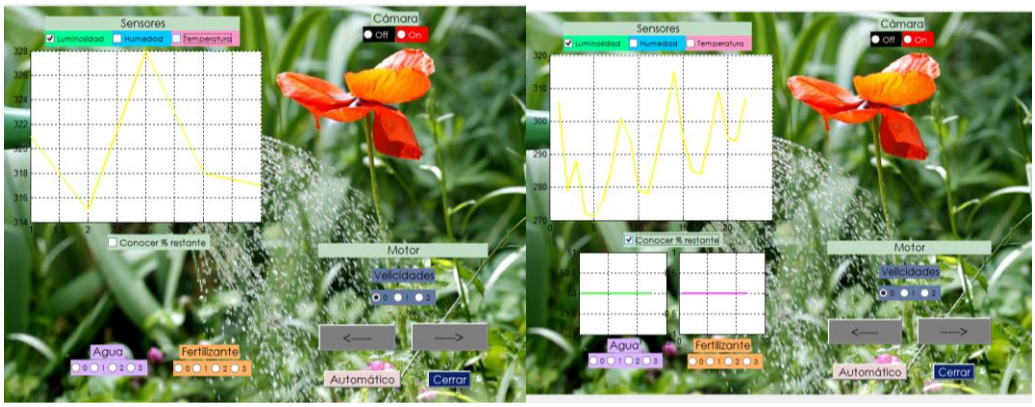

FIGURA 56. LUMINOSIDAD Vs TIEMPO

FIGURA 57. SENSORES DE NIVEL DEL TANQUE

En la anterior figura se puede observar las variaciones de la luminosidad y el nivel en los tanques en tiempo real.

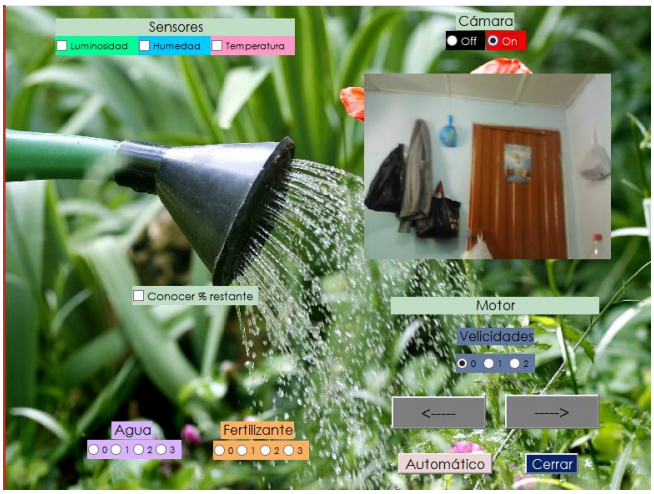

FIGURA 58. ACTIVACIÓN DE LA CÁMARA.

En la anterior figura se puede observar la visualización de una cámara ip.

Teniendo en cuenta la importancia de las conexiones con el Arduino, se decide realizar una tarjeta que permita tales interconexiones de manera adecuada.

En la siguiente figura se verá el esquema del circuito:

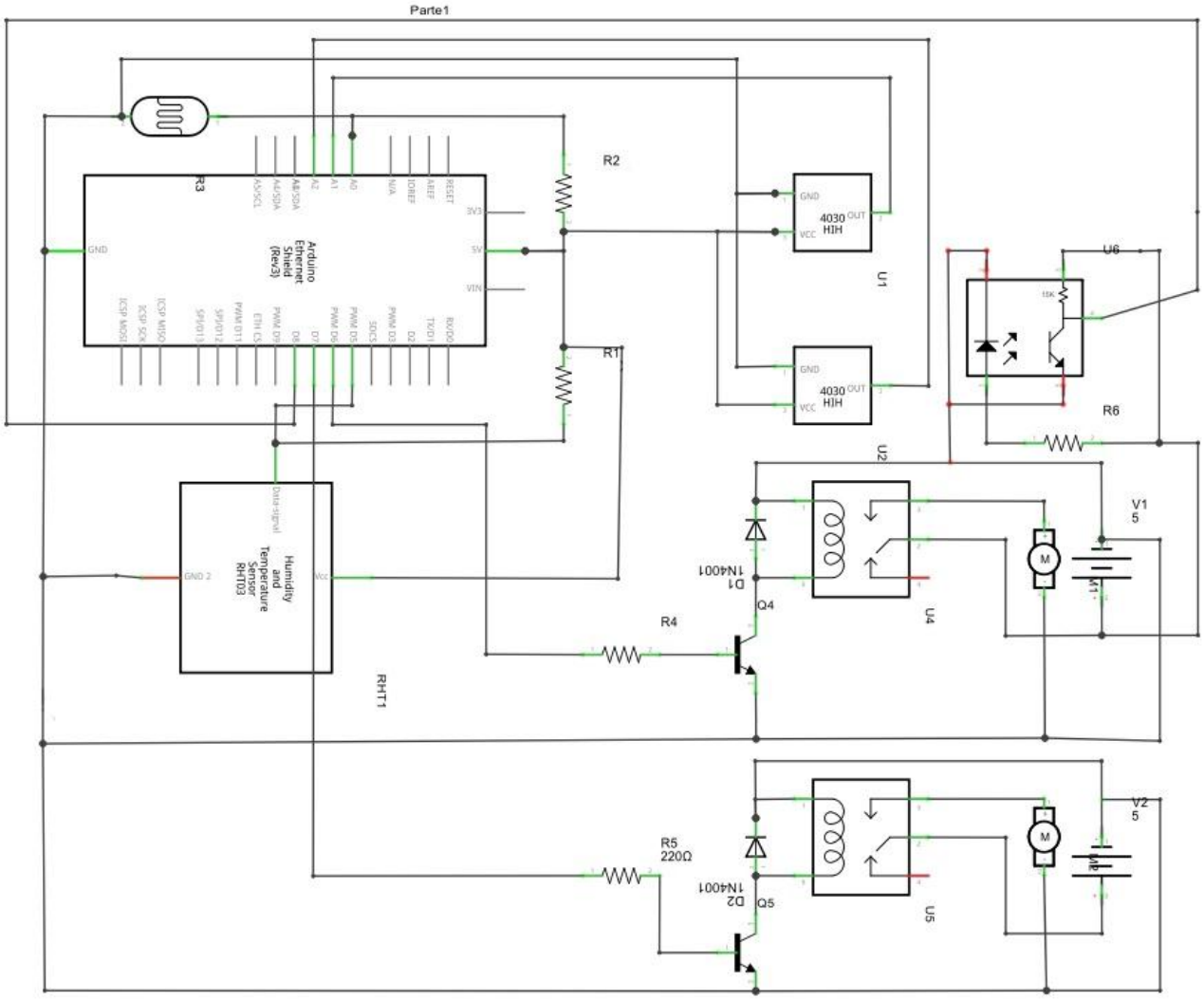

FIGURA 59. DIAGRAMA ELECTRÓNICO.

Como se puede observar en el circuito anterior, se destaca la conexión de cada elemento al Arduino, lo que nos permite tener como resultado la siguiente PCB:

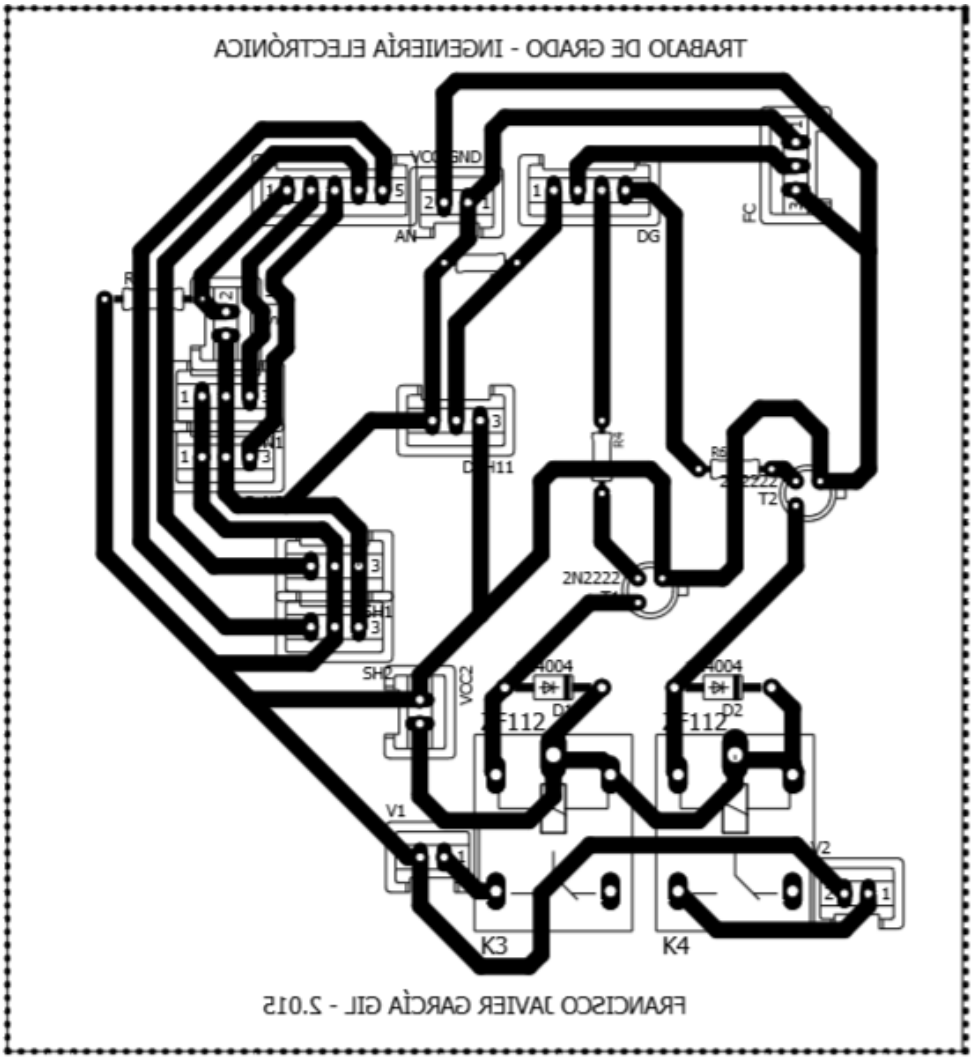

FIGURA 60. PCB TARJETA DE DESARROLLO

Ya en un plano real, la tarjeta de desarrollo es la siguiente:

FIGURA 61. TARJETA DE DESARROLLO REAL

Para finalizar, se presenta la totalidad de los dispositivos desarrollados en funcionamiento.

Teniendo en cuenta que los dispositivos desarrollados sirven de base para el proyecto cultivo robótico remoto, se presentan las siguientes imágenes con el dispositivo de desplazamiento lineal funcionando, lo mismo que el sistema de riego.

Las siguientes imágenes son tomadas del informe de este proyecto:

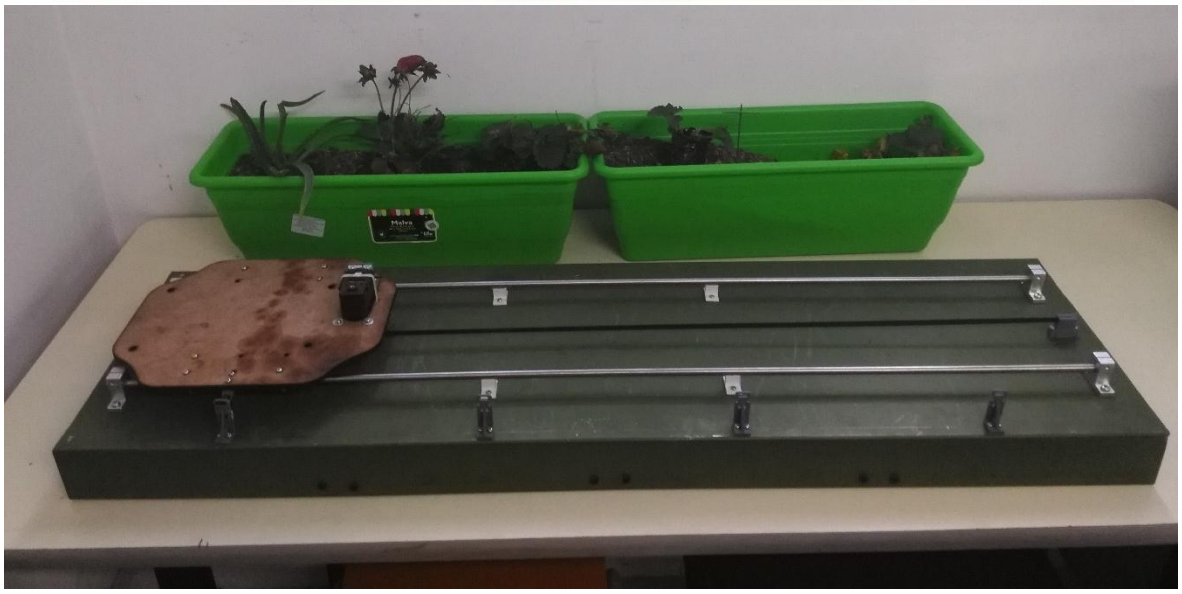

FIGURA 62. PRUEBA DE MOVIMIENTO SOBRE EL RIEL

En la figura anterior se observa el riel con las materas correspondientes.

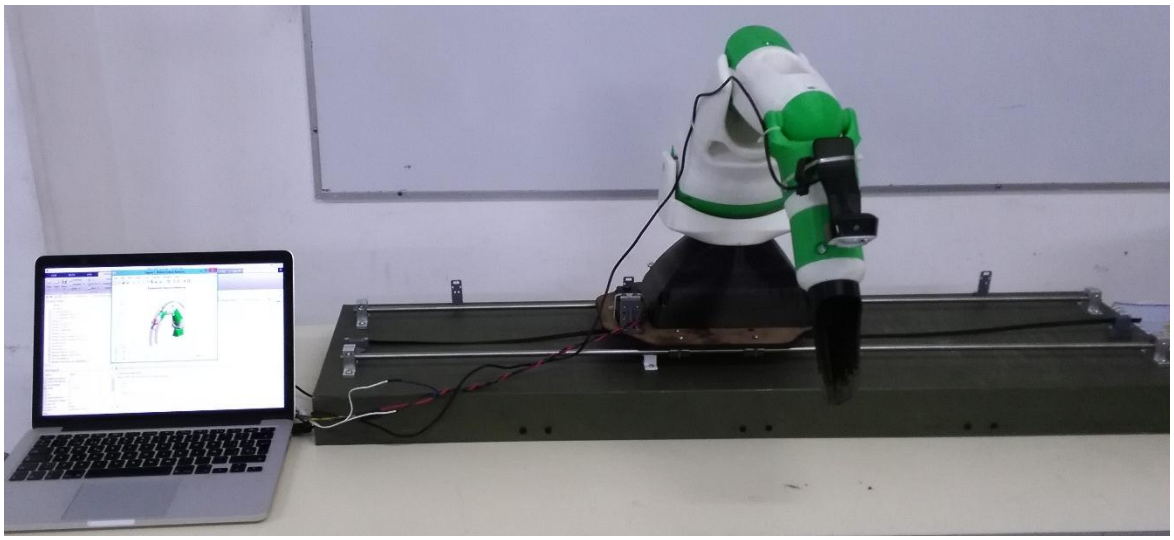

FIGURA 63. PRUEBA DE MOVIMIENTO SOBRE EL RIEL CON EL ROBOT

En la figura anterior se realiza la prueba de movimiento del robot sobre el riel.

FIGURA 64. ROBOT AGRICOLA

Se puede observar en esta figura al robot en movimiento sobre el riel, ubicándose para realizar el riego.

FIGURA 65. RIEGO ROBOT AGRÍCOLA

En esta figura se puede ver el prototipo final desarrollado, realizando el riego de las plantas.

