

ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

**Facultad de Ingeniería en Mecánica y Ciencias de la
Producción**

"Diseño de un robot móvil para riego automático de plantas"

PROYECTO INTEGRADOR

Previo la obtención del Título de:

Ingeniero Mecánico

Presentado por:

Kevin Billy León Suarez

Lorena Lisseth Montes Ortega

GUAYAQUIL - ECUADOR

Año: 2018

DEDICATORIA

Este trabajo lo dedico principalmente a mis abuelos que desde el cielo me han guiado en este arduo camino; a mi familia y amigos.

Lorena Lisseth Montes Ortega

AGRADECIMIENTOS

Agradezco a mis padres por todo su apoyo incondicional a lo largo de este camino, a las grandes amistades formadas en estos cinco años de universidad que me han ayudado a seguir a delante en todo momento, por último, a las amistades que perduran desde la niñez y te alientan a seguir adelante.

Lorena Lisseth Montes Ortega.

AGRADECIMIENTOS


Agradeciendo primeramente a Dios por la bendición de ayudarme a cumplir este gran objetivo de mi vida, a mis padres porque gracias a sus esfuerzos y apoyo he podido superarme, a familiares y amigos por su apoyo incondicional en todo momento durante toda mi formación profesional.

Kevin Billy León Suárez.

DECLARACIÓN EXPRESA

"Los derechos de titularidad y explotación, nos corresponde conforme al reglamento de propiedad intelectual de la institución; **Kevin Billy León Suárez** y **Lorena Lisseth Montes Ortega** y damos nuestro consentimiento para que la ESPOI realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"



Kevin Billy León Suárez


Lorena Lisseth Montes
Ortega

EVALUADORES



Francis Loayza
PROFESOR DE LA MATERIA



Efraín Terán, M.Sc
PROFESOR TUTOR

RESUMEN

El elevado crecimiento de la población mundial en los últimos años genera una mayor demanda de alimentos que debe ser cubierto por el campo agrícola, por lo tanto, se proyecta el uso de robots móviles que agilicen el proceso de agricultura. Se construyó un robot automatizado para riego de plantas en base a dos matrices de decisión: diseño y comunicación inalámbrica, para lo cual se usó el Lego Mindstorms EV3 con sus respectivos sensores, además de otros sensores adicionales. Se realizó la adquisición de datos mediante una tarjeta Arduino UNO, se adaptó un reservorio y una bomba pequeña que trabaja con una batería de 9 V. Para los resultados se realizaron tres pruebas: la primera consistió en la comunicación planta-robot, la segunda el movimiento del robot y la última para la bomba. El robot logró ser estable y transportarse con el peso extra del reservorio-bomba y circuito, además de moverse con facilidad en superficies lisas e irregulares.

Palabras Clave: Mindstorms EV3, Riego, Agricultura, Robótica.

ABSTRACT

The high growth of the world population in recent years generates a greater demand for food that must be covered by the agricultural field, therefore, the use of mobile robots is projected to speed up the agricultural process. An automated robot was built to irrigate plants based on two decision matrices: design and wireless communication, for which the Lego Mindstorms EV3 was used with their respective sensors, in addition to other additional sensors. The data was acquired by means of an Arduino UNO card, a reservoir and a small pump that works with a 9 V battery were adapted. For the results, three tests were carried out: the first consisted of the plant-robot communication, the second the movement of the robot and the last one for the pump. The robot managed to be stable and transported with the extra weight of the reservoir-pump and circuit, in addition to moving easily on smooth and irregular surfaces.

Keywords: *Mindstorms EV3, Irrigation, Agriculture, Robotics.*

ÍNDICE GENERAL

RESUMEN.....	I
ABSTRACT	II
ÍNDICE GENERAL	III
ABREVIATURAS.....	V
SIMBOLOGÍA	VI
ÍNDICE DE FIGURAS	VII
ÍNDICE DE TABLAS.....	VIII
ÍNDICE DE PLANOS.....	IX
CAPÍTULO 1.....	1
1. Introducción	1
1.1 Descripción del problema	2
1.2 Justificación del proyecto	2
1.3 Objetivos	3
1.3.1 Objetivo General.....	3
1.3.2 Objetivos Específicos.....	3
1.4 Marco teórico.....	4
1.4.1 Robot MINDSTORMS EV3	4
CAPÍTULO 2.....	7
2. Metodología	7
2.1 Procedimiento esquemático	7
2.2 Requerimientos del diseño	8
2.3 Identificación y selección de las alternativas	9
2.3.1 Alternativas para el prototipo	9
2.3.2 Alternativas para el sistema de comunicación (Módulo inalámbrico).....	11

2.4	Diseño Conceptual	14
2.5	Diseño detallado.....	15
2.5.1	Diseño detallado del reservorio y bomba.....	15
2.5.2	Diseño detallado del sistema de control de posicionamiento y riego	16
2.5.3	Programación del equipo	21
2.6	Pruebas.....	25
2.6.1	Prueba A: Comunicaciones	25
2.6.2	Prueba B: Bomba.....	26
2.6.3	Prueba C: Movimiento del prototipo.....	27
CAPÍTULO 3.....		28
3.	Resultados Y ANÁLISIS	28
3.1	Resultados	28
3.1.1	Resultados del diseño del reservorio y bomba	28
3.1.2	Conexiones y códigos.....	29
3.1.3	Ensamble de los componentes del robot.....	32
3.1.4	Resultados de pruebas.....	35
3.2	Análisis de costos.....	41
CAPÍTULO 4.....		43
4.	Conclusiones Y RECOMENDACIONES.....	43
	Conclusiones.....	43
	Recomendaciones.....	43
BIBLIOGRAFÍA		
APÉNDICES		

ABREVIATURAS

PWM	Modulación por ancho de pulsos (pulse width modulation)
Tx	Transmisor
Rx	Receptor

SIMBOLOGÍA

Hz	Hertzios
Pw	Potencia Hidráulica
Q	Caudal
Pf	Potencia del Motor
η	Eficiencia
ρ	Densidad
g	Gravedad
∇	Volumen
ft	Pies
lt	Litros
V	Voltaje
I	Corriente
H	Altura
m	Metro
Ncm	Newton centímetros
KB	Kilobyte
bps	Bits por segundos
dBm	Decibelios milivattios

ÍNDICE DE FIGURAS

Figura 1.1. Sensores y componentes del LEGO MINDSTORMS EV3	5
Figura 2. 1. Proceso de diseño esquemático para el prototipo.....	8
Figura 2. 2. Sketch de la alternativa A.....	9
Figura 2. 3. Sketch de la alternativa B.....	10
Figura 2. 4. Diseño conceptual del robot móvil para riego.....	14
Figura 2. 5. Diagrama de flujo del sistema de control de posicionamiento.....	17
Figura 2. 6 Diagrama de flujo para código del Arduino A	22
Figura 2. 7. Diagrama de flujo para el código del Arduino B.....	23
Figura 2. 8. Diagrama de flujo para el código del Brick EV3.....	24
Figura 3. 1. Modelo 3D del reservorio para agua.....	28
Figura 3. 2. Imagen de reservorio y bomba ensamblados.....	29
Figura 3. 3. Esquema de conexión del Arduino A (Emisor)	30
Figura 3. 4 Esquema de conexión para el Arduino B (receptor)	31
Figura 3. 5 Esquema de conexión de los sensores del EV3	31
Figura 3. 6. Ensamble de los motores con el Brick y los sensores.....	32
Figura 3. 7. Ensamble de la canasta donde se coloca el reservorio bomba.....	33
Figura 3. 8. Cajas para los dispositivos electrónicos	33
Figura 3. 9. El robot con todos sus componentes.....	34
Figura 3. 10. Señal de “l” y “h” que llega al receptor.....	37
Figura 3. 11. Señal del higrómetro que es enviada por los XBees a los Arduinos	37
Figura 3. 12. Señal del Arduino → Brick.....	38
Figura 3. 13. Señal del Brick → Arduino.....	39
Figura 3. 14. Gráfica de Cabezal vs Caudal de la bomba	40
Figura 3. 15. Curva de trabajo de la bomba (Potencia hidráulica vs Revoluciones)..	40

ÍNDICE DE TABLAS

Tabla 1. 1. Especificaciones del Large motor	6
Tabla 1. 2. Especificaciones del Midium motor.....	6
Tabla 2. 1. Criterios para el prototipo con la respectiva ponderación.....	10
Tabla 2. 2. Matriz de selección para la mejor alternativa de prototipo	11
Tabla 2. 3. Especificaciones de cada alternativa	13
Tabla 2. 4. Criterios para el módulo inalámbrico con la respectiva ponderación.	13
Tabla 2. 5. Matriz de selección para la mejor alternativa de módulo inalámbrico.....	14
Tabla 2. 6. Especificaciones de diseño para el reservorio.....	15
Tabla 2. 7. Características técnicas del Digi XBee	19
Tabla 2. 8. Especificaciones de la tarjeta ARDUINO UNO	20
Tabla 2. 9. Especificaciones del sensor HC-SR04	21
Tabla 3. 1. Pruebas realizadas para la comunicación entre XBee.....	35
Tabla 3. 2. Costo del circuito del Robot	41
Tabla 3. 3. Costo del circuito de la planta.....	42

ÍNDICE DE PLANOS

PLANO 1 Ensamble y plano del reservorio.

PLANO 2 Ensamble y plano de las cajas de circuito.

CAPÍTULO 1

1. INTRODUCCIÓN

La robótica ha presentado un gran avance en los últimos años, hoy en día se considera como una de las áreas de mayor interés de la ciencia y la tecnología. Su principal desarrollo se ha dado al servicio de la automatización de industrias avanzadas, de manera especial la automotriz y la Aero-espacial, pero recientemente se la encuentra en casi todos los rubros de la actividad económica, desde actividades cotidianas en el hogar hasta servicios de rescate. A su vez de manera paralela los avances en las tecnologías de materiales, inteligencia artificial, sensores, servo-mecanismos, posicionamiento satelital y telecomunicaciones han contribuido de forma directa a la creación de técnicas y dispositivos para solucionar una infinidad de problemas de formas más precisas, seguras, en menor tiempo y haciendo uso de menos recursos.

Actualmente los robots son capaces de realizar una diversidad de trabajos y desempeñar funciones específicas, ya sean tareas peligrosas, repetitivas o de difícil acceso para los seres humanos. Por ejemplo, son utilizados en el monitoreo de actividades volcánicas, exploración en otros planetas, estudiar comportamientos en aguas profundas, etc.

En las últimas décadas la robótica móvil se ha involucrado en el campo de la agricultura creando así la agrorobotic, área que investiga y busca la eficiencia en el uso de los principales recursos de la agricultura por medio de la implementación de robots móviles. Existen ya algunos robots móviles que se encuentran trabajando en el campo. Como ejemplo se tiene el VineRobot, este es un robot que ha revolucionado la industria del vino desde el 2016, consta de sensores no invasivos que permiten obtener y enviar la información sobre el viñero con gran precisión, determinando que zona del viñero debe ser regada y cuanto de agua. Así como este, existen otros robots que determinan que nutrientes hacen falta en el cultivo y prepara la mezcla y lo surten, también encontramos drones que cumplen diferentes funciones como esparcir pesticida o controlar el estado de las hectáreas.

Se han diseñado y construido numerosos robots móviles autónomos como instrumentos para la introducción de la robótica y mecatrónica, tales como robots de juguete basados en Merccano (Parkin, 2002), los robots LEGO Mindstorms (Erwin, Cyr, & Rogers, 200; Hirst, Johnson, Petre, Price, & Richards, 2003), vehículos comerciales de pequeña escala (Buju. 2008), y desarrollo de equipos aficionados (Das, Yost, & Krishman, 2010).

El término en inglés Agrobotics es abreviatura de agricultural robot o agricultural robotics y aparece en artículos de revistas especializadas, noticias, blogs, workshops e incluso en alguna empresa utilizado como nombre comercial.

Agrobotics se refiere al trabajo en el campo agrícola de sistemas incorporados con sensores y actuadores cuya operación puede ser autónoma o semiautónoma en cooperación con el ser humano. Tenemos algunos desarrollos que han marcado tendencia en el sector de producción primaria: (Bosch, 2013)

- Robotización de tambos (alimentación, ordeño, limpieza, etc.)
- Invernáculos automatizados/inteligentes
- Logística en grandes instalaciones (viveros, invernáculos, hidro/aeroponia)
- Máquinas autónomas y de precisión (tractores, cosechadoras, podadoras, fumigadoras, desmalezadoras, etc.)
- Robots de vigilancia (depredadores, plagas)
- Riego inteligente

Para este proyecto se trabaja con el equipo LEGO Mindstorms EV3, el cual es un kit robótico de piezas ensamblables y sensores acoplables con un dispositivo programable llamado bloque EV3 que permite programar variedad de acciones y configuraciones, las cuales se programan en el bloque EV3 y/o software.

1.1 Descripción del problema

Debido a la necesidad de obtener mayor producción a menor tiempo en el campo agrícola, la implementación de las áreas de automatización y robótica

en los últimos años han sido de gran ayuda para tener mayor eficiencia. Por lo que el diseño de un robot móvil con capacidad de moverse a través de una superficie irregular evadiendo obstáculos y que pueda surtir agua a cualquier tipo de plantas ayudará a procesos agrícolas.

1.2 Justificación del proyecto

En las últimas décadas el avance de la robótica ha permitido métodos innovadores que les permiten intervenir en amplios campos y realizar diversas actividades. Por ejemplo, actualmente el concepto de agricultura precisión es de gran importancia debido que se requiere cubrir todas las necesidades de las plantaciones de forma más personalizada. La aplicación de la robótica y automatización en el área agrícola permite brindar esa atención “personalizada” que los cultivos solicitan.

En todo proceso se busca reducir costos de producción y utilizar los recursos con mayor eficiencia. Los sistemas de riego actuales presentan precios muy elevados de instalación o mantenimiento, y pueden desperdiciar grandes cantidades de agua.

Debido a esto surge la necesidad de realizar un robot de riego a bajo costo capaz de regar plantas de manera que racionalice el uso de recursos hídricos.

1.3 Objetivos

1.3.1 Objetivo General

Adaptar un sistema de riego agrícola a un robot móvil, para que pueda regar plantas en terrenos irregulares.

1.3.2 Objetivos Específicos

- Construir el robot de riego con las piezas de MINDSTORMS EV3 de la forma que más favorezca al proyecto.

- Diseñar el sistema de riego.
- Adaptar el sistema de riego al MINDSTORMS EV3.
- Comprobar el buen funcionamiento del diseño.

1.4 Marco teórico

1.4.1 Robot MINDSTORMS EV3

Los robots Mindstorms son juguetes educativos orientados a niños y adolescentes que aparecieron en el mercado en 1998, desarrollado gracias a la colaboración entre el laboratorio de ingeniería MIT y la empresa LEGO. El primer robot surgió con el nombre de RCX (Robotic Command eXplorers), el cual se conformaba de un brick con 32kb de capacidad en almacenamiento en la memoria RAM.

En el año 2006 LEGO lanza el Lego Mindstorms NXT al mercado, en 2009 se mejora el software de desarrollo incluyendo sensores nuevos, como el sensor de color RGB reemplazante del sensor de luz, versión denominada NXT 2.0 y se inicia su comercialización.

