UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA: INGENIERÍA ELECTRÓNICA

Trabajo de titulación previo a la obtención del título de: INGENIERA E INGENIERO ELECTRÓNICOS

TEMA:

CONTROL DE UN ROBOT BÍPEDO HUMANOIDE EN UN ESPACIO 3D MEDIANTE EL SENSOR KINECT

AUTORES:
ARAUJO VÁSQUEZ VLADIMIR GIOVANNY
TIPÁN RIOFRÍO MYRIAM MERCEDES

DIRECTOR:
VÍCTOR VINICIO TAPIA CALVOPIÑA

Quito, abril de 2015

DECLARATORIA DE RESPONSABILIDAD Y AUTORIZACIÓN DE USO DEL TRABAJO DE GRADO

Nosotros, autorizamos a la Universidad Politécnica Salesiana, la publicación total o parcial de este trabajo de grado y su reproducción sin fines de lucro.

Además, declaramos que los conceptos, análisis desarrollados y las conclusiones del presente trabajo son de exclusiva responsabilidad de los autores.

Quito, abril 2015

Vladimir Giovanny Araujo Vásquez	Myriam Mercedes Tipán Riofrío
CI: 1718330242	CI: 1500679517

DEDICATORIA

Dedico este proyecto a mis padres Mariana y Guido por el apoyo durante toda mi carrera de ingeniería, tiempo en el cual pude seguir adelante gracias a su ejemplo, confianza y ayuda, a mi hermana Mayra quien siempre ha sido un modelo a seguir y que siempre fue un apoyo en mi vida estudiantil, y a su hijo Steven que llego recientemente a nuestra familia y a ser una motivación más, también a mi abuelita quien desde hace muchos años nos acompaña incondicionalmente siendo parte importante de la familia, finalmente a Myriam mi amiga, compañera y amor de mi vida quien con su cariño, trabajo y apoyo incondicional hizo posible que lográramos cumplir este proyecto y muchas metas más durante nuestra vida universitaria.

Vladimir Giovanny Araujo Vásquez

Dedico este trabajo de Titulación a Dios por cada día darnos una oportunidad para seguir adelante, a mis padres, Myriam y Alfredo, por la paciencia y confianza entregada durante el desarrollo del proyecto, a mi hermano Juan Carlos por darme siempre sus consejos, y ayudarme a escoger lo mejor para mis estudios y presionarme para terminarlos lo más pronto posible, a mis abuelitos de parte de papa, Mercedes y Luis que desde el cielo me han ayudado a tomar las mejores decisiones y a mis abuelitos de parte de mama, Bertha y Carlos por sus oraciones para que todo marche bien, a mis demás familiares y compañeros por sus palabras de aliento y buenos deseos, y finalmente pero no menos importante a Vladimir, mi aby, mi amigo y compañero incondicional, por su amor, apoyo y esfuerzo durante todo este tiempo, sin el este proyecto no hubiese avanzado.

Myriam Mercedes Tipán Riofrío

AGRADECIMIENTO

Agradecemos a la Universidad Politécnica Salesiana por habernos acogido durante nuestros estudios de pregrado, así también a todos los profesores que estuvieron a nuestro lado en el transcurso de este tiempo ya que con sus conocimientos nos han ayudado a formarnos como profesionales de bien, mil gracias.

A nuestro tutor el Ing. Víctor Vinicio Tapia que a lo largo de este proceso ha sabido guiarnos de forma concisa para la culminación de este trabajo, gracias por el apoyo y confianza durante el desarrollo de este proyecto.

Vladimir Giovanny Araujo Vásquez

&

Myriam Mercedes Tipán Riofrío

ÍNDICE

INTRODUCCIÓN **1**
CAPÍTULO 1 ..... **2**
PLANTEAMIENTO DEL PROBLEMA **2**
1.1 Planteamiento del problema 2
1.2 Objetivos 2
1.2.1 Objetivo general 2
1.2.2 Objetivos específicos 2
1.3 Justificación 3
1.4 Alcance 3
CAPÍTULO 2 ..... **4**
MARCO TEÓRICO **4**
2.1 Origen y desarrollo de la robótica 4
2.2 Definición y clasificación del robot 5
2.2.1 Robots de servicio y teleoperados 6
2.3 Plataformas humanoides 7
2.3.1 All-new Asimo (2011) 7
2.3.2 Pepper 8
2.3.3 Proyecto Romeo 9
2.3.4 DARwIn-OP robot (2012) 10
2.3.5 NAO robot 10
2.4 Cinemática del robot 11
2.4.1. Grados de libertad criterio de Kutzbach-Grübler 12
2.4.1.1 Cálculo de los grados de libertad del robot RoboPhilo 13
2.4.2 Cinemática directa 15
2.4.3 Cinemática inversa 15
2.5 Interfaz natural de usuario (NUI) 19
2.6 Kinect 21
2.6.1 ¿Qué es Kinect? 21
2.6.2 Componentes del sensor Kinect 21
2.6.3 ¿Cómo funciona Kinect? 22
2.7 SDK de Kinect para Windows 24
2.7.1 Requerimientos 25

