UNIVERSIDAD HISPANOAMERICANA

INGENIERÍA EN ELECTRÓNICA

PARA OPTAR POR EL GRADO DE BACHILLERATO

DESARROLLO E IMPLEMENTACIÓN DE UN PROTOTIPO DE SILLA DE RUEDAS ELECTRICA CONTROLADA POR ARDUINO PARA II CUATRIMESTRES DE AÑO 2016

Nervin Rivas Jerez

Tutor: Ing. Jorge Villalobos Cascante

Noviembre, 2016

UNIVERSIDAD HISPANOAMERICANA
Seriedad y Prestigio

DECLARACIÓN JURADA

Yo Nervin Jose Rivas Jerez , cédula de identidad número 155812068608, en condición de egresado de la carrera de Ingeniería en electrónica de la Universidad Hispanoamericana, y advertido de las penas con las que la ley castiga el falso testimonio y el perjurio, declaro bajo la fe del juramento que dejo rendido en este acto, que: A) mi trabajo de graduación, para optar por el título de Bachillerato titulado "DESARROLLO E IMPLEMENTACIÓN DE UN PROTOTIPO DE SILLA DE RUEDAS ELECTRICA CONTROLADA POR ARDUINO PARA II CUATRIMESTRES DEL AÑO 2016" es una obra original y para su realización he respetado todo lo preceptuado por las Leyes Penales, así como la Ley de Derechos de Autor y Derecho Conexos, número 6683 del 14 de octubre de 1982 y sus reformas, publicada en la Gaceta número 226 del 25 de noviembre de 1982; especialmente el numeral 70 de dicha ley en el que se establece: *"Es permitido citar a un autor, transcribiendo los pasajes pertinentes siempre que éstos no sean tantos y seguidos, que puedan considerarse como una producción simulada y sustancial, que redunde en perjuicio del autor de la obra original".* B) conozco y acepto que la Universidad se reserva el derecho de protocolizar este documento ante Notario Público.C) conozco los reglamentos y procedimientos que rigen la modalidad de Proyectos y acepto los términos de estos. Firmo, en fe de lo anterior, en la ciudad de Llorente de Tibás, San Jose, el 19 de noviembre de 2016.

Nervin Jose Rivas Jerez

UNIVERSIDAD HISPANOAMERICANA
SERIEDAD Y PRESTIGIO

CARTA DEL TUTOR

San José, 18 de noviembre del 2016

Señores
Departamento de Registro
Universidad Hispanoamericana

Estimado señor:

El estudiante Nervin Rivas Jerez, cédula de identidad número 155812068608, me ha presentado, para efectos de revisión y aprobación, el trabajo de investigación denominado "*Desarrollo e Implementación de un prototipo de silla de ruedas eléctrica controlada por Arduino en el segundo cuatrimestre del año 2016*", el cual ha elaborado para optar por el grado académico de Bachillerato.

En mi calidad de tutor, he verificado que se han hecho las correcciones indicadas durante el proceso de tutoría y he evaluado los aspectos relativos a la elaboración del problema, objetivos, justificación; antecedentes, marco teórico, marco metodológico, tabulación, análisis de datos; conclusiones y recomendaciones.

De los resultados obtenidos por el postulante, se obtiene la siguiente calificación:

Tabla 1 Calificación del proyecto

#	Rubro	% Teórico	% Asignado
a	Original del tema.	10	9
b	Cumplimiento de entrega de avances.	20	18
c	Coherencia entre los objetivos, los instrumentos aplicados y los resultados de la investigación.	30	30
d	Relevancia de las conclusiones y recomendaciones.	20	19
e	Calidad, detalle del marco teórico.	20	19
Total:		100	95

En virtud de la calificación obtenida, se avala el traslado al proceso de lectura.

Atentamente,

Ing. Jorge Villalobos Cascante
Cédula de identidad: 1-1185-0467
Carné colegio profesional: IEL-22656

UNIVERSIDAD HISPANOAMERICANA
SERIEDAD Y PRESTIGIO

CARTA DEL LECTOR

San José, 02 de ENERO del 2017

Señores
Departamento de Registro
Universidad Hispanoamericana

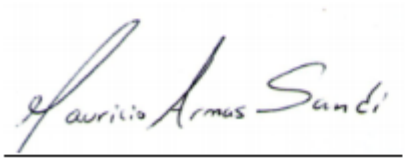
Estimado señor:

El estudiante NERVIN JOSE RIVAS JEREZ, cédula de identidad número 15581206808, me ha presentado, para efectos de revisión y aprobación, el trabajo de investigación denominado *"DESARROLLO E IMPLEMENTACIÓN DE UN PROTOTIPO DE SILLA DE RUEDAS ELECTRICA CONTROLADA POR ARDUINO PARA II CUATRIMESTRES DE AÑO"*, el cual ha elaborado para obtener su grado de Bachillerato.

He revisado y he hecho las observaciones relativas al contenido analizado, particularmente lo relativo a la coherencia entre el marco teórico y análisis de datos, la consistencia de los datos recopilados y la coherencia entre éstos y las conclusiones; asimismo, la aplicabilidad y originalidad de las recomendaciones, en términos de aporte de la investigación. He verificado que se han hecho las modificaciones correspondientes a las observaciones indicadas.

Por consiguiente, este trabajo cuenta con mi aval para ser presentado en la defensa pública.

Atentamente,



Nombre del profesor: Mauricio Daniel Armas Sandí
Cédula de identidad: 1-1361-0843
Carné colegio profesional: IEL-22358

CARTA DE REVISION FILOLÓGICA

Jueves 11 de enero, 2017

Dirección de Registro
Facultad de Ingeniería
Universidad Hispanoamericana

Estimados señores:

Por este medio yo, Karol Jiménez García, mayor, casada, filóloga y profesora de español, incorporada al Colegio de Licenciados y Profesores, con el número de carné: 039257, vecina de Desamparados, portadora de la cédula de identidad 1-1101-0902, hago constar:

1. Que he revisado el trabajo final de graduación para optar por el grado académico de Bachillerato denominado: **"DESARROLLO E IMPLEMENTACIÓN DE UN PROTOTIPO DE SILLA DE RUEDAS ELECTRICA CONTROLADA POR ARDUINO PARA II CUATRIMESTRES DE AÑO 2016".**

2. Que el trabajo final de graduación es sustentado por el estudiante: Nervin Rivas Jerez, cédula: 155812068608
3. Que se le han hecho las correcciones pertinentes en acentuación, ortografía, puntuación, concordancia gramatical y otras del campo filológico.

En espera de que mi participación satisfaga los requerimientos de la Universidad Hispanoamericana, se suscribe atentamente,

Karol Jiménez García
Máster
Carné No. 039257
Filóloga

DEDICATORIA

Este trabajo está dedicado a mis padres Coralia Jerez Salgado y Jose Luis Rivas Jerez, por apoyarme en la realización de este trabajo con su gran amor y consejo. A mi novia, Dessire Jiménez Chacón que valientemente me ha apoyado sin esperar nada a cambio en este trayecto tan importante en mi vida. Además, agradezco a los profesores y demás personas que de una u otra forma han tenido vocación para transmitir y compartir su conocimiento y aportaron con la realización de este proyecto.

AGRADECIMIENTOS

Agradezco a mi novia Dessire Jiménez Chacón que con su apoyo dedicación y entrega pude lograr la finalización de este proyecto.

A mis cuñados Rafael Jiménez Chacón y Gerardo Soto Chacón que me apoyaron con su experiencia en la realización de la estructura del prototipo.

A todas aquellas personas que de una u otra manera me dieron su consejo y ayuda.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO I. PLANTEAMIENTO DEL PROBLEMA

1.1 Introducción al tema del proyecto.

La silla de ruedas es una herramienta para el trasporte de personas que no puedan desplazarse por sí mismas sobre sus extremidades inferiores, de estas hay dos grandes grupos, de acuerdo con su funcionamiento: las sillas manuales (Figura 1.1a) y las sillas eléctricas (figura 1.1b). Las sillas manuales son utilizadas por personas que no presentan ninguna discapacidad en sus extremidades superiores y presentan la suficiente fuerza en éstas para poder moverse de forma independiente, sin embargo, hay personas que por diferentes razones no disponen de la fuerza en sus brazos para mover ese tipo de sillas, por ello deben recurrir a comprar una silla de ruedas eléctrica, la cual entre otras cosas consta de motores que generan la propulsión necesaria para provocar el desplazamiento.

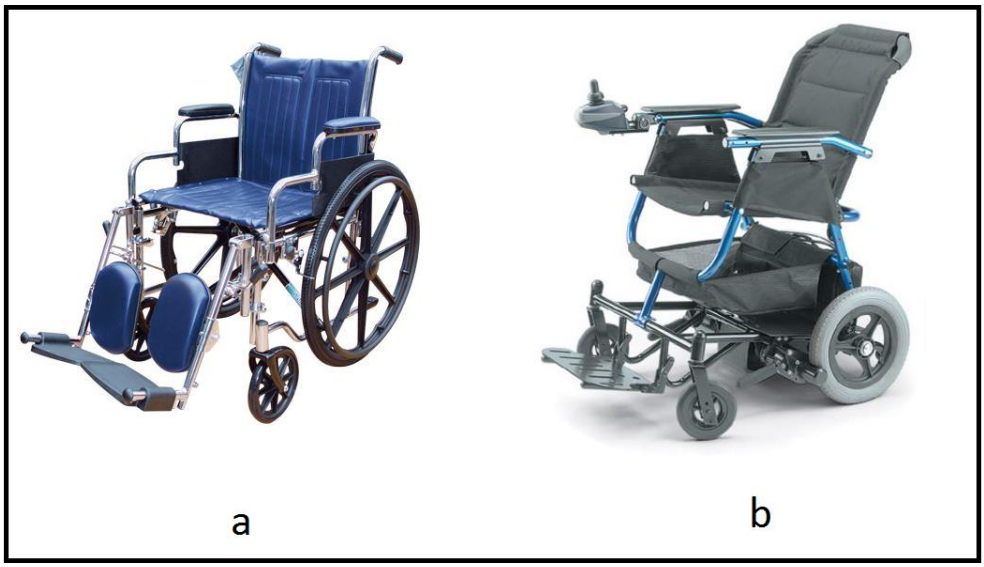


Figura 1 Silla de rueda manual y eléctrica

Estas sillas no son de fácil adquisición debido a su costo, el cual varía dependiendo de los materiales en los que está construida y algunos aspectos

como peso, altura e incluso en cuanto a la personalización debido al acondicionamiento, dependiendo del tipo de discapacidad que tenga el usuario de la misma. Muchas de estas personas se ven obligadas a hacer una gran inversión parar conseguir estas sillas, en el caso de no poder hacerlo, se puede considerar la posibilidad de comprarlas de segunda mano, con el inconveniente de que en muchos casos se encuentran algo dañadas o con algún problema de funcionamiento, lo cual disminuye su funcionalidad.

1.2 Antecedentes de la institución.

Asociación para personas con discapacidad para el progreso de Santa Ana, conocida por sus siglas APEDISTROFA fue fundada en el año 1991, no obstante, cuenta con personería jurídica a partir del año 1996 cuenta con un promedio de 50 personas en la cual hay personas con diferentes discapacidades de estas personas 25 personas utilizan sillas de ruedas y un total de 8 tienen sillas de ruedas eléctricas estas sillas todas son modelos viejos, ya que se compraron de segunda mano y están presentando problemas en sus partes mecánicas y también en la electrónica.

El principal objetivo de esta asociación es dar herramientas para que las personas con discapacidad puedan incorporarse a la sociedad de una forma más independiente, por medio de cursos de auto ayuda, superación y de cómo involucrarse activamente en su entorno, también se imparten talleres de computación, artesanías, ejercicios físicos y planes de reciclaje.

Actualmente su presidenta es Auria Valverde Alfaro, quien es una usuaria de silla de ruedas eléctrica y ha experimentado problemas en su funcionamiento.

Misión

Somos un movimiento asociativo de y para personas discapacitadas sin fines de lucro que promueve la plena integración, la mejora de la calidad de vida, mediante el fomento de iniciativas y actividades que generen oportunidades de empleo y aportan valor, contribuyendo al desarrollo y a la igualdad de oportunidades en la sociedad.

Visión

Ser la organización social líder en innovación del abordaje integral para el desarrollo del potencial de las personas con discapacidad.

1.3 Justificación del proyecto.

El precio excesivo de las sillas de ruedas eléctricas imposibilita a personas de bajos recursos con discapacidades motoras el adquirir una de estas para facilitar su movilidad diaria. Es importante comprender la discapacidad como un tema que involucra a personas, familias, organizaciones y comunidades; lo anterior convierte este sector de la población en un foco investigativo

multidisciplinario, generador y promotor de acciones tendientes a mejorar la calidad de vida de todas las personas.

También se debe tomar en cuenta que muchas de estas sillas de ruedas eléctricas son personalizadas de fábrica, esto quiere decir que el fabricante diseña y construye una silla de ruedas especial para un paciente específico, tomando en cuenta su tamaño, peso, tipo de discapacidad y principalmente adecuándola a sus necesidades, por ello algunos pacientes deben tener cuidado a la hora de adquirir una silla de ruedas de segunda mano, pues ésta podría no adaptarse a sus necesidades.

Es meritorio recalcar como las personas con discapacidad cuentan con los mismos derechos que las demás personas, razón por la cual todos debemos contribuir en la búsqueda de lograr una inclusión de este sector poblacional.

El presente proyecto presenta un enfoque de ayuda social mediante el rediseño de lo relacionado con el control de una silla de ruedas eléctrica, de forma que cuando está presente un daño provocado por problemas en la computadora central y en el *joystick*, se pueda tener acceso a una solución pronta y factible económicamente, con el fin de evitar incurrir en la compra de una nueva.

1.4 Definición del problema.

En Costa Rica, cuando los usuarios de sillas de ruedas eléctricas sufren un desperfecto en la tarjeta principal de las mismas (también llamada computadora central) o en el dispositivo de control, se ven obligados a adquirir repuestos en el extranjero “si los hay” o descartar por completo, pues en el país existen muy pocas empresas que vendan repuestos y que tengan personal técnico capacitado para reparar las sillas.

Esto nos lleva a la pregunta de ¿Cómo implementar un sistema que garantice el funcionamiento de sillas de ruedas eléctricas descontinuadas, a partir de la reutilización de partes?

Este proyecto se propone para ayudar un grupo de apoyo comunal localizado en la municipalidad de Santa Ana, que cuenta con 8 personas usuarias de sillas de ruedas, las cuales provienen de diferentes partes del país para el año 2016.

1.5 Objetivo general y los específicos.

1.5.1 Objetivo General.

Desarrollar un prototipo para el control de sillas de ruedas eléctricas, utilizando una tarjeta Arduino UNO y que además tenga la opción de un control secundario por medio de una aplicación móvil, para el aprovechamiento de tres tipos de modelos diferentes de las sillas descontinuadas, los cuales consisten en Quikie pulse 6, Jassy pride, Invacare Pronto, de la Asociación APEDISTROFA.

1.5.2 Objetivos específicos.

i. Determinar las partes reutilizables de las sillas eléctricas actuales y las tecnologías involucradas en su funcionamiento, mediante un inventario y verificación de los diferentes componentes eléctricos y mecánicos.
ii. Elaborar un prototipo mediante la utilización de una tarjeta Arduino que permita el control centralizado para sillas de ruedas eléctricas por medio de *joystick* y aplicaciones móviles.
iii. Realizar pruebas controladas de los diferentes componentes del prototipo, así como también de la correcta comunicación entre el software y hardware que permitan la verificación de la funcionalidad del prototipo.
iv. Analizar el costo – beneficio de la solución propuesta mediante una tabla comparativa.

1.6 Alcance y limitaciones.

1.6.1 Alcances

- Este proyecto se realizará para un grupo de 8 personas en la municipalidad de Santa Ana que provienen de diferentes partes de territorio nacional que tengan la posibilidad de movimiento a voluntad de alguna de sus extremidades superiores (brazos) y también que requieran de la ayuda de otra persona para su desplazamiento.
- Este proyecto deja las puertas abiertas para otras asociaciones de la misma índole que tengan sillas de ruedas en abandono por daños en la parte electrónica (tarjeta de control o tarjeta de potencia).

1.6.2 Limitaciones

- Las mejoras no solo serán en la parte electrónica, sino en la parte mecánica y de acondicionamiento para cada usuario, por ello se deberá consultar a expertos en mecánica de precisión para la fabricación y adaptación de piezas.

- La aplicación Google App inventor solo se encuentra disponible para dispositivos con sistema operativo Android.

- Coordinación de visitas y de reuniones con los diferentes usuarios para poder realizar el inventario de partes y componentes.

CAPÍTULO II. MARCO TEÓRICO

2.1 Marco conceptual general

2.1.1 Silla de ruedas

Desde tiempos antiguos el hombre ya intentaba suplir la ausencia o falta de funcionalidad de alguno de sus miembros ya sean superiores o inferiores. Algunos ejemplos los encontramos en Kazajstán, donde fue hallada una huella de prótesis (2300 AC), y en Egipto, lugar en el que apareció un príncipe conocido como Tutankamon (Figura 2) con una pierna atrofiada apoyado en una especie bastón o muleta (2.000 AC).

Figura 2 Tutankamon

Fuente: https://thecuriositypost.wordpress.com/2014/10/22/el-verdadero-aspecto-de-tutankamon/

Los intentos por crear una silla de ruedas han sido numerosos a lo largo de la historia de la humanidad, pero es importante remontarnos en el tiempo para encontrar los verdaderos inicios de la silla de ruedas. Los investigadores creen que el primer intento de ponerle ruedas en una silla fue alrededor del año 4000

AC, pues tanto la silla como la rueda se descubrieron en esa época, aunque se debe referir a la primera representación gráfica de una silla con ruedas, que data del año 525 AC en la China antigua. En este grabado se muestra en la figura 3. Lo cual parece un sillón con tres ruedas destinado a ser impulsado por terceras personas.

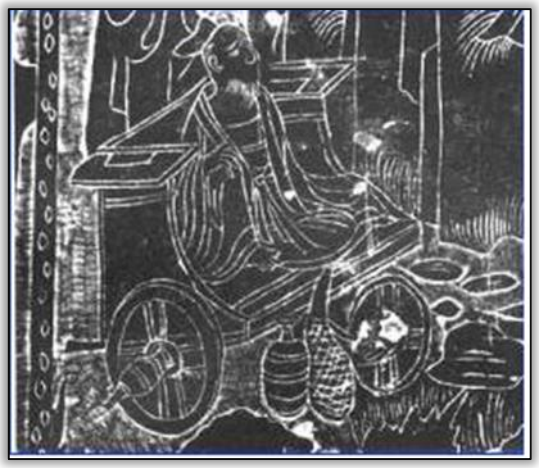


Figura 3 Silla China

Fuente: http://www.minusval2000.com/otros/reportajes/historia_silla_de_ruedas/index.html

De la misma época también se encuentra la imagen encontrada en una vasija griega figura (Figura 4) donde se puede apreciar una camilla para un niño con ruedas. Aunque no es una silla, su importancia se fundamenta en ser la prueba arqueológico más antiguo de utilización de ruedas en la movilización de un paciente.

Figura 4 Camilla Griega

Fuente: http://www.minusval2000.com/otros/reportajes/historia_silla_de_ruedas/index.html

Aunque la fecha exacta o nombre de los inventores de la primera silla de ruedas moderna no estén claros, la primera silla de ruedas concebida para el fin de transportar a una persona, con un diseño similar a las sillas actuales, fue fabricada nada menos que para el monarca Felipe II por un inventor desconocido.

Estaba equipada con cuatro ruedas pequeñas, reposapiés e incluso respaldo reclinable, según puede verse en un dibujo fechado en 1595.

Figura 5 Silla rey Felipe III

Fuente: http://www.minusval2000.com/otros/reportajes/historia_silla_de_ruedas/index.html

El año 1665, Stephen Farfler, un relojero alemán de solo 22 años construyo el primer vehículo autopropulsable para su propio uso (Figura.6). Este artefacto tiene un increíble parecido con un *handbike* de nuestra época.

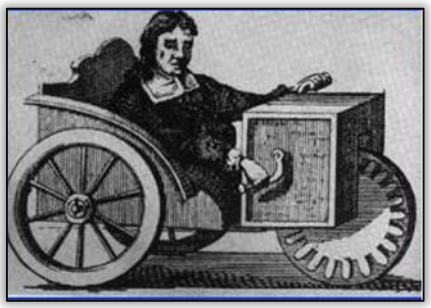


Figura 6 Silla de Stephen Farfler

Fuente: http://www.minusval2000.com/otros/reportajes/historia_silla_de_ruedas/index.html

En 1783 aparece la Silla "Bath", inventada por el fabricante John Dawson en la ciudad inglesa de Bath, de donde toma su nombre. Este modelo de tres ruedas dominará el mercado hasta el siglo XIX. Sin embargo, la silla "Bath" no era cómoda y durante el siglo siguiente fueron añadiéndose mejoras, pensando sobretodo en el confort del usuario, como respaldo y reposapiés ajustables. Una patente de 1869 describe una silla con ruedas traseras autopropulsable y ruedas delanteras pequeñas, llegando por fin a verdaderas sillas de ruedas impulsadas por el propio usuario.

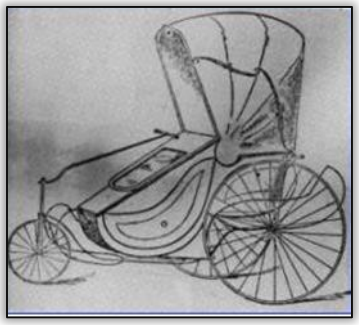


Figura 7 Silla Bath

Fuente: http://www.minusval2000.com/otros/reportajes/historia_silla_de_ruedas/index.html

Entre 1867 y 1875 se siguieron añadiendo mejoras, como los aros de propulsión y ruedas de goma. Cabe destacar, que gran parte de estas mejoras se produjeron gracias a la invención de la bicicleta en el siglo XIX y su posterior evolución.

En 1900 se introdujeron las ruedas radiadas en las sillas manuales y en 1916 se fabricó en Londres la primera silla de ruedas motorizada. Las primeras sillas motorizadas eran sillas manuales adaptadas con diversos sistemas de engranajes

muy rudimentarios y poco eficientes difíciles de manejar, más adelante se adoptaron los motores de tracción directa y sistemas de control más precisos.

Las últimas dos décadas han supuesto un enorme avance, tanto para las sillas manuales como las eléctricas. Nuevos materiales, mejor rendimiento y sobretodo la posibilidad de personalizar las sillas de acuerdo a las necesidades individuales de cada persona.

Otro invento que ha influido decisivamente en la evolución de las sillas de ruedas, es sin duda el automóvil. La necesidad de transportar la silla, determinó la invención de la silla plegable.