En el mes de julio del año 2013 sale a la venta el tercer modelo nombrado EV3 (cuyo nombre proviene de la palabra EVolution). La capacidad del brick de este modelo es de 16Mb de memoria flas y 64 Mb de memoria RAM, además cuenta con una ranura para memoria expandible con capacidad máxima de 32Gb. Se incluyen nuevos sensores como sensor infrarrojo digital (IR), sensor giroscópico, un sensor color digital capaz de detectar ausencia de color y hasta siete diferentes colores y un generador de señal infrarroja, que puede usarse también como control remoto. Acerca de la comunicación entre el brick y la computadora, el robot EV3 posee un puerto USB 2.0, Bluetooth, y con una tarjeta de red inalámbrica (WI-FI) conectada mediante el puerto USB, lo cual permiten que se programe el robot de manera inalámbrica. Su sistema operativo se basa en Linux con capacidad de interconexión con dispositivos

móviles inteligentes por medio de la compatibilidad del sistema operativo con IOS Y Android. (Tello Leal, 2013)



Figura 1.1. Sensores y componentes del LEGO MINDSTORMS EV3 (León, 2013)

1.4.1.1 EV3 Brick

El BRICK es la parte más importante, ya que es el cerebro del EV3. Se alimenta con 6 pilas de 1.5 [V] AA o una batería de Litio recargable. Tiene cuatro puertos de entrada y cuatro de salida para conectar los sensores, además de un puerto para PC Mini-USB y un puerto de USB Host y SD Card. Por último, se tiene un display que muestra lo que pasa dentro el EV3 Brick y permite usar su interface, en la parte del Anexo A se muestra la data sheet del EV3 Brick. (LEGO, 2013)

1.4.1.2 Actuadores

LEGO Technic tiene una variedad de motores que van desde pequeños micromotores, motores medianos hasta los motores largos. Estos dos últimos son los que posee el EV3, los cuales tienen buenas características en lo que es la respuesta y rendimiento, la siguiente tabla muestra algunas especificaciones de estos tipos de actuadores. (LEGO, 2013).

Tabla 1. 1. Especificaciones del Large motor [LEGO, 2013]

LEGO MINDSTORMS EV3 Large motor	
9 Volt, sin carga	175 [RPM]
	60 [mA]
Stalled	15 [Ncm]
	1.8 [A]

Tabla 1. 2. Especificaciones del Medium motor [LEGO, 2013]

LEGO MINDSTORMS EV3 Medium motor	
9 Volt, sin carga	260 [RPM]
	80 [mA]
Stalled	15 [Ncm]
	800 [mA]

1.4.1.3 Sensores

En el Kit de LEGO MINDSTORMS se tiene incluido los siguientes sensores:

- Sensor de color: Reconoce hasta siete diferentes colores y mide la intensidad de luz.
- Touch sensor (sensor táctil): reconoce la presión que se ejerce con la punta del sensor permitiendo al robot sentir las cosas o golpes.
- Sensor infrarrojo: permite que el robot pueda ver, puede funcionar como un sensor de proximidad, como sensor de búsqueda y como un receptor de control remoto.

CAPÍTULO 2

2. METODOLOGÍA

El proyecto constó en montar un sistema de riego agrícola en un robot móvil (LEGO MINDSTORMS EV3), de manera que se tenga una atención personalizado para cada planta. En este capítulo se tomaron las decisiones para la ubicación y/o arreglo de las partes del MINDSTORMS EV3, el diseño el sistema de riego y el programa para el correcto funcionamiento del prototipo.

La identificación del problema, los objetivos y los resultados de las investigaciones que se presentaron en el primer capítulo sirvieron para la especificación de los requerimientos de diseño y la identificación de alternativas de diseño del prototipo. Más adelante se detallan las alternativas y las matrices de decisión, las cuales se realizaron para determinar la opción más favorable según cumplan mejor con los requerimientos de diseño.

Seleccionada la mejor opción, se procedió con el desarrollo del diseño conceptual para el robot de riego, indicando sus componentes y brevemente cómo será el funcionamiento. Luego se comienza a realizó el diseño detallado, donde se detalló cada elemento a utilizar para la construcción del prototipo, identificando los sensores y el sistema operativo que se utilizó para que el prototipo funcione de manera adecuada.

2.1 Procedimiento esquemático

La *figura 2.1* muestra la metodología de diseño que permitió la realización del prototipo. El proceso comenzó con la definición del problema y la propuesta de objetivos. Luego se realizó un análisis de las alternativas de diseño según los requerimientos del diseño, una vez seleccionada la mejor alternativa se procedió a realizar el diseño conceptual o de forma. Después, se armó el prototipo y se realizaron las respectivas pruebas de desempeño. Finalmente se tuvo el modelo 3D.



Figura 2. 1. Proceso de diseño esquemático para el prototipo.

2.2 Requerimientos del diseño

Costos de fabricación y mantenimiento

Es un punto muy importante debido a que en el mercado existen ya algunos robots que cumplen esta función, además de diferentes sistemas de riego, por lo que se buscó una alternativa con la misma funcionalidad y con elementos de bajos costos.

Estabilidad

El robot debe tener una buena estabilidad al momento de desplazarse, debido a que trabajará en áreas con superficies irregulares.

Control de humedad y descarga de agua

En el sistema de riego que se adaptará al robot debe contar con un sensor de humedad capaz de sensar y mandar la señal al equipo para que riegue, además de un controlador de descarga de agua para no desperdiciar el recurso.

Disponibilidad

La elaboración del prototipo debe realizarse con instrumentos disponibles en el mercado nacional.

Fácil funcionamiento

El equipo debe brindar al trabajador la facilidad de operarlo, para que este pueda enfocarse en otras actividades.

2.3 Identificación y selección de las alternativas

2.3.1 Alternativas para el prototipo

Alternativa A

Es un robot con carga estática, se adapta un reservorio al cuerpo del robot para que transporte el fluido (agua) necesario y pueda regar las plantas, el tanque será la carga estática. El reservorio tendrá una capacidad determinada según el área de riego. A continuación se muestra un sketch de la alternativa.

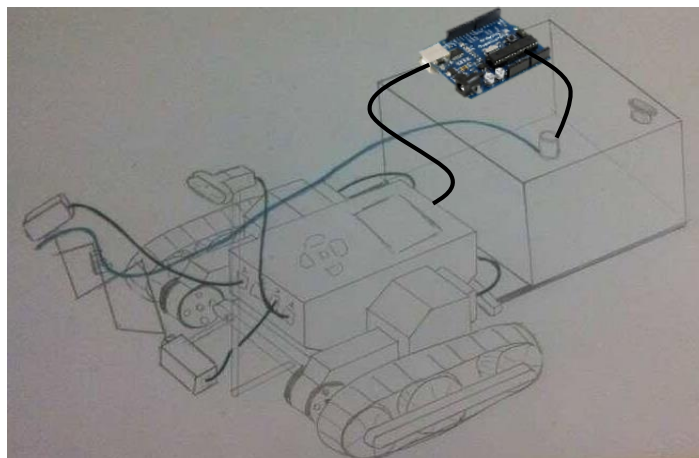


Figura 2. 2. Sketch de la alternativa A

Alternativa B

Robot con presión dinámica, este lleva una manguera retráctil conectada directamente a la llave de agua corriente, por lo que tendría que cargar con una presión cambiante (dinámica); el largo de la manguera sería una limitante. En la siguiente figura se muestra un boceto de la alternativa.

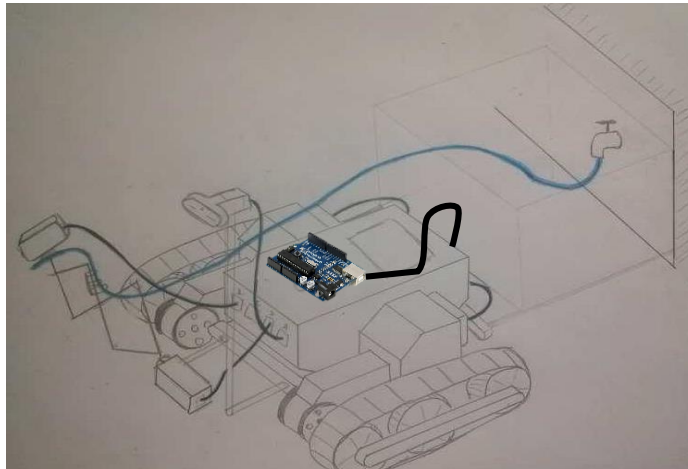


Figura 2. 3. Sketch de la alternativa B.

2.3.1.1 Selección del prototipo

Para la selección de la mejor alternativa se elaboró una matriz de decisión estableciendo una escala del 1 al 5 para calificar cada criterio establecido, siendo 1 la calificación más baja y 5 la máxima. Primero se procedió a dar una ponderación a cada criterio según la importancia (*tabla 2. 1*). Luego, en la matriz de decisión se calificó las alternativas en cada criterio y se las sumó, se obtuvo el puntaje mostrado en la *tabla 2.2*, lo cual indicó que opción era la mejor.

Tabla 2. 1. Criterios para el prototipo con la respectiva ponderación

Criterio	Ponderación [%]
1) Mantenimiento	20
2) Estabilidad	30
3) Costo	15
4) Fácil funcionamiento	15
5) Control de descarga de agua	20

Tabla 2. 2. Matriz de selección para la mejor alternativa de prototipo

Criterio a calificar	Alternativa A		Alternativa B	
	Calificación	Valor Ponderado	Calificación	Valor Ponderado
1	3	0.6	4	0.8
2	5	1.5	3	0.9
3	3	0.45	4	0.6
4	5	0.75	4	0.6
5	4	0.8	3	0.6
Resultados	Total= 4.1		Total= 3.5	

Como se muestra en la tabla anterior, la opción ganadora fue la del robot con una carga estática (Alternativa A), esto debido a que el robot tendrá mejor estabilidad al movilizarse. Además, con esta opción se tiene un funcionamiento más fácil para el operario en comparación con la otra alternativa.

2.3.2 Alternativas para el sistema de comunicación (Módulo inalámbrico)

Alternativa 1

Wi-Fi: Es un mecanismo que conecta de manera inalámbrica dispositivos electrónicos. Es necesario contar con un enrutador para poder utilizar este módulo, una ventaja es que al conectarse al internet el usuario es capaz de controlar y monitorear el dispositivo desde cualquier lugar del mundo mediante algún dispositivo portátil. Los estándares 802.11 utilizan la banda de 2.4 GHz, cuenta con 32 nodos, se definen 11 canales, los cuales no son completamente dependientes, es recomendable utilizar canales con una separación de 5 canales entre ellos. Trabaja con una velocidad de transmisión de datos mayores a 50Mbps, presenta una configuración bastante compleja, además de un alto consumo de batería llegando a los 20 mA en reposo y su precio es relativamente alto (aproximadamente \$56).

Alternativa 2

Bluetooth: La tecnología Bluetooth utiliza ondas de radio para el intercambio de datos. No necesita de enrutador como el Wi-Fi y ofrece la posibilidad de controlar el proyecto desde el propio móvil. Está diseñado para la comunicación a corta distancia, la cual tiene un alcance de entre 5 a 10 metros y una velocidad de transmisión de 1 Mbps. Este sistema posee 8 nodos y su consumo de batería es menor que la del Wi-Fi, la cual puede durar varios días y consume un máximo de 0.2 mA estando en reposo y su configuración no es muy compleja. No necesita de enrutador como el Wi-Fi, y ofrece la posibilidad de controlar el proyecto desde el propio móvil.

Es necesario contar con un dispositivo maestro, el cual permite un máximo de 7 dispositivos sincronizados para la transmisión de datos.

Alternativa 3

ZigBee: Módulo inalámbrico que ofrece una comunicación inalámbrica perfecta entre dispositivos, puertas de enlace y adaptadores. Están adecuados para cualquier tipo de arquitectura de red, incluyendo celulares, hilos de ejecución, 802.15.4 y estándares abiertos de Wi-Fi. Permite la interconexión de dispositivos remotos y crear redes, a pesar del extremadamente bajo consumo de energía (3 μ A en reposo) tiene un largo alcance. Debido a que cada dispositivo puede comunicarse con sus pares, así como con el centro inteligente. Un dispositivo que se encuentre fuera del alcance del controlador recibe de todos modos sus instrucciones a través de cualquier número de vías, esta configuración se conoce como red de mallas. El módulo inalámbrico ZigBee brinda una velocidad de transmisión de datos relativamente baja en comparación con las alternativas anteriores (<250 kbps). Sin embargo, la red Zigbee podría conformarse de hasta 65535 equipos, lo cual representa una gran cantidad de equipos, posee un alcance de hasta 30 metros y su precio es accesible.

2.3.2.1 Selección del módulo inalámbrico

Para las alternativas del módulo inalámbrico se realizó la siguiente tabla resumiendo las especificaciones de cada uno.

Tabla 2. 3. Especificaciones de cada alternativa

Alternativa Parámetro	Alternativa 1 Wi-Fi	Alternativa 2 Bluetooth	Alternativa 3 ZigBee
Velocidad	<50 Mbps	1 Mbps	<250 kbps
Núm. Nodos	32	8	255 – 65535
Duración de batería	Horas	Días	Años
Consumo transm.	400 mA	40 mA	30 mA
Consumo reposo	20 mA	0.2 mA	3 μ A
Precio	Caro	Medio	Barato
Configuración	Compleja	Compleja	Simple

Para la selección de la mejor alternativa se elaboró una matriz de decisión estableciendo una escala del 1 al 5 para calificar cada criterio establecido, siendo 1 la calificación más baja y 5 la máxima. Primero se procedió a dar una ponderación a cada criterio según la importancia (*tabla 2.4*). Luego, en la matriz de decisión se calificó las alternativas en cada criterio y se las sumó, se obtuvo el puntaje mostrado en la *tabla 2.2*, lo cual indicó que opción era la mejor.

Tabla 2. 4. Criterios para el módulo inalámbrico con la respectiva ponderación

Criterio	Ponderación [%]
1) Configuración	25
2) Precio	25
3) Duración de batería	20
4) Consumo reposo	15
5) Número de nodos	15

Tabla 2. 5. Matriz de selección para la mejor alternativa de módulo inalámbrico

Criterio a calificar	Alternativa 1		Alternativa 2		Alternativa 3	
	Calificación	Valor Ponderado	Calificación	Valor Ponderado	Calificación	Valor Ponderado
1	1	0.25	1	0.25	5	1.25
2	1	0.25	3	0.75	5	1.25
3	1	0.20	2	0.40	5	1.00
4	1	0.15	3	0.45	5	0.75
5	2	0.30	1	0.15	5	0.75
Resultados	Total= 1.15		Total= 2.00		Total=5.00	

2.4 Diseño Conceptual

La siguiente *figura 2.4* muestra el diseño conceptual del robot móvil.