2.8 Microsoft Visual Studio 26
2.8.1 Visual C# 26
2.8.2 Windows Presentation Foundation en Visual Studio 26
2.9 Robot humanoide RoboPhilo 27
2.9.1 Características del robot RoboPhilo RTW 28
2.9.2 Philo gripper 29
2.9.3 Controlador RoboPhilo 29
2.9.4 Servomotores 30
2.10 WiFi 34
CAPÍTULO 3 36
ANÁLISIS Y ADECUACIÓN DEL ROBOT 36
3.1 USC servo controller 36
3.1.1 Características 36
3.1.2 Montaje del controlador 37
3.2 Generación de secuencias de movimiento 41
3.3 Módulo WiFi HLK-RM04 44
3.3.1 Configuración módulo WiFi HLK-RM04 46
3.4 Análisis cinemático del robot humanoide: RoboPhilo 49
3.4.1 Cinemática inversa 49
CAPÍTULO 4 53
DISEÑO E IMPLEMENTACIÓN DE LA APLICACIÓN DE CONTROL 53
4.1 Procesamiento de señales 53
4.1.1 Detección de esqueleto 53
4.1.2 Reconocimiento de voz 55
4.2 Comunicación serial 57
4.2 Diseño de la aplicación 58
4.2.1 Ventana splash 58
4.2.2 Ventana principal 59
4.2.3 Ventana de ángulos 60
4.3 Diagramas de flujo 61
4.3.1 Diagrama principal 61
4.3.2 Subproceso iniciar ventana principal 63
4.3.3 Subproceso obtención datos Kinect 64
4.3.4 Subproceso control robot 65

4.3.5 Subproceso comunicación 66
CAPÍTULO 5 67
PRUEBAS Y RESULTADOS 67
5.1 Pruebas en modo imitar 68
5.2 Pruebas en modo controlar 72
5.2.1 Medición modo mover 72
5.2.2 Medición modo maniobrar 74
5.2.3 Prueba de desempeño del robot 75
5.3 Análisis financiero 78
5.3.1 Costo de materiales 78
5.3.2 Costo de ingeniería 78
5.3.3 Costo total 78
CONCLUSIONES 79
RECOMENDACIONES 81
LISTA DE REFERENCIAS 82
ANEXOS 84

ÍNDICE DE FIGURAS

Figura 1. Telemanipulador diseñado como una extensión de los brazos del operador 4
Figura 2. Robot Asimo diseñado por la Empresa Honda 8
Figura 3. Robot Pepper diseñado por Aldebaran empresa Francesa 9
Figura 4. Robot Romeo diseñado por Aldebaran 9
Figura 5. Robot DARwIn-OP 10
Figura 6. Robot Nao diseñado por Aldebaran 11
Figura 7. Diagrama relación entre cinemática directa e inversa 12
Figura 8. Brazo derecho grados de libertad y eslabones del robot Robophilo 13
Figura 9. Pierna derecha grados de libertad y eslabones del robot Robophilo 14
Figura 10. Robot articular 3GDL. 17
Figura 11. Elementos 2 y 3 del robot en a) configuración codo arriba, b) configuración codo abajo 18
Figura 12. Tecnología de interfaces naturales de usuario. 20
Figura 13. Sensor Kinect diseñado por Microsoft. 21
Figura 14. Partes del sensor Kinect. 22
Figura 15. Emisor y sensor de IR en funcionamiento. 23
Figura 16. a) Imagen de Profundidad. b) Imagen de Cámara RGB 23
Figura 17. Evolución del SDK de Kinect 25
Figura 18. RoboPhilo 27
Figura 19. Versiones: a) RoboPhilo Kit y b) RoboPhilo RTW 28
Figura 20. RoboPhilo con Pinzas 29
Figura 21. Controlador RoboPhilo con distribución de pines 30
Figura 22. Servo Motor SV3031A 31
Figura 23. Posición del servomotor en función del ancho de pulso 33
Figura 24. USC servo controller de 32 canales versión 2014 36
Figura 25. Diagrama de distribución de servomotores 37
Figura 26. USC-32 en RoboPhilo. 38
Figura 27. TOROBOT RIOS USC 39
Figura 28. Comunicación WiFi HLK-RM04 – PC 45
Figura 29. Diagrama conexión módulo WiFi y controlador de servos. 46
Figura 30. Establecimiento de conexión a traves de la red TOLNK 47
Figura 31. Portal de conexión para el dispositivo WiFi 47