Figura 8 Silla de Everest & Jennings

Fuente: http://www.minusval2000.com/otros/reportajes/historia_silla_de_ruedas/index.html

En 1932 el ingeniero Harry Jennings, construye en Nueva York la primera silla de estructura tubular plegable, para su amigo parapléjico Herbert Everest. Juntos fundan Everest & Jennings, una compañía que monopolizaría las ventas de sillas de ruedas durante muchos años. Hasta tal punto que el gobierno de Estados

unidos interpuso una demanda antimonopolio contra Everest & Jennings, por controlar el precio de las sillas de ruedas.

El diseño original de esta primera silla plegable se sigue utilizando al día de hoy en sillas básicas por todo el planeta, por supuesto con algunas mejoras.

Que avances se esperan en el futuro:

En sillas manuales menos significa más. Los fabricantes apuntan por hacer sillas cada día más ligeras y compactas, adaptadas al ritmo de vida moderno. En el futuro más cercano es fácil pensar que se siga por ese camino, investigando, inventando e implementado nuevos materiales, fuertes pero ligeros e incorporando innovaciones de diseño.

El deporte y el ocio juegan un papel importante en el futuro de este sector, muy rápidamente los fabricantes introducen nuevos modelos específicos para cada deporte y otras actividades, creando nuevos campos de investigación y desarrollo.

El campo de las sillas eléctricas es mucho más abierto, debido al rapidísimo avance de la electrónica. La meta más cercana podría ser el desarrollo de motores con menor consumo y la implantación en el mercado de baterías más eficientes, que recarguen más deprisa y tengan mayor capacidad. Todo destinado a ofrecer la mayor autonomía diaria al usuario.

De nuevo la industria automovilística podría ayudar a conseguir estos avances, esta vez gracias a su reciente apuesta por los coches eléctricos, lo que podría hacer realidad con nuevas tecnologías en baterías y otros componentes.

Diversas innovaciones como asistencia por GPS ya están en desarrollo y podrían estar disponibles en pocos años.

Los kits de motorización para sillas manuales, previsiblemente serán cada día más discretos y ligeros, siendo útiles para mayor cantidad de personas.

Uno de los grandes avances en este campo, aunque fue un fracaso a nivel de ventas y comercialización es el *ibot* una silla de ruedas eléctrica de alta tecnología con el potencial para amortiguar y de pasar sobre superficies irregulares, montaje escaleras y levantar la silla de ruedas, viene con equipo de alto nivel, los modos y velocidades. El *ibot* es la creación del Dr. Dean Kamen, quien fundó DEKA Investigación y la Corporación. El costo de esta silla es de alrededor de 25.000 dólares al entrar en el mercado. Sin embargo, debido a su alto precio y alguna operación defectuosa, no está disponible.

A pesar del fracaso de la *ibot*, las sillas de ruedas eléctricas hoy están equipadas con la última tecnología. Los avances han hecho sillas de ruedas eléctrica autónomas pensando en la comodidad y agilidad, permitiendo a las personas con discapacidad una amplia gama de movilidad adaptada a sus necesidades.

Figura 9 IBOT 3000

Fuente: http://wedurewe.parknhost.com/ibot-transporter.html

2.1.2 Arduino

Arduino es una plataforma de prototipos electrónica de código abierto (*open-source*) basada en hardware y software flexibles y fáciles de usar. Está pensado para artistas, diseñadores, estudiantes de electrónica, computación y robótica, también como hobby y para cualquiera interesado en crear objetos o entornos interactivos.

Arduino puede percibir su entorno mediante la recepción de entradas desde una variedad de sensores y puede interactuar a su alrededor mediante el control de luces, motores y otros artefactos. El microcontrolador de la placa se programa usando el Arduino *Programming Language* (basado en *Wiring*) y el Arduino

Development Environment (basado en *Processing*). Los proyectos de Arduino pueden ser autónomos o se pueden comunicar con software en ejecución en un ordenador (por ejemplo con Flash, *Processing*, *MaxMSP*, entre otros).

Arduino fue inventado en el año 2005 por el entonces estudiante del instituto IVRAE Massimo Banzi, quien, en un principio, pensaba en hacer Arduino por una necesidad de aprendizaje para los estudiantes de computación y electrónica del mismo instituto, pues en ese entonces, adquirir una placa de micro controladores eran bastante caro y no ofrecían el soporte adecuado; no obstante, nunca se imaginó que esta herramienta se llegaría a convertir en años más adelante en el líder mundial de tecnologías DIY (*Do It Yourself*). Inicialmente fue un proyecto creado no solo para economizar la creación de proyectos escolares dentro del instituto, sino que además, Banzi tenía la intención de ayudar a su escuela a evitar la quiebra de la misma con las ganancias que produciría vendiendo sus placas dentro del campus a un precio accesible (1 euro por unidad).

El primer prototipo de Arduino fue fabricado en el instituto IVRAE. Inicialmente estaba basado en una simple placa de circuitos eléctricos, donde estaban conectados un micro controlador simple junto con resistencias, además de que únicamente podían conectarse sensores simples como leds u otras resistencias, incluso no contaba con el soporte de algún lenguaje de programación para manipularla.

Años más tarde, se integró al equipo de Arduino Hernando Barragán, un estudiante de la Universidad de Colombia que se encontraba haciendo su tesis, y tras enterarse de este proyecto, contribuyó al desarrollo de un entorno para la programación del procesador de esta placa: *Wiring*, en colaboración con David Mellis, otro integrante del mismo instituto que Banzi, quien más adelante, mejoraría la interfaz de software.

Tiempo después, se integró al "*Team* Arduino" el estudiante español David Cuartielles, experto en circuitos y computadoras, quien ayudó Banzi a mejorar la interfaz de hardware de esta placa, agregando los micro controladores necesarios para brindar soporte y memoria al lenguaje de programación para manipular esta plataforma.

Más tarde, Tom Igoe, un estudiante de Estados Unidos que se encontraba haciendo su tesis, escuchó que se estaba trabajando en una plataforma de *open-source* basada en una placa de microcontroladores preensamblada. Después se interesó en el proyecto y fue a visitar las instalaciones del Instituto IVRAE para averiguar en que estaban trabajando. Tras regresar a su país natal, recibió un e-mail donde el mismo Massimo Banzi invitó a Igoe a participar con su equipo para ayudar a mejorar Arduino. Aceptó la invitación y ayudó a mejorar la placa haciéndola más potente, agregando puertos USB para poder conectarla a un ordenador. Además, él le sugirió a Banzi la distribución de este proyecto a nivel mundial.

Cuando creyeron que la placa estaba al fin lista, comenzaron su distribución de manera gratuita dentro de las facultades de electrónica, computación y diseño del mismo instituto. Para poder promocionar el proyecto Arduino dentro del campus, debieron consultar con un publicista que más parte pasaría a formar parte del equipo Arduino: Gianluca Martino, quien la distribuyo dentro del instituto y promocionándola a algunos conocidos y amigos suyos. Al ver su gran aceptación por parte de los alumnos y maestros y tomando en cuenta el consejo de Igoe, pensaron en su distribución a nivel mundial, para lo cual contactaron a un amigo y socio de Banzi, Natan Sadle, quien se ofreció a producir en masa las placas tras interesarse en el proyecto. (Historia de Arduino y su nacimiento, s.f.)

Arduino está constituido en el hardware por un microcontrolador (que es programable con un lenguaje de alto nivel), presente en la mayoría de los modelos de Arduino, encargado de realizar los procesos lógicos y matemáticos dentro de la placa, además de controlar y gestionar los recursos de cada uno de los componentes externos conectados a la misma. Consta además de una amplia variedad de sensores eléctricos como cámaras VGA, sensores de sonido, seguidores de línea, botones de control de sensores, e incluso, otras placas de microcontroladores (mejor conocidos como *Shields*), que pueden adaptarse fácilmente gracias a que Arduino cuenta con entradas de pines analógicos y digitales para integrar estos componentes sin necesidad de alterar el diseño original de esta placa. Además, Arduino cuenta con la ventaja de tener entre sus elementos principales puertos seriales de entrada /salida (input/output), esto le permite conectarse por medio de un cable USB a una computadora para poder

trabajar con ella desde nivel software, ya que es dónde se le darán las “ordenes” que ejecutarán cada uno de los componentes conectados a la placa, e incluso, para operar como un dispositivo más (dependiendo de la configuración que hayamos establecido y para que se quiere utilizar). Además, Arduino para operar necesita de una fuente de alimentación externa, pues por desgracia, no cuenta con una propia, por ello también se encuentra incorporada una entrada para conectar un cable con entrada similar al USB, donde será conectado a un otro dispositivo que tenga entrada USB, o hasta en el mismo dispositivo. (Historia de Arduino y su nacimiento, s.f.)

2.1.3 Microcontrolador

Un microcontrolador puede usarse para muchas aplicaciones algunas de ellas son: manejo de sensores, controladores, juegos, calculadoras, agendas, avisos lumínicos, secuenciador de luces, cerrojos electrónicos, control de motores, relojes, alarmas, robots, entre otros.

¿Cómo funciona un microcontrolador?

Como el hardware ya viene integrado en un solo chip, para usar un microcontrolador se debe especificar su funcionamiento por software a través de programas, los cuales indiquen las instrucciones que el microcontrolador debe realizar. En una memoria se guardan los programas y un elemento llamado CPU se encarga de procesar paso por paso las instrucciones del programa. Los lenguajes de programación típicos que se usan para este fin son ensamblador y C,

pero antes de grabar un programa al microcontrolador se debe compilarlo a hexadecimal que es el formato con el que funciona el microcontrolador. (MikroElectronika, 2016)

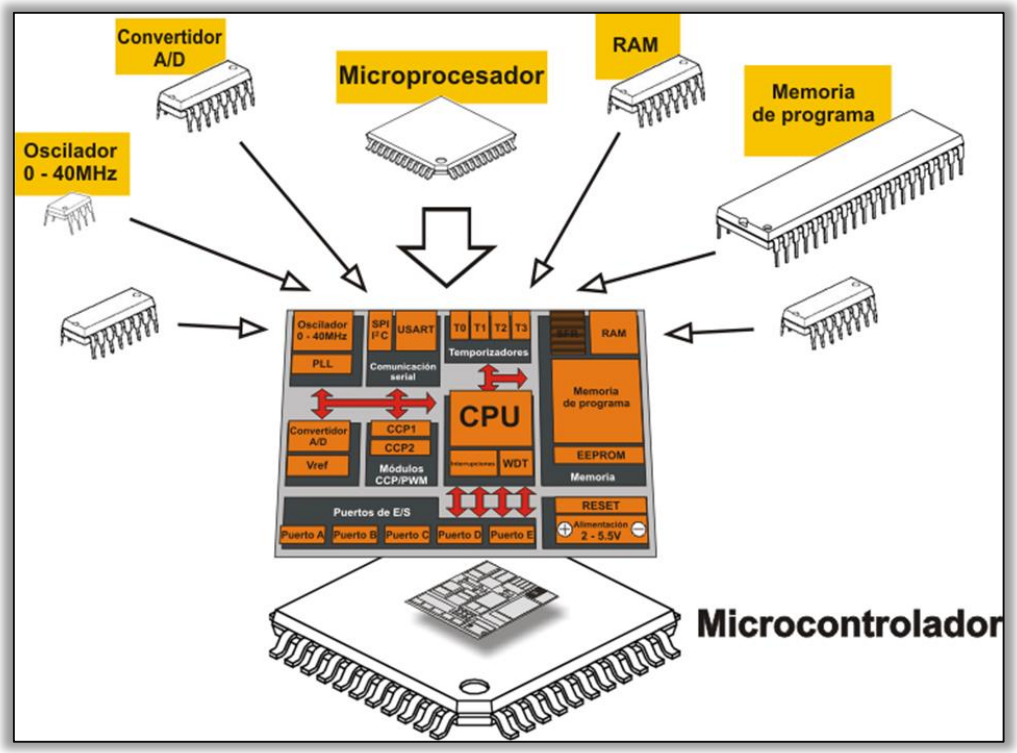


Figura 10 Microcontrolador

Fuente: http://learn.mikroe.com/ebooks/microcontroladorespicc/chapter/introduccion-al-mundo-de-los-microcontroladores/

2.1.4 Smartphone

El término *smartphone* viene del habla inglesa y hace referencia a aquello que, en el idioma español, se conoce como teléfono inteligente. Se trata de un teléfono celular (móvil) el cual ofrece prestaciones similares a las que brinda una computadora (PC) y se destaca por su conectividad.

Es normal que los expertos mencionen que el *smartphone* está a mitad de camino entre un teléfono celular convencional y una computadora portátil. El smartphone cuenta con todas las funciones básicas del celular (permite realizar llamadas telefónicas, enviar mensajes de texto, etc.) y le agrega características avanzadas (conexión a Internet, capacidad multimedia, pantalla táctil).

En la actualidad existen diversos tipos de *smartphones*, algunos con mayor cantidad de funciones que otros. Con un *smartphone*, es posible que una persona pueda conectarse a la Web a través de una red 3G, 4G o WiFi; consultar una ubicación mediante un GPS; reproducir archivos MP3 o de video; sacar fotografías y grabar videos; utilizar juegos; gestionar una agenda: y, en algunos casos, hasta visualizar documentos de trabajo creados en PDF u otros formatos.

Entre los sistemas operativos que se emplean en los *smartphones*, se puede mencionar a iOS, Android, BlackBerry OS, *Symbian* OS y Windows Phone. Respecto a los fabricantes de esta clase de dispositivos, entre los más populares se encuentran Samsung, Sony, Nokia, LG, Motorola, Alcatel, BlackBerry y Apple.

2.1.5 Aplicación móvil

Una aplicación móvil, o app (en inglés) es un software específico para ser ejecutados en teléfonos inteligentes, tabletas y otros dispositivos móviles y que permite al usuario efectuar una tarea concreta de cualquier tipo profesional, de

ocio, educativa, de acceso a servicios, etc., facilitando las gestiones o actividades por desarrollar.

Por lo general se encuentran disponibles a través de plataformas de distribución, operadas por las compañías propietarias de los sistemas operativos móviles como Android, iOS, BlackBerry OS, Windows Phone, entre otros. Existen aplicaciones móviles gratuitas u otras de pago, donde en promedio el 20-30 % del costo de la aplicación se destina al distribuidor y el resto es para el desarrollador. El término app se volvió popular rápidamente, tanto que en 2010 fue listada como Word *of the Year* (Palabra del Año) por la American *Dialect Society*

2.1.6 APP inventor

El App inventor es una aplicación de *Google Labs* para crear aplicaciones (Valga la redundancia) de Android, estas apps pueden crearse desde cualquier *smartphone* con sistema operativo Android, además también puede ser utilizado desde una computadora de cualquier marca.

Esta aplicación ha sido una de las grandes innovaciones de los últimos años por parte de Google debido a que ha permitido a los principiantes en materia de programación realizar sus propias aplicaciones con diferentes objetivos, ya sea para satisfacción y comodidad personal o incluso para llenar sus bolsillos y monetizarlas en Android *Market*.

Gracias a App Inventor ahora es una realidad que cualquiera programe sus propias funcionalidades en su *smartphone* e incluso como ya se ha mencionado en su propia computadora gracias a Android para computadores. (moviles, s.f.)

2.1.7 Bluetooth

Bluetooth es una tecnología de comunicación entre dispositivos de corto alcance. En 1994, Ericsson inició el desarrollo de esa tecnología, investigando una forma barata de comunicación inalámbrica entre el móvil y sus accesorios. Después de esas investigaciones iniciales, quedó clara la potencialidad de ese tipo de conexión. En 1998, seis grandes empresas: Sony, Nokia, Intel, Toshiba, IBM y Ericsson, realizaron un consorcio para conducir y profundizar el estudio de esa forma de conexión, formando el llamado *Bluetooth Special Interest Group*.

El nombre "Bluetooth" es un homenaje al rey de Dinamarca y Noruega, Harald Bltand, que en la lengua inglesa es llamado de Harold Bluetooth. El nombre del rey fue escogido por el hecho de haber unificado las tribus de su país, semejantemente a lo que la tecnología pretende hacer: unificar tecnologías diferentes. El símbolo del Bluetooth es la unión de dos runas nórdicas para las letras H y B, sus iniciales.

La tecnología es bastante ventajosa, pues permite la comunicación entre diversos dispositivos sin la necesidad de cables. Además de eso, es una tecnología barata. Por esos motivos, el Bluetooth ganó popularidad, haciéndose uno de los

principales métodos de conexión entre dispositivos de la actualidad. Entre los dispositivos que pueden ser conectados vía *bluetooth*, se pueden citar: teléfonos celulares, ordenadores, videojuegos, impresoras, escáneres, mouses, teclados, etc.

La desventaja de esta tecnología es el hecho de su alcance corto. Además de eso, el número máximo de dispositivos conectados al mismo tiempo también es limitado. (bluetooth, 2016)

2.1.8 Módulos de Bluetooth HC-05

Este módulo de conexión inalámbrica y es exclusivo para trabajar con las tarjetas Arduino estas tarjetas tiene un par de salidas de comunicación serial TX y RX a las cuales se les conecta el módulo HC-05 lo que hace es trasformar esas señales eléctricas en señales de radio frecuencia específicamente con el Protocolo conocido como Bluetooth.

Cabe mencionar que también existen otros módulos de conexión inalámbrica no solo en Bluetooth sino en WIFI, ZIGBEE etc. Pero para efectos de este proyecto se analizarán: la trasmisión por Bluetooth y el módulo HC-05.

El módulo HC-05 (figura 12) puede funcionar tanto como maestro, como esclavo, por lo que en el sistema debe existir al menos uno de estos. También se puede lograr que la comunicación se dé entre dos de estos módulos, pero teniendo en

cuenta que uno debe ser el maestro y otro el esclavo. En contraste, el módulo HC-06 solamente tiene la funcionalidad de esclavo. (GeekFactory, 2016)

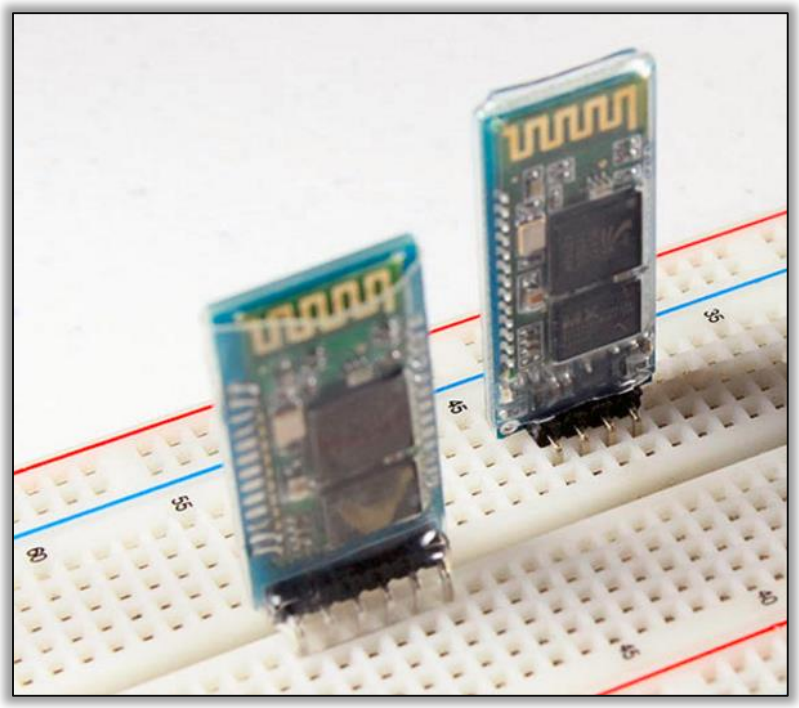


Figura 11 módulo HC-05

Fuente: http://www.geekfactory.mx/tutoriales/bluetooth-hc-05-y-hc-06-tutorial-de-configuracion/

2.1.9 Señales analógicas y digitales

Es importante definir que son señalas analógicas y digitales, pues el prototipo funciona con ambos tipos. Las señales analógicas son aquellas que provienen de un sistema analógico, según (Tocci, 1996, sistemas digitales principios y aplicaciones), “Un sistema analógico contiene dispositivos que manipulan cantidades físicas representadas en forma analógica”. Las señales analógicas varían su magnitud en el tiempo, como por ejemplo la diferencia de potencial generada por las unidades motoras.

Las señales digitales son el producto de un sistema digital; este es un grupo de dispositivos que son capaces de procesar información representada de manera discreta (que solo tiene valores enteros) o cantidades físicas, un ejemplo de señal digital son los datos provenientes de un microcontrolador (Tocci, 1996, sistemas digitales principios y aplicaciones, p. 4).

En la figura 11 se observa claramente la diferencia entre una señal digital y una analógica, en cuanto la señal digital no varía su amplitud en función del tiempo, mientras que la analógica.

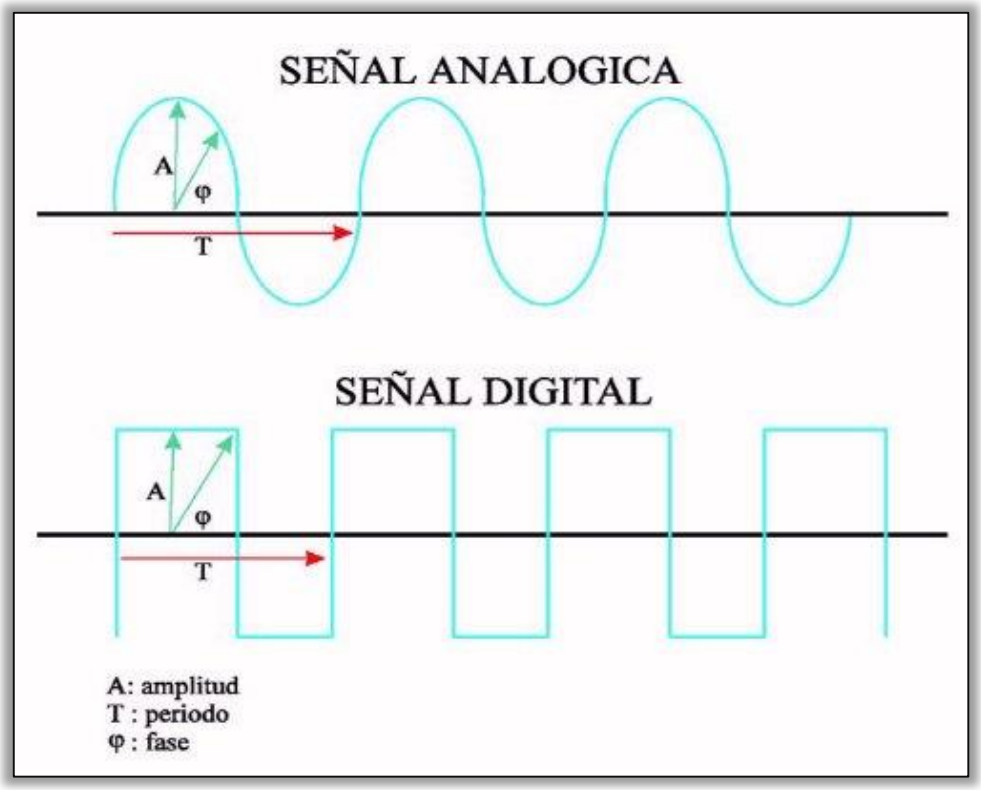


Figura 12 Señal analógica vs señal digital

Fuente: http://www.angelfire.com/ak2/karenteamo/tdatos.html

2.1.10 Modulación de ancho de pulso

La modulación de ancho de pulso o PWM, por sus siglas en inglés, es el método utilizado para controlar motores eléctricos y, en el caso de este proyecto, motores de corriente continua, con este método se puede controlar el voltaje de entrada al motor así por controlar su velocidad.