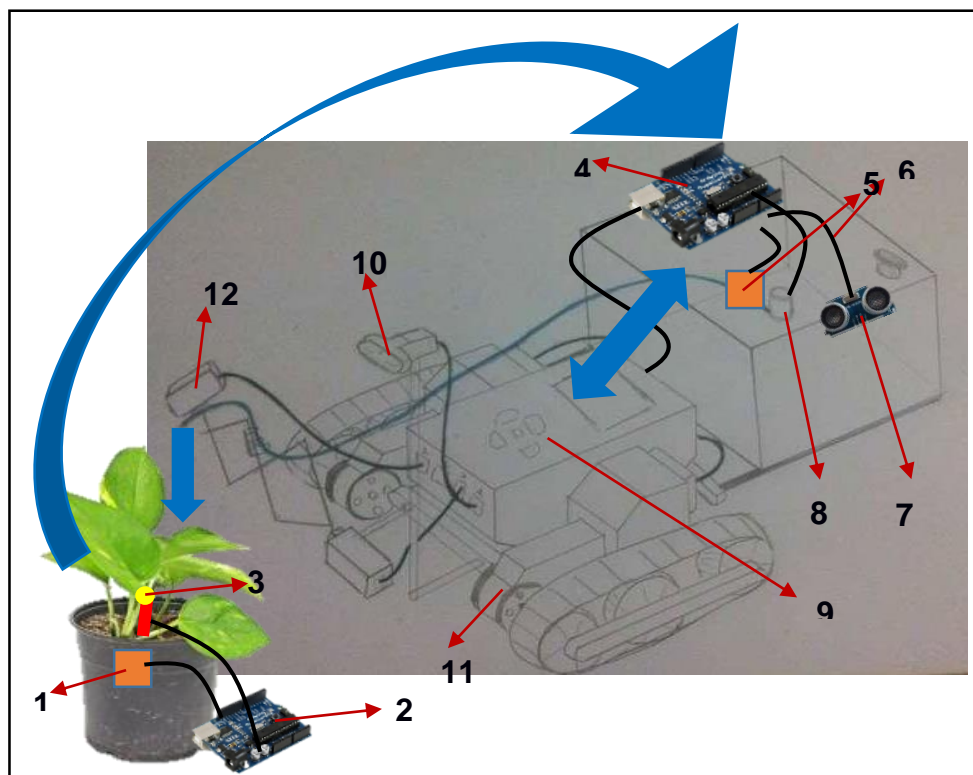


Figura 2. 4. Diseño conceptual del robot móvil para riego. 1) Módulo inalámbrico (emisor). 2) Tarjeta Arduino A. 3) Sensor de humedad. 4) Tarjeta Arduino B. 5) Módulo inalámbrico (receptor). 6) Reservorio de agua. 7) Sensor ultrasónico. 8) Bomba. 9) Brick EV3. 10) Sensor ultrasónico EV3. 11) Actuadores largos del EV3. 12) Sensor de color EV3.

Se tiene como señal de entrada la del sensor de humedad, la cual es enviada por medio del módulo inalámbrico XBee de la tarjeta A (planta) a la tarjeta B (robot). Se transmite la señal de humedad desde la tarjeta B hasta el Brick EV3 para que el robot comience a buscar a la planta. Por medio del sensor de color EV3 y sensor ultrasónico EV3 el prototipo es capaz llegar a la maseta y esquivar obstáculos. Cuando llegue a la planta el Brick EV3 manda la señal a la tarjeta B para que esta comience a bombear agua. Además, en la tarjeta B controla la cantidad de agua en el reservorio por medio de un sensor de nivel.

2.5 Diseño detallado

El diseño detallado se lo dividió en dos partes para mejor facilidad de estudio, diseño de: reservorio y bomba para el agua; y el sistema de control.

2.5.1 Diseño detallado del reservorio y bomba

2.5.1.1 Reservorio

Para el diseño del reservorio se fijó su capacidad a partir del conocimiento del consumo de agua al día en un área de 2000 [m²] es de unos 360 [lt]; por lo tanto, se decidió que la superficie de riego sea de 5 [m²], lo que equivaldría a 0.9 [lt]. Entonces, se estableció como capacidad del reservorio 1 [lt] de agua, con este volumen se selecciona por facilidad una botella plástica de gaseosa, debido a que es liviana y resistente a líquidos.

En la siguiente tabla se muestra en resumen las especificaciones del diseño del reservorio para agua.

Tabla 2. 6. Especificaciones de diseño para el reservorio

Capacidad	1 [lt]
Material	Plástico PVC
Dimensionamiento	H=12, Ø=11 [cm]

2.5.1.2 Bomba

Para la bomba se decidió que tenga la capacidad de regar un aproximado de 2.5 [cm] de nivel de agua en 10 segundos por cada bombeo. Se identificaron dos alternativas viables: primero la de trabajar con una bomba casera y, segundo utilizar una bomba de limpia parabrisas. Ambas fueron buenas opciones por su peso liviano y costos, pero se optó por usar la bomba de limpia parabrisas por su eficiencia; además de que con la bomba casera se tuvieron inconvenientes.

Se acopló el aparato al reservorio y se adaptaron los componentes necesarios para el correcto funcionamiento con Arduino. Una vez terminada las respectivas adaptaciones se realizaron pruebas para obtener la curva de trabajo de la bomba y su eficiencia, lo cual se detalla en el apartado de *Prueba B*.

2.5.2 Diseño detallado del sistema de control de posicionamiento y riego

Para que el robot pueda saber a qué planta tiene que ir, éste recibe primero la señal del sensor de humedad de dicha planta, la cual se envía por medio de los módulos inalámbricos desde el Arduino A al Arduino B. Después se envía la señal de humedad de la tarjeta B al Brick EV3, con esta señal el robot comienza a moverse y activa los sensores: de color en modo que identifique sólo el blanco para que llegue a la planta y; ultrasónico para que pueda evitar los obstáculos.

El sistema de riego se acciona en el momento que el sensor de color mande la señal de que está en la planta al Brick EV3 y este a su vez lo mande al Arduino para que comience a bombear agua, la cantidad de agua surtida se controla por medio de un sensor ultrasónico.

Los sensores y tarjetas Arduino que se usaron fueron los siguientes:

- Sensor de humedad: higrómetro FC-28
- Dos módulos inalámbricos: XBee serie 2

- Dos tarjetas Arduino UNO
- Sensores ultrasónicos HC-SR04
- Sensor ultrasónico EV3
- Sensor de color EV3

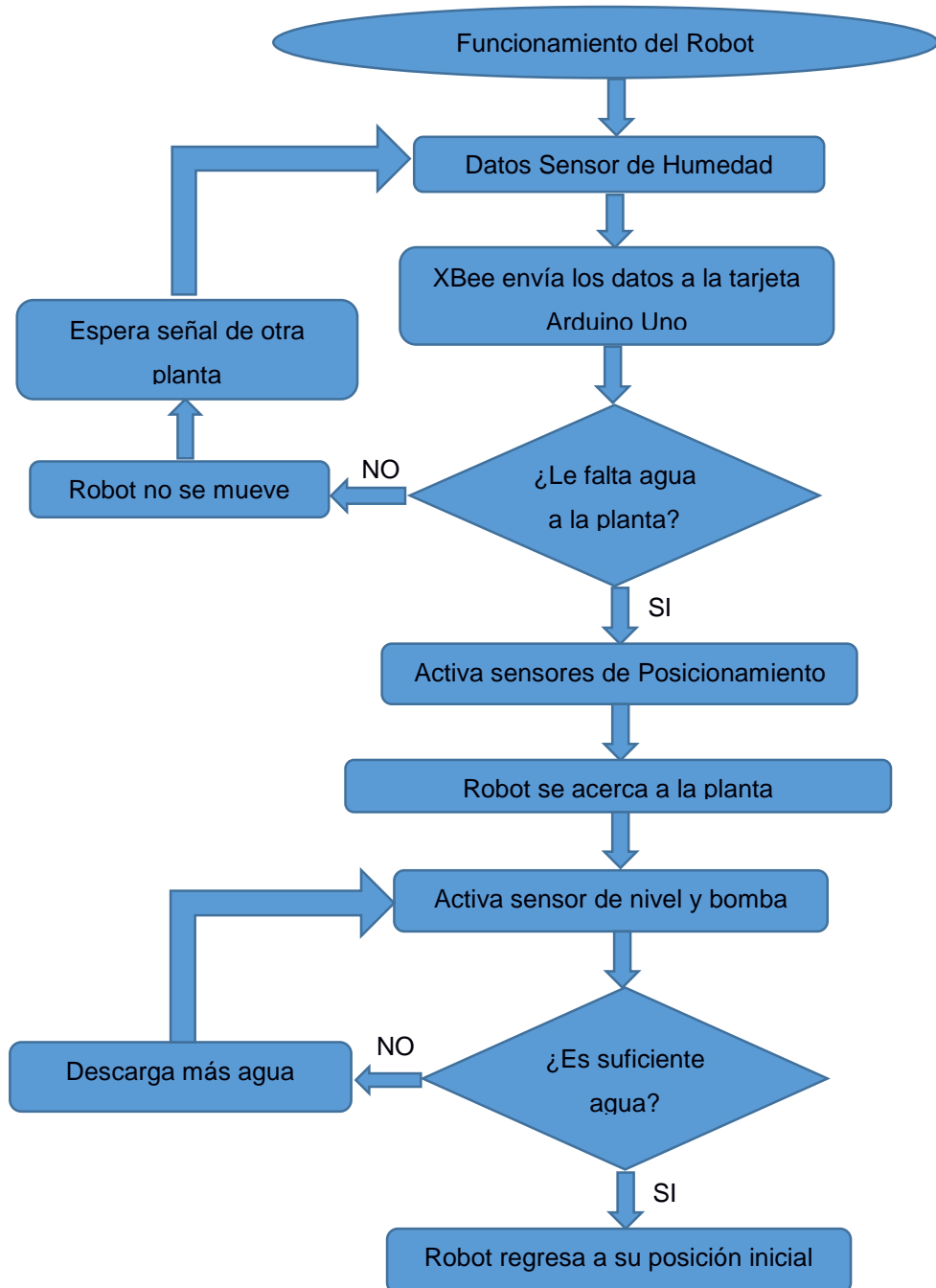


Figura 2. 5. Diagrama de flujo del sistema de control de posicionamiento y riego.

2.5.2.1 Sensor de humedad (higrómetro FC-28)

El FC-28 es un sensor medidor de humedad del suelo por medio de la variación de su conductividad. Son bastantes utilizados en sistemas automáticos de riego enviando la señal para activar el sistema de bombeo cuando sea necesario.

El sensor se lo coloca en la maceta de la planta para que mida la humedad de dicha planta entregando como resultado un valor de 0 o 1, siendo 0 el indicador de que la planta si tiene agua (sensor sumergido en agua o tierra húmeda) y 1 para cuando falte agua (sensor en el aire o en suelo muy seco), dichos valores se envían mediante el módulo inalámbrico al robot.

2.5.2.2 Digi XBee

Módulo inalámbrico que ofrece una comunicación inalámbrica perfecta entre dispositivos, puertas de enlace y adaptadores. Están adecuados para cualquier tipo de arquitectura de red, incluyendo celulares, hilos de ejecución, 802.15.4 y estándares abiertos de Wi-Fi.

El modelo a utilizar en el proyecto es el Digi XBee 802.15.4, el cual cumple con la función de transmitir la humedad de la planta otorgada por el higrómetro FC-28 a la señal inalámbrica del robot. La tabla a continuación muestra las características del modelo.

Tabla 2. 7. Características técnicas del Digi XBee [(Digi)]

Velocidad de datos	250 kbps
Rango de alcance	100 ft (30 m)
Poder de transmisión	1 mW
Sensibilidad del receptor (1% PER)	-92 dBm
DIGI hardware	Si
Chip transceptor	Freescale MC13212
Frecuencia	2.4 GHz
Entradas ADC	(6) 10-bit ADC entradas
Interfaz de datos en serie	3.3 V CMOS UART
Velocidad de datos en serie	1200 bps – 250 kbps
Digital I/O	8
Opciones de Antena	Chip, chip de alambre, U.FL, & RPSMA
Fuente de alimentación	2.8 – 3.4 VDC
Transmitir corriente	45 mA @ 3.3 VDC
Recibir corriente	50 mA @ 3.3 VDC
Corriente de apagado	<10 uA @ 25°C

Batería

Para la elección de la batería se debe tomar en cuenta aspectos como los voltajes de alimentación del módulo XBee, el sensor de humedad y el tamaño de la batería. Tomando en cuenta esto, el empleo de una batería alcalina de 9V es la mejor opción, ya que se tienen las alimentaciones de 5V del sensor de humedad y 3.3V del XBee.

2.5.2.3 Tarjeta ARDUINO UNO R3

Placa micro-controladora basada en la hoja de datos ATmega328. Cuenta con 14 pines de entrada/salida digital (6 de los cuales pueden ser usados como salidas PWM), 6 entradas analógicas, un oscilador de cristal de 16 MHz, conector de alimentación, conexión USB, un encabezado ICSP y un botón de reinicio. En este trabajo, se usó una tarjeta Arduino para ejecutar el código que permita captar la señal de humedad de la planta para ser transmitida. Otra segunda tarjeta para que reciba la señal de humedad y envíe al Brick EV3, aparte de activar la bomba cuando sea necesario. A continuación, se tienen las especificaciones de la tarjeta:

Tabla 2. 8. Especificaciones de la tarjeta ARDUINO UNO [(static.rapidonline)]

Voltaje de operación	5 V
Voltaje de entrada (recomendado)	7-9 V
Voltaje de entrada (límite)	6-20 V
Pines digitales I/O	14
Entradas de pines análogos	6
Corriente DC Pin I/O	40 mA
Corriente DC por Pin 3.3 V	50 mA
Memoria flash	32 KB
SRAM	2 KB
EEPROM	1 KB
Velocidad de reloj	16 MHz

2.5.2.4 Sensor de Color EV3

Este sensor puede medir la cantidad de luz reflejada, medir la luz del ambiente y medir los colores. En este caso, se utilizó al sensor para que identifique los colores, cuando mida blanco a 3 [mm] de distancia se indica que está en la planta. Esto provoca que el robot se detenga y active la bomba de riego.

2.5.2.5 Sensor Ultrasónico NXT

El ultrasónico NXT permite que el prototipo se pueda mover con libertad, detectando objetos en el camino para no chocar, logra realizar esto midiendo la distancia a la que está el objeto en un rango de [0 – 255] centímetros. Trabaja como lo hacen los radares o sonares, enviando ondas de sonidos y controla en tiempo en que tardar en rebotar dicha onda.