Figura 32. Portal de conexión para el dispositivo WiFi 48
Figura 33. Configuración en menú “Serial Settings” 48
Figura 34. Ángulo brazo derecho de RoboPhilo 50
Figura 35. Ángulo codo y hombro derecho de RoboPhilo 51
Figura 36. Ventana splash de aplicación de control 59
Figura 37. Ventana principal de aplicación de control 60
Figura 38. Ventana de ángulos de aplicación de control 60
Figura 39. Diagrama de flujo principal 61
Figura 40. Diagrama de flujo principal (continuación) 62
Figura 41. Diagrama de flujo de subproceso iniciar ventana principal 63
Figura 42. Diagrama de flujo de subproceso obtención datos Kinect 64
Figura 43. Diagrama de flujo de subproceso control robot 65
Figura 44. Diagrama de flujo de subproceso comunicación 66
Figura 45. Diagrama de funcionamiento del sistema completo 67
Figura 46. Posición para medición del brazo izquierdo 68
Figura 47. Posición para medición del brazo derecho 69
Figura 48. Posición para medición de la pierna izquierda 70
Figura 49. Posición para medición de la pierna derecha 71
Figura 50. Control por voz – trayectoria de prueba 76
Figura 51. Modo maniobrar – traslado del objeto 77

ÍNDICE DE TABLAS

Tabla 1.Desarrollo de los robots 5
Tabla 2. Clasificación de los robots según la AFRI..... 6
Tabla 3. Clasificación de robots de servicio y teleoperados 7
Tabla 4. Modelo de cinemática directa-matriz Denavit Hartenberg y teleoperados.. 16
Tabla 5. Parámetros de un servo motor analógico y digital..... 32
Tabla 6. Características de servomotores..... 33
Tabla 7. Pila de protocolos TCP/IP..... 35
Tabla 8. Valores de ancho de pulso y ángulo correspondiente para RoboPhilo..... 40
Tabla 9. Secuencia de caminado para RoboPhilo 42
Tabla 10. Ángulos brazo izquierdo 69
Tabla 11. Ángulos brazo derecho..... 70
Tabla 12. Ángulos pierna izquierda| 71
Tabla 13. Ángulos pierna derecha..... 72
Tabla 14. Valores de tiempo en modo mover 73
Tabla 15. Valores de confianza en modo mover..... 74
Tabla 16. Valores de confianza en modo maniobrar..... 74
Tabla 17. Valores de confianza en modo maniobrar..... 75
Tabla 18. Listado de costo de materiales 78
Tabla 19. Listado total de costos..... 78

INDICE DE ANEXOS

ANEXO 1. Datasheet módulo WiFi HLK-RM04 84
ANEXO 2. Datasheet plataforma RoboPhilo 87
ANEXO 3. Datasheet servo controller 89

RESUMEN

Este proyecto comprende de un sistema de asistencia para discapacitados y trabajadores, que capture y replique los movimientos de las extremidades del cuerpo humano de tal manera que el autómata sea el medio por el cual la persona pueda desempeñar sus actividades laborales libre de peligro o en el caso de las personas discapacitadas asistiéndoles.

El sistema comprende 2 modos de trabajo los cuales son: modo imitar el cual sirve para replicar los movimientos de las 4 extremidades del cuerpo humano, para este modo es necesario que el robot este colgado en su pedestal. Modo controlar el cual permite el desplazamiento del robot mediante comandos de voz y adicional maniobrar objetos haciendo uso la réplica de movimientos de los brazos.

Para todo este sistema se ha empleado como intérprete del usuario el sensor Kinect que se utiliza en las consolas de video juegos, el cual tiene la capacidad de capturar los movimientos del usuario y su voz, también se ha utilizado el lenguaje de programación C# de Microsoft debido a su compatibilidad y estabilidad con Kinect y en cuanto a hardware se usó el robot RoboPhilo.

ABSTRACT

This program is about an assistance system for disabled people and workers. This will capture and replicate the movements of the human body limbs, the robot is the way through which the person can perform their occupational activities without danger or in the case of the disabled by assisting them.

The system includes two working modes which are: 1) Imitate mode which allows to replicate the movements of the 4 extremities of the human body; for this, the robot is hung on its pedestal. 2) The control mode which allows the movement of the robot by using voice commands. Furthermore, it allows the user to direct the robot for taking specific objects through the replica of arm movements.

As the interpreter of the user, the Kinect sensor has been used. This sensor is used in video game consoles, and has the ability to capture the user's movements and voice, additionally, Programming language named C # development by Microsoft has been used in this project because of both compatibility and stability with Kinect.The platform called RoboPhilo was used in the hardware.

INTRODUCCIÓN

Las personas discapacitadas sufren de limitantes en el día a día, como no poder caminar, tomar objetos y transportarlos, esto representa una problemática grande ya que llegan a sentirse frustrados e inútiles. Así mismo en el ámbito laboral existen trabajos que son peligrosos de realizar poniendo en riesgo a cada minuto a la persona.

La tecnología ha avanzado mucho en los últimos años por lo que ya existen varias soluciones a estos problemas, permitiendo a las personas discapacitadas valerse por sí mismas para realizar actividades cotidianas para que así también puedan integrarse a la sociedad o en el caso de trabajadores evitar que puedan salir lastimados o incluso morir debido al riesgo de su actividad.