En la figura 13 se presenta un ejemplo donde se muestra como la pequeña variación en el ancho del pulso se logra variar la tensión de entrada al motor y por ende su velocidad.

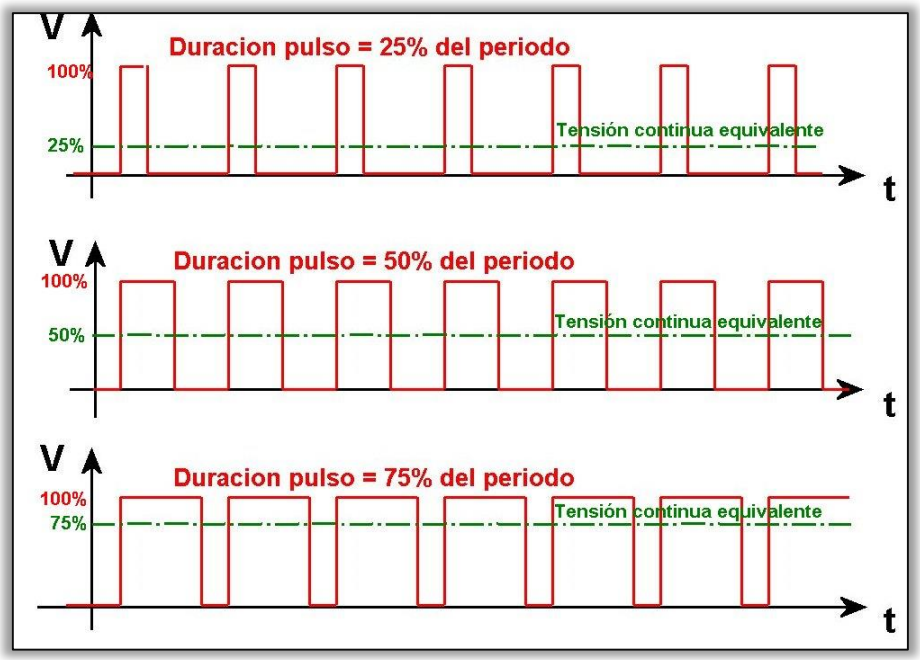


Figura 13 PWM

Fuente: http://www.arduino.utfsm.cl/modulacion-por-ancho-de-pulso-pwm/

2.1.11 Puente H

El puente H o puente en H es un circuito electrónico que permite a un motor eléctrico de corriente continua girar en ambos sentidos, ya viene construido con algunos circuitos integrados utilizados para tal fin, pero también se pueden construir a partir de componentes discretos.

Un puente H se construye con 4 interruptores (mecánicos o mediante transistores). Figura 14. Cuando los interruptores S1 y S4 están cerrados (S2 y S3 abiertos) se aplica una tensión positiva en el motor, haciéndolo girar en un sentido. Abriendo los interruptores S1 y S4 (cerrando S2 y S3), el voltaje se invierte, permitiendo el giro en sentido inverso del motor.

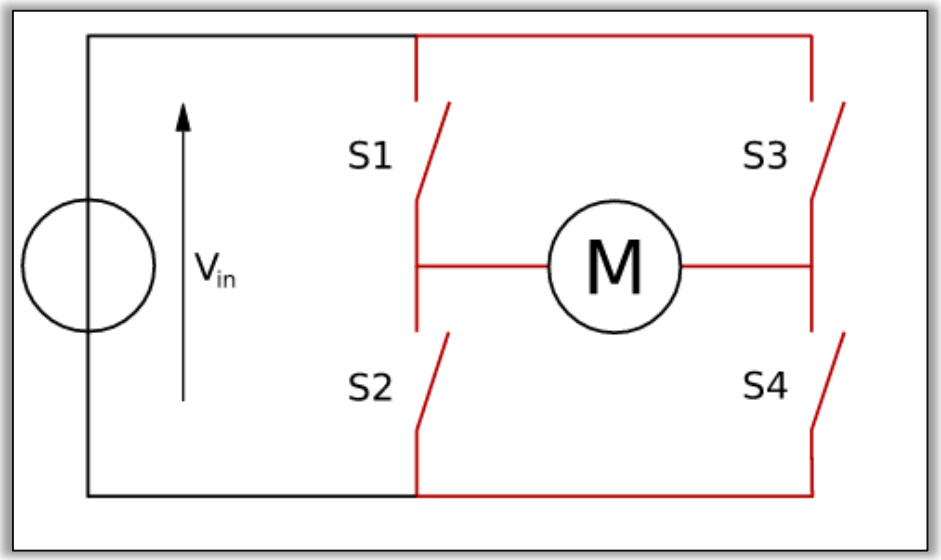


Figura 14 Puente H con interruptores

Fuente: http://blutintegrado.blogspot.com/2012/05/puente-h.html

2.1.12 Processing

Este consiste en un lenguaje de programación de código abierto orientado a objetos y su respectivo entorno integrado de desarrollo (Figura 15) construido para las artes electrónicas, medios artísticos y comunidades de diseño visual con el propósito de enseñar las bases fundamentales de la programación de computadoras en un contexto visual. El lenguaje en sí se construye sobre Java, pero utiliza una sintaxis simplificada y un modelo de programación de gráficos (Processing, s.f.).

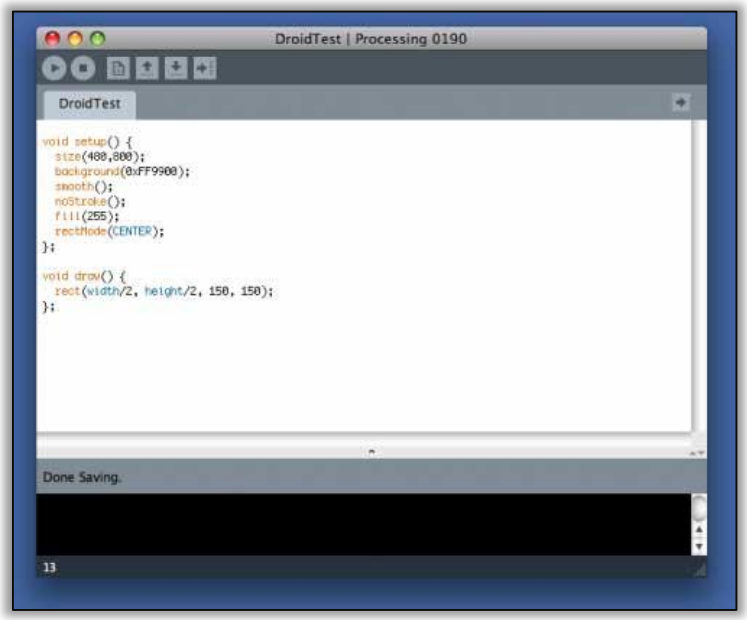


Figura 15 Interfaz gráfica Processing

Fuente: http://blog.bricogeek.com/noticias/tutoriales/como-programar-para-android-con-processing/

CAPÍTULO III: MARCO METODOLÓGICO

3.1 Tipo de investigación:

3.1.1 Finalidad. Aplicada.

“Investigación aplicada: su finalidad es la solución de problemas prácticos para trasformar las condiciones de un hecho que nos preocupa. El propósito fundamental no es aportar al conocimiento teórico.” (Barrantes Echavarría, 2004)
La aplicada fundamental, se entiende como aquella investigación relacionada con la generación de conocimientos en forma de teoría o métodos estimados en un período mediato, los cuales podrían desembocar en aplicaciones al sector productivo.

Este trabajo de investigación tiene una finalidad aplicada, porque se propone transformar el conocimiento 'puro' en conocimiento útil; la búsqueda y consolidación del saber y la aplicación de los conocimientos para el enriquecimiento del acervo cultural y científico, así como la producción de tecnología al servicio del desarrollo integral de las naciones. La investigación aplicada puede ser Fundamental o Tecnológica, aunque el propósito principal de esta investigación servirá como precedente para desarrollar nuevas implementaciones por medio de prototipos que brinden soluciones a diferentes problemáticas de personas discapacitadas usuarios de sillas de ruedas eléctricas, logrando satisfacer las necesidades de un segmento de la población costarricense que no goza hasta el momento de alternativas económicas para su movilidad diaria.

3.1.2 Dimensión temporal trasversal.

"Estudios transversales (sincrónicos): estudia aspectos de desarrollo de los sujetos en un momento dado." (Barrantes Echavarría, Investigación Un camino al conocimiento, 2004)

Se tomará como muestra todas las partes de las sillas de ruedas en su totalidad para determinar cuáles pueden ser remplazadas, reparadas y reutilizadas mediante un inventario previo

3.1.3 Marco de la investigación.

La investigación micro es de tipo práctico, busca el estudio exhaustivo de un caso en concreto o de un conjunto.

Se toma en cuenta que las empresas dedicadas a la fabricación de sillas de ruedas eléctricas tienen como mercado la clase alta, sin embargo, en este caso, la Asociación atiende a personas de estratos medio y bajo.

3.1.4 Naturaleza.

"La investigación cualitativa postula una concepción fenomenológica, inductiva, orientada al proceso. Busca descubrir o generar teorías. Pone énfasis

en la profundidad y sus análisis no necesariamente, son traducidos a términos matemáticos". (Barrantes Echavarría R. , Investigación Un camino al conocimiento, 2004)

"La investigación cuantitativa pone una concepción global positivista, hipotética-deductiva, objetiva, particularista y orientada a los resultados. Se desarrolla más directamente en la tarea de verificar y comprobar teorías por medio de estudio muéstrales representativos." (Barrantes Echavarría R. Investigación Un camino al conocimiento, 2004)

De acuerdo con lo expuesto anterior, por lo tanto la naturaleza de la investigación es cuantitativa

3.1.5 Carácter.

El carácter de esta investigación es de un proyecto debido a que se realizará un prototipo para solventar la problemática

3.2 Diseño metodológico:

3.2.1 Metodología para la propuesta de mejora.

1. Determinación de las partes reutilizables de las sillas eléctricas actuales y las tecnologías involucradas, para lo cual, se elaborará un inventario de partes de acuerdo con el estado de cada una de las componentes de las sillas, organizando en una tabla los siguientes aspectos (marca, serie, partes, estado, cantidad).

2. Se seleccionará la tarjeta Arduino más apropiada para el proyecto dependiendo de sus entradas y salidas digitales y analógicas también salidas con opción para PMW.

3.2.2 Metodología para la implementación del proyecto

Se realizarán las pruebas de funcionamiento con respecto a la compilación del programa de Arduino y pruebas eléctricas a la tarjeta.

3.2.3 Evaluación del costo beneficio

Se investigará el costo beneficio de las sillas de ruedas en el mercado actual, tanto el precio de las sillas completas y sus repuestos comparándolas con los componentes necesarios para realizar el proyecto.

CAPÍTULO IV. DIAGNÓSTICO

4.1 Determinación de la situación actual.

Para determinar la situación actual en APEDISTROFA se debe realizar un inventario de partes y componentes de todas las sillas de ruedas en general para determinar cuántas sillas de ruedas, componentes o partes se pueden reutilizar.

4.2 Recolección de datos.

La recolección de los datos es por medio de inventario, el cual se muestra en la tabla.1

Marca	Modelo	Serie	Parte	Estado	Cantidad
Quickie	pulse 6	125kl03	tarjeta principal / llantas desgaste	dañado	1
Quickie	pulse 6	125zh15	Joystick / llantas desgaste / cargador dañado	desgaste	1
Quickie	Qm-710	125xxx	Joystick / reposa pies dañados / desgaste de faja de tracción	dañado	1
Jassy	Pride	112304	tarjeta principal / llantas desgaste / Joystick	dañado	1
Jassy	Pride	11387	Joystick / llantas desgaste / desgaste de faja de tracción	desgaste	1

Invacare	Pronto	11xxx	Joystick / Baterías con sulfato	desgaste	1
Invacare	Pronto	N/A	tarjeta / desgaste de llantas / principal	buena	1
Invacare	Veranda	N/A	tarjeta / desgaste de llantas principal	buena	1

Tabla 1 Inventario de partes de sillas de ruedas.

4.3. Objetivos de las actividades

Los objetivos del inventario fueron los siguientes:

- Determinar la cantidad de partes reutilizables de cada una de las sillas de ruedas dañadas.
- Obtener un estimado de partes que se pueden reparar o comprar.
- Conocer los materiales utilizados en la fabricación de sillas de ruedas eléctricas.

4.3 Desarrollo de prototipo.

A continuación, en las siguientes imágenes se demuestra la fabricación del prototipo.