2.5.2.6 Sensor ultrasónico (HC-SR04)

Sensor ultrasónico con capacidad de detectar la presencia de un objeto y a su vez puede sentir y transmitir la distancia al objeto. Posee dos transductores, un altavoz y un micrófono, su funcionamiento no se ve afectado por la presencia de luz solar o de cuerpos negros como telémetro óptico. Pines:

- VCC: Alimentación +5V (4.5V min – 5.5V max)
- TRIG: Trigger entrada (input) del sensor (TTL)
- ECHO: Echo salida (output) del sensor (TTL)
- GND

Para el proyecto se lo emplea para medir el nivel del agua en el reservorio, el cual está colocado en la parte superior, adaptado en la tapa. Conociendo la distancia de vacío y la distancia entre el sensor y el agua con el reservorio lleno en su capacidad máxima, se elabora un código para determinar la cantidad de agua liberada y por ende la cantidad de agua restante. Las características principales de este sensor se muestran en la siguiente tabla:

Tabla 2. 9. Especificaciones del sensor HC-SR04 [(Soria, 2013)]

Corriente de reposo	<2 mA
Corriente de trabajo	15 mA
Ángulo de medición	30°
Ángulo de medición efectivo	<15°
Detección	De 2 a 400 cm
Dimensiones	45 mm x 20 mm x 15 mm
Precisión	Varía entre 3 mm o 0.3 cm
Frecuencia	40 KHz

2.5.3 Programación del equipo

La programación se lo realizó en tres partes, una de ellas en lenguaje Python y las otras dos en Arduino; además, para los XBee se usó el programa *XCTU* para configurar uno como emisor y otro receptor en modo API. El código de la parte del masetero (higrómetro y XBee) se le llamó emisor y fue escrita en Arduino, esta permite que la señal de humedad sea transmitida a donde se encuentra el robot. El segundo código que corresponde al que va en la tarjeta Arduino que se encuentra en el robot, recepta la señal del emisor y la envía al Brick EV3, en el momento requerido activa la bomba y controla el nivel de agua del reservorio con el ultrasónico HC-SR04. El tercer código que va en el

Brick EV3 permite la comunicación bilateral entre el robot y la tarjeta Arduino, logrando que se mueva el MINDSTORMS y busque la planta cuando esta lo necesite y, avisar que se accione la bomba cuando ya estén en el sitio. Para este último se hizo uso el programa *ev3dev* que permite lenguajes Python, JAVA, C++ entre otros, este programa es un sistema operativo basado en Debia-Linux que permite ejecutar varias plataformas compatibles con MINDSTORMS.

A continuación se presentan el diagrama de flujo para el código que ejecuta el Arduino A (emisor), el que se encuentra en la planta.

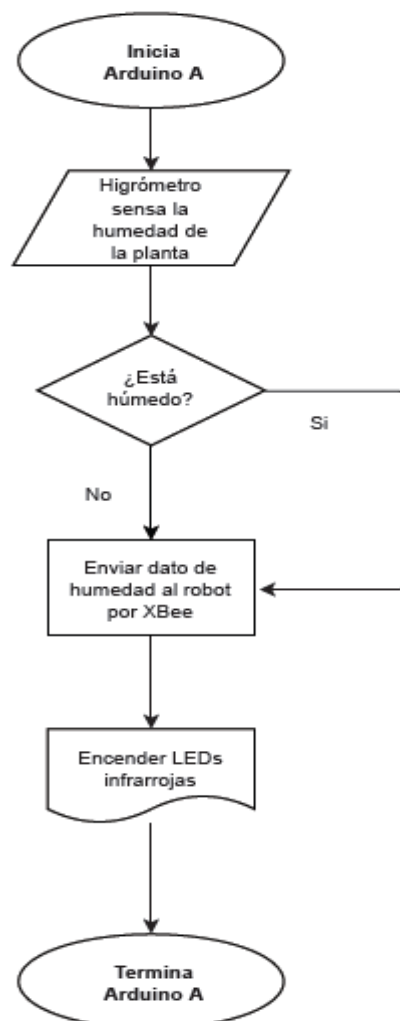


Figura 2. 6 Diagrama de flujo para código del Arduino A

El siguiente diagrama de flujo es del código que es ejecutado por la tarjeta Arduino B, la cual está en el robot.

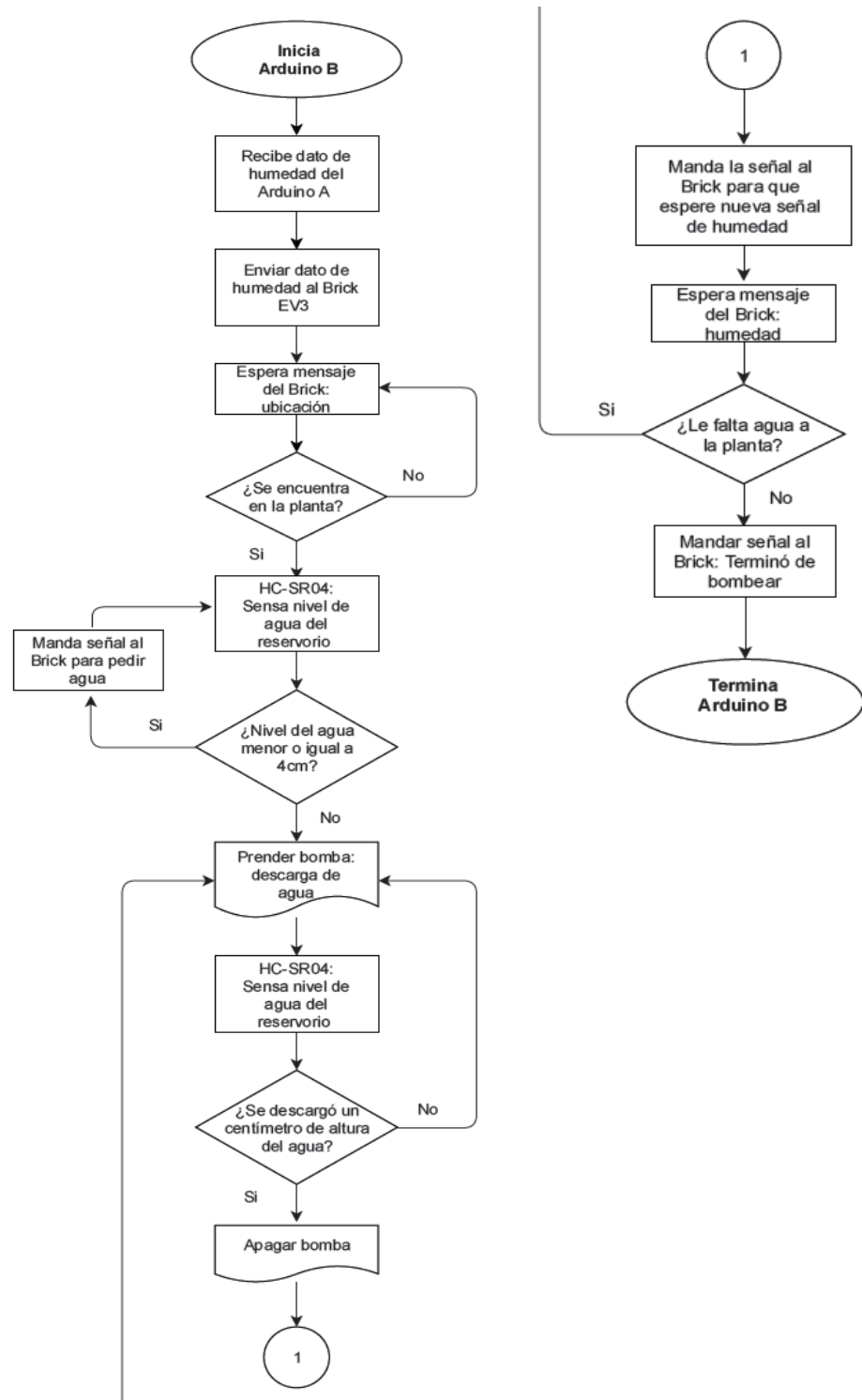


Figura 2. 7. Diagrama de flujo para el código del Arduino B.

El último diagrama de flujo que se presenta en la *figura 2.8*, es para el código en Python que corre el Brick EV3.

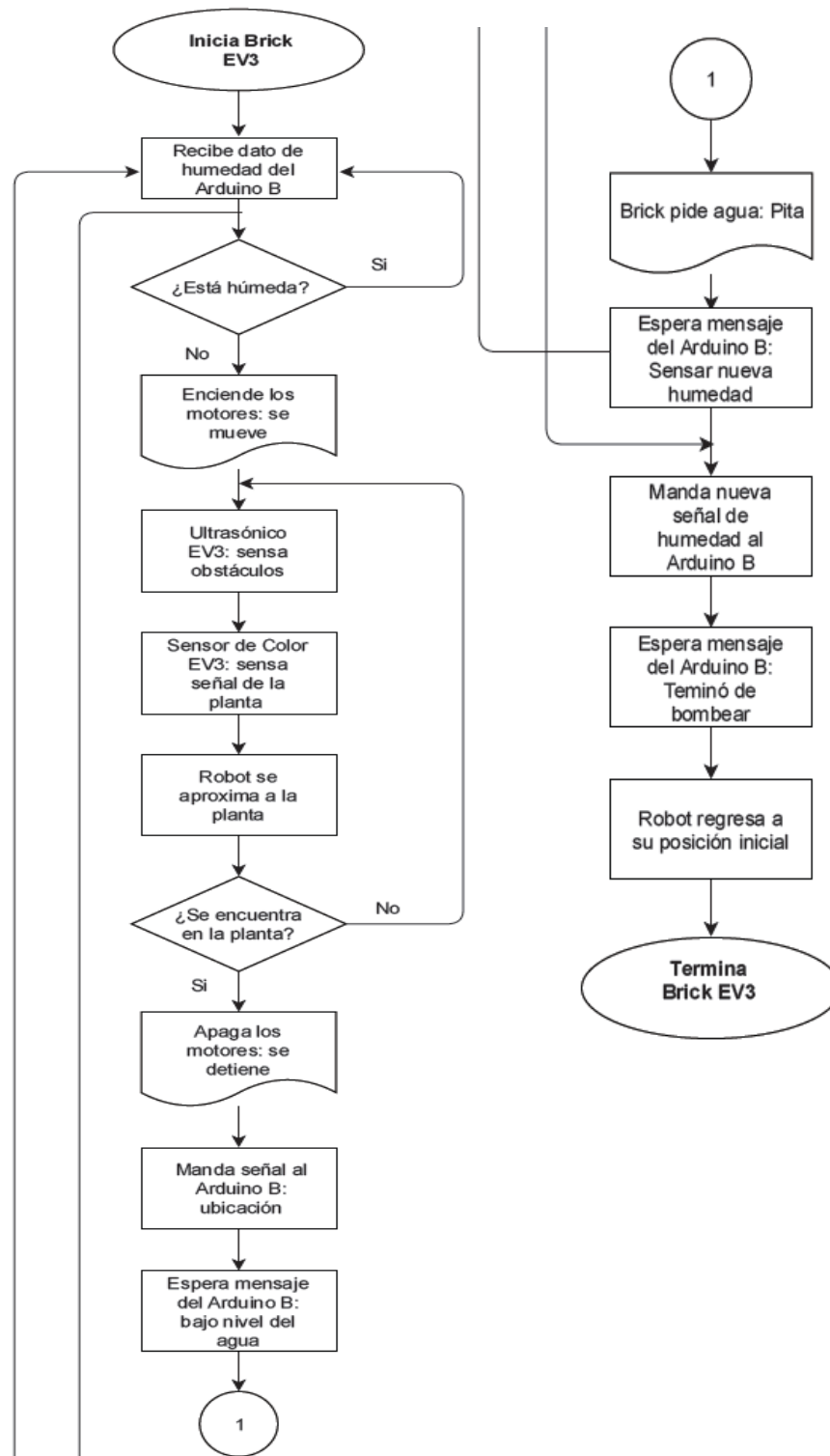


Figura 2. 8. Diagrama de flujo para el código del Brick EV3

2.6 Pruebas

En este apartado se detalló la metodología de las pruebas que se realizaron para verificar el correcto funcionamiento del robot.

2.6.1 Prueba A: Comunicaciones

Se probaron la correcta comunicación entre los dispositivos:

- a) ***XBee (emisor) ↔ XBee (receptor)***: Para comprobar que exista una correcta comunicación entre los XBee se hizo uso del programa XCTU, en el cual primero se programó a un módulo como Coordinator AT (emisor) y al otro como Router AT (receptor). Posteriormente se realizó la conexión entre los dos y se comenzaron a enviar mensajes como: “Hallo”, “H L” y “1 0”. Al momento que se tuvo una comunicación efectiva se continuó con la siguiente comunicación.

- b) ***XBee ↔ Arduino***: Una vez programado los XBee, se procedió a conectar cada uno a un Arduino y se cargó primero un código donde solo se envíen “h” de HIGH y “l” de LOW, luego se lo ejecutó y se probó que se transmita correctamente los mensajes. A partir de que si funcionó ese primer código se lo adaptó para que envíe la señal de humedad del higrómetro con un segundo código, y también se probó la correcta transmisión de la señal de humedad.

- c) ***Arduino ↔ Brick EV3***: La comunicación entre estos dos dispositivos se la realizó por medio del puerto serial del Brick, configurando correctamente en el código del robot la paridad, el baud rate, etc. Primero se realizó un código simple donde el Brick pueda guardar los datos del Arduino en un archivo y luego lo muestre en la pantalla. Una vez que este primer código funcionó se procedió a que el robot lea el archivo y muestre cada carácter como uno solo, para comparar cada uno y darle una orden, si es “1” comience a mover y si es “0” no se mueva. Teniendo la comunicación Arduino->Brick, se adaptó el código para que ahora se envíe señal del Brick al Arduino.

2.6.2 Prueba B: Bomba

Primero se realizaron los cálculos de la potencia hidráulica que se necesita en la bomba para bombear lo que se especificó en la sección de diseño de bomba. Además, al escoger una bomba de limpiaparabrisas se sabe que su eficiencia está alrededor del 80%. Por lo tanto, se tiene que adaptar una batería con el voltaje y corriente necesario tal que, la potencia del motor alcance una eficiencia aproximada a la real. Para esto se usaron las ecuaciones descritas más adelante.

Una vez conocida la batería que se utilizó, se la conectó a la bomba junto con el módulo RELAY para que funcione correctamente con la tarjeta Arduino UNO. Además, se colocó en la parte superior del reservorio el sensor ultrasónico HC-SR04, para medir las alturas necesarias para la curva de trabajo. Con las correctas conexiones se procedió a realizar las pruebas para obtener la curva de trabajo del equipo.

Las curvas que se trazaron fueron de Cabezal vs Caudal y Potencia hidráulica vs Revoluciones, para lo cual se programó el Arduino que midiera la altura y el volumen de descarga, además de variar las revoluciones del motor. Con los valores obtenidos del código y el tiempo, se procedió a realizar los cálculos de caudal y potencia hidráulica, necesarios para las gráficas.

- *Caudal del fluido*

$$Q = V * t \quad (2.1)$$

V : Volumen de descarga.

t : Tiempo en que descargó ese volumen.

- *Potencia del fluido*

$$P_w = \rho g Q h \quad (2.2)$$

Donde:

h : Altura.

- *Potencia motor*

$$P_f = V * I \quad (2.3)$$

Donde:

V : Voltaje.

I : Corriente.

- *Eficiencia*

$$\eta = \frac{P_w}{P_f} \quad (2.4)$$

2.6.3 Prueba C: Movimiento del prototipo

Por último, se realizaron pruebas con el robot ensamblado, las cuales fueron en superficies lisas e irregulares, ejecutando el código realizado en PYTHON 3 desde el BRICK. Además, se probó la parte del código en la que el Brick emite el sonido de aviso para recargar de agua el reservorio cuando este lo necesitó.

- Prueba en superficie lisa:** El robot se mueve desde su posición inicial hasta donde se encuentra la planta seca a una velocidad constante de 157.5 [rpm], el 90% de la velocidad máxima a la que se pueden mover el MINDSTORMS EV3. Una vez que llegó al punto de destino se detiene y acciona la bomba, al terminar de regar la planta el robot retrocede a la misma velocidad y, espera a que vuelva a llegar nuevamente la señal de humedad baja. Esta primera prueba se la realizó a diferentes distancias entre planta y robot: 1 [m], 10 [m] y, 15 [m].
- Prueba en superficie irregular:** Al igual que en el ensayo anterior, el robot comienza a moverse cuando la señal de humedad es “1” (seco), a una velocidad constante de 148.75 [rpm], el 85% de la velocidad máxima del MINDSTORMS EV3. Nuevamente al llegar a la planta se detiene y acciona la bomba, al terminar de regar la planta el robot retrocede a velocidad de 140 [rpm] y espera nueva señal de humedad. En este caso se realizaron las pruebas a distancias de: 0.5 [m], 5 [m] y, 10 [m].