Actualmente existen muchos sistemas de iteración con el usuario como es la interfaz natural la cual ha sido pensada para que el usuario no tenga contacto con ningún dispositivo extra para controlar el sistema. Esta tecnología es reciente y es muy usada en videojuegos que han demostrado ser muy interesantes pero a la vez han despertado la creatividad de desarrolladores de todo el mundo, los cuales han creado sistemas no solo de entretenimiento sino también de asistencia, domótica, realidad aumentada, etc.

Conociendo los problemas y teniendo la tecnología se ha pensado en realizar un sistema de control de un robot humanoide mediante un sensor de interfaz natural que permita el control del robot mediante imitación y comandos de voz, para que pueda asistir a una persona; el sensor Kinect fue muy conocido en los últimos años y Microsoft lanzo su kit de desarrollo sin costo alguno por lo cual fue el dispositivo escogido para el desarrollo de la aplicación.

CAPÍTULO 1
PLANTEAMIENTO DEL PROBLEMA

1.1 Planteamiento del problema

En la actualidad en ciertos sectores laborales los factores de riesgo ocupacionales exponen al trabajador a estar en peligro constante, causando daños materiales, lesiones físicas o hasta la muerte.

Registros actuales de la información de la población, permiten conocer que en el país existe una cantidad considerable de personas con discapacidad en sus extremidades inferiores, dicho grupo requiere la asistencia de otra persona, lo que les limita realizar ciertas actividades cotidianas por sí mismas.

En el Ecuador se ha garantizado la inclusión social para personas con discapacidad y el cuidado de la integridad física de trabajadores, por tal motivo se ve la necesidad del uso sistemas de asistencia a personas con el fin de mejorar y proteger la vida de estos grupos sociales.

1.2 Objetivos

1.2.1 Objetivo general

Controlar un Robot Bípedo humanoide mediante una aplicación con sensor Kinect, para realizar actividades de asistencia en un espacio 3D.

1.2.2 Objetivos específicos

- Desarrollar una aplicación en un lenguaje de programación de alto nivel para que el robot replique los movimientos de las extremidades superiores e inferiores del cuerpo humano a través del sensor Kinect.
- Integrar el robot bípedo y la aplicación desarrollada con el sensor Kinect, por medio de un controlador electrónico y comunicación inalámbrica.
- Programar rutinas en el controlador electrónico que permita el desplazamiento del robot en una superficie plana.
- Implementar el reconocimiento de voz en la aplicación desarrollada, para controlar el desplazamiento del robot y los movimientos de las manos que no pueden ser captados por la cámara del sensor Kinect.

1.3 Justificación

Actualmente los grandes avances y desarrollos de la tecnología respecto a los sistemas de interfaz con el usuario permiten generar aplicaciones novedosas como el reconocimiento de voz, de gestos e incluso de los movimientos del cuerpo humano, facilitando en cierto grado la realización de tareas cotidianas como es el caso de personas con discapacidad de movimiento en sus extremidades inferiores o trabajadores que realizan actividades de riesgo.

Este proyecto propone el desarrollo de un sistema de asistencia para discapacitados y trabajadores, que capture y replique los movimientos de las extremidades del cuerpo humano de tal manera que el autómata sea el medio por el cual la persona pueda desempeñar sus actividades laborales libre de peligro o asistiendo a las personas con discapacidad.

El desarrollo de este sistema permitirá generar posibles soluciones a problemas que se presentan en personas discapacitadas con dificultad de movilidad en sus extremidades inferiores y a su vez a trabajadores que realizan actividades de alto riesgo. Este proyecto es la base y podrá ser mejorado agregando más funciones en futuras investigaciones de asistencia para discapacitados o trabajadores en el Ecuador.

1.4 Alcance

Este proyecto tiene como objetivo controlar el autómata RoboPhilo por medio de una aplicación con el sensor Kinect, el usuario podrá hacer uso de funciones como réplica de extremidades principales del cuerpo humano y reconocimiento de voz para ejecutar rutinas pre-programadas y movimientos que el sensor no puede captar fácilmente. Esto permitirá que el usuario sea capaz de manejar al robot para realizar sus actividades libre de peligro.