5. CONCLUSIONES:

- ✓ Este proyecto es de gran relevancia ya que teniendo en cuenta las variables obtenidas y conociendo las condiciones o los requerimientos que tenga una planta, se puede tomar la mejor decisión para que haya un buen desarrollo de la misma.
- ✓ Se logró el acceso remoto para la adquisición de las variables que intervienen en el cultivo.
- ✓ Se logra desarrollar los diferentes dispositivos de apoyo para el proyecto cultivo robótico remoto tales como la base para desplazar linealmente el robot, y como la monitorización de las diferentes variables físicas.
- ✓ El desarrollo de la interfaz gráfica con el control de todos los dispositivos se hizo de forma adecuada, mostrando resultados satisfactorios.
- ✓ Se logra obtener resultados satisfactorios en cuanto al posicionamiento del robot ya que la comunicación a través del USB2Dynamixel es apropiada.
- ✓ El sistema de riego que se adaptó al robot cumple satisfactoriamente los requerimientos del diseño.
- ✓ El proyecto cultivo robótico remoto abre las puertas para empezar a implementar en el área de la agricultura los conocimientos aprendidos durante el diplomado.
- ✓ La implementación de los sensores de presencia de agua y de humedad en tierra de Arduino, teniendo en cuenta su buena respuesta son recomendados para ésta y futuras prácticas.
- ✓ Las pruebas realizadas en los dispositivos nos muestran que los materiales utilizados fueron adecuados y permiten obtener resultados satisfactorios.

BIBLIOGRAFÍA

[1]ALEX BARRIENTOS. Automatización en la granja: bienvenido a la década de la agricultura robot [en línea]. < http://hipertextual.com/2015/11/agricultura-robot > [citado el 30 de Noviembre de 2015]

[2]Protocolo UDP [en línea]. <http://es.ccm.net/contents/284-protocolo-udp>. Noviembre de 2015.
[3]Diferencias entre los protocolos UDP y TPC [en línea]. <http://es.ccm.net/faq/1559-diferencias-entre-los-protocolos-tcp-y-udp>. Noviembre de 2015.
[4] Arduino MEGA 2560 [en línea]. <https://www.arduino.cc/en/Main/arduinoBoardMega2560#>. Noviembre de 2015.
[5] Getting Started w/ Arduino on Mac OS X[en línea]. <https://www.arduino.cc/en/Guide/MacOSX>. Noviembre de 2015.
[6] Arduino Ethernet Shield[en línea]. <https://www.arduino.cc/en/Main/ArduinoEthernetShield>. Noviembre de 2015.
[7] Arduino Ethernet Shield [en línea]. <http://www.tuelectronica.es/tutoriales/arduino/arduino-ethernet-shield.html>. Noviembre de 2015.
[8] USB2Dynamixel [en línea]. <http://support.robotis.com/en/product/auxdevice/interface/usb2dxl_manual.htm>.Noviembre de 2015.
[9] Eyectores de agua del lavaparabrisas [en línea].<http://www.actualidadmotor.com/eyectores-de-agua-del-lavaparabrisas>. Noviembre de 2015.
[10] DHT11: Sensor de humedad/temperatura para Arduino [en línea]. <http://panamahitek.com/dht11-sensor-de-humedadtemperatura-para-arduino/>. Noviembre de 2015.
[11] LDR - Resistencia dependiente de la luz – Fotoresistencia [en línea]. <http://electronica-electronics.com/info/LDR-fotoresistencia.html>. Noviembre de 2015.
[12] La agro-robótica, un concepto que va camino a la consolidación [en línea]. <http://contextoganadero.com/ganaderia-sostenible/la-agro-robotica-un-concepto-que-va-camino-la-consolidacion>. Noviembre de 2015.
[13] En Argentina crean robot todo terreno para cultivos intensivos [en línea]. <http://www.contextoganadero.com/internacional/en-argentina-crean-robot-todo-terreno-para-cultivos-intensivos>. Septiembre de 2015.

[14] El tema de tecnología Agricultura Urbana empieza a abrir campo[en línea].<http://www.larepublica.co/el-tema-de-tecnolog%C3%ADa-agricultura-urbana-empieza-abrir-campo_155796>. Septiembre de 2015.