CAPÍTULO 3

3. RESULTADOS Y ANÁLISIS

3.1 Resultados

A continuación se presentan los resultados obtenidos en el proceso de diseño del robot para riego. Resultado de los cálculos realizados, modelo 3D del robot, conexiones y códigos utilizados.

3.1.1 Resultados del diseño del reservorio y bomba

La siguiente *figura 3.1* se muestra el modelo 3D del reservorio para agua con las medidas especificadas en el capítulo anterior. El tanque se compone de un tubo y una tapa PVC, además de una tapa superior diseñada e impresa en 3D con material PLA. En el diseño se tiene tres aberturas, la de abajo es donde se adaptó una bomba de agua con una manguera de $\frac{1}{4}$ " para bombear; en la parte de arriba, se colocó el sensor ultrasónico en la abertura rectangular y, por el orificio es para recargar de agua.

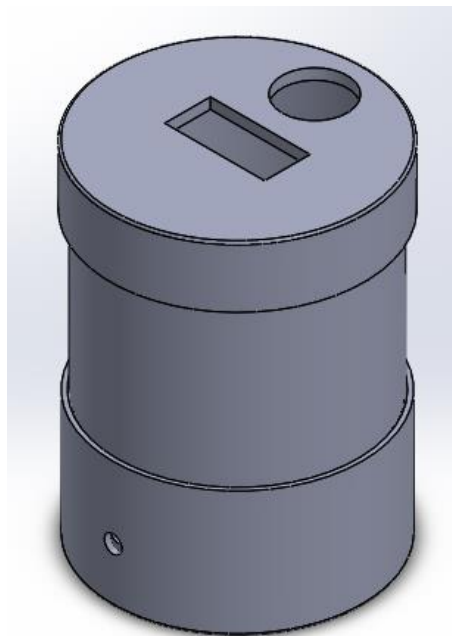


Figura 3. 1. Modelo 3D del reservorio para agua

En la siguiente imagen se muestra el reservorio que se usó para realizar todas las pruebas respectivas ya construido y acoplado a la bomba en la parte de abajo, además de tener ensamblado en la parte superior el sensor de nivel.



Figura 3. 2. Imagen de reservorio y bomba ensamblados

3.1.2 Conexiones y códigos

3.1.2.1 Diagrama de conexiones

En esta parte se detallarán las conexiones que se realizaron entre los sensores y las tarjetas Arduino, así como las del Arduino y sensores EV3 con el Brick EV3.

En la figura 3.2 se muestra la conexión del higrómetro y XBee con el Arduino A, esto permitirá que se envíe la señal de humedad al robot, en otras palabras esta primera conexión es la del emisor. En la parte derecha del Arduino A se encuentra el higrómetro con los cables rojo, negro y amarillo que corresponden al puerto de voltaje, tierra y señal digital (PIN 4) respectivamente. El lado izquierdo se tiene al XBee en su adaptador USB con los cables azul, rojo y blanco que van conectados a los puertos de voltaje, tierra y digital 9 (como TX) respectivamente. Esta tarjeta va conectada a una batería de 9V.

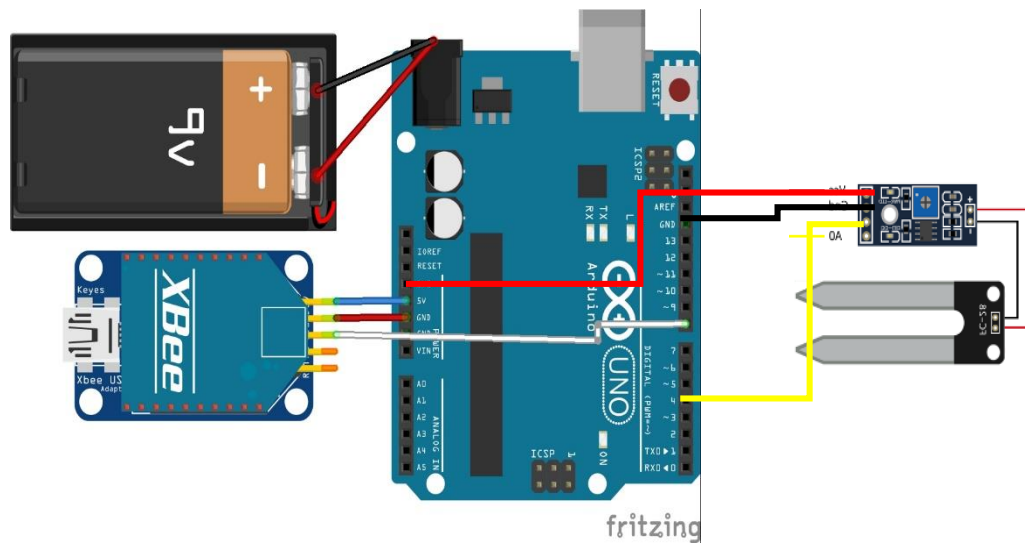


Figura 3. 3. Esquema de conexión del Arduino A (Emisor)

Por otra parte la figura 3.3 muestra la conexión de la tarjeta Arduino B, la cual cumplirá la función de recibir la señal de la tarjeta A, de comunicarse con el MINDSTORMS EV3 y accionar la bomba. En la tarjeta B se conectará el XBee (receptor) y el ultrasónico HC-SR04, por el puerto USB se conectarán el Brick EV3 y el Arduino.

En la parte izquierda (bajo) se encuentra el XBee (receptor) conectado a su adaptador y este a su vez se conecta al Arduino B con los cables verde (PIN 11 \equiv voltaje), gris (PIN \equiv tierra) y negro (PIN 8 \equiv RX). La bomba está conectada a un relé y a una batería de 9V, los cables que salen del relé al Arduino, amarillo y verde, son de conexión a tierra y voltaje respectivamente, los verdes de voltaje y el cable rojo conecta al PIN 3. A la derecha tenemos al sensor HC-SR04 con los cables de conexión verde (voltaje), morado en el pin 7 (trig), rojo en el pin 6 (echo) y amarillo (tierra).

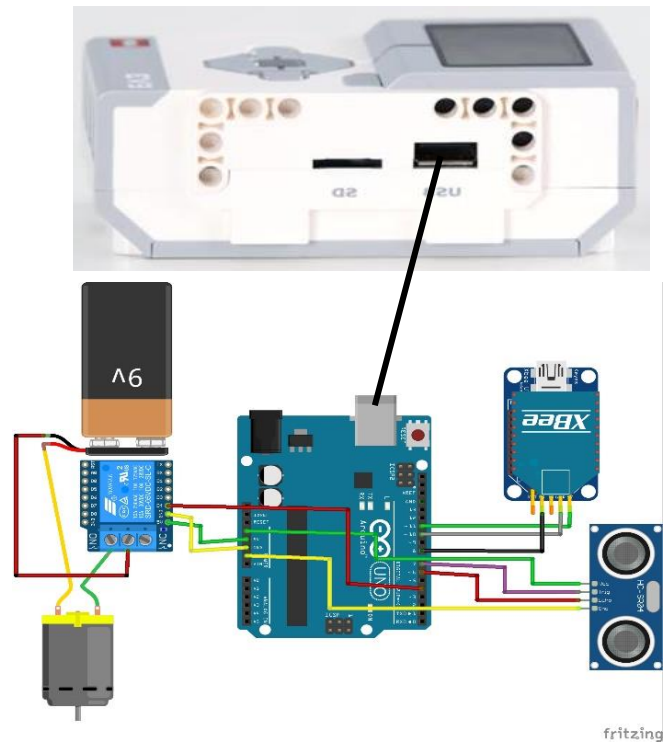


Figura 3. 4 Esquema de conexión para el Arduino B (receptor)

La siguiente figura muestra las conexiones realizadas en el Brick EV3. Los motores largos se encuentran conectados en los puertos de salida B y C, mientras que los sensores están conectados en los puertos de entrada 3 (Color) y 4 (Infrarrojo).



Figura 3. 5 Esquema de conexión de los sensores del EV3

3.1.3 Ensamble de los componentes del robot

Se comenzó ensamblando la parte de los motores con el Brick y los sensores del EV3, el cual se lo realizó en modo tanque como se aprecia en la *figura 3.6*. Posteriormente se realizó una especie de canasta en la parte superior del robot como muestra la *figura 3.7*, en donde se colocó el reservorio y la bomba. Además, se construyeron dos cajas de madera para colocar los dispositivos electrónicos, una que va a un lado de la planta y el otro que está en la parte trasera del robot, la *figura 3.8* se presentan las cajas con los circuitos.

Por último, en la *figura 3.9* se muestra el robot terminado, con todos los ensambles realizados.



Figura 3. 6. Ensamble de los motores con el Brick y los sensores

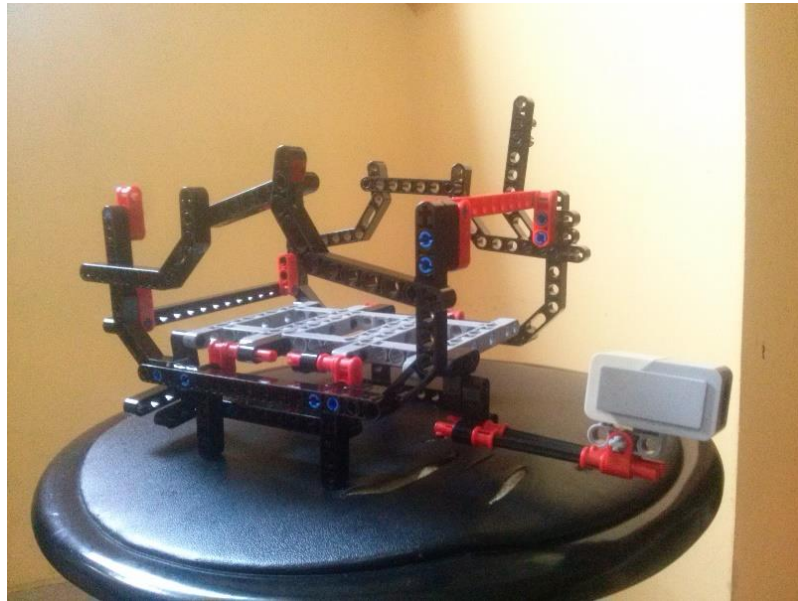
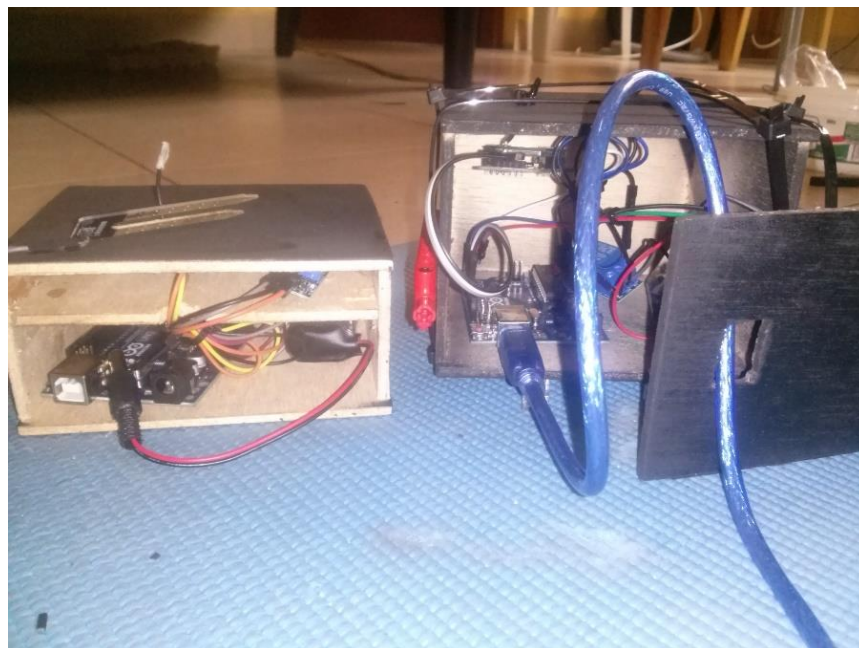