CAPÍTULO 2
MARCO TEÓRICO

2.1 Origen y desarrollo de la robótica

A lo largo del tiempo, el hombre se ha interesado por máquinas y dispositivos que son capaces de imitar movimientos y acciones de los seres vivos, mucho antes del siglo XIII, existían mecanismos que realizaban movimiento de seres animados, es por eso que los griegos definían a estas máquinas como *automatos,* de esta primera definición proviene la palabra *autómata.*

En el siglo XX, El escritor checo Karel Capek (1890-1938), usó la palabra *robot* en la obra teatral R.U.R (Robots Universales Rossum) por el año de 1920. Su origen es la palabra eslava *robota,* que se refiere al trabajo realizado de forma forzada. (Barrientos, 2007)

El desarrollo proveniente tras los primeros autómatas que en su mayoría tenían aspecto humano, revelan que los precursores de los robots fueron telemanipuladores (Figura 1). Raymond Goertz fue uno de los pioneros en el campo de la robótica. En 1948, mientras trabajaba para la Comisión de Energía Atómica en el Laboratorio Nacional de Argonne, desarrolló el primer telemanipulador. (Barrientos, 2007)

Este consistía en un sistema maestro-esclavo con el fin de manejar material radiactivo, años más tarde, en 1954 Goertz usó tecnología electrónica y de servocontrol, sustituyendo la transmisión mecánica por una eléctrica, desarrollando así un telemanipulador con servo control bilateral.

Telemanipulador

Figura 1. Telemanipulador diseñado como una extensión de los brazos del operador
Fuente: (Telemanipuladores Argentinos para el mundo, 2012)

El robot como máquina lleva un desarrollo independiente del término robot, y aunque el concepto popular establece que este, tiene una apariencia humana y que actúa como tal, es necesario conocer los avances de la robótica desde sus inicios, en la tabla 1 se presenta una cronología sobre los hechos más relevantes del desarrollo de los robots.

Tabla 1.
Desarrollo de los robots

Año	**Autor**	**Robot**
1946	G.C Devol	Controlador magnético que accionaba una máquina mecánica.
1948	R.C. Goertz	Manipulador (Mecánico) de componentes radioactivos
1954	R.C. Goertz	Telemanipulador de servo control bilateral.
1958	Ralph Mosher	Handy-Man, brazos mecánicos Teleoperados
1959	Planet Corporation	Robot controlado por interruptores de fin de carrera.
1960	G.C Devol, J.F. Engelberger	Unimate, Brazo robótico industrial.
1966	Trallfa Industrier AS	Robot de Pintura por pulverización
1982	Makino	SCARA(Selective Compliance Asembly Robot Arm)

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

2.2 Definición y clasificación del robot

Los robots de hoy en día son considerados como medios que permiten realizar tareas repetitivas generando un rendimiento óptimo y brindando seguridad al operador, Sin embargo la mayoría de los expertos en robótica dirían que es complicado ofrecer una definición de robot universalmente aceptada. De hecho, distintas organizaciones y asociaciones internacionales ofrecen definiciones distintas pero que si bien, es cierto, estas son próximas entre sí. La definición de robot que ofrece el Robot Industries Association (RIA), es:

“Manipulador funcional reprogramable, capaz de mover material, piezas, herramientas o dispositivos especializados mediante movimientos variables programados, con el fin de realizar tareas diversas.” (Barrientos, 2007)

A continuación se definirá algunas de las características citadas en la definición anterior:

Multifuncional: versatilidad para llevar a cabo distintas tareas, lo cual involucra una considerable auto-adaptabilidad al entorno.

Programable: capacidad para modificar la tarea mediante el cambio de programa, que favorece su adaptación rápida a diferentes aplicaciones.

En la tabla 2 se puede apreciar la clasificación de los robots según la Asociación Francesa de Robótica Industrial (AFRI), el presente proyecto de titulación está relacionado con la categoría Tipo C.

Tabla 2.
Clasificación de los robots según la AFRI

Tipo A	Manipulador con control manual o telemando
Tipo B	Manipulador automático con ciclos reajustados: se detiene mediante fines de carrera o topes; control por PLC accionamiento neumático eléctrico o hidráulico
Tipo C	Robot programable con trayectoria continua o punto a punto. Carece de conocimiento sobre su entorno
Tipo D	Robot capaz de adquirir datos de su entorno, readaptando su tarea en función de estos.

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

2.2.1 Robots de servicio y teleoperados

Se define como robot de servicio a los dispositivos controlados por un programa de ordenador, estos dispositivos son móviles o estacionarios y cuentan con uno o varios brazos para realizar tareas predefinidas, el robot de servicio no realiza tareas industriales.

En cuanto a los robots Teleoperados la NASA los define como dispositivos robóticos con brazos manipuladores y sensores con cierto grado de movilidad, controlados remotamente por un operador humano de manera directa, o mediante un ordenador, A continuación se muestra la clasificación de robots de servicio y teleoperados.

Tabla 3.
Clasificación de robots de servicio y teleoperados

1	Robots submarinos
2	Robots domésticos
3	Insectoides
4	Funcionoides
5	Humanoides

Nota. Fuente: (Barrientos, 2007)

2.3 Plataformas humanoides

La sustitución del operador por un programa de computadora que controle al manipulador estableció el concepto del robot actual en este documento nos centraremos en los robots humanoides. En el siguiente punto se detallaran ejemplos de los robots humanoides exponiendo sus principales características y funcionalidades.