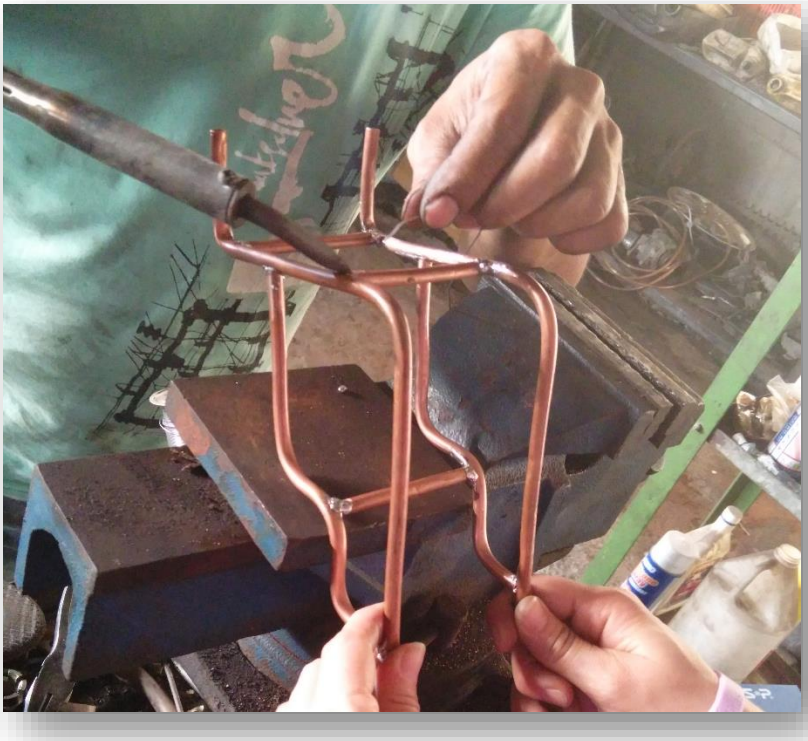


Figura 16 Soldado de partes del armazón

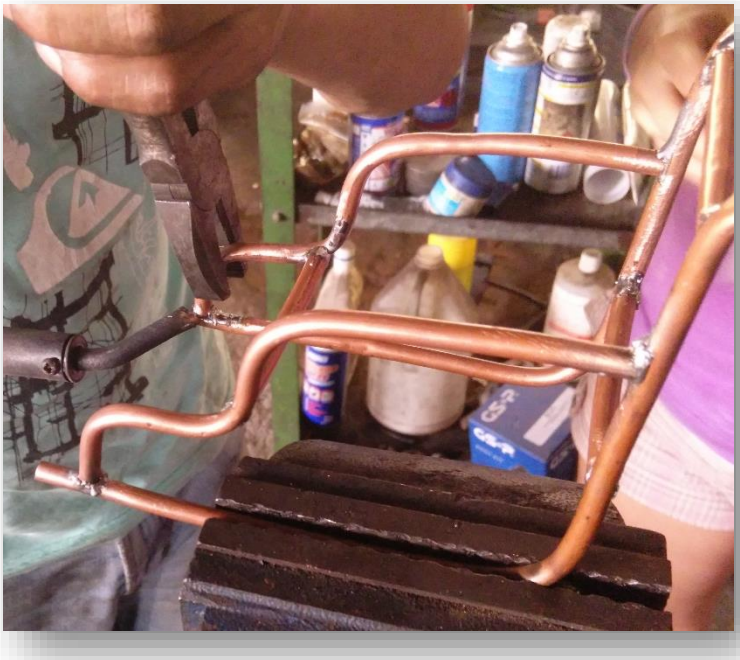


Figura 17 Soldado de partes del armazón

Figura 18 Revisión y enfriamiento de los puntos

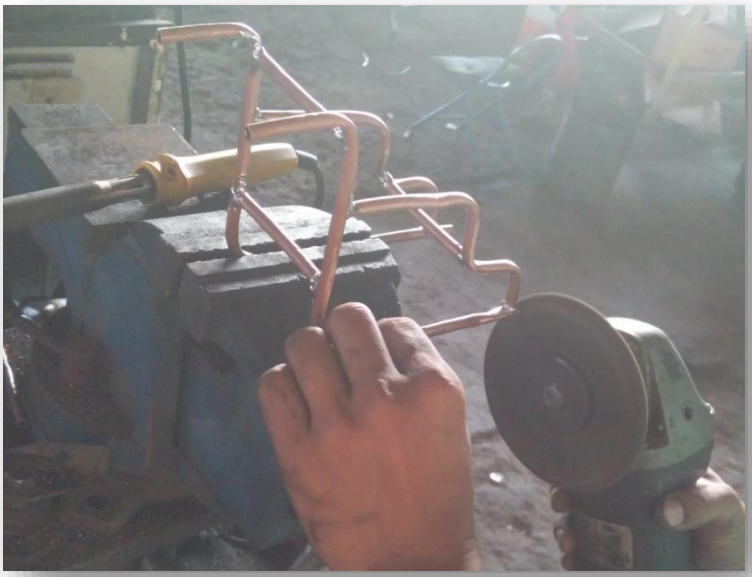


Figura 19 Corte de partes y afinado de puntas

Figura 20 Revisión general

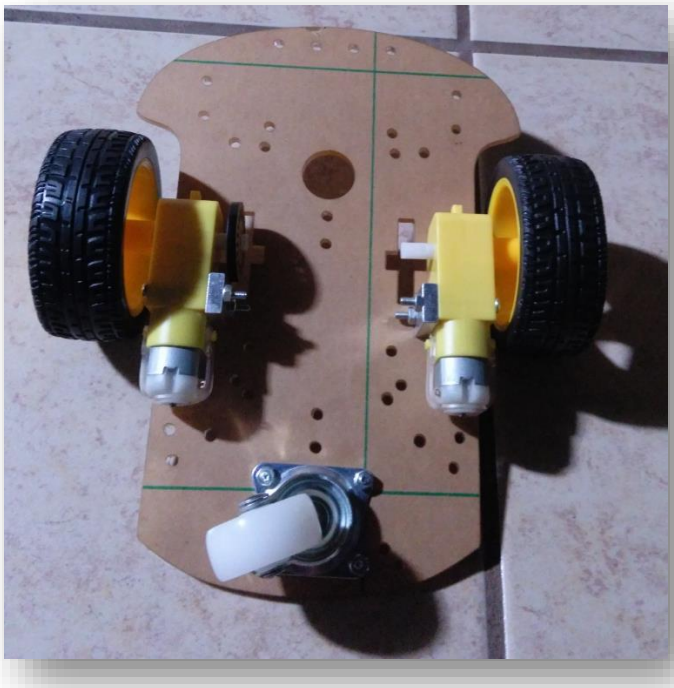


Figura 21 Vista inferior instalacion de motores

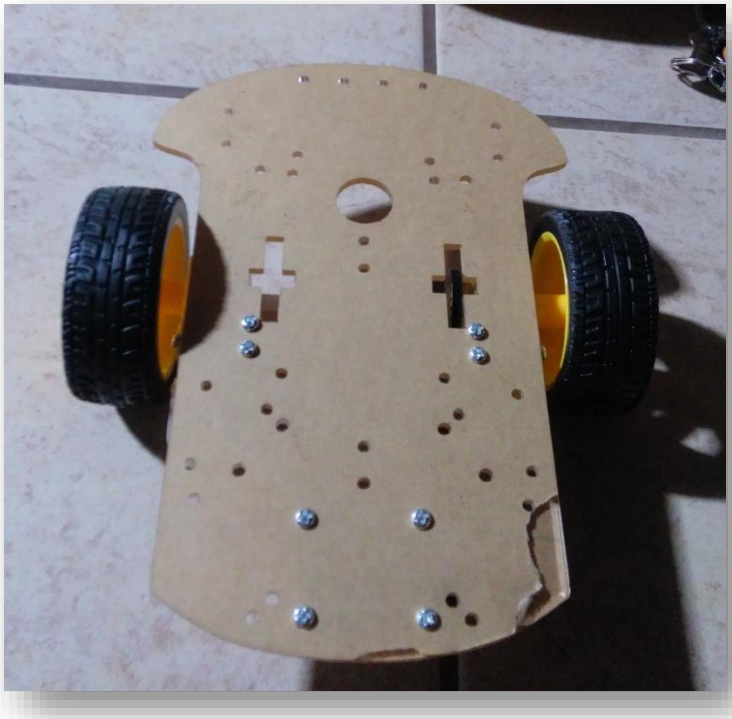


Figura 22 Vista superior

Figura 23 Revisión de componentes

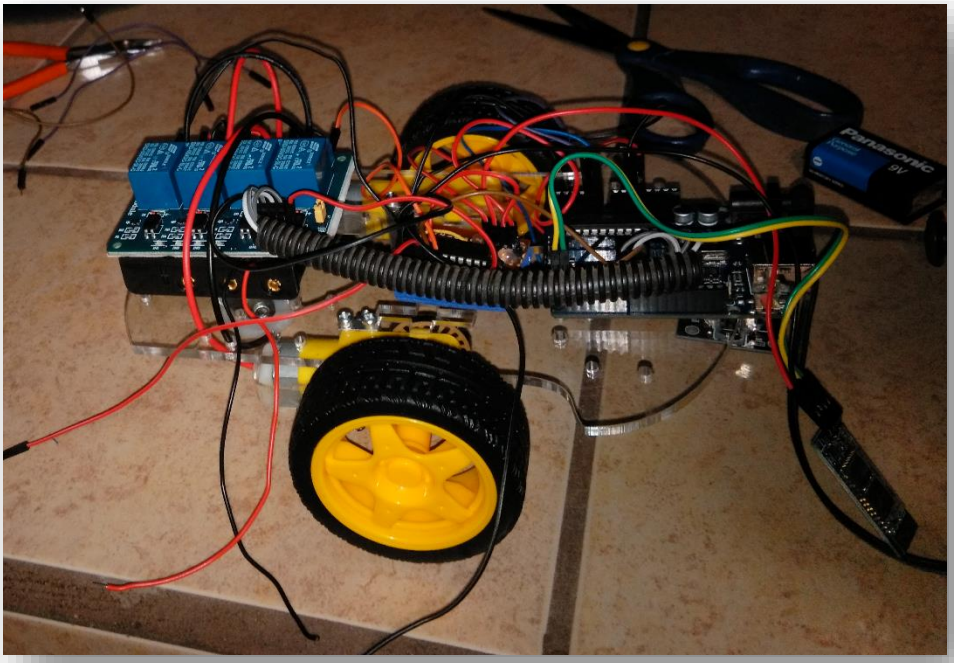


Figura 24 Colocación de componentes

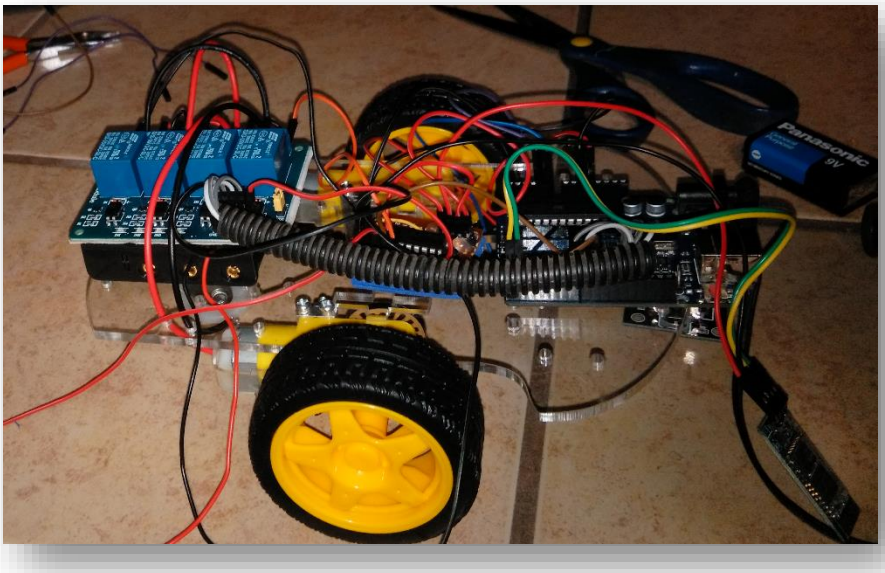


Figura 25 Colocación de componentes

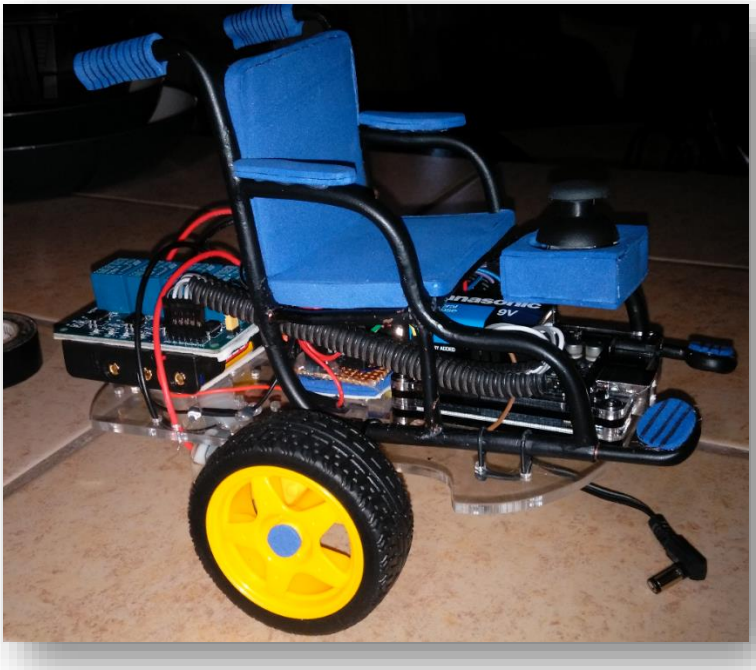


Figura 26 Prototipo terminado vista lateral

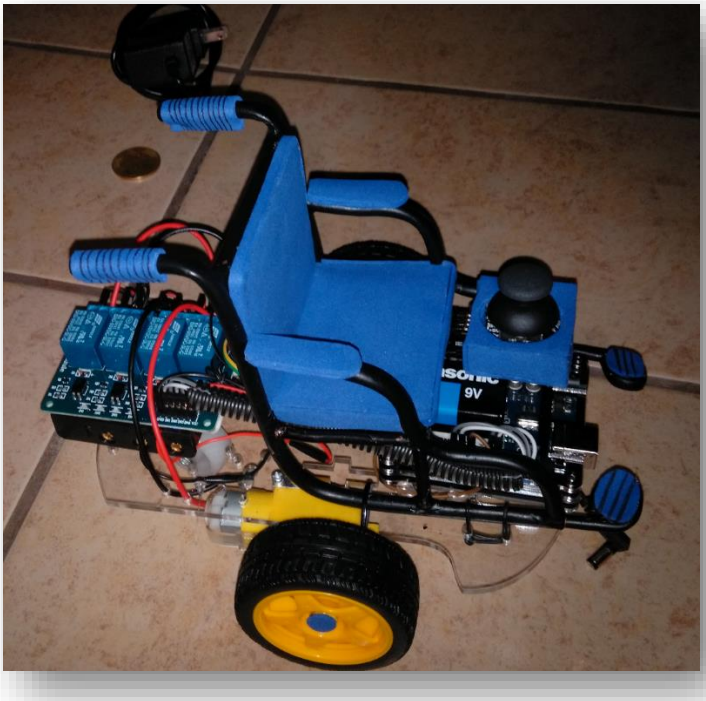


Figura 27 Prototipo vista superior

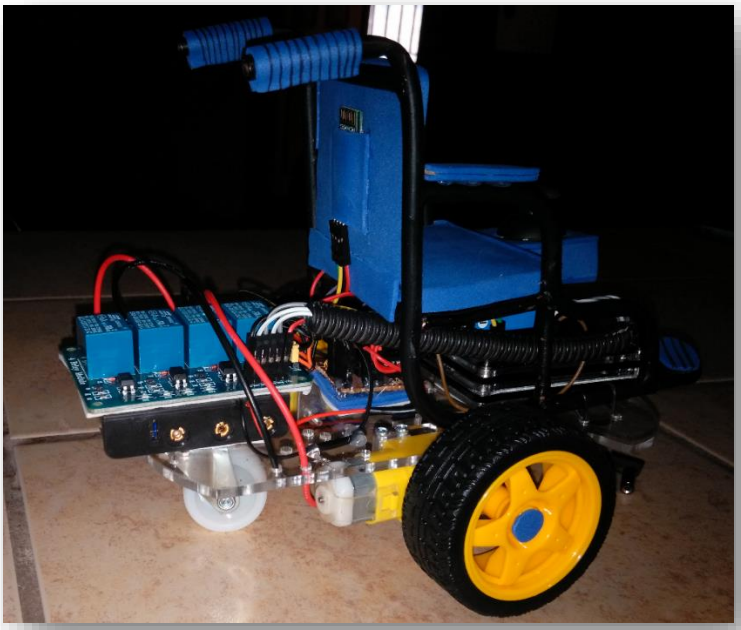


Figura 28 Prototipo vista trasera

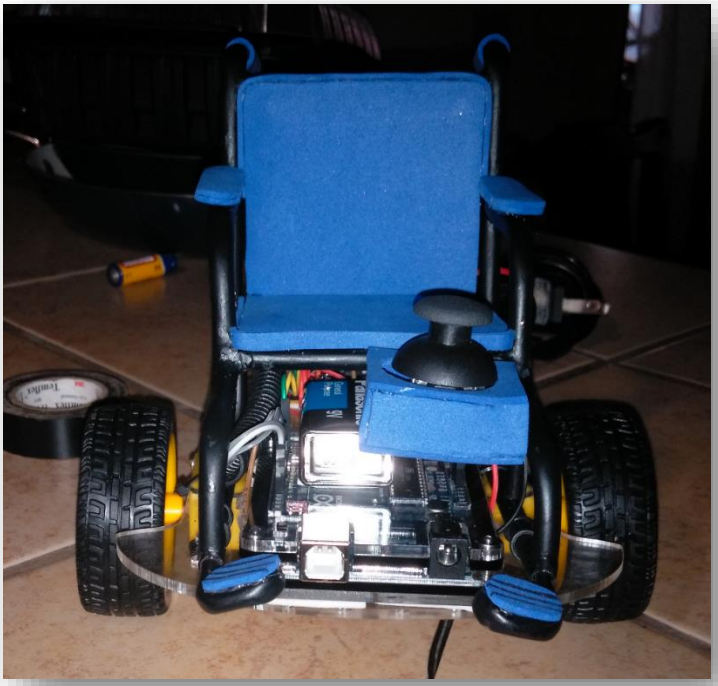


Figura 29 Prototipo vista frontal

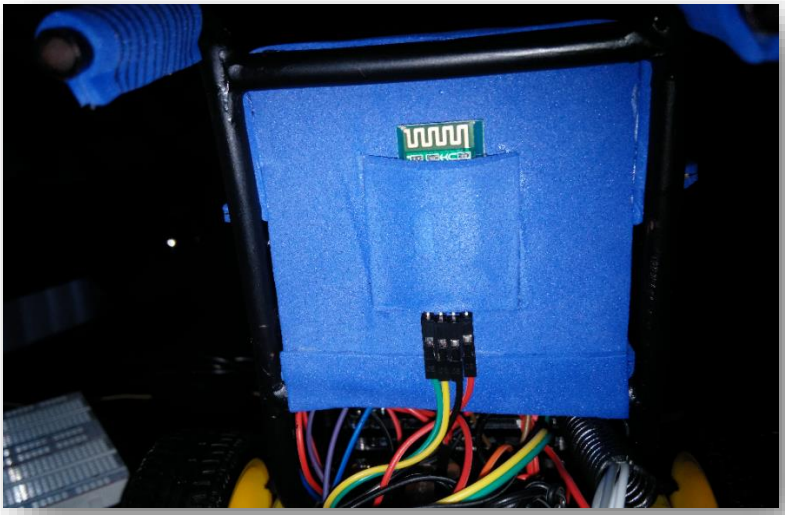


Figura 30 Colocación del módulo HC05

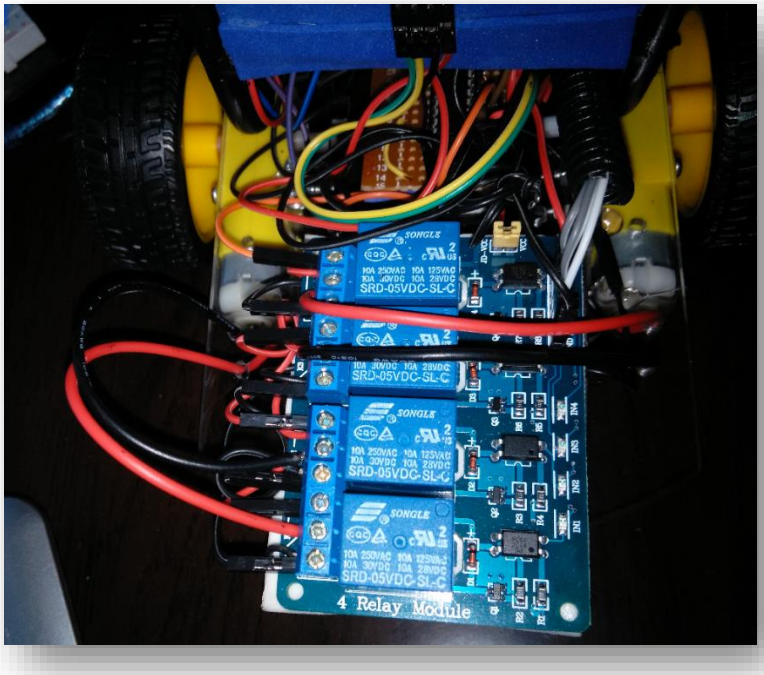


Figura 31 Posición de módulo de relés

CAPÍTULO V. DISEÑO Y DESARROLLO DEL PROYECTO

5.1. Selección de la propuesta

Después de una investigación de los diferentes tipos de tarjetas para proyectos, es necesario identificar cuál se adapta a los requerimientos de este proyecto, se determina que las tarjetas de Arduino son las que mejor se adaptan, debido a sus componentes y fácil programación, sobre todo cuando se observan los costos de los componentes del prototipo, detallados en el apartado 5.3

Debido a la naturaleza del proyecto, algunos de los componentes no se pueden conseguir en el mercado nacional como la propia tarjeta Arduino.

5.2. Detalle de la propuesta

5.2.1 Microcontrolador Arduino Uno

Se decidió utilizar la tarjeta Arduino como unidad central o de control, debido a su versatilidad y fácil programación, también porque permite el uso de módulos, los cuales son tarjetas especializadas para una única función como por ejemplo comunicación telefónica, inalámbrica por WiFi o Bluetooth, tarjetas de driver para control de motores, entre otros.

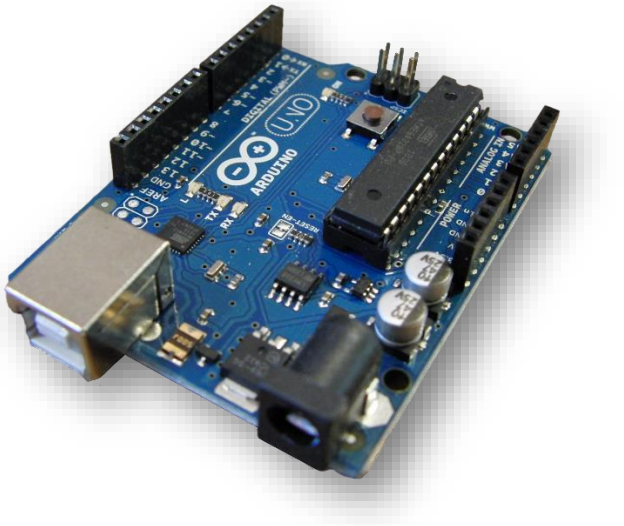


Figura 32 Arduino Uno Rev3.

Fuente: **http://www.instructables.com/file/FK71BAQIMTEB29Q/**

El Arduino Uno es una tarjeta microcontroladora basada en el ATmega328, entre sus características principales se encuentran las siguientes: seis entradas analógicas, catorce salidas digitales entre las cuales seis se pueden utilizar como salidas de PWM, un cristal de 16 MHz, puerto USB para comunicación serial con una computadora, así como para la programación, entre otras.

Una de las características determinantes es su voltaje de operación de 5 V, lo que lo hace idóneo para su uso con los motores de corriente directa elegidos para el prototipo. Existen otras tarjetas de Arduino como el “Due” que poseen una mayor cantidad de entradas y salidas, pero esta por ejemplo opera a 3.3 V y es de un costo superior.

A manera de síntesis, el Arduino Uno posee las siguientes características básicas:

- Microcontrolador: ATmega328

- Voltaje de operación: 5 V
- Voltaje de entrada (recomendado): 7-12 V
- Voltaje mínimo y máximo de entrada: 6-20 V
- Patillas de entrada y salida (I/O): 14 (de las cuales 6 se pueden utilizar como salidas PWM)
- Entradas analógicas: 6
- Corriente CD por pin I/O: 40 mA
- Corriente CD para la patilla de 3.3 V: 50 mA
- Memorias integradas dentro del ATmega 328: • Flash: 32 kB (de los cuales 0.5 kB son utilizados para el programa de arranque o "*bootloader*")
- SRAM: 2 kB
- EEPROM: 1 kB
- Velocidad del reloj: 16 MHz

A continuación, se muestra el diagrama de bloques del microcontrolador del Arduino Uno.

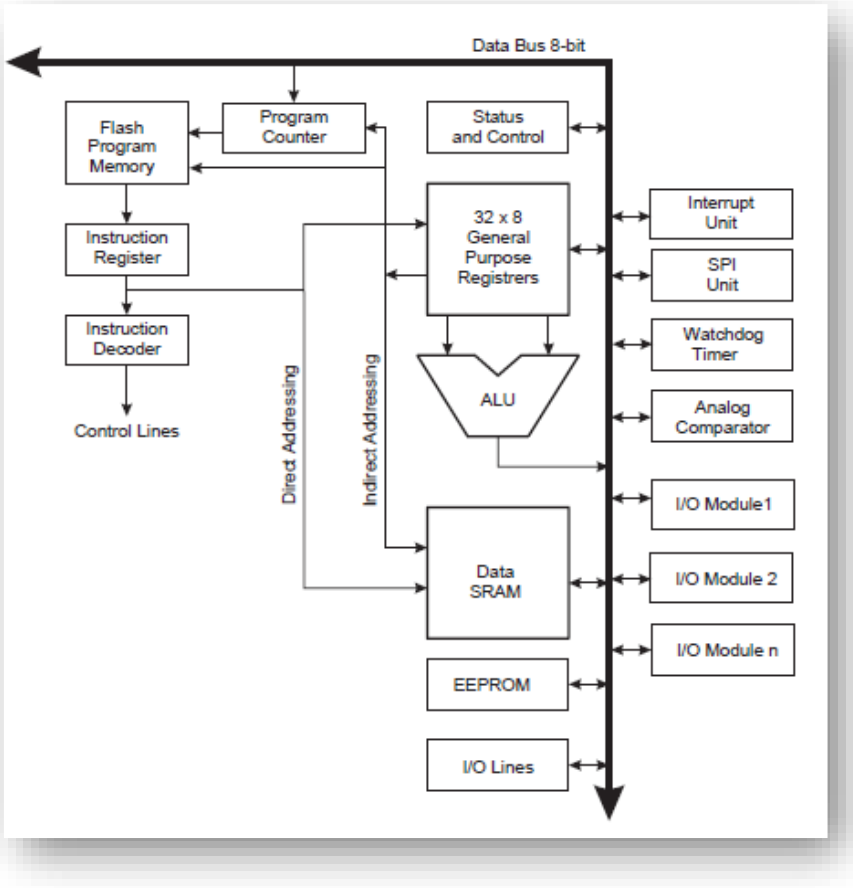


Figura 33 Diagrama de bloques del ATmega328

Fuente: http://www.atmel.com/Images/doc8161.pdf, página 8

En cuanto a la programación, el Arduino Uno utiliza un lenguaje basado en C y un compilador propietario que se puede descargar de manera gratuita de la página del fabricante (www.arduino.cc). La plataforma Arduino es considerada "*Open-Source*", esto significa que es de carácter gratuito y disponible para mejoras por parte de los usuarios.

5.2.2. Etapa de control para motores

Para efectos del prototipo se eligió el integrado L293D *Driver controller* que es un puente H para motores de corriente continua el cual ayudara con el control de la velocidad y un juego de 4 *relés* para el control de dirección y giro de los motores en la siguiente figura 18. Se ve un circuito simplificado de la conexión para un solo motor.

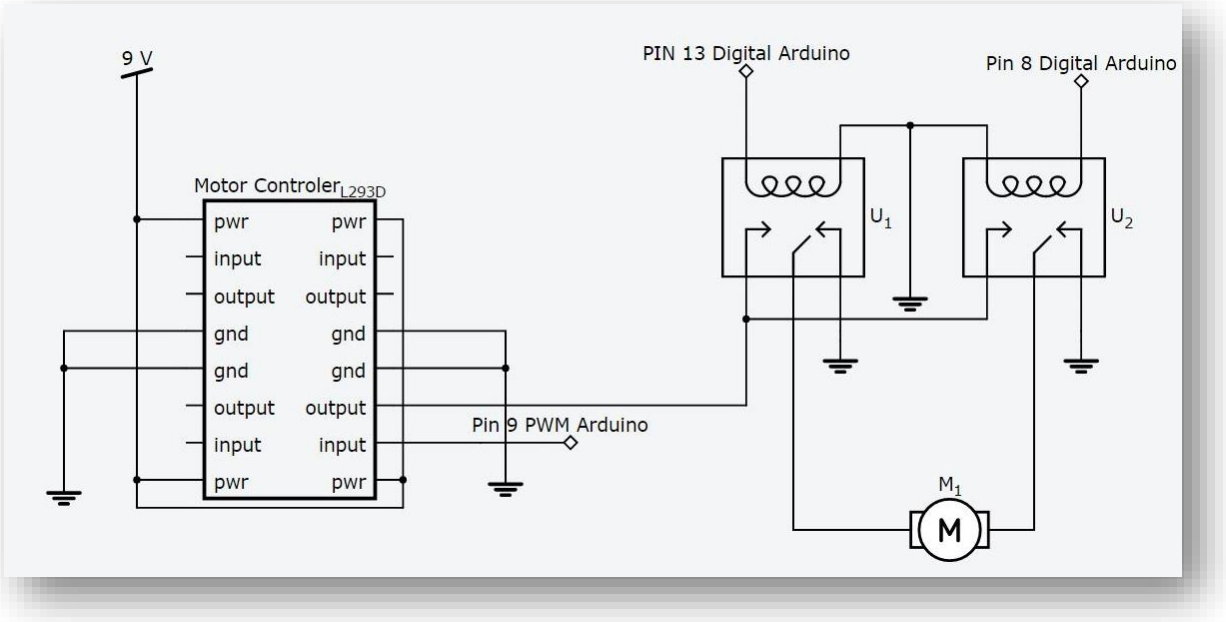


Figura 34 Etapa simplificada de control de motores

Fuente: Elaboración propia

5.2.3. Módulo HC-05 *Bluetooth*

En la siguiente figura se demuestra la conexión básica para el uso del módulo HC-05, en la cual se puede ver que puede ser alimentado directamente desde la tarjeta Arduino.

El módulo *bluetooth* HC-05 viene configurado de fábrica para trabajar como maestro o esclavo. En el modo maestro puede conectarse con otros módulos *bluetooth*, mientras en el modo esclavo queda a la escucha peticiones de conexión.