Figura 3. 7. Ensamble de la canasta donde se coloca el reservorio bomba



**Figura 3. 8. Cajas para los dispositivos electrónicos (i) Circuito de la planta,
(d) Circuito del robot**

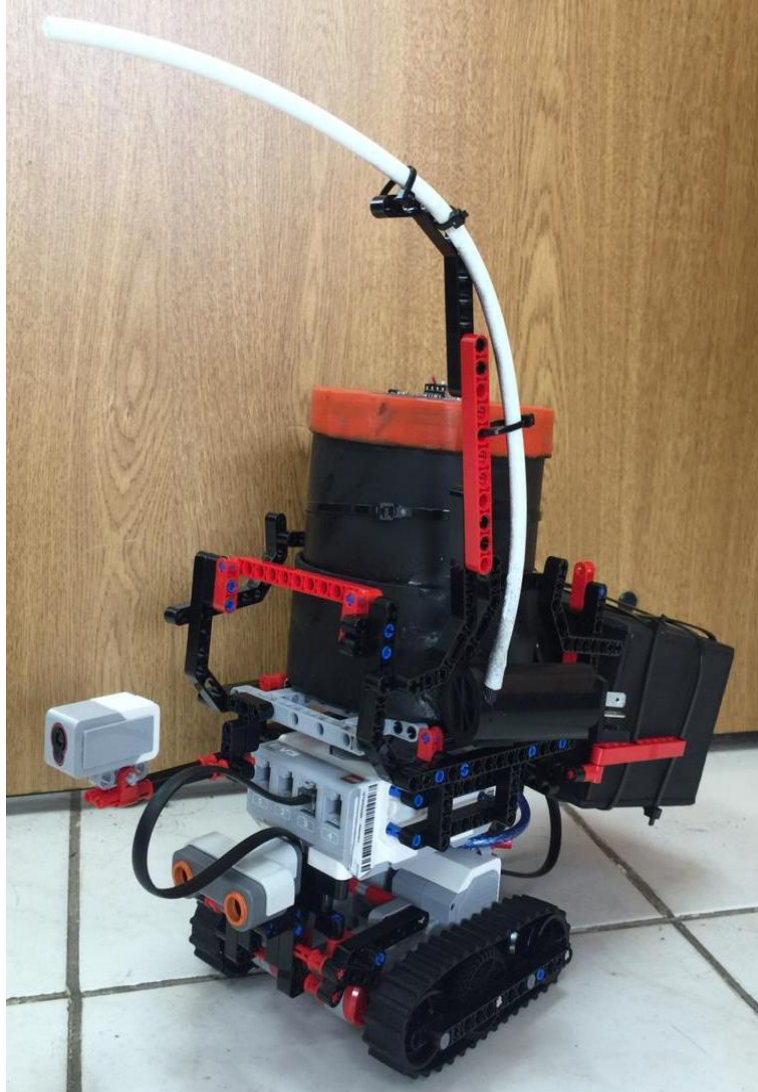


Figura 3. 9. El robot con todos sus componentes

3.1.4 Resultados de pruebas

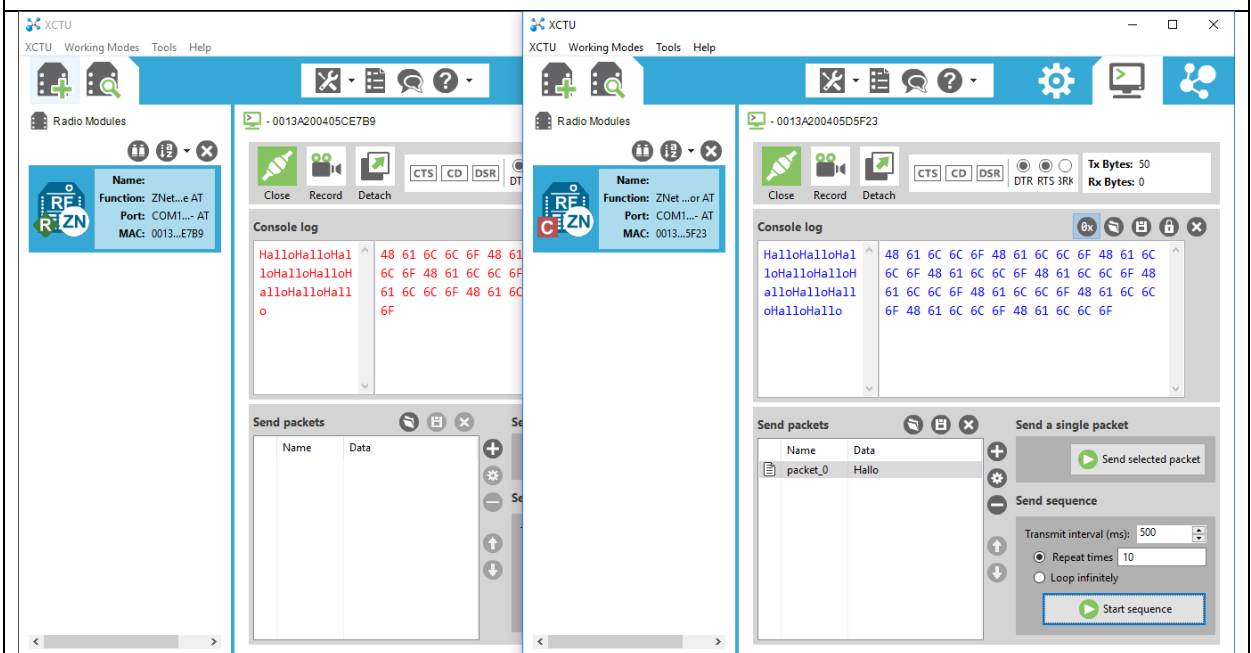
3.1.4.1 Prueba A

a) XBee ↔ XBee

Tabla 3. 1. Pruebas realizadas para la comunicación entre XBee

Programación de los XBee en XCTU	
<p>Configuración del XBee receptor, el cual está ubicado en el robot.</p>	<p>Configuración del XBee emisor, que se encuentra en la planta.</p>
Pruebas de comunicación entre los XBee: (prueba 1)	
<p>El receptor los recibió los caracteres correctamente.</p>	<p>El emisor envió los caracteres de "hallo", "h", "l", "1" y "0" una sola vez cada uno.</p>

Pruebas de comunicación entre los XBee: (prueba 2)

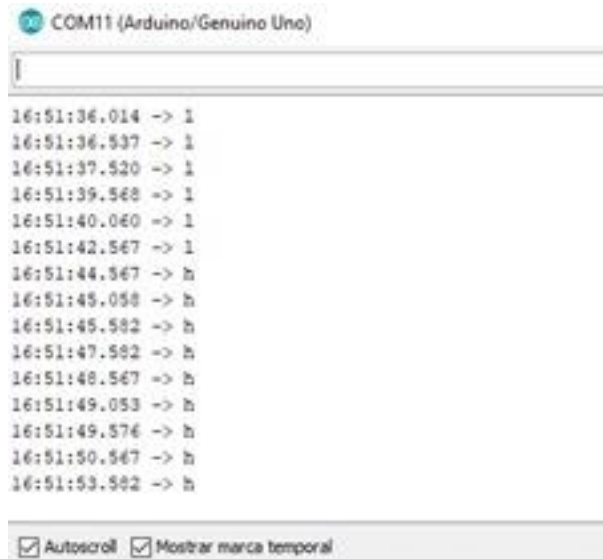


El receptor recibe con éxito la palabra “hallo” infinitas veces, lo cual da un porcentaje alto de seguridad de que la comunicación entre los XBee no fallará.

Ahora el emisor envía la palabra “hallo” infinitas veces, esto es para comprobar que no se vaya a trabar o colgar en el envío de la señal, ya que en el proyecto la señal de humedad se estará enviando constantemente.

b) XBee ↔ Arduino

Se realizaron los códigos mostrados en la sección de ANEXO B para realizar la primera prueba de enviar sólo “H” en HIGH y “L” en LOW. La figura 3.10 muestra el monitor del receptor donde se aprecia las señales que le son enviadas varias veces por un periodo de 5 segundos. Mientras que en la figura 3.11 se muestra la señal de humedad del higrómetro de “1” equivale a seco y “0” húmedo, para esto se modificó unos parámetros en el primer código.

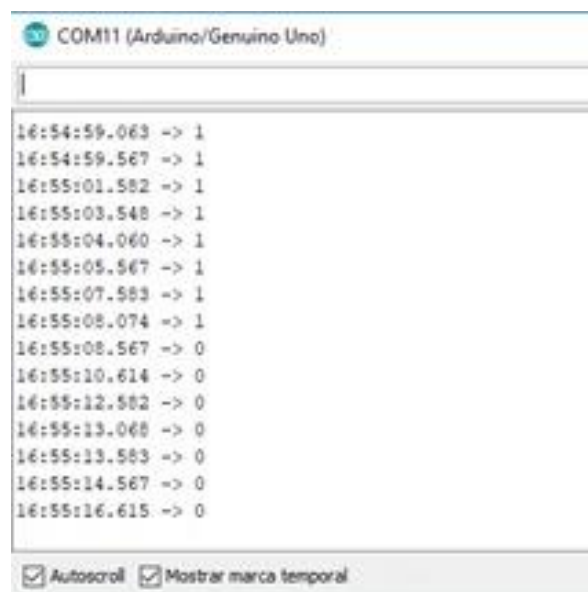


```
COM11 (Arduino/Genuino Uno)

16:51:36.014 -> l
16:51:36.537 -> l
16:51:37.520 -> l
16:51:39.568 -> l
16:51:40.060 -> l
16:51:42.567 -> l
16:51:44.567 -> h
16:51:45.050 -> h
16:51:45.582 -> h
16:51:47.582 -> h
16:51:48.567 -> h
16:51:49.053 -> h
16:51:49.576 -> h
16:51:50.567 -> h
16:51:53.582 -> h

 Autoscroll  Mostrar marca temporal
```

Figura 3. 10. Señal de “l” y “h” que llega al receptor



```
COM11 (Arduino/Genuino Uno)

16:54:59.063 -> 1
16:54:59.567 -> 1
16:55:01.582 -> 1
16:55:03.548 -> 1
16:55:04.060 -> 1
16:55:05.567 -> 1
16:55:07.583 -> 1
16:55:08.074 -> 1
16:55:08.567 -> 0
16:55:10.614 -> 0
16:55:12.582 -> 0
16:55:13.068 -> 0
16:55:13.583 -> 0
16:55:14.567 -> 0
16:55:16.615 -> 0

 Autoscroll  Mostrar marca temporal
```

Figura 3. 11. Señal del higrómetro que es enviada por los XBees a los Arduinos

Al tener una comunicación confiable entre los dispositivos, se procedió a adaptar en el código que va al Arduino B para que pueda accionar la bomba cuando esta sea necesaria y mida el nivel de agua en el reservorio.

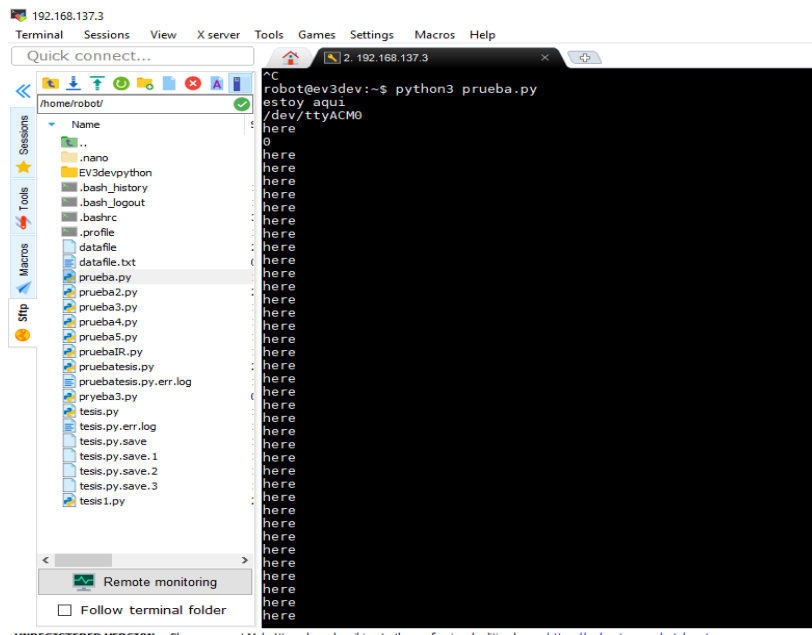


Figura 3. 13. Señal del Brick → Arduino

Se prefirió trabajar con el segundo código (Brick→Arduino), debido a que con el primer código se tenía muchas fallas, entre ellas:

- A veces no reconocía el serial donde se encuentra conectado el Arduino, por lo tanto no se leía nada.
- Además la señal se guardaba en un documento por lo que se complicaba el comparar la señal del Arduino en el código de Python.

3.1.4.2 Prueba B

Con los requerimientos y ecuaciones descritas en el capítulo anterior, además de que la altura de descarga es de 25 [cm], se tuvo que la potencia hidráulica (P_w) debe ser igual a 0.058 [W]. La bomba con la que se trabajó fue de 12 [V], pero la batería que se conectó era de 9 [V] y con corriente de 10 [mA], por lo que la potencia del motor fue de 0.09 [W]. Con las dos potencias se obtuvo que la eficiencia con la que trabaja la motor-bomba es del 65%, 15% menor a la que normalmente trabajaría una de esta clase conectándola directamente a una batería de 12 [V] de autos. Los cálculos se detallan en la sección de ANEXO A.

Las pruebas que se realizaron con la bomba permitieron obtener los datos necesarios para trazar las curvas de Cabezal vs Caudal (*figura 3.14*) y posteriormente la curva de Potencia hidráulica vs Revoluciones (*figura 3.15*). Los cálculos necesarios que se realizaron para dichas curvas se muestran en la sección de *ANEXOS A*.

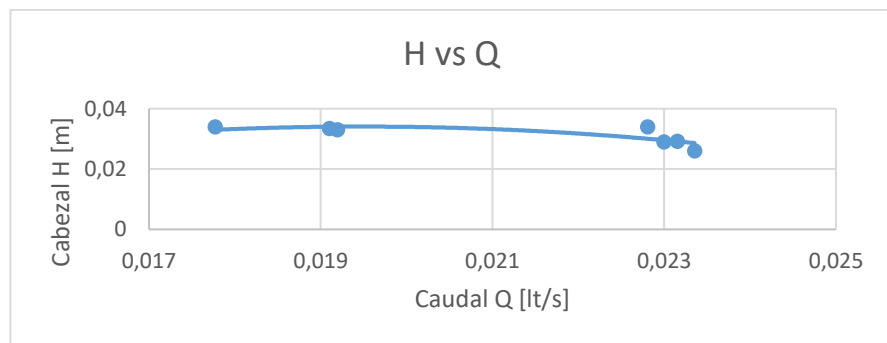


Figura 3. 14. Gráfica de Cabezal vs Caudal de la bomba

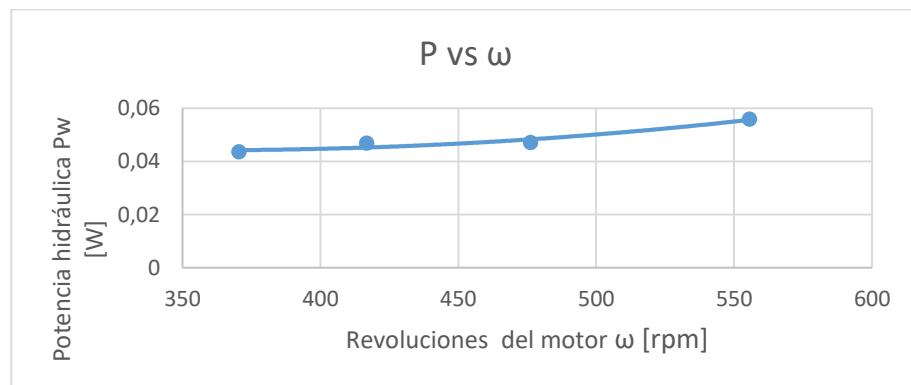


Figura 3. 15. Curva de trabajo de la bomba (Potencia hidráulica vs Revoluciones)

3.1.4.3 Prueba C

a.) Prueba en liso

Se logró realizar las pruebas detalladas en el capítulo anterior con éxito. El robot no tuvo ningún problema en llegar hasta la planta y regar en ninguna de las tres distancias propuestas, lo que comprueba el alcance que los módulos de XBee si tienen para su comunicación. El único problema que se tuvo fueron los motores, los cuales se programaron para que ambos

anden a la misma velocidad, lo que provocó que el robot se moviera hacia un lado. Para corregir el problema se cambiaron las revoluciones por minutos, para un motor se lo programó al 90% y para el otro al 85% de sus velocidades.

b.) Prueba en irregular

Para esta prueba se buscó primero un área con irregularidades muy pronunciadas, en la cual el robot tuvo algunos problemas al andar; por lo tanto, se decidió buscar otra superficie sin irregularidades tan prominentes. Para este caso el robot logró moverse con más facilidad y llegar al objetivo y regar, como en la prueba anterior también se cambiaron las velocidades de los motores al 85% uno y 80% el otro.

3.2 Análisis de costos

En esta sección se analizan todos los costos relacionados con los componentes para la construcción del equipo. Se decidió analizar de manera independiente el circuito de la planta y el del robot, debido a que para aumentar el número de plantas sólo es necesario el circuito ubicado en la planta. Por otra parte los materiales y sensores ubicados en el robot no requieren cambios para trabajar con una mayor cantidad de plantas.

Tabla 3. 2. Costo del circuito del Robot

Materiales	Cantidad	Costo unidad	Costo total
Mindstorms EV3	1	339.46	339.46
Arduino UNO R3	1	10	10
Sensor ultrasónico HC-SR04	1	4	4
Xbee	1	43	43
Motor-Bomba 12V	1	10	10
Batería 9V	1	0.85	0.85
Módulo Relay	1	1	1
Reservorio	1	8	8
Caja del circuito	1	3	3
TOTAL			419.31

El costo para la construcción del robot es de \$419.31, el cual es relativamente elevado debido que el equipo Mindstorms EV3 tiene un precio alto, sin embargo, si sólo tomamos en cuenta los sensores el precio disminuye considerablemente.