2.3.1 All-new Asimo (2011)

Honda-Robotics desde 1986 ha introducido en el mercado innovaciones tecnológicas, que apoyan en la vida diaria de las personas, entre estas se destaca el robot ASIMO, que ha pasado de ser una "máquina automática" a una "máquina autónoma" con la capacidad de toma de decisiones para determinar su comportamiento en función del entorno, tales como predecir hacia dónde se dirige una persona y evitar chocarla, El nuevo Asimo tiene una altura de 130 centímetros, pesa 48 kilos (6 menos que el modelo anterior), el grado de libertad de movimiento es de 57 grados (23 grados más que su predecesor) y la velocidad máxima de avance es de 9 km/h.

Robot Asimo

Figura 2. Robot Asimo diseñado por la Empresa Honda
Fuente: (infobae, 2014)

Las tecnologías necesarias que se han desarrollado para lograr la autonomía de Asimo;

a) Estabilidad de alto nivel que permite al robot mantener su postura moviendo rápidamente sus extremidades para recuperar el equilibrio.

b) Reconocimiento externo a partir de múltiples sensores que permite al robot integrar la información, como por ejemplo los movimientos de las personas, y así modificar su ruta rápidamente.

c) Para generar un comportamiento autónomo y determinar la siguiente acción sin ser controlado por un operador, el sistema realiza predicciones a partir de la información recogida del entorno.

2.3.2 Pepper

Creado por Aldebaran, Pepper es el primer robot humanoide diseñado para vivir con seres humanos su principal objetivo es interactuar con los miembros de la familia, pesa alrededor de 28 Kg y mide 120 cm; consta de 17 articulaciones y 3 ruedas omnidireccionales que le permiten moverse libremente y con fluidez.

Pepper es un robot social y esto gracias al uso de nueva tecnología como por ejemplo su increíble cámara 3D para percibir su entorno, su sistema de ultrasonido para evitar obstáculos, también dispone de sensores táctiles para que pueda sentir su tacto y reaccionar en consecuencia.

Robot Pepper

Figura 3. Robot Pepper diseñado por Aldebaran empresa Francesa
Fuente: (Aldebaran, Who is Pepper?)

2.3.3 Proyecto Romeo

Romeo es un robot de investigación en colaboración con laboratorios y empresas como Aldebaran. Iniciado en enero de 2009, el Proyecto ROMEO tiene como objetivos; ser una plataforma de mecatrónica y software interactivo, además de realizar acciones de Robot Asistente, vigilancia y HMI.

Este robot humanoide de 140 cm tiene la intención de proporcionar asistencia a las personas mayores y los que han perdido autonomía. Su altura le permite abrir puertas, subir escaleras, e incluso tomar objetos sobre la mesa.

Robot Romeo

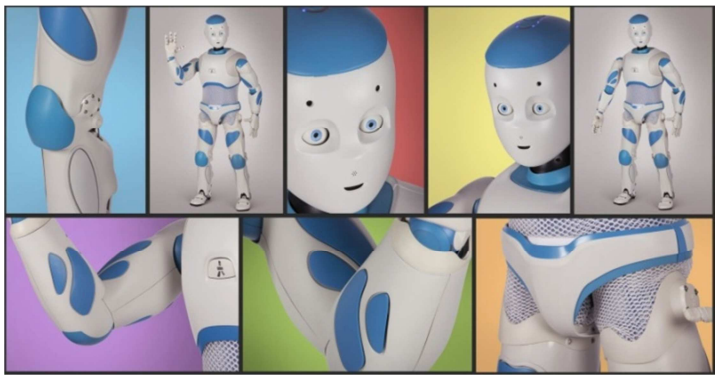


Figura 4. Robot Romeo diseñado por Aldebaran
Fuente: (Aldebaran, Romeo)

2.3.4 DARwIn-OP robot (2012)

ROBOTIS en colaboración con Virginia Tech ha diseñado y desarrollado un robot humanoide para investigación y educación de alto nivel, su significado es Robot dinámico antropomórfico con inteligencia abierta, que abarca campos de humanoide, inteligencia artificial, algoritmo de marcha, la visión, cinemática inversa y la lingüística.

Con tan solo 45 cm y un peso de 2,9 Kg consta de sensores de alta capacidad de carga, actuadores 20MX-28T, con 6 grados de libertad en cada pierna ,3 en ambos brazos y 2 en el cuello, teniendo un total de 20 grados de liberta que le proporciona capacidad de movimiento.

Robot DARwIn-OP

Figura 5. Robot DARwIn-OP
Fuente: (Robotis)

2.3.5 NAO robot

Desde su nacimiento en 2006, NAO ha estado en constante evolución, Aldebaran creó un sistema operativo dedicado llamado Naoqi, que permite que el pequeño humanoide entienda e interprete los datos recibidos por sus sensores. Con 58 cm de altura NAO, permite a los estudiantes aprender a programar de una manera práctica y divertida, pueden programarle para caminar, coger objetos pequeños e incluso bailar.