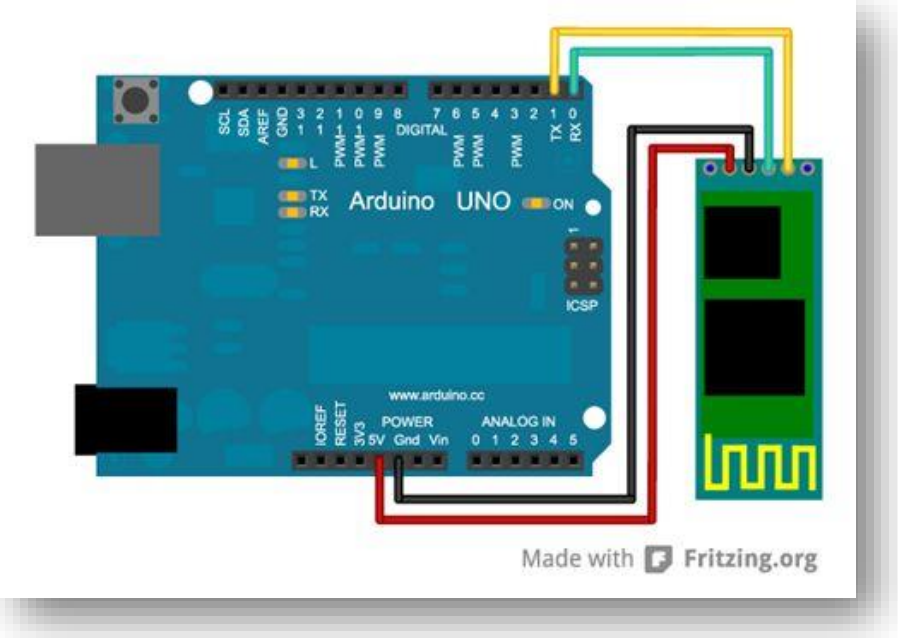


Figura 35 Conexión de HC05

Fuente: Frigzi.org

Características:

- Especificación *bluetooth* v2.0 + EDR (*Enhanced Data Rate*)
- Puede configurarse como maestro, esclavo, y esclavo con auto conexión (*Loopback*) mediante comandos AT
- Chip de radio: CSR BC417143
- Frecuencia: 2.4 GHz, banda ISM

- Modulación: GFSK (*Gaussian Frequency Shift Keying*)
- Antena de PCB incorporada
- Potencia de emisión: ≤ 4 dBm, Clase 2
- Alcance 5 m a 10 m
- Sensibilidad: ≤ -84 dBm a 0.1% BER
- Velocidad: Asincrónica: 2.1 Mbps (max.)/160 kbps, sincrónica: 1 Mbps/1 Mbps
- Seguridad: Autenticación y encriptación (*Password* por defecto: 1234)
- Perfiles: Puerto serial Bluetooth
- Módulo montado en tarjeta con regulador de voltaje y 6 pines suministrando acceso a VCC, GND, TXD, RXD, KEY y status LED (STATE)
- Consumo de corriente: 50 mA
- El pin RX del módulo requiere resistencia de *pull-up* a 3.3 V (4.7 k a 10 k). Si el microcontrolador no tiene resistencia de *pull-up* interna en el pin Tx se debe poner externamente.
- Niveles lógicos: 3.3 V. Conectarlos a señales con voltajes mayores, como por ej. 5 V, puede dañar el módulo.
- Voltaje de alimentación: 3.6 V a 6 V
- Dimensiones totales: 1.7 cm x 4 cm aprox.
- Temperatura de operación: -20 ºC a +75 ºC

Configuración:

El módulo suele venir configurado como esclavo, con velocidad de transmisión serial de 38400 bps ó 9600 bps (Dependiendo del nivel de KEY al alimentarlo), 1 bit de parada, y sin bit de paridad, nombre: HC-05, *password*: 1234

Para su configuración se puede conectar al viejo puerto serial RS232 de la computadora a través de un convertidor TTL a RS232, o mejor empleando un conversor USB a serial TTL y utilizando el *Hyperterminal* de Windows u otro programa similar para enviar los comandos AT (Por ej. el SSCOM32, *PuTTY*, entre otros.). (A partir de Win Vista el *hyperterminal* ya no está incluido en el SO)

Por supuesto también se pueden enviar los comandos AT desde cualquier microcontrolador sin ayuda de computadoras.

Con Arduino de 3.3 V también se puede hacer fácilmente y sin ningún convertidor con un pequeño sketch que utiliza el monitor serial del IDE de Arduino para escribir los comandos AT y observar la respuesta del módulo. Como este monitor emplea la comunicación serial que el Arduino utiliza para comunicarse con la computadora en los pines 0 y 1 digitales, se crea un puerto serial por software para pasar los datos al módulo Bluetooth, empleando los pines restantes del Arduino como los pines digitales 10 y 11.

A continuación, se presenta un código básico de programación para el puerto serial.

```
#include <SoftwareSerial.h>  // incluir la librería correspondiente a la comunicación
SoftwareSerial BT(10,11);
```

```
void setup()
{
  BT.begin(9600); //Velocidad del puerto del módulo Bluetooth
  Serial.begin(9600); //Abrimos la comunicación serie con el Pc
}

void loop()
{
  if(BT.available())
  {
    Serial.write(BT.read());
  }


  if(Serial.available())
  {
     BT.write(Serial.read());
  }
}
```

5.2.5. Diagrama de bloques

En la figura 20 se puede ver el diagrama de bloques del sistema completo: el Joystick y el teléfono celular envían comandos a la tarjeta Arduino esta los interpreta y los ejecuta habilitando y deshabilitando las salidas apropiadas. El usuario puede realizar el control, tanto con el joystick como con la aplicación del teléfono celular.

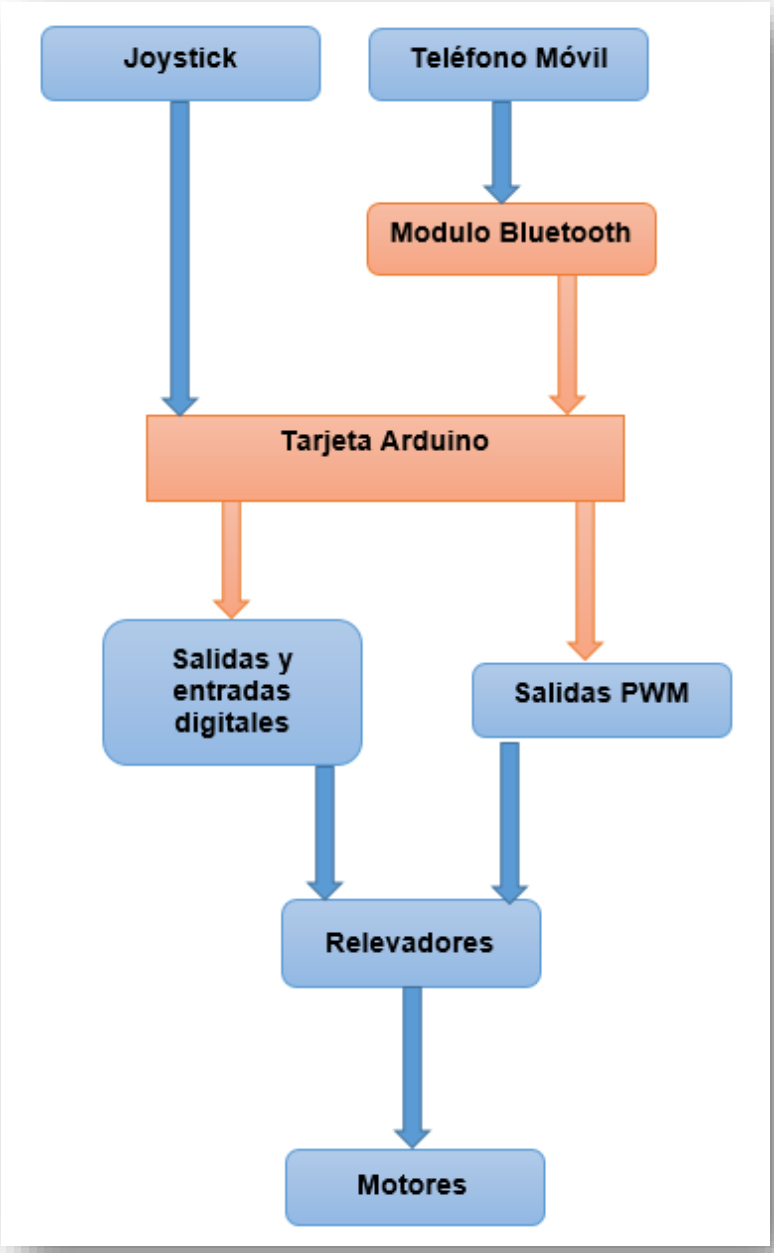


Figura 36 Diagrama de Bloques

Fuente: Elaboración propia

5.2.6. Circuito

A continuación, en la figura 21 se muestra el circuito del prototipo en una versión simplificada, donde se muestran las conexiones por utilizar en al Arduino, las entradas analógicas A0 y A1 son para el *Joystick,* las salidas/entradas digitales D0 y D1 son para el modulo *Bluetooth,* las salidas *D13 y D12 son para los* relés que controlan uno de los motores y las salidas D8 y D7 son para las r*elés* que controlan el otro motor, la salida D9 es la salida PWM para el control de velocidad que va a la entrada 10 del integrado L293D.

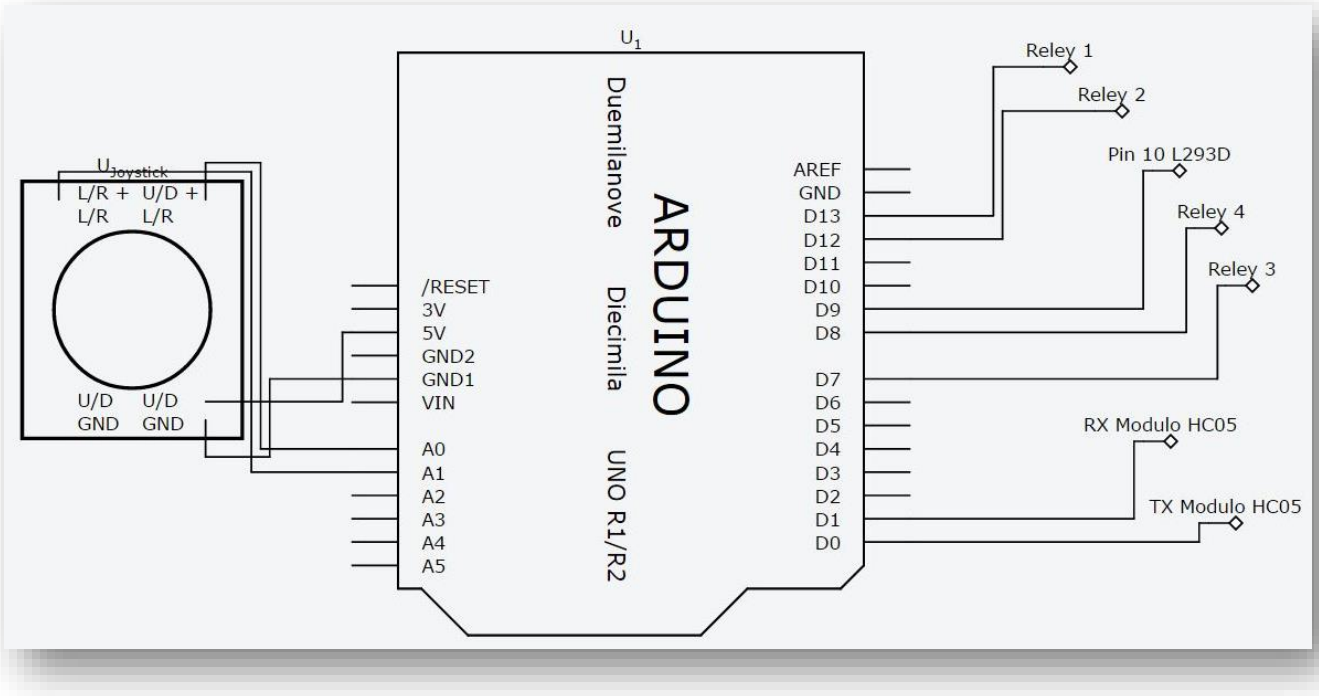


Figura 37 Conexión Simplificada de Arduino

Fuente: Elaboración propia

5.2.7. Programación del Arduino

Como se describe en la página del fabricante de las tarjetas Arduino la estructura básica del lenguaje de programación de Arduino es bastante simple y

se compone de al menos dos partes. Estas dos partes necesarias, o funciones, encierran bloques que contienen declaraciones, estamentos o instrucciones.

```
void setup() //Primera Parte
{
  estamentos;
}
void loop() //Segunda Parte
{
  estamentos;
}
```

En donde setup() es la parte encargada de recoger la configuración y loop() contiene el programa que se ejecutará cíclicamente (de ahí el término loop –bucle-). Ambas funciones son necesarias para el funcionamiento del programa.

La función de configuración (setup) debe contener la declaración de las variables. Es la primera función por ejecutar en el programa, se ejecuta sólo una vez, y se utiliza para configurar o inicializar pinMode (modo de trabajo de las E/S), configuración de la comunicación en serie y otras.

La función bucle (loop) siguiente contiene el código que se ejecutará continuamente (lectura de entradas, activación de salidas, etc) Esta función es el núcleo de todos los programas de Arduino y la que realiza la mayor parte del trabajo.

- **setup()**

La función setup() se invoca una sola vez cuando el programa empieza. Se utiliza para inicializar los modos de trabajo de los pins, o el puerto serie. Debe ser

incluido en un programa, aunque no haya declaración que ejecutar. Así mismo se puede utilizar para establecer el estado inicial de las salidas de la placa.

```
void setup()
{
  pinMode(pin, OUTPUT); // configura el 'pin' como salida
  digitalWrite(pin, HIGH);  // pone el ‘pin’ en estado HIGH
}
```

- **loop()**

Después de llamar a setup(), la función loop() hace precisamente lo que sugiere su nombre, se ejecuta de forma cíclica, esto posibilita que el programa esté respondiendo continuamente ante los eventos producidos en la placa.

```
void loop()
{
digitalWrite(pin, HIGH); // pone en uno (on, 5v) el 'pin'
delay(1000);             // espera un segundo (1000 ms)
digitalWrite(pin, LOW); // pone en cero (off, 0v.) el 'pin'
delay(1000);
}
```

- **Funciones**

Una función es un bloque de código que tiene un nombre y un conjunto de instrucciones ejecutadas cuando se llama a la función. Son funciones setup() y loop() de las cuales ya se ha hablado. Las funciones de usuario pueden ser escritas para realizar tareas repetitivas y para reducir el tamaño de un programa. Las funciones se declaran asociadas a un tipo de valor “*type*”. Este valor será el que devolverá la función, por ejemplo '*int*' se utilizará cuando la función devuelve un dato numérico de tipo entero. Si la función no devuelve ningún valor entonces se colocará delante la palabra “*void*”, esto significa “función vacía”. Después de declarar el tipo de dato que devuelve la función se debe escribir el nombre de la

función y entre paréntesis se escribirán, si es necesario, los parámetros para que la función se ejecute.

```
type nombreFunción(parámetros)
{
instrucción;
}
```

La función siguiente devuelve un número entero, delayVal() se utiliza para poner un valor de retraso en un programa que lee una variable analógica de un potenciómetro conectado a una entrada de Arduino. Al principio se declara como una variable local, 'v' recoge el valor leído del potenciómetro que estará comprendido entre 0 y 1023, luego se divide el valor por 4 para ajustarlo a un margen comprendido entre 0 y 255, finalmente se devuelve el valor 'v' y se retornaría al programa principal. Esta función cuando se ejecuta devuelve el valor de tipo entero 'v'.

```
int delayVal()
{
int v;                  // crea una variable temporal 'v'
v= analogRead(pot);     // lee el valor del potenciómetro
v /= 4;                 // convierte 0-1023 a 0-255
return v;               // devuelve el valor final
}
```

- **{} entre llaves**

Las llaves sirven para definir el principio y el final de un bloque de instrucciones. Se utilizan para los bloques de programación setup(), loop(), if.., entre otros.

```
type funcion()
{
instrucciones;
}
```

Una llave de apertura “{“ siempre debe ir seguida de una llave de cierre “}”, si no es así el programa dará errores.

El entorno de programación de Arduino incluye una herramienta de gran utilidad para comprobar el total de llaves. Sólo se debe hacer *click* en el punto de inserción de una llave abierta e inmediatamente se marca el correspondiente cierre de ese bloque (llave cerrada).

- **; punto y coma**

El punto y coma “;” se utiliza para separar instrucciones en el lenguaje de programación de Arduino. También se utiliza para separar elementos en una instrucción de tipo *“bucle for*”.

```
int x = 13;   /* declara la variable 'x' como tipo entero de valor 13 */
```

Nota: es necesario poner fin a una línea con un punto y coma o se producirá en un error de compilación. El texto de error puede ser obvio, y se referirá a la falta de una coma, o puede que no. Si se produce un error raro y de difícil detección, lo primero por hacer es comprobar que los puntos y comas están colocados al final de las instrucciones.

- **/*… */ bloque de comentarios**

Los bloques de comentarios, o comentarios multi-línea son áreas de texto ignorados por el programa, utilizados para las descripciones del código o

comentarios que ayudan a comprender el programa. Comienzan con / * y terminan con * / y pueden abarcar varias líneas.

```
/* esto es un bloque de comentario no se debe olvidar
cerrar los comentarios estos deben estar equilibrados */
```

Debido a que los comentarios son ignorados por el compilador y no ocupan espacio en la memoria de Arduino pueden ser utilizados con generosidad. También pueden utilizarse para "comentar" bloques de código con el propósito de anotar informaciones para depuración y hacerlo más comprensible para cualquiera.

Nota: Dentro de una misma línea de un bloque de comentarios NO se puede escribir otro bloque de comentarios (usando /*..*/).

- **// línea de comentarios**

Una línea de comentario empieza con // y terminan con la siguiente línea de código. Al igual que los comentarios de bloque, los de línea son ignoradas por el programa y no ocupan espacio en la memoria.

```
// esto es un comentario
```

Una línea de comentario se utiliza a menudo después de una instrucción, para proporcionar más información acerca de lo que hace esta o para recordarla más adelante.

5.2.7.1 Código de programación Arduino para el prototipo.

A continuación de muestra la programación utilizada para efectos del prototipo.