Tabla 3. 3. Costo del circuito de la planta

Materiales	Cantidad	Costo unidad	Costo total
Sensor higrómetro FC-28	1	2	2
Xbee	1	43	43
Batería 9V	1	0.85	0.85
Arduino UNO	1	10	10
Caja del circuito	1	3	3
TOTAL			58.85

El costo para el circuito de la planta es de \$58.85, si se desea trabajar con más plantas se multiplica el número de plantas por el valor total obtenido en la *tabla 3.3*.

Por lo tanto se obtiene un costo total neto de \$478.16 para una planta o maceta, si se requiere regar más de una planta o diferentes tipos de plantas, se necesitará obtener más kits para sensar la humedad en las plantas.

CAPÍTULO 4

4. CONCLUSIONES Y RECOMENDACIONES

Conclusiones

- Se realizó de manera exitosa la comunicación entre la planta y el robot, mediante los módulos inalámbricos zigbee, así como la conexión entre el Arduino B y el Brick.
- El modelo EV3 logró ser estable con el diseño propuesto y transportar la carga adicional del reservorio/bomba y circuito con facilidad, tanto en superficies lisas como irregulares.
- Se programó los controles con el programa Arduino IDE y el movimiento del robot con Python 3.
- El costo total neto para la construcción de este robot fue de \$478.16, en el que se incluye el kit completo de los MINDSTORMS.
- Debido a que no se logró encontrar la librería para colocar al sensor infrarrojo en modo de búsqueda, se trabajó con el sensor de color para que encuentre la planta, programando el blanco como el color predeterminado para indicar que ya llegó a la planta.
- Se obtuvo una eficiencia de 65% para el motor-bomba y se determinó la curva de trabajo para la bomba a diferentes revoluciones del motor, midiendo el caudal a cada revolución.

Recomendaciones

- Se recomienda utilizar el sensor infrarrojo en modo de búsqueda (seek) para que se dirija hacia la planta con mayor eficiencia.
- Extender la utilidad del proyecto aumentando el número y variedad de plantas a sensar, por lo que se debería proponer como un proyecto integrado multidisciplinario.
- Se recomienda que el programa usado en el Brick sea un archivo ejecutable desde el robot sin necesidad de tenerlo conectado a la computadora.
- Es recomendable tener conocimientos previos acerca de dispositivos electrónicos, ya que se necesita realizar varias conexiones durante el

proyecto, por lo tanto, es importante identificar las terminales positiva y negativa de cada elemento, así como realizar conexiones en serie y/o paralelo.

- Programar de manera correcta los XBee para tener una comunicación exitosa, se necesita un coordinador (módulo situado en el robot) y el número de emisores dependerá directamente del número de plantas, ya que por cada planta necesitamos un módulo emisor.
- Debido a que se construyó con un prototipo con el Lego Mindstorms EV3 y no se conocía la capacidad máxima de carga del equipo se trabajó con reservorio de 1 lt de capacidad máxima, sería recomendable trabajar con un equipo de mayor potencia para así poder aumentar la capacidad del reservorio con el fin de mejorar el proyecto.

BIBLIOGRAFÍA

Frank, R. H., & Bernanke, B. (2007). Principles of macro- economics (3rd ed.). Boston, MA: McGraw-Hill/Irwin.

Dynamics of content propagation in BitTorrent like P2P file exchange systems. Artículo presentado en 50th IEEE Conference on Decision and Control and European Control Conference (IEEE CDC – ECC 2011), Orlando, Estados Unidos.

Kidpsych is an excellent website for young children. Accedido el 4 de abril, 2012, desde <http://www.kidpsych.org>.

ryant, P. (1999). Biodiversity and Conservation. Accedido el 21 de agosto, 2012 desde <http://darwin.bio.uci.edu/~sustain/bio65/Titlpage.htm>

Koo, D. J., Chitwoode, D. D., & Sanchez, J. (2008). Violent victimization and the routine activities/lifestyle of active drug users. *Journal of Drug Issues*, 38, 1105-1137. Accedido el 4 de abril, 2012, desde <http://www2.criminology.fsu.edu/~jdi/>

Keller, T. E., Cusick, G. R., & Courtney, M. E. (2007). Approaching the transition to adulthood: Distinctive profiles of adolescents aging out of the child welfare system. *Social Services Review*, 81, 453-484.

APÉNDICES

APÉNDICE A

A1 Cálculos para la eficiencia

- *Caudal del fluido*

$$Q = V/t \quad (2.1)$$

$$Q = 2.38 \times 10^{-4} [m^3] / 10 [s]$$

$$Q = 2.38 \times 10^{-5} [m^3/s]$$

$$Q = 0.0238 [lt/s]$$

- *Potencia del fluido*

$$P_w = \rho g Q H \quad (2.2)$$

$$P_w = 1000 [kg/m^3] * 9.8 [m/s^2] * 2.38 \times 10^{-5} [m^3/s] * 0.25 [m]$$

$$P_w = 0.0583 [W]$$

- *Potencia motor*

$$P_f = V * I \quad (2.3)$$

$$P_f = 9 [V] * 0.01 [A]$$

$$P_f = 0.09 [W]$$

- *Eficiencia*

$$\eta = \frac{P_w}{P_f} \quad (2.4)$$

$$\eta = \frac{0.0583 [W]}{0.09 [W]} \%$$

$$\rightarrow \eta = 65\% \leftarrow$$

A2 Cálculos representativos para las curvas de trabajo del motor-bomba

- *Caudal del fluido*

$$Q = \forall / t \quad (2.1)$$

$$Q = 1.78 \times 10^{-4} [m^3] / 10 [s]$$

$$Q = 1.78 \times 10^{-5} [m^3/s]$$

$$Q = 0.0178 [lt/s]$$

- *Potencia del fluido*

$$P_w = \rho ghQ \quad (2.2)$$

$$P_w = 1000 [kg/m^3] * 9.8 [m/s^2] * 0.25 [m] * 1.78 \times 10^{-5} [m^3/s]$$

$$P_w = 0.0436 [W]$$

Nota: Los datos de Cabezal y Revoluciones se los obtuvieron del código.

APÉNDICE B

B1 Códigos para pruebas Xbee ↔ Arduino

Emisor

```
#include <SoftwareSerial.h>

int ledpin =13;
SoftwareSerial mySerial(8, 9); // RX, TX

void setup(){
  mySerial.begin(9600);
  pinMode(ledpin, OUTPUT);
  Serial.begin(9600);
}

void loop(){
  mySerial.println('h');
  mySerial.println('h');
  mySerial.println('h');
  mySerial.println('h');
  mySerial.println('h');
  mySerial.println('h');
  digitalWrite(ledpin,HIGH);
  delay(5000);

  mySerial.println('l');
  mySerial.println('l');
  mySerial.println('l');
  mySerial.println('l');
  mySerial.println('l');
  digitalWrite(ledpin,LOW);
  delay(5000);
}
```

Receptor

```
#include <SoftwareSerial.h>

const int ledPin = 13;
int mensaje;
SoftwareSerial mySerial(8, 9); // RX, TX

void setup(){
  mySerial.begin(9600);
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
}

void loop(){
  while(mySerial.available() > 0)
    Serial.println("dentro de my serial");
  {
    mensaje = mySerial.read();
  }
  if(mensaje == 'h'){
    Serial.println('h');
    digitalWrite(ledPin, HIGH);
  }
  else if(mensaje == 'l'){
    Serial.println('l');
    digitalWrite(ledPin, LOW);
  }
  else{}
}
```


B2 Código para prueba Arduino ↔ Brick

<i>Arduino</i>	Brick
<pre>const int ledPin = 13; int controlMoverse; const int VCCPin = 11; const int GNDPin = 10; SoftwareSerial mySerial(8, 9); // RX, TX void loop() { Serial.begin(9600); pinMode(VCCPin, OUTPUT); pinMode(GNDPin, OUTPUT); digitalWrite(VCCPin,HIGH); digitalWrite(GNDPin,LOW); pinMode(ledPin, OUTPUT); mySerial.begin(9600); } void loop() { while(mySerial.available(>0){ controlMoverse=mySerial.read(); } if(controlMoverse == '1' && aviso==0)//SECO { Serial.println('1'); Serial.println('1'); Serial.println('1'); Serial.println('1'); delay(100);</pre>	<pre>#!/usr/bin/env python3 from ev3dev.ev3 import * from threading import Thread import os def iniciar(): os.system('stty -F /dev/ttyACM0 cs8 9600 ignbrk -brkint -icrnl -imaxbel -opost -onlcr -isig -icanon -iexten -echo -echoe -echok -echoctl -echoke noflsh -ixon - crtcts ') print(os.system(' cat /dev/ttyACM0 > datafile')) t = Thread(target=iniciar) t.start() with open('datafile','r') as data: while True: a = int(data.read(1)) print(a) data.close()</pre>

```

    digitalWrite(ledPin, HIGH);
    controlMoverse=NULL;
    aviso=0;
}
else if(controlMoverse == '0' &&
aviso==0)//HUMEDO
{
    Serial.println('2');
    Serial.println('2');
    Serial.println('2');
    Serial.println('2');
    Serial.println('2');
    delay(100);
    digitalWrite(ledPin, LOW);
    controlMoverse=NULL;
    aviso=0;
}
}

```

Brick

```

#!/usr/bin/env python3
from ev3dev.ev3 import *
from threading import Thread
import time
import os
import serial

ser= serial.Serial('/dev/ttyACM0',9600)
print(ser.name)
bandera=1
while True:
    for i in range(25):
        ser.write(b'1')

```

Arduino

```

const int pruebaled= 4;
int luz=0;

void setup(){

    Serial.begin(9600);
    digitalWrite(pruebaled, LOW);
}

void loop(){
    luz= Serial.read();
    if (luz=='1'){

```

```
ser.write(b'0/n')  
print("here")  
ser.close()
```

```
digitalWrite(pruebaled, HIGH);  
delay(5000);  
}  
else{  
digitalWrite(pruebaled, LOW);
```

B3 Código para pruebas de bomba

```
const int bombaPin = 3;
const int PinTrig = 7;
const int PinEcho = 6;

const float VelSon = 34000.0;
const float pi = 3.14;

float distancia;
float actual=0;
float temporal_d;
float resta=0;
float volumen;
float presion1;
float presion2;
int bandera=1;

void setup()
{
  Serial.begin(9600);
  pinMode(PinTrig, OUTPUT);
  pinMode(PinEcho, INPUT);
  pinMode(bombaPin, OUTPUT);
  digitalWrite(bombaPin, LOW);
  Serial.println("Eliga de 0 a 9 la velocidad de giro del motor");
}

void loop()
{
  digitalWrite(bombaPin, LOW);
  if (Serial.available()){
```

```

char a = Serial.read();

if (a>='0' && a<='9'){

    iniciarTrigger();
    unsigned long tiempo = pulseIn(PinEcho, HIGH);
    temporal_d = tiempo * 0.000001 * VelSon / 2.0;
    Serial.print("h1: ");
    Serial.println(temporal_d);

    int velocidad = map(a,'0','9',0,255);
    analogWrite(bombaPin,velocidad);
    Serial.print("El motor esta girando a la velocidad ");
    Serial.println(a);
    delay(10000);

    iniciarTrigger();
    tiempo = pulseIn(PinEcho, HIGH);
    actual = tiempo * 0.000001 * VelSon / 2.0;
    Serial.print("h2: ");
    Serial.println(actual);
    resta=actual-temporal_d;
    Serial.println(resta);
    volumen=pi*11*11*resta;
    Serial.print("V: ");
    Serial.println(volumen);
    Serial.println("Eliga de 0 a 9 la velocidad de giro del motor");
    delay(1000);
}
else {

```

```
    Serial.print("Velocidad invalida");  
    Serial.println(a);  
  }  
}  
}
```

```
void iniciarTrigger()  
{  
  digitalWrite(PinTrig, LOW);  
  delayMicroseconds(2);  
  
  digitalWrite(PinTrig, HIGH);  
  delayMicroseconds(10);  
  
  digitalWrite(PinTrig, LOW);  
}
```

B4 Código Emisor, Arduino A

```
#include <SoftwareSerial.h>
/*
  XBEE - Emisor

*/
const int VCCPin = 11;
const int GNDPin = 10;
SoftwareSerial mySerial(8, 9); // RX, TX
//const int RESETPin = 8;
const int ledPin = 13;
const int sensorPin = 4;
void setup(){

  Serial.begin(9600); //Iniciar el Serial
  pinMode(sensorPin, INPUT);
  pinMode(ledPin, OUTPUT);
  pinMode(VCCPin, OUTPUT);
  pinMode(GNDPin, OUTPUT);
  // pinMode(RESETPin,OUTPUT);
  digitalWrite(VCCPin,HIGH);
  digitalWrite(GNDPin,LOW);
  //digitalWrite(RESETPin,HIGH);
  mySerial.begin(9600);
}

void loop()
{
  int humedad = digitalRead(sensorPin);
  Serial.println(humedad);
```

```
if (humedad ==1)
{
  mySerial.print('1'); //seco
  digitalWrite(ledPin,HIGH);
}else{
  //Serial.println(humedad);
  mySerial.print('0'); //humedo
  digitalWrite(ledPin,LOW);
}
delay(1000);
}
```


B5 Código Receptor, Arduino B

```
#include <SoftwareSerial.h>

const int ledPin = 13;
const int bombaPin = 4;
const int PinTrig = 7;
const int PinEcho = 6;

int mensaje;
int controlMoverse;
const int VCCPin = 11;
const int GNDPin = 10;
SoftwareSerial mySerial(8, 9); // RX, TX

const float VelSon = 34000.0;
float distancia;

float actual=0;
int bandera=1;
unsigned long tiempo;
float temporal_d;
float resta=0;
int a=0;

void setup() {

  Serial.begin(9600);
  pinMode(bombaPin, OUTPUT);
  digitalWrite(bombaPin, LOW);
```

```
int velocidad = map(a,'0','9',0,255);
digitalWrite(bombaPin,velocidad);
```

```
pinMode(VCCPin, OUTPUT);
pinMode(GNDPin, OUTPUT);
digitalWrite(VCCPin,HIGH);
digitalWrite(GNDPin,LOW);
```

```
pinMode(ledPin, OUTPUT);
mySerial.begin(9600);
```

```
pinMode(PinTrig, OUTPUT);
pinMode(PinEcho, INPUT);
}
```

```
void iniciarTrigger()
{
  digitalWrite(PinTrig, LOW);
  delayMicroseconds(2);

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

  digitalWrite(PinTrig, LOW);
}
```

```
int aviso=0;
void loop() {

if (Serial.available() > 0) {
  mensaje = Serial.read();
  if (mensaje == '1') {
```

```
aviso=1;
bandera=1;

iniciarTrigger();
unsigned long tiempo = pulseIn(PinEcho, HIGH);
distancia = tiempo * 0.000001 * VelSon / 2.0;

while(distancia>9) // Señal que falta agua
{
    Serial.println('3');
    Serial.println('3');
    Serial.println('3');
    Serial.println('3');
    Serial.println('3');
    Serial.println('3');
    Serial.println('3');
    Serial.println('3');
    Serial.println('3');
    Serial.println('3');
    Serial.println('3');
    Serial.println('3');

    iniciarTrigger();
    tiempo = pulseIn(PinEcho, HIGH);
    distancia = tiempo * 0.000001 * VelSon / 2.0
}

iniciarTrigger();
tiempo = pulseIn(PinEcho, HIGH);
temporal_d = tiempo * 0.000001 * VelSon / 2.0;

if (resta > 1.2){
    digitalWrite(bombaPin, LOW); //Bomba off
```

```

    Serial.println('4');
    Serial.println('4');
    Serial.println('4');
    Serial.println('4');
    Serial.println('4');
    Serial.println('4');
    Serial.println('4');
    Serial.println('4');
    Serial.println('4');
    Serial.println('4');
    aviso=0;
    bandera=0;
}
else{
    digitalWrite(bombaPin, HIGH); //Bomba on
    iniciarTrigger();
    tiempo = pulseIn(PinEcho, HIGH);
    actual = tiempo * 0.000001 * VelSon / 2.0;
    resta=actual-temporal_d;
}
}