Cuenta con 25 grados de libertad, para su movimiento, dos cámaras con las que reconoce el entorno, una unidad de medición inercial, lo que le permite saber si está de pie o sentado y 4 micrófonos direccionales para que pueda oír alguna orden.

Robot Nao

Figura 6. Robot Nao diseñado por Aldebaran
Fuente: (Aldebaran, Who is NAO?)

2.4 Cinemática del robot

La cinemática del robot trata con el estudio analítico de la geometría del movimiento de un robot con respecto a un sistema de coordenadas de referencia fijo con una función de tiempo sin considerar las fuerzas/momentos que originan dicho movimiento. La cinemática se interesa por la descripción analítica del movimiento espacial del robot en función del tiempo, en particular las relaciones entre las variables espaciales de tipo articulación y la posición y la orientación del efector final del robot.

Existen dos problemas que resolver en la cinemática del robot:

Problema cinemático directo: determinar la posición y orientación del extremo final del robot, con respecto a un sistema de coordenadas de referencia, conocidos los ángulos de las articulaciones y los parámetros geométricos de los elementos del robot.

Problema cinemático inverso: determinar los ángulos de giro y movimientos de junturas que debe adoptar el robot para una posición y orientación del extremo conocidas.

Diagrama cinemática directa e inversa.

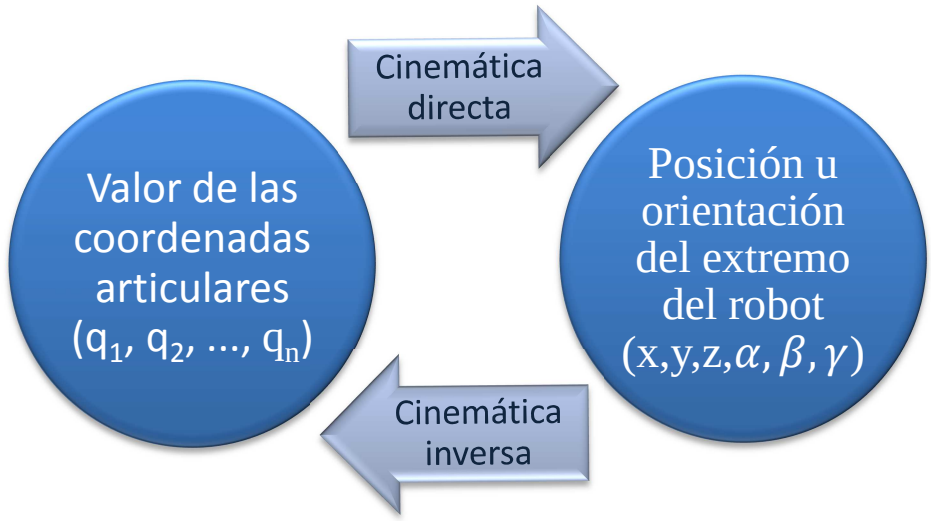


Figura 7. Diagrama relación entre cinemática directa e inversa
Fuente: (Barrientos, 2007)

2.4.1. Grados de libertad criterio de Kutzbach-Grübler

Los grados de libertad (GDL), se refieren al número de variables independientes que determinan la posición o movimiento de un sistema, el número de grados de libertad se determina mediante la fórmula de Kutzbach-Grübler.

- Para el cálculo de los grados de libertad en sistemas planos se utiliza la siguiente expresión:

$$GDL = 3(n-1) - 2j_1 j_2 - j_2 \qquad [1]$$

Dónde:

GDL = Grados de libertad totales del mecanismo

n = Número de eslabones, incluyendo el eslabón fijo o de referencia

j_1= Número de pares inferiores

j_2= Número de pares superiores

- Para el cálculo de grados de libertad en sistemas tridimensionales se utiliza:

$$GDL = 6(n-1) - 5j_1 - 4j_2 - 3j_3 - 2j_4 - j_5 \qquad [2]$$

Dónde:

GDL = Grados de libertad totales del mecanismo

n = Número de eslabones, incluyendo el eslabón fijo o de referencia

j_i= Número de pares o uniones de clase "i" (i grados de libertad de las junturas)

2.4.1.1 Cálculo de los grados de libertad del robot RoboPhilo

Grados de libertad y eslabones del RoboPhilo

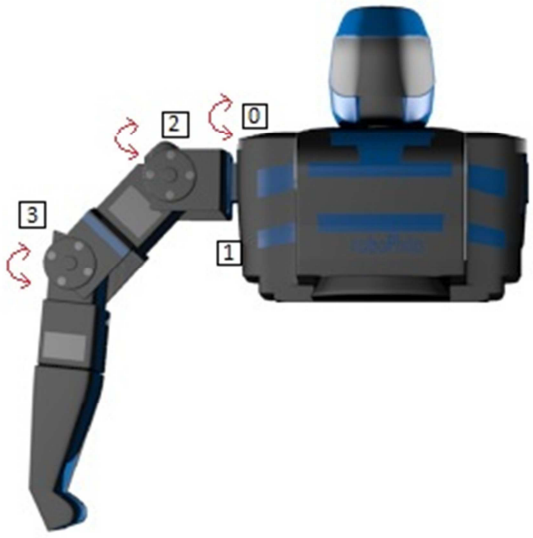


Figura 8. Brazo derecho grados de libertad y eslabones del robot Robophilo
Elaborado por: Vladimir Araujo y Myriam Tipán