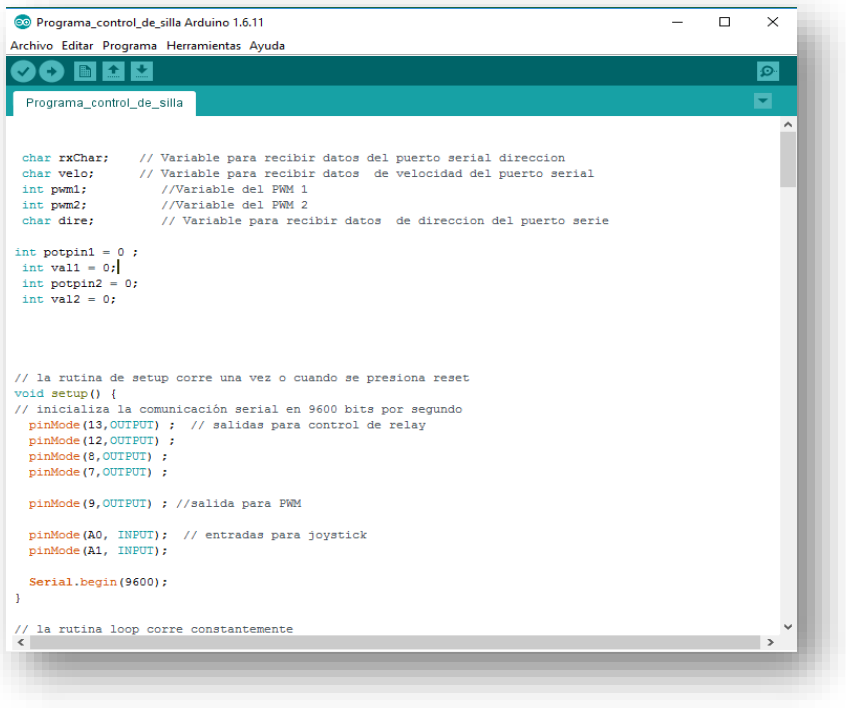


Figura 38 Programación Arduino para el prototipo

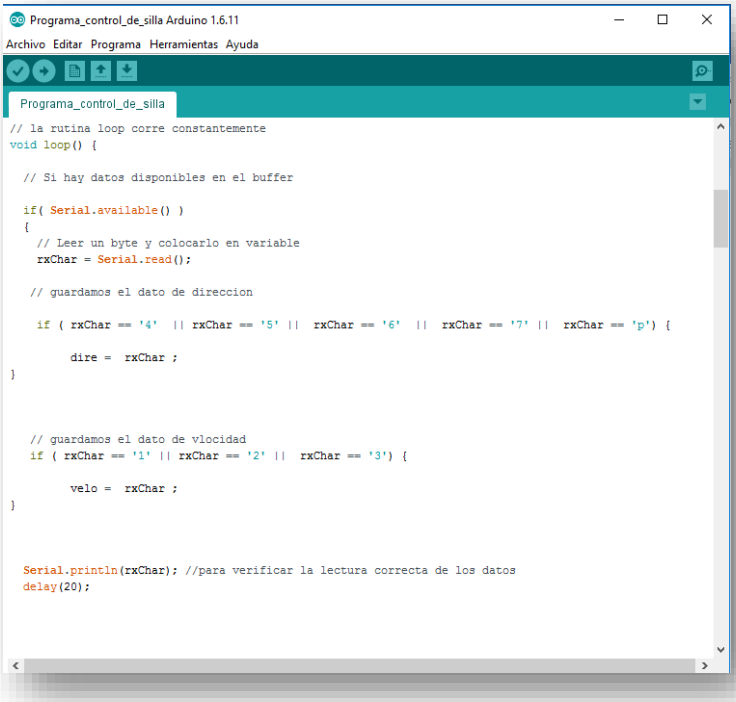


Figura 39 Programación Arduino para el prototipo

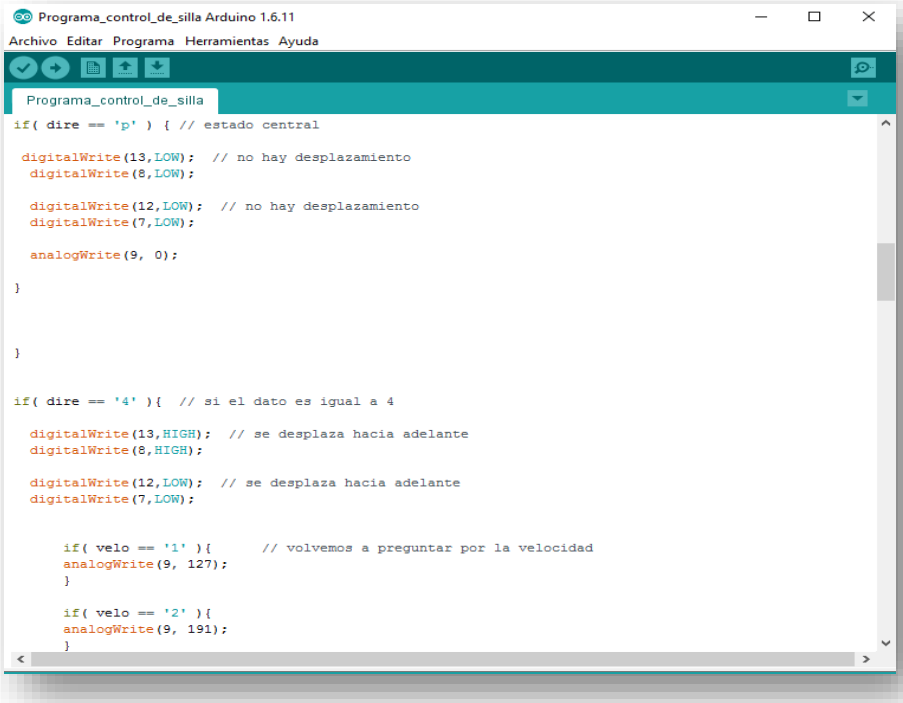


Figura 40 Programación Arduino para el prototipo

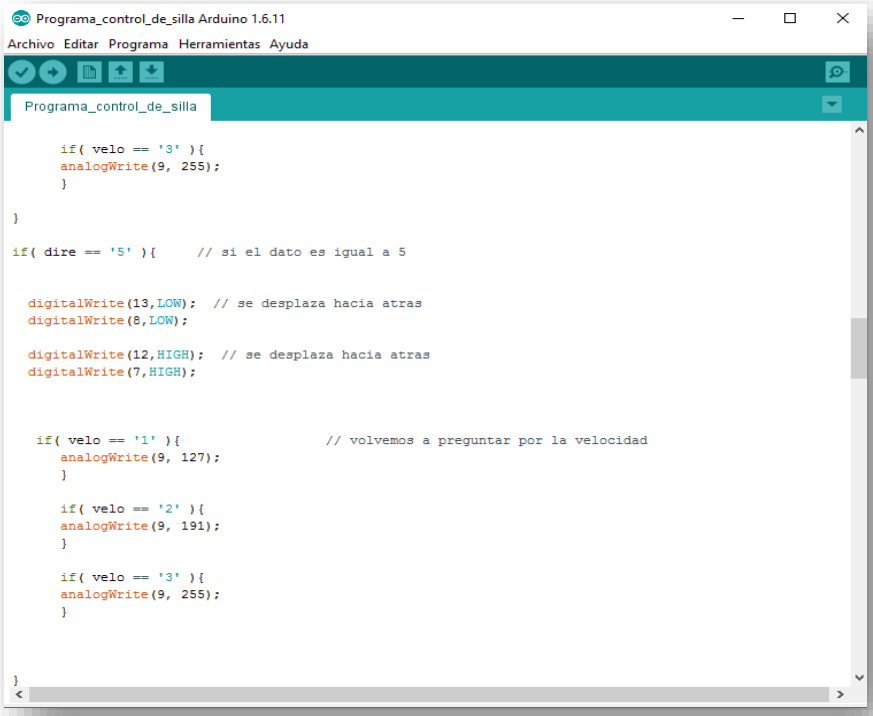


Figura 41 Programación Arduino para el prototipo

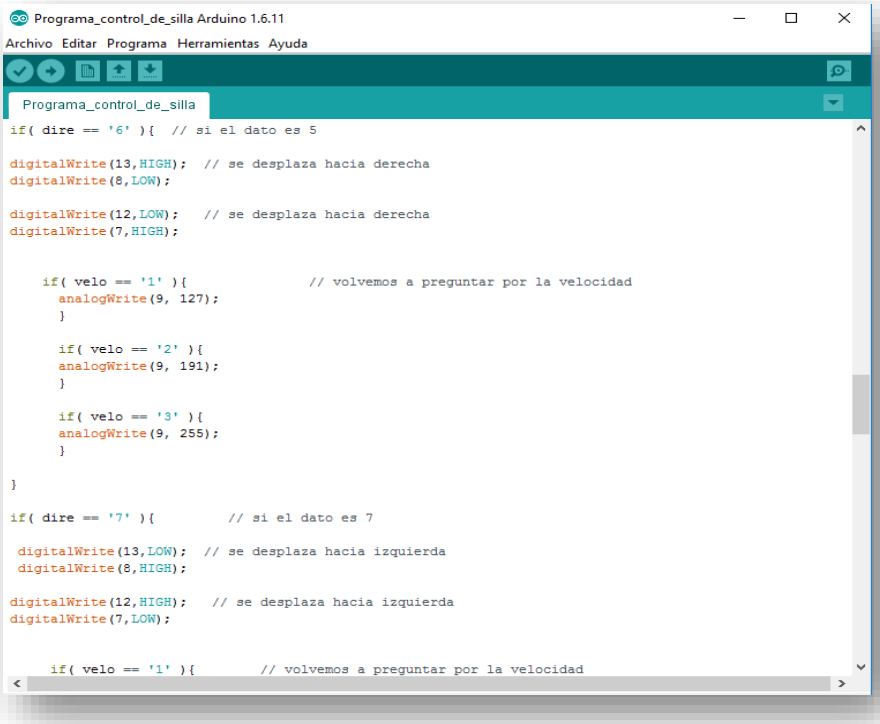


Figura 42 Programación Arduino para el prototipo

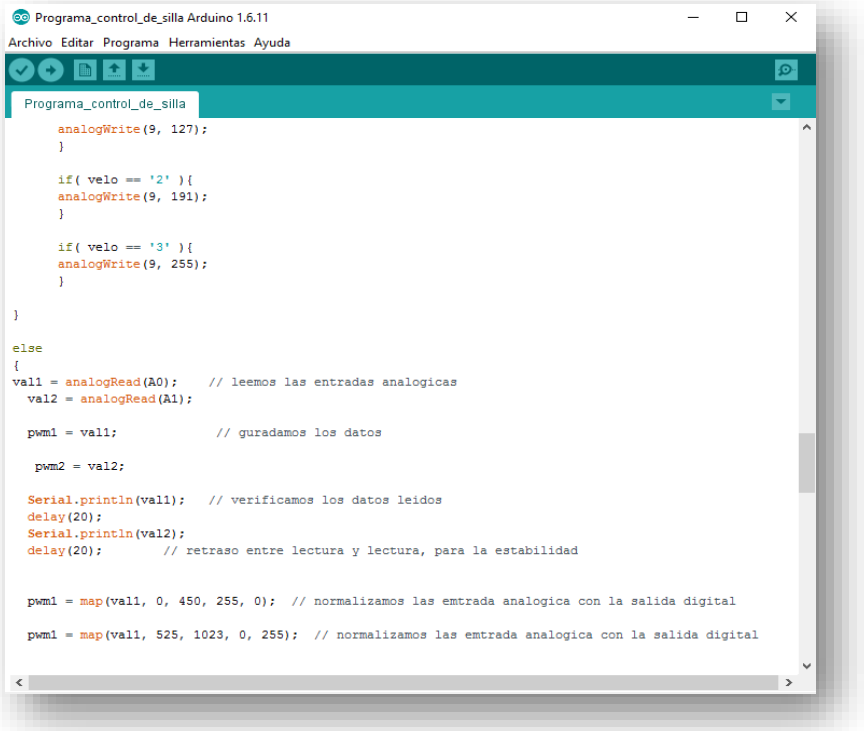


Figura 43 Programación Arduino para el prototipo

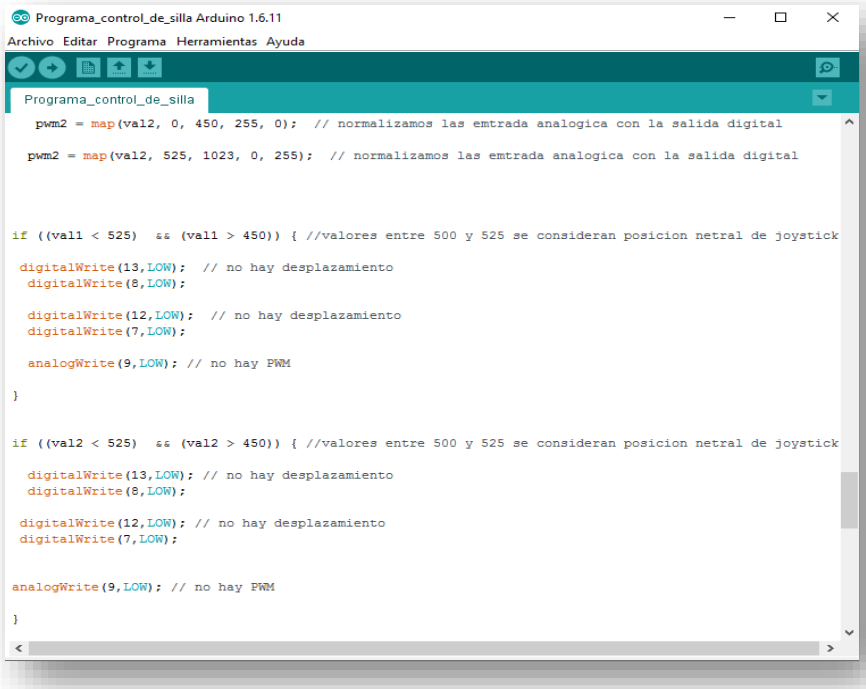


Figura 44 Programación Arduino para el prototipo

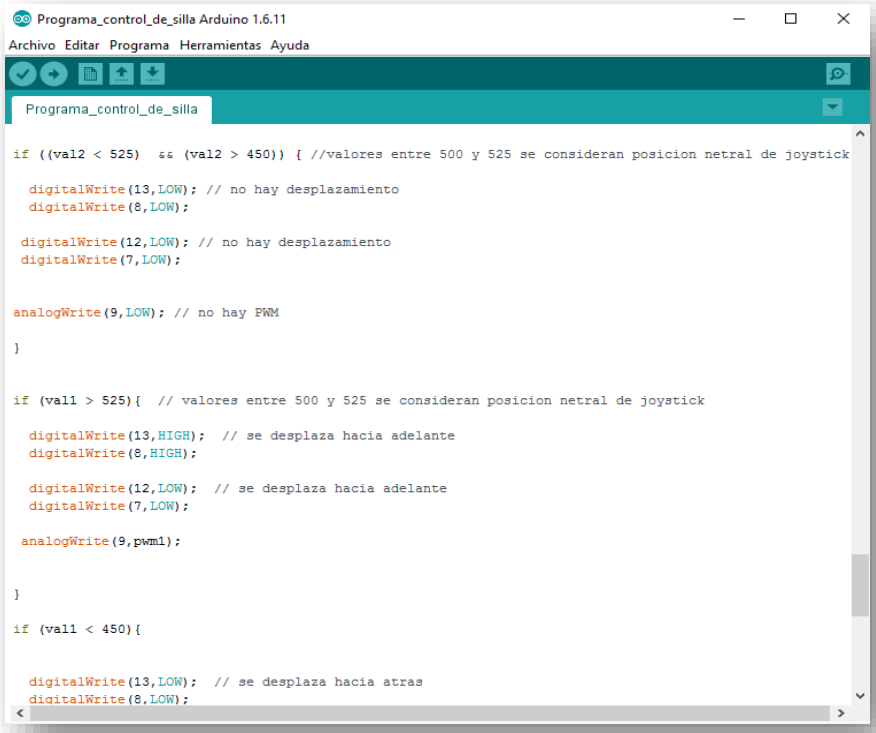


Figura 45 Programación Arduino para el prototipo

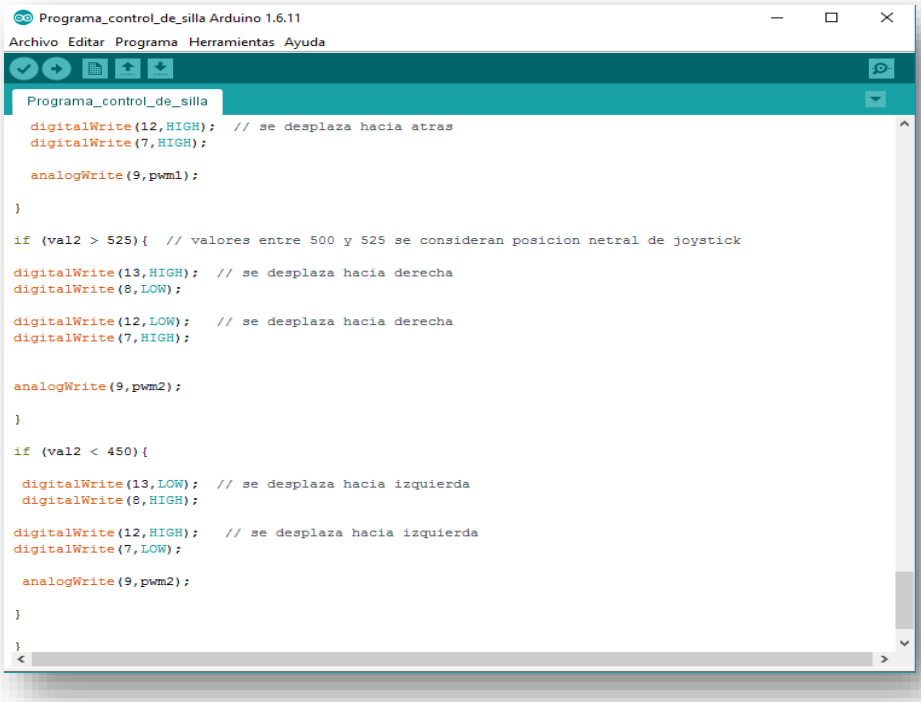


Figura 46 Programación Arduino para el prototipo

5.2.7.2 Código de programación para la Aplicación móvil.

A continuación, se muestra la programación utilizada para efectos del prototipo. La aplicación se programa con un código de bloques como se muestra en las siguientes imágenes, ello lo hace muy fácil de comprender y utilizar, en el anexo 2. Se pondrá u manual básico de programación para el APP inventor.

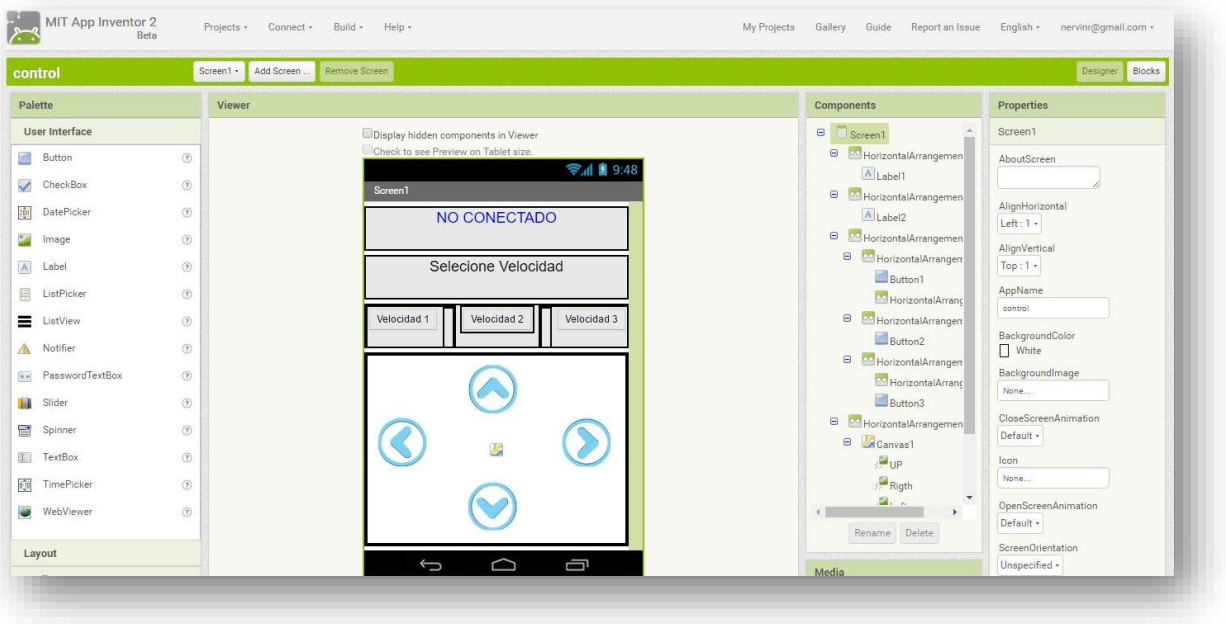


Figura 47 Ventana principal APP Inventor 2

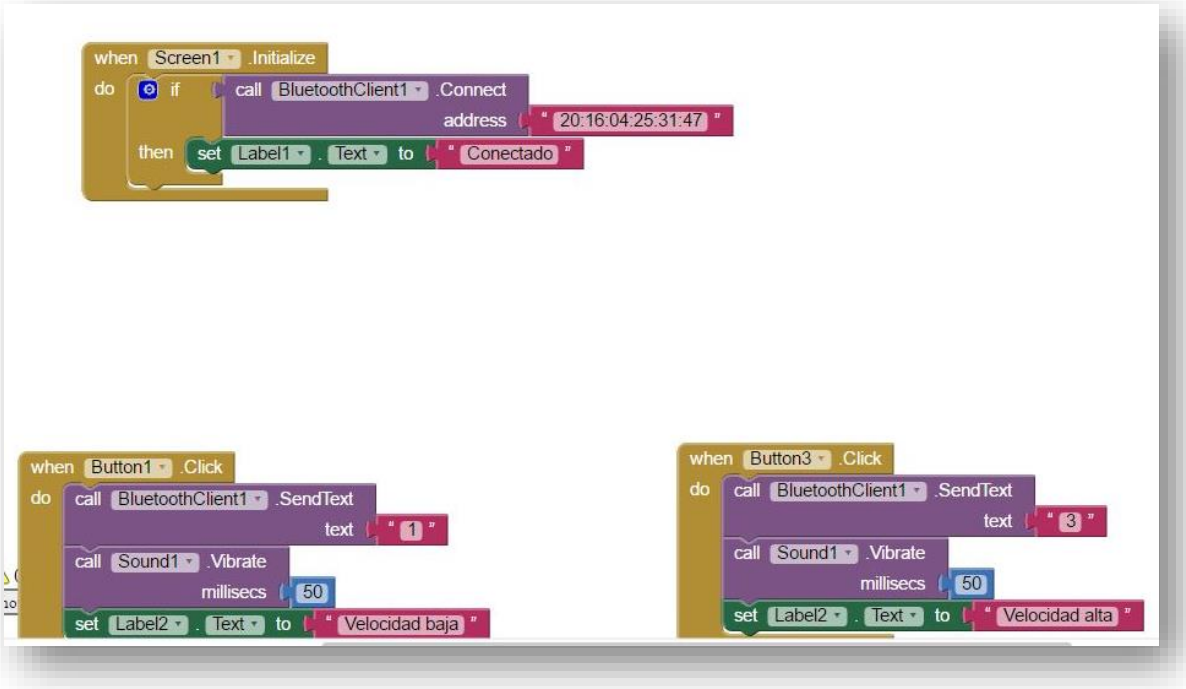


Figura 48 Inicialización y selección de velocidad

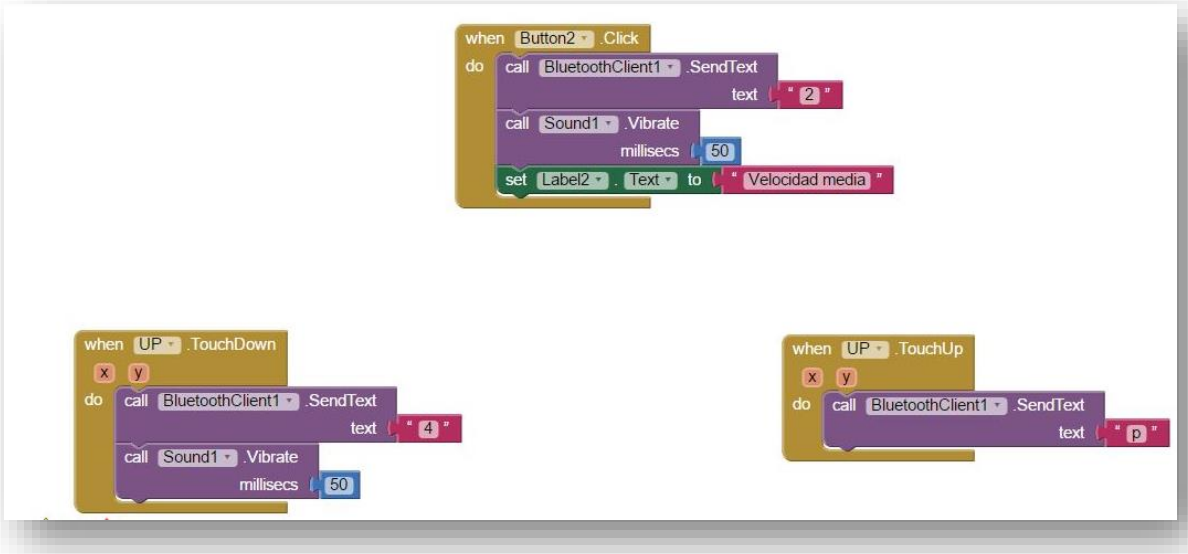


Figura 49 Selección de velocidad, desplazamiento hacia adelante

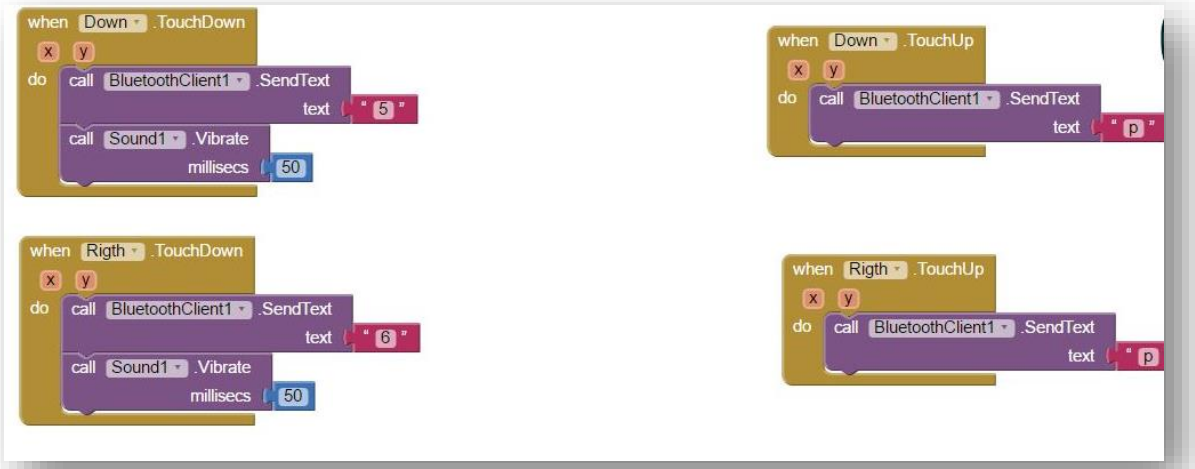


Figura 50 desplazamiento hacia atrás y derecha

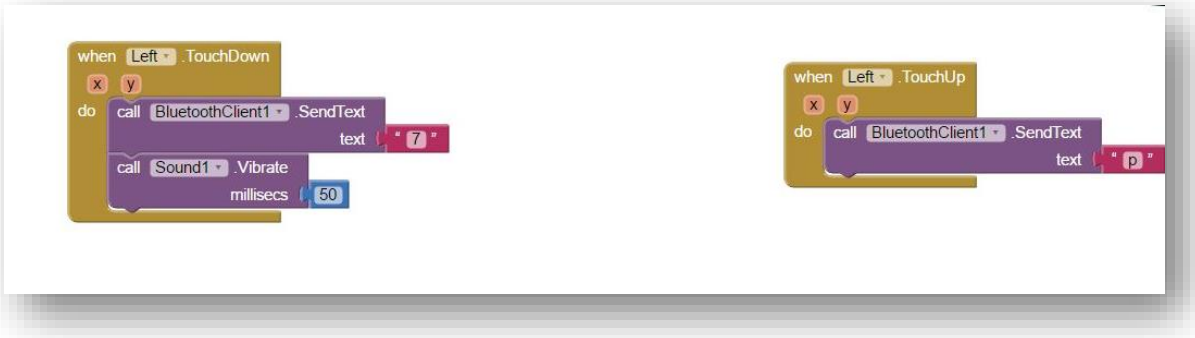


Figura 51 Desplazamiento hacia izquierda y parar

Figura 52 Visualización del usuario en el celular

5.3. Costos de implementación

En la Tabla 2. se exponen los costos de los componentes utilizados para la fabricación del prototipo, a eso se le agrega aproximadamente más 80 horas en el desarrollo del código para el Arduino, 180 horas aproximadamente de la elaboración de la aplicación móvil y unas 25 horas aproximadamente en el ensamble del prototipo y pruebas generales.