```

```

while(mySerial.available(>0){

```

```

    controlMoverse=mySerial.read();
}

```

```

if(controlMoverse == '1' && aviso==0) //SECO

```

```

{
    Serial.println('1');
    Serial.println('1');
    Serial.println('1');
}

```

```
Serial.println('1');
Serial.println('1');
Serial.println('1');
Serial.println('1');
Serial.println('1');
Serial.println('1');
Serial.println('1');
digitalWrite(ledPin, HIGH);
controlMoverse=NULL;
}
else if(controlMoverse == '0' && aviso==0) //HUMEDO
{
Serial.println('2');
Serial.println('2');
Serial.println('2');
Serial.println('2');
Serial.println('2');
Serial.println('2');
Serial.println('2');
Serial.println('2');
Serial.println('2');
Serial.println('2');
Serial.println('2');
Serial.println('2');
digitalWrite(ledPin, LOW);
controlMoverse=NULL;
}
}
```

Código de movimiento del robot, Brick EV3

```
#!/usr/bin/env python3
from ev3dev.ev3 import *
from threading import Thread
import time
from time import sleep
import os
import serial

mA = LargeMotor('outA')
mB = LargeMotor('outB')

ser= serial.Serial('/dev/ttyACM0',9600)

bandera=0
while(True):

    boton=ser.read()

    print("boton",boton)
    print(type(boton))

    if boton == b'1':      #SECO
        print("estoy en a")
        bandera=1
        mA.run_forever(speed_sp=990)
        mB.run_forever(speed_sp=900)

    elif boton == b'2':    #HUMEDO
        print("estoy en z")
        bandera=0
        mA.stop(stop_action="hold")
```

```

mB.stop(stop_action="hold")

elif boton == b'3':      #NO HAY AGUA
    Sound.beep()
    Sound.tone(2000, 1500)

elif boton == b'4':      #TERMINÓ DE BOMBLEAR
    mA.run_timed(time_sp=4000, speed_sp=-900)
    mB.run_timed(time_sp=4000, speed_sp=-700)
    sleep (2)
    mA.stop(stop_action="hold")
    mB.stop(stop_action="hold")
    mA.run_forever(speed_sp=0)
    mB.run_forever(speed_sp=0)

else:
    print('nada')
    mA.stop(stop_action="hold")
    mB.stop(stop_action="hold")
    mA.run_forever(speed_sp=0)
    mB.run_forever(speed_sp=0)

cl = ColorSensor()
cl.mode='COL-COLOR'
colors=('0','1','2','3','4','5','6','7')
a = str (colors[cl.value()])
print("color",a)
if a=='6' and bandera==1:
    mA.run_timed(time_sp=4000, speed_sp=0)
    mB.run_timed(time_sp=4000, speed_sp=0)
    sleep(4)
    mA.stop(stop_action="hold")

```

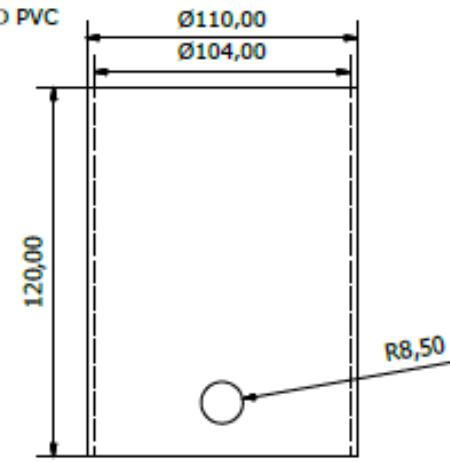
```
mB.stop(stop_action="hold")
for i in range(30):
    ser.write(b'1')
print("llegaste a la planta")
mA.run_forever(speed_sp=0)
mB.run_forever(speed_sp=0)
sleep(4)
```

```
ser.close()
mB.run_forever(0)
mC.run_forever(0)
```

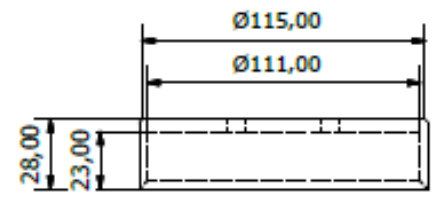

APÉNDICE C
PLANOS

6 5 4 3 2 1

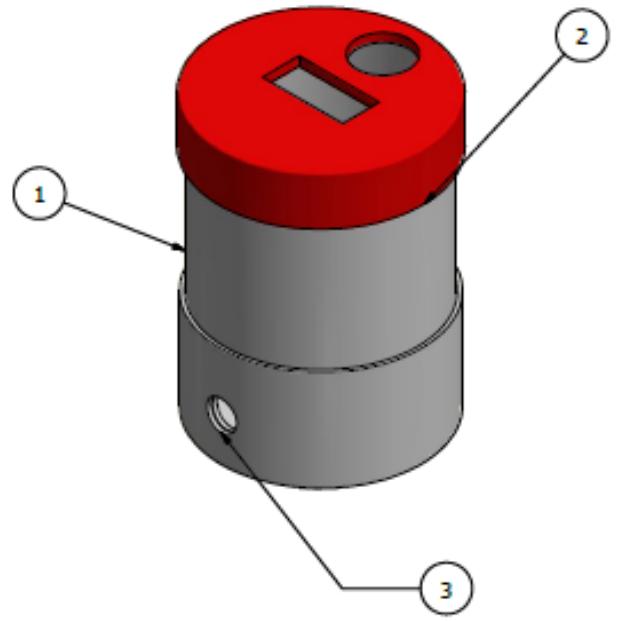
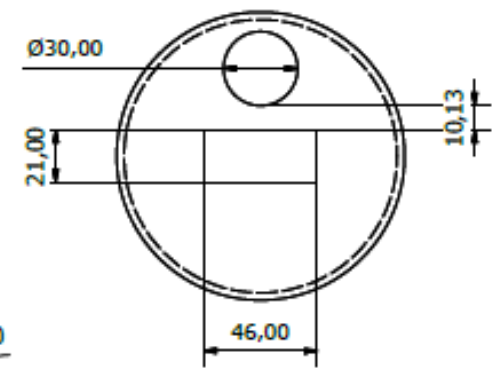
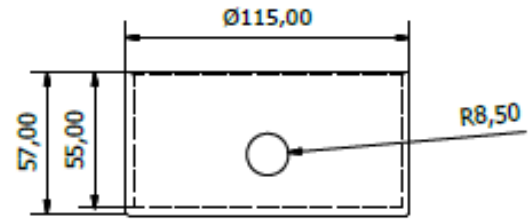
1. TUBO PVC



2. TUBO SUPERIOR

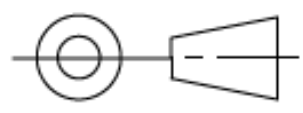


3. TAPA INFERIOR



D
C
B
A

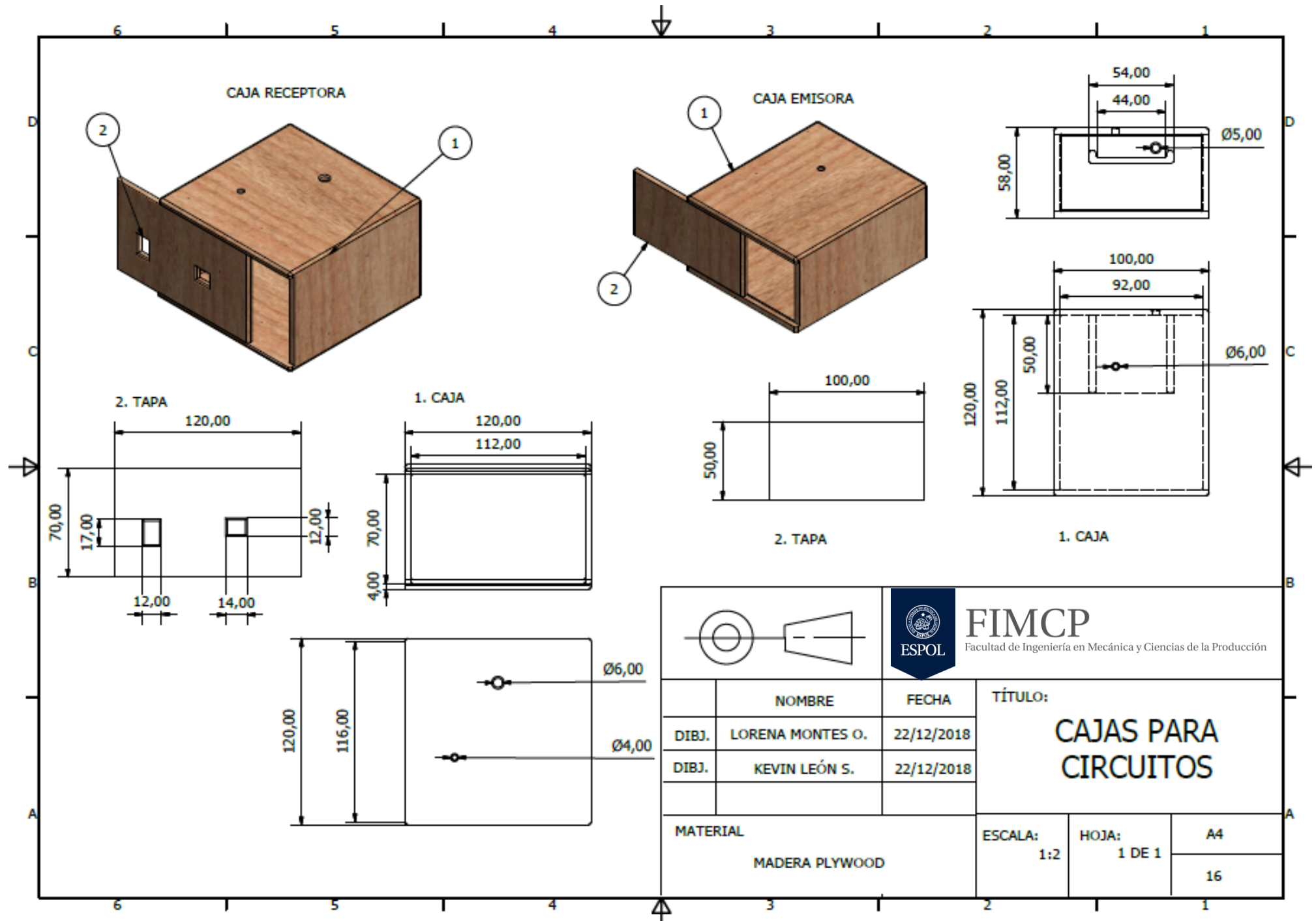
D
C
B
A



FIMCP
Facultad de Ingeniería en Mecánica y Ciencias de la Producción

NOMBRE		FECHA	TÍTULO:	
DIBJ.	LORENA MONTES O.	22/12/2018	RESERVORIO PARA AGUA	
DIBJ.	KEVIN LEÓN S.	22/12/2018		
MATERIAL			ESCALA:	HOJA:
1. PVC 2. PLA 3. PVC			1:2	1 DE 1
				A4
				16

6 5 4 3 2 1



CAJA RECEPTORA

CAJA EMISORA

2. TAPA

1. CAJA

2. TAPA

1. CAJA

				FIMCP Facultad de Ingeniería en Mecánica y Ciencias de la Producción	
	NOMBRE	FECHA	TÍTULO:		
DIBJ.	LORENA MONTES O.	22/12/2018	<h1>CAJAS PARA CIRCUITOS</h1>		
DIBJ.	KEVIN LEÓN S.	22/12/2018			
MATERIAL			ESCALA:	HOJA:	A4
MADERA PLYWOOD			1:2	1 DE 1	16

REGISTRO DE REUNIONES Y ENTREVISTAS



FIMCP
Facultad de Ingeniería en Mecánica
y Ciencias de la Producción

Materia Integradora de Ingeniería Mecánica MECG1026, 2018 - Termino I
Registro de Reuniones y Actividades del Proyecto

Paralelo: 3

Profesor Coordinador: Francis Roderich Loayza Paredes

Proyecto: "Diseño de un robot móvil para riego automático de plantas"

Reunión No.	Fecha	Estudiante	Profesor Tutor	Actividad / Avance / Comentarios
1	11/10/2018	Kevin León Suárez Lorena Montes Ortega	Francis Loayza	- Revisión & entrega del pcta de constitución del Proyecto
2	18/10/2018	Kevin León Suárez Lorena Montes Ortega	Efraín Terán	- Entrega del Robo MINDSTORMS EV3
3	25/10/2018	Kevin León Suárez Lorena Montes Ortega	Francis Loayza	- Primera revisión del capítulo 1
4	09/11/2018	Kevin León Lorena Montes	Francis Loayza	- Segunda revisión del capítulo 1
5	15/11/2018	Kevin León Lorena Montes	Francis Loayza	- Entrega final del capítulo 1 - Primera revisión del capítulo 2
6	14/11/2018	Lorena Montes	Cristhian Sacarello	- Asesoría para la comunicación entre los XBers con Arduino
7	20/11/2018	Kevin León Lorena Montes	Francis Loayza	- Segunda revisión del capítulo 2 (Diseño detallado)
8	06/12/2018	Kevin León Lorena Montes	Francis Loayza	- Tercera revisión del capítulo 2
9	14/12/2018	Kevin León Lorena Montes	Cristhian Sacarello	- Realización de Pruebas de comunicación entre los XBee con arduino
10	20/12/2018	Kevin León Lorena Montes	Efraín Terán	- Prestación de filamento para impresión 3D del reservorio
11	10/01/2018	Kevin León Lorena Montes	Francis Loayza	- Revisión del capítulo 2 - Primera entrega del capítulo 3
12	11/01/2018	Kevin León Lorena Montes	Geoncarlos Zamora	- Asesoría para la construcción del reservorio & para la realización de pruebas para la Bomba
13	15/01/2018	Kevin León Lorena Montes	Cristhian Sacarello	- Pruebas de comunicación entre Arduino & Brick EV3
14	17/01/2018	Kevin León Lorena Montes	Francis Loayza	- Revisión del cap. 2 (Metodología de Pruebas) - Revisión del cap 3 (Resultados / Conexión)
15	21/01/2018	Kevin León Lorena Montes	Cristhian Sacarello	- Segunda prueba de comunicación entre Arduino & Brick ev3
16	25/01/2018	Kevin León	Francis Loayza	- Revisión del cap. 2 & 3 - Revisión de la construcción de Reservorio/Bomba