Solo considerando los costos de los componentes, se está muy por debajo de los precios de las sillas de ruedas eléctricas en el mercado que están entre los

$1.000 a $7.000 dólares, dependiendo de las características de las mismas y del fabricante.

Componente	Costo en dólares	Costo en colones	Gastos de envió	IV	Subtotal
Arduino Uno R3	$9,99	₡5.395	₡605	₡700	₡6.700
Adaptador de poder 9V	$4,88	₡2.635	₡605	₡342	₡3.582
Bluetooth HC-05	$7,99	₡4.315	₡605	₡561	₡5.481
L293D Motor driver	$5,50	₡2.970	₡605	₡387	₡3.962
Kit car chassis	$11,78	₡6.361	₡605	₡827	₡7.793
Caja para arduino	$12,25	₡6.615	₡605	₡860	₡8.080
Cables para batería 9V	$1,34	₡724	₡605	₡96	₡1.425
Batería de 9 V		₡3.535	₡0	₡0	₡3.535
Batería 1.5V AA		₡1.325	₡0	₡0	₡1.325
Programación arduino		₡30.000	₡0	₡0	₡30.000
Programación Aplicación móvil		₡20.000	₡0	₡0	₡20.000
Ensamble del prototipo		₡15.000	₡0	₡0	₡15.000
			TOTAL		**₡106.882**

Tabla 2 Costos de componentes

Fuente: Elaboración propia

5.4. Descripción de actividades

5.4.1. Descripción de las actividades llevadas a cabo

Fecha	Horas	Actividad	Objetivo	Resultado
2-May-2016	12	Elaboración de inventario	Verificación estado de partes que se podrían reutilizar	concluido
14-May-2016	16	Elaboración de marco de silla	dar la estructura para la apariencia de una silla	concluido
16-May-2016	1	Ensamble de partes del kit car chasis para proyectos	determinar si el kit llego completo y funcionamiento	concluido
16-May-2016	3	Colocación de los diferentes componentes	determinar el peso y balance de las cargas sobre la silla	concluido
21-May-2016	1	Fijación de componentes	posición final de cada componente	concluido
21-May-2016	3	Prueba de conexión Bluetooth modulo HC05	Determinar el estado del modulo y pre-programarlo para el prototipo	concluido
28-May-2016	3	Prueba de tarjeta arduino	Determinar el estado del modulo y pre-programarlo para el prototipo	concluido
4-Jun-2016	160	Programación de código arduino	Programa principal de control	concluido
11-Jun-2016	180	Programación de aplicación Móvil	Aplicación para el celular	concluido
18-Jun-2016	8	Pruebas generales		concluido

Tabla 3 Actividades realizadas

Fuente: Elaboración propia

CAPÍTULO VI: CONCLUSIÓNES Y RECOMENDACIONES

6.1 Conclusiones.

Gracias a la elaboración de la propuesta se puede concluir como los componentes que comúnmente presentan más deterioro son el *Joystick* y la tarjeta de control. En razón de lo anterior, resulta posible lograr controlar una silla de ruedas por medio de una tarjeta Arduino y además poderla controlar por un dispositivo móvil a través de una aplicación móvil, diseñada en una herramienta gratuita con el APP Inventor, ello dará la posibilidad de rescatar cualquier tipo de silla sin importar la marca o modelo y que esté en abandono por deterioro de su etapa de control, ya sea por la tarjeta principal o por el Joystick. También se puede agregar que gracias a la versatilidad de la tarjeta Arduino Uno, la cual de acuerdo con sus características permitiría la adición de nuevas funciones para proyectos futuros, facilitando una mejor interacción con los usuarios. Sin dejar a un lado el tema económico y como se demuestra en la tabla 2 del capítulo 5.3. Costos de componentes, se concluye que la realización de esta propuesta representa un ahorro de 80% con respecto al costo de la adquisición de la silla más económica presente en el mercado, la cual para el momento de la consulta era de al menos $1.000.

6.3 Recomendaciones.

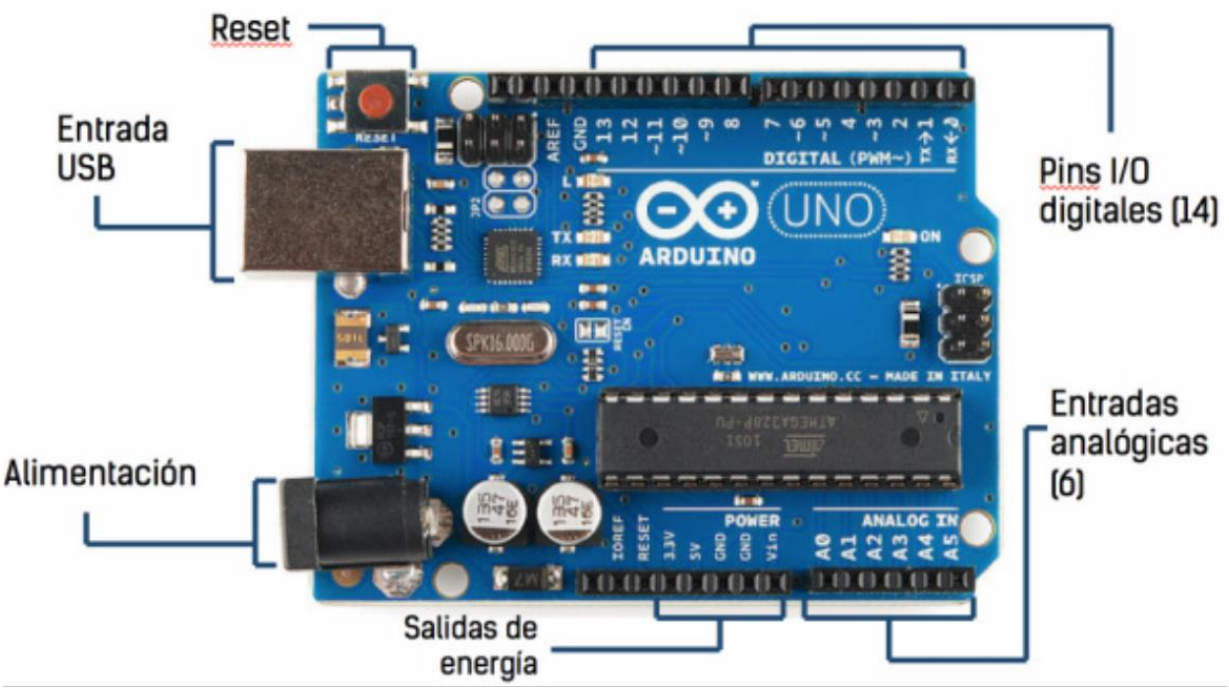
De acuerdo con los conocimientos adquiridos durante la elaboración de este prototipo se dan las siguientes recomendaciones para realizar mejoras a proyectos futuros.

1. Diseñar la parte electrónica y mecánica en conjunto para facilitar la integración de ambas partes.
2. Construir o comprar un joystick, pensando en las exigencias o necesidades de cada usuario.
3. Evaluar un sistema de frenado para soportar el peso de la silla y el usuario en una pendiente leve.
4. El APP inventor es una herramienta gratuita, pero solo funciona para teléfonos con sistema operativo Android, por lo tanto, se deben tomar en cuenta los sistemas operativos más usados para los teléfonos celulares con el fin de que la solución sea más fácil de comercializar, hay software que hacen aplicaciones para IOS de Apple, se debe investigar cuál es el mejor a la hora de ejecutarlos en estos dispositivos.

ANEXOS

Anexo 1

Arduino UNO pinout



Anexo 2

Manual básico de APP inventor

BÁSICOS APP INVENTOR

Manual de Introducción a AppInventor

VNiVERSiDAD D SALAMANCA
CAMPUS DE EXCELENCIA INTERNACIONAL

Oficina de Conocimiento Abierto

ServicioInnovación y ProducciónDigital

Manual básico de APP inventor

Contenido

Manual básico de APP inventor

1. ¿Qué es AppInventor?

AppInventor es una aplicación web de Google que permite crear aplicaciones para el sistema operativo de dispositivos móviles Android. Utiliza un editor Drag and Drop (Arrastrar y soltar) para la generación de interfaces gráficas y un sistema de bloques para gestionar el comportamiento de la aplicación. Los proyectos generados a través de esta herramienta se almacenan automáticamente en los servidores de App Inventor, permitiendo llevar en todo momento un seguimiento y control de todo nuestro trabajo.

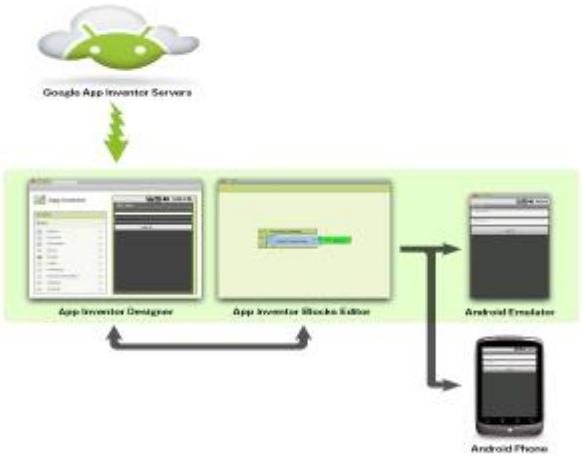


Figura 1. Visión global de AppInventor

La principal característica de *AppInventor* es que no es necesario tener ningún conocimiento de programación para desarrollar las aplicaciones. Simplemente basta con disponer de un navegador web y una cuenta de usuario de *Google*.

Manual básico de APP inventor

2. ¿Qué tipo de aplicaciones pueden crearse con AppInventor?

AppInventor ofrece un amplio abanico de posibilidades para los desarrolladores:

- ***Juegos***

  Puede crearse juegos sencillos haciendo uso incluso del acelerómetro incluido en el dispositivo móvil.
- ***Aplicaciones educativas***

  Es posible desarrollar aplicaciones útiles para educación, como por ejemplo tests de respuestas múltiples o preguntas directas.
- ***Aplicaciones de geolocalización***

  Puede hacerse uso del dispositivo GPS incluido en el móvil para crear aplicaciones de geolocalización.
- ***Aplicaciones multimedia complejas***

  Pueden crearse aplicaciones que van desde reconocimiento de códigos de barras hasta reproducir vídeos y música o tomar fotografías.
- ***Aplicaciones orientadas a la Web***

  Pueden desarrollarse aplicaciones que se comuniquen con la web (Twitter, RSS, etc.).

Manual básico de APP inventor

3. ¿Cómo se construye una aplicación en AppInventor?

Las aplicaciones construidas mediante AppInventor están compuestas por los elementos que se muestran en el siguiente diagrama:

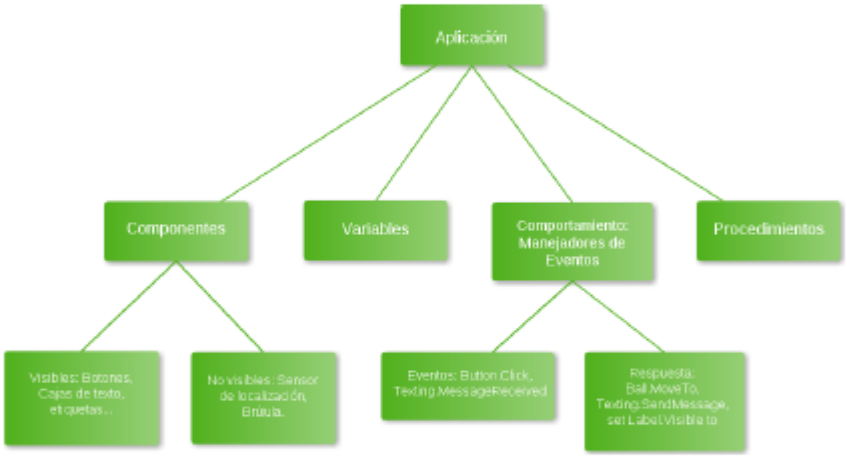


Figura 2. Arquitectura interna de una aplicación creada con AppInventor

Una buena manera de entender una aplicación, es descomponerla en dos partes, por un lado los componentes y por otro los comportamientos.

Manual básico de APP inventor

3.1. Componentes

Hay dos tipos de componentes principales en cualquier aplicación: los visibles y los no visibles.

Los componentes visibles son aquellos que podemos ver una vez hemos ejecutado nuestra aplicación (botones, cajas de texto, etiquetas, etc.). El conjunto de estos elementos se denomina comúnmente como la interfaz de usuario de la aplicación.

Por otro lado, los componentes no visibles son aquellos que no podemos ver en la aplicación, ya que no son parte de la interfaz de usuario. Proporcionan acceso a la funcionalidad interna de los dispositivos; por ejemplo, el componente *Texting* permite enviar y procesar mensajes de texto, y el componente *LocationSensor* permite determinar la localización del dispositivo.

Ambos componentes están definidos mediante una serie de propiedades. Las propiedades son fragmentos de memoria que permiten almacenar información relativa al componente al que referencian. Los componentes visibles, por ejemplo, disponen de propiedades relativas a su posición, altura y anchura, y alineación, que definen conjuntamente su aspecto dentro de la aplicación global. Todas estas propiedades se definen dentro del diseñador de componentes de AppInventor.

Manual básico de APP inventor

Figura 3. Ejemplo de un componente y sus propiedades

3.2. Comportamiento

El comportamiento define como una aplicación debe responder ante una serie de eventos, los producidos por la interacción del usuario (un clic de botón) y los externos (un SMS recibido en nuestro dispositivo). En este punto es donde reside la mayor complejidad en el desarrollo de aplicaciones. Afortunadamente, AppInventor proporciona un lenguaje visual de bloques que nos permite definir comportamientos de una forma muy precisa.

Normalmente, podemos identificar el desarrollo de aplicaciones con la elaboración de una "receta", es decir, siguiendo una secuencia lineal de instrucciones.

Manual básico de APP inventor

Sin embargo, la mayoría de las aplicaciones actuales, no cumple estrictamente este tipo de paradigma. No se ejecutan una serie de instrucciones en un orden predeterminado, sino que, la aplicación reacciona a una serie de eventos, normalmente iniciados por el usuario final de la aplicación. Por ejemplo, si el usuario hace clic sobre un botón, la aplicación responde realizando alguna operación (enviar un mensaje de texto, confirmar una determinada operación, etc.).

Este tipo de aplicaciones se pueden interpretar como un conjunto de componentes que reaccionan ante unos determinados eventos. Las aplicaciones incluyen una serie de "recetas" (secuencias de instrucciones), las cuales se ejecutan cuando se producen los eventos asociados a las mismas.

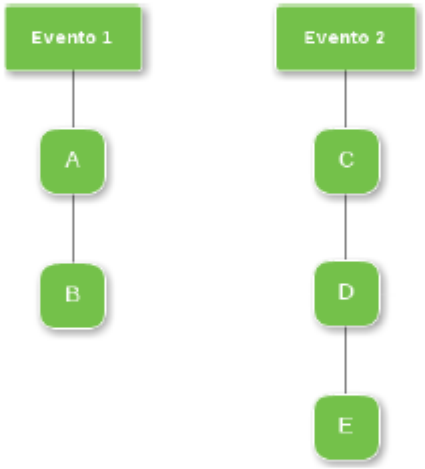


Figura 4. Ejemplo secuencial de eventos e operaciones

La mayoría de los eventos son generados por el usuario final de la aplicación, salvo en algunos casos. Una aplicación puede reaccionar a eventos que suceden en el propio dispositivo, como por ejemplo: cambios en el sensor de orientación, eventos generados en otros dispositivos, datos que llegan desde la web, etc.

Manual básico de APP inventor

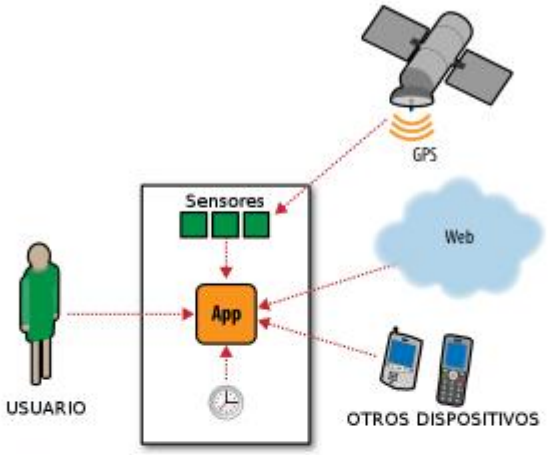


Figura 5. Una aplicación puede responder tanto a eventos internos, externos o generador por el usuario.

El motivo principal por el cual podemos afirmar que las aplicaciones desarrolladas mediante AppInventor son intuitivas, es porque están basadas en el paradigma evento-respuesta.

Manual básico de APP inventor

3.2.1. Eventos

A continuación, explicaremos brevemente los diferentes tipos de eventos que pueden desencadenar diferentes acciones en este tipo de aplicaciones.

Tipo de Evento	Ejemplo
Evento iniciado por el usuario	*Cuando* el usuario hace clic en el *boton_1*, *hacer*...
Evento de inicialización	*Cuando* la aplicación se inicia, *hacer*...
Evento de temporización	*Cuando* han pasado 20 milisegundos, *hacer*...
Evento de animación	*Cuando* dos objetos colisionan, *hacer*...
Evento externo	*Cuando* el teléfono recibe un SMS, *hacer*...

Eventos iniciados por el usuario

Son el tipo más común de eventos. Reflejan la interacción del usuario final con la aplicación. Por ejemplo, en la siguiente figura podemos observar como se ha definido que cuando el usuario hace click sobre el botón *"SpeakItButton"* se reproduce oralmente el contenido escrito en la caja de texto *"TextBox1"*.

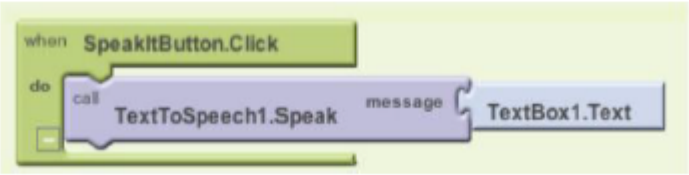


Figura 6. Ejemplo de evento iniciado por la interacción del usuario

Manual básico de APP inventor

Eventos de inicialización

En algunas ocasiones nuestras aplicaciones requieren realizar ciertas funciones en el momento en el que se inician. AppInventor considera este inicio de la aplicación como un evento.
En el siguiente ejemplo, podemos observar como en el momento en el que se inicia la aplicación realizamos una determinada acción, en este caso, invocar a la función *"MoveMole"*.

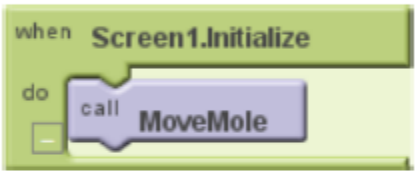


Figura 7. Ejemplo de evento de inicialización

Eventos de temporización

Podemos necesitar en algunos casos que cierta actividad de nuestra aplicación se realice en un cierto momento. AppInventor dispone de un componente denominado *"Clock"*, mediante el cual podremos programar la ejecución de determinadas acciones en un determinado momento.
Por ejemplo, en la siguiente figura podemos observar como se ha definido un temporizador para que en un determinado tiempo mueva una bola desplazándola 10 unidades sobre el eje de las X.

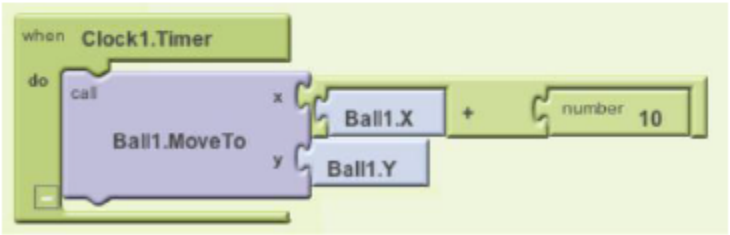


Figura 8.. Ejemplo de evento de temporización

Manual básico de APP inventor

Eventos de animación

Las actividades en las que se ven envueltos objetos gráficos también pueden producir eventos. De esta manera podemos crear juegos o aplicaciones con animaciones interactivas, controlando en todo momento que debería ocurrir.

En el siguiente ejemplo, se ha definido un comportamiento que controla las colisiones entre el objeto *"Ball1"* y el objeto denominado *"other"*, de tal modo que cuando se produce una colisión entre ambos, el objeto *"Ball1"* cambia su color a rosa y se reproduce un sonido de explosión.

Figura 9. Ejemplo de evento de animación

Eventos externos

Cualquier entrada externa hacia el dispositivo es considerada también como un evento. Por ejemplo, el hecho de recibir un mensaje de texto o la posición GPS son interpretados como eventos externos.

En el siguiente ejemplo, podemos observar como se ha definido un conjunto de bloques que reaccionan al evento de recibir un mensaje de texto. Una vez recibido, se responde al emisor con el mensaje de *"Stop texting me!"*.

Manual básico de APP inventor

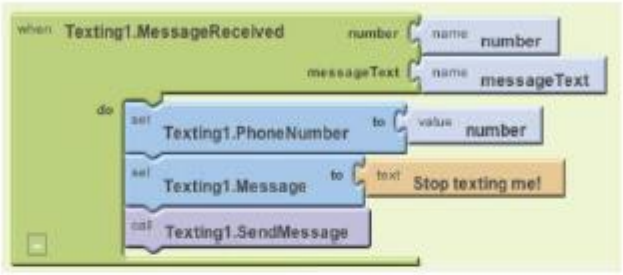


Figura 10. Ejemplo de evento externo

Para resumir, podemos decir que todas las aplicaciones que desarrollemos estarán formadas por un conjunto de eventos y sus respectivas respuestas. Nuestro trabajo será conceptualizar y definir las respuestas a los eventos que queramos manejar.

Manual básico de APP inventor

3.2.2. Manejadores de Eventos

Existen varios tipos de manejadores de eventos, los cuales explicaremos a continuación.

Condicionales

Las respuestas a eventos no son siempre secuencias lineales. En algunos casos, se pueden evaluar ciertas condiciones y elegir en función de ello que hacer.

Figura 11. Ejemplo de manejador condicional

Manual básico de APP inventor

Bloques de repetición

En algunas ocasiones podemos desear que nuestra aplicación repita ciertas operaciones. En estos casos utilizaremos los bloques de repetición proporcionados por AppInventor, que nos permitirán ejecutar ciertas tareas un número determinado de veces o mientras sea verdadera alguna condición.

Recordar valores

Debido a que los manejadores de eventos pueden ejecutar bloques, en algunas ocasiones es necesario mantener cierta información almacenada. Esta información se almacena en unas unidades de memoria denominadas variables, las cuales podremos definir en nuestras aplicaciones. Por ejemplo en una aplicación para jugar, podemos definir una variable puntuación para almacenar y modificar su valor en función de los aciertos y errores del usuario.

Podemos decir que las variables son como las propiedades , pero no están asociadas a ningún componente en particular. Las variables almacenan datos temporalmente mientras la aplicación se está ejecutando; cuando cerramos la aplicación, los datos ya no están disponibles.

En algunas ocasiones, puede ser necesario que ciertos datos estén disponibles no sólo mientras se ejecuta la aplicación, sino también cuando se cierra y vuelve a ejecutarse. Es en este caso particular, cuando necesitaremos utilizar un almacenamiento persistente, utilizando algún tipo particular de base de datos.

Manual básico de APP inventor

Comunicaciones con la web

Algunas aplicaciones se comunican a través de internet enviando peticiones hacia servicios web.

Twitter es un ejemplo de servicio web con el cual AppInventor puede comunicarse. Podemos crear aplicaciones que sean capaces de enviar peticiones y mostrar los tweets de nuestros seguidores e incluso actualizar nuestro estatus.

4. Requisitos Previos

Antes de comenzar con el desarrollo de aplicaciones para Android, necesitamos seguir una serie de pasos.

Comprobar que cumplimos todos los requisitos para poder utilizar AppInventor

- Ordenador y Sistema Operativo
  - Macintosh (con procesador Intel): Mac OS X 10.5, 10.6
  - Windows: Windows XP, Windows Vista, Windows 7
  - GNU/Linux: Ubuntu 8+, Debian 5+
- Navegador Web
  - Mozilla Firefox 3.6 o superior
  - Nota: Si estamos utilizando Firefox con la extensión NoScript, deberemos deshabilitarla para evitar problemas de utilización.
  - Apple Safari 5.0 o superior
  - Google Chrome 4.0 o superior
  - Microsoft Internet Explorer 7 o superior

Manual básico de APP inventor

Para poder empezar a utilizar el servicio de AppInventor necesitaremos tener una *cuenta de google*. Si no disponemos de ella, podremos crear una de manera gratuita en el siguiente enlace.

Comprobar nuestra configuración de la máquina virtual de Java

- Nuestro ordenador necesita tener instalado Java 6. Para descargar la última versión de Java visitaremos la web www.java.com.
- Visitar la página de prueba de Java. Deberíamos observar un mensaje que indique que Java está funcionando correctamente y que la versión de Java es la 1.6 o superior.
- Ejecutar el test Java de AppInventor haciendo clic en este enlace. Este proceso comprobará que nuestro navegador está configurado apropiadamente para ejecutar Java y que nuestro ordenador puede ejecutar aplicaciones mediante Java Web Start.

AppInventor no funcionará en nuestro ordenador si estos tests no han tenido éxito. Antes de comenzar a utilizar la aplicación debemos comprobar todos los pasos anteriores.

Instalar el software de AppInventor en nuestro ordenador

Antes de utilizar AppInventor, debemos instalar el software adecuado al sistema operativo que vayamos a utilizar. En los siguientes enlaces se proporcionan instrucciones detalladas de como instalar el software:

- Instrucciones para Mac OS X
- Instrucciones para GNU/Linux
- Instrucciones para Windows

Manual básico de APP inventor

A partir de este momento, ya podremos empezar a utilizar y crear nuestras aplicaciones mediante AppInventor.

5. El Entorno de Desarrollo

EL entorno de programación de AppInventor tiene tres partes fundamentales:

- El *Diseñador de Componentes* se ejecuta en el navegador web. Es utilizado para seleccionar los componentes de nuestra aplicación y especificar sus propiedades. A través de él, iremos definiendo el aspecto visual.
- El *Editor de bloques* se ejecuta en una ventana independiente del *Diseñador de Componentes*. Nos permitirá crear los comportamientos necesarios de nuestra aplicación y asociarlos a sus respectivos componentes.
- Un dispositivo Android nos permitirá ejecutar y comprobar nuestras aplicaciones mientras las estamos desarrollando. Si no disponemos de ningún dispositivo adecuado, podremos probar nuestras aplicaciones utilizando el emulador de Android, el cual viene integrado dentro del sistema.

En la siguiente figura se puede visualizar cada una de las partes mencionadas anteriormente.

Manual básico de APP inventor

Figura12. El Diseñador de Componentes, el Editor de Bloques y el Emulador de Android.

Para empezar a utilizar AppInventor, comenzaremos introduciendo la siguiente dirección web en nuestro navegador http://appinventor.googlelabs.com. La primera vez que iniciemos sesión en la aplicación, podremos ver nuestra Página de Proyecto, la cual estará vacía porqué aún no hemos desarrollado ninguna aplicación. Para crear un proyecto, haremos clic sobre el botón *New* situado en la parte superior izquierda de la pantalla, introduciremos el nombre que queramos para el proyecto y haremos clic en OK.

La primera ventana que nos aparecerá a continuación es el *Diseñador de Componentes*. Una vez que estamos dentro, haremos clic sobre el botón *Open Blocks Editor* situado en la parte superior derecha. En este momento, el Editor de Bloques aparecerá en una nueva ventana. Este proceso puede tardar en torno a 30 segundos.

En la ventana del Editor de Bloques podemos observar dos botones en la parte superior derecha, como muestra la siguiente figura.

Manual básico de APP inventor

Figura 13. Parte superior del editor de bloques

Si disponemos de un dispositivo Android y un cable USB, conectaremos el dispositivo al ordenador y seleccionaremos *"Connect to Device"*. Por el contrario sino disponemos de ningún dispositivo Android o si queremos probar nuestras aplicaciones utilizando un emulador, haremos clic en *"New Emulator"* y esperaremos en torno a 30 segundos hasta que el emulador termina de cargar. Cuando haya terminado el proceso, haremos clic en *"Connect to Device"* y seleccionaremos el emulador creado previamente.

5.1. Diseñando los Componentes

Los componentes son los elementos que combinamos para crear nuestras aplicaciones. Algunos son muy simples, como por ejemplo una etiqueta que muestra un texto en la pantalla, o un botón, mediante el cual podemos iniciar una determinada acción. Otros componentes son más elaborados; un componente *Canvas* que permite visualizar imágenes o animaciones; el acelerómetro, un sensor de movimiento que detecta cuando movemos o agitamos el teléfono; o componentes que nos permiten enviar mensajes de texto, reproducir música y video, obtener información desde páginas web, etc.

Cuando abrimos el *Diseñador de Componentes*, aparecerá la siguiente ventana.

Manual básico de APP inventor

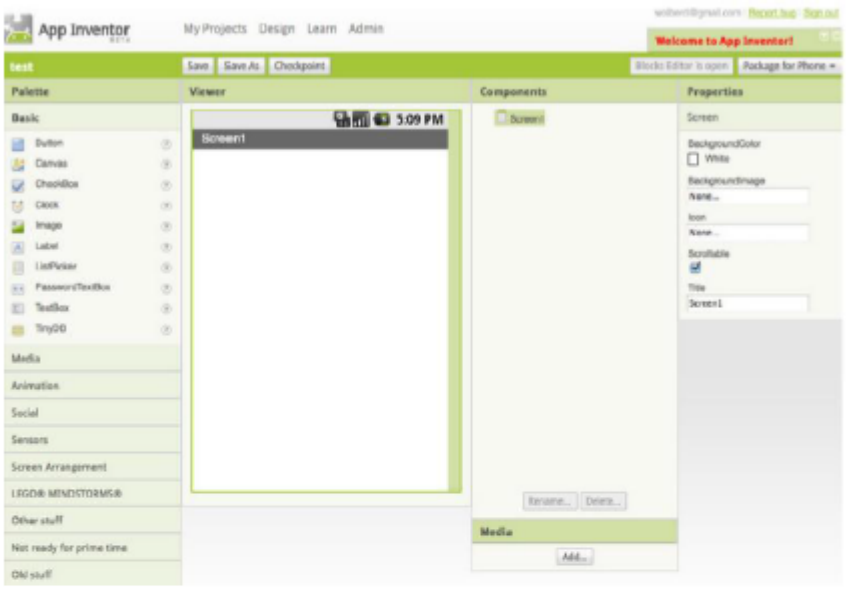


Figura 14. Ventana principal del Diseñador de Componentes

El *Diseñador de Componentes* está dividido en varias partes:

En la parte central encontramos un área denominada *Viewer*. Aquí es donde colocaremos nuestros componentes en función de cómo queramos que sea el diseño visual de nuestra aplicación. Una vez especificados y añadidos los componentes, debemos probar como se visualiza realmente nuestra aplicación en un dispositivo Android, utilizando para ello un dispositivo físico o el emulador.

A la izquierda de la aplicación encontramos un área denominada *Palette*, en la que podemos encontrar la lista completa de componentes de AppInventor. La *Palette* está dividida en varias

categorías. Haciendo clic en cada una de ellas podremos ir viendo todos los componentes disponibles.

A la derecha del *Viewer* podemos ver el listado de *Componentes,* el cual muestra una lista de todos los componentes que vamos añadiendo a nuestra aplicación. Todos los componentes que vayamos añadiendo al *Viewer,* se irán mostrando automáticamente en el listado. Por defecto, siempre aparece el componente *Screen1,* que hace referencia a la pantalla del teléfono.

En la parte inferior podemos observar un área denominada *Media* que muestra todos los componentes multimedia (imágenes y sonidos) que incluyamos en nuestro proyecto.

Por último en la parte derecha de la aplicación, encontramos la sección de propiedades, mediante la cual podremos ir modificando, según nuestras necesidades, todos y cada uno de los elementos que vayamos incluyendo en la aplicación. La lista de propiedades irá cambiando en función del elemento que estemos editando, para ello haremos clic sobre el elemento deseado en el listado y aparecerán las propiedades que podemos modificar para el elemento actual seleccionado.

5.2. Añadiendo Comportamientos

El *Editor de Bloques* (*Blocks Editor*) nos permite añadir y asignar tareas específicas a los componentes individuales de nuestra aplicación, de tal modo que podamos ir creando de manera conjunta la funcionalidad global que pretendemos que tenga nuestra aplicación.

El *Editor de Bloques* está implementado como una aplicación de *Java Web Start,* que se ejecuta en nuestro ordenador.

Para iniciarlo, debemos hacer clic sobre el botón *"Open the blocks editor"* situado en la parte superior derecha del *Diseñador de Componentes.* Una vez pulsado, el texto del botón cambiará y nos indicará que se está cargando el *Editor de Bloques.* Nuestro navegador web nos

Manual básico de APP inventor

preguntará en una nueva ventana que queremos hacer con la aplicación. Aceptamos y automáticamente se mostrará la ventana del *Editor de Bloques*.

Una vez cargado el *Editor de Bloque* aparecerá la siguiente ventana.

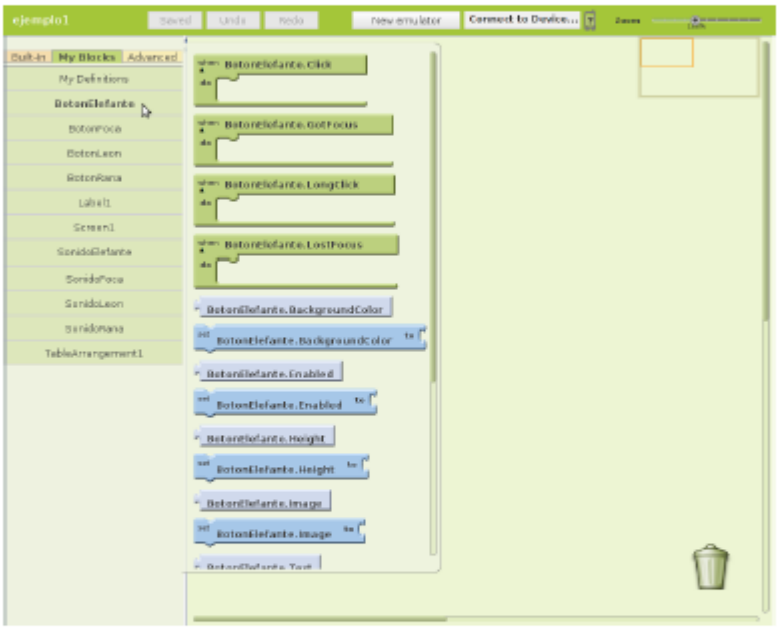


Figura 15. Ventana principal del Editor de bloques

Manual básico de APP inventor

Se divide en las siguientes partes:

- En la parte izquierda de la pantalla encontramos la zona para seleccionar bloques. A su vez, incluye tres pestañas: *Built-In, My Blocks y Advanced*. La pestaña *Built-In* contiene siete grupos genéricos de bloques: *Definition, Text, Lists, Math, Logic, Control y Colors*. Todos estos bloques estarán siempre disponibles para incluirlos en nuestras aplicaciones, independientemente de los componentes utilizados y añadidos a través del diseñador. La pestaña *My Blocks* nos permite acceder a todos los componentes definidos en el diseñador, con el fin de asignarles la funcionalidad y el comportamiento que deseemos. Por último, la pestaña de *Advanced,* nos permite aplicar comportamientos sobre grupos de componentes (botones, etiquetas, etc.) de manera global.
- En la parte central se encuentra el editor de bloques, mediante el cual definiremos las tareas y el comportamiento de los componentes de nuestra aplicación. Iremos combinando bloques según nuestras necesidades para completar el funcionamiento de nuestra aplicación.

Manual básico de APP inventor

6. Referencias

http://code.google.com/p/taiic/

http://appinventorintro.com/

http://www.tuappinvetorandroid.com/

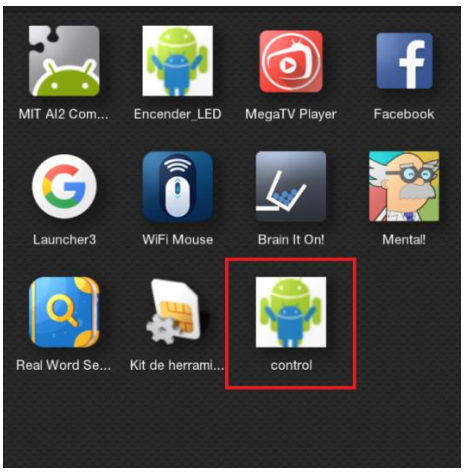
http://www.appinventorbeta.com/forum/

http://appinventorblog.com/

Anexo 3

Guía básica para el uso de la aplicación de control

1. Se selecciona el icono de la aplicación de control.



2. Cuando inicie la aplicación se debe esperar que pase de no conectado a conectado.

3. Una vez lograda la conexión se puede seleccionar la velocidad a la que se desea realizar el desplazamiento.

4. La velocidad va de forma ascendente el botón de velocidad 1 se refiere a la velocidad más baja del sistema.

5. Cuando se selecciona velocidad 2 se pone el sistema a una velocidad media.

6. Cuando se selecciona la velocidad 3 se pone el sistema a máxima velocidad.

7. Las flechas de control son las que se ven en color azul claro y dan el movimiento hacia adelante, atrás, izquierda y derecha, según lo requiera el usuario.

BIBLIOGRAFIA

Barrantes Echavarría, R. (2004). Un camino al conocimiento, ENFOQUE CUANTITATIVO Y CUALITATIVO. . San Jose: Universidad Estatal a Distancia.

Barrantes Echavarría, R. (2004). BarrantUn camino al conocimiento, ENFOQUE CUANTITATIVO Y CUALITATIVO. San Jose: Universidad Estatal a Distancia.

bluetooth. (10 de junio de 2016). Obtenido de https://www.bluetooth.com/what-is-bluetooth-technology/bluetooth

GeekFactory. (03 de 05 de 2016). Obtenido de http://www.geekfactory.mx/tutoriales/bluetooth-hc-05-y-hc-06-tutorial-de-configuracion/

Historia de Arduino y su nacimiento. (s.f.). Obtenido de https://botscience.wordpress.com/2012/06/05/historia-de-arduino-y-su-nacimiento/

MikroElectronika. (20 de Junio de 2016). MikroElectronika. Obtenido de http://learn.mikroe.com/ebooks/microcontroladorespicc/chapter/introduccion-al-mundo-de-los-microcontroladores/

moviles, A. (s.f.). APPmoviles. Obtenido de http://appmoviles.net/que-es-el-app-inventor-para-que-sirve/

Tocci, R. J. (1996). Sistemas Digitales Principios y Aplicaciones.

MINOS. Catálogo [en línea]: de Sillas Eléctricas de la marca Minos. [España]: Innova Advanced Software Solutions. <http://www.minos97.com/pdf/SillasElectricas.pdf> [Consultada el 14 de Agosto de 2016].

Iniciativa Múltiple de Atención de Gaps a la Integración, Normalización y Accesibilidad. Catálogo [en línea]: de la silla de ruedas F55 de la marca Powertec. [España] Antonio Rodríguez. <http://www.imagina.org/mercado/fotos/Powertwch_F55_S.pdf> [Consultada el 17 de Agosto de 2016].

Instituto Tecnológico de Massachsetts. Documento [en línea]: Proyecto Vehículo Autónomo con Brazo Manipulador Controlado a Distancia Vía TCP/IP. José Antonio Domínguez Caballero. [Estados Unidos]. MIT.<http://mit.edu/jadc/www/memoriavacbmcd.pdf> [Consultada el 19 de agosto de 2016].

Universidad del País Vasco. Fuerza [en línea] de rodamiento en un plano inclinado. [España]: Ángel Franco García. Última actualización: 3 de Octubre de 2006. <http://www.sc.ehu.es/sbweb/fisica/dinamica/rozamiento/plano_inclinado/plano_inclinado.htm > [Consultada el 23 de agosto de 2016].

Renato Masías. Tutorial [en línea]: Diseño de Puentes H. Icy Phoenix Robots Perú. [Estados Unidos]. <http://www.robotsperu.org/foros/3-vt248.html?postdays=0&postorder=asc&start= 30> [Consultada el 03 de septiembre de 2016].

Ing. Francisco José Terrón Dáz. Documento [en línea]: Carrito controlado por la calculadora TI-92 Plus y el CBL. [Estados Unidos] <http://mx.geocities.com/calcti/recre98.htm> [Consultada el 06de septiembre de 2016].

D. Emilio J. Bueno Peña. Documento [en línea]: Montaje de un convertidor DC-DC en puente completo para excitar un motor DC de imán permanente. Departamento de Electrónica de la Universidad de Alcalá [España] <http://www.depeca.uah.es/docencia/ITT-SE/lep/curso_online/P4/P4_6.htm> [Consultada el 15 de septiembre de 2016].

TodoRobot. Documento [en línea]: MOSFET H-Bridge esquema y teoría de operación. [Argentina]. <http://www.todorobot.com.ar/documentos/hbridge-MOSFET.pdf> [Consultada el 20 de septiembre de 2016].

Lon Glazner. Documento [en línea]: Stamp-Controlled High Power H-Bridge. Parallax [Estados Unidos].

<www.parallax.com/dl/docs/cols/nv/vol2/col/nv52.pdf> [Consultada el 28 de septiembre de 2016]

F.L Castro. Documento [en línea]: Cálculo de disipadores. Terra [España]. <http://www.terra.es/personal2/equipos2/disipadores.htm> [Consultada el 02 de Octubre de 2016]

Luis M. Cárdaba. Documento [en línea]: Disipadores térmicos. [España]. http://www.lcardaba.com/articles/heatsinks/heatsinks.htm [Consultada el 10 de Octubre de 2016].

José Luis Molina. Documento [en línea]: Dispadores de calor. Iespana[Francia] <http://profesormolina2.iespana.es/electronica/componentes/disip/disip.htm> [Consultada el 19 de Octubre de 2016].

Rod Elliott. Documento [en línea]: The Heatsinks. Elliott Sound Products [Estados Unidos]. Publicada: 11 de febrero de 2005. <http://sound.westhost.com/articles/diy-heatsink.htm> [Consultada el 8 de octubre de 2016].

Miguel Ángel Montejo Ráez. Documento [en línea]: Introducción a los disipadores de calor en semiconductores de potencia. REDEYA [EstadosUnidos].

<http://www.redeya.com/electronica/cursos/disipa/disipa.htm> [Consultada el 24 de octubre de 2016].

Felipe Calvo Álvarez. Documento [en línea]: MOTOR DE IMANES PERMANENTES COMO PROPULSOR NAVAL. Revista Marina de la armada de Chile. [Chile]. <http://www.revistamarina.cl/revistas/1999/3/calvo.pdf> [Consultada el 26 de octubre de 2016]

Rafael Rus Palacios. Documento [en línea]: Diseño e implantación de un acelerador electrónico controlable en un vehículo monoplaza. Laboratorio de Automoción del Departamento de Ingeniería Mecánica. [España]. <www.tecnun.es/automocion/proyectos/acelerador_electronico/memoria.pdf> [Consultada el 01 de noviembre de 2016].