Para el cálculo de los grados de libertad para el brazo derecho se determinó el número de eslabones, incluyendo como elemento fijo el tronco del robot, como se observa en la figura 8 se tiene 4 eslabones y 3 movimientos de 1 grado de libertad.

Cálculo:

$$GDL = 6(n-1) - 5j_1 - 4j_2 - 3j_3 - 2j_4 - j_5 \qquad [2]$$

$$n = 4\,, j_1 = 3$$

$$GDL = 6(4-1) - 5(3)$$

$$GDL = 3$$

El sistema del brazo derecho consta de 3 grados le libertad, el mismo resultado da el análisis para el sistema del brazo izquierdo.

Sumando los grados de libertad del brazo derecho e izquierdo da un total de 6 grados de libertad.

Grados de libertad y eslabones del RoboPhilo

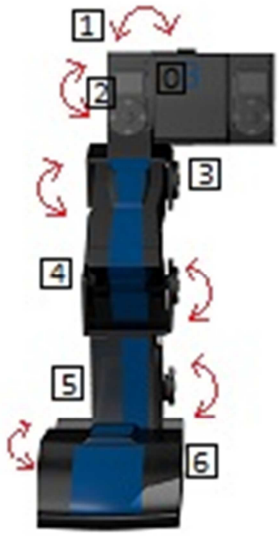


Figura 9. Pierna derecha grados de libertad y eslabones del robot Robophilo
Elaborado por: Vladimir Araujo y Myriam Tipán

Para el cálculo de los grados de libertad para la pierna derecha se determinó el número de eslabones, incluyendo como elemento fijo el tronco del robot, como se observa en la figura 9 se tiene 7 eslabones y 6 movimientos de 1 grado de libertad.

Cálculo:

$$GDL = 6(n-1) - 5j_1 - 4j_2 - 3j_3 - 2j_4 - j_5 \qquad [2]$$

$$n = 7\,, j_1 = 6$$

$$GDL = 6(7-1) - 5(6)$$

$$GDL = 6$$

El sistema de la pierna derecha consta de 6 grados le libertad, el mismo resultado da el análisis para el sistema del brazo izquierdo.

Sumando los grados de libertad de la pierna derecha e izquierda da un total de 12 grados de libertad.

La estructura del RoboPhilo consta adicionalmente de 2 tenazas, con 2 grados de libertad cada uno, teniendo como resultado 4 grados de libertad, además de un servomotor en la cabeza y cintura, de 1 grado de libertad cada uno, por lo tanto:

$$GDL\ del\ RoboPhilo\ = 24$$

2.4.2 Cinemática directa

Se usa algebra vectorial y matricial para describir y representar la localización de los elementos de un robot en un espacio tridimensional con respecto a un sistema de referencia fijo. Como los elementos de un robot son objetos rígidos o eslabones unidos entre sí por articulaciones, se puede establecer un sistema de referencia fijo situado en la base del robot y describir cada uno de los elementos con respecto a dicho sistema de referencia.
A continuación se enlistan las soluciones para la cinemática directa:

- Resolución mediante relaciones geométricas.
- Resolución mediante matrices de transformación homogénea.
- Resolución mediante el uso de cuaternio.

2.4.3 Cinemática inversa

Básicamente el objetivo del problema cinemático inverso consiste en encontrar los valores que deben adoptar las coordenadas articulares del robot $q = [q1, q2, \dots, qn]^T$ para que su extremo se posicione y oriente según una determinada localización espacial.

El problema cinemático inverso se puede resolver por diversos métodos tales como:

- Transformación inversa.
- Algebra de tornillo.
- Matrices duales
- Cuaterniones duales
- Iterativo
- Métodos geométricos.

Este documento se centra en la resolución por métodos geométricos.

2.4.3.1 Resolución del problema cinemático inverso a partir de la matriz de transformación homogénea

Para el uso de este método de debe conocer el modelo de cinemática directa, conociendo las relaciones del valor de posición y orientación del extremo del robot en función de sus coordenadas articulares, obtener por manipulación de aquellas las relaciones inversas. Sin embargo, en la práctica este método no es muy usado ya que en muchas ocasiones es tan complejo que obliga a descartarlo.

Mediante este método se puede calcular las coordenadas del punto P, del robot de la figura 10 para el sistema de coordenadas se establece sobre las siguientes reglas.

- $\boldsymbol{\theta_i}$: es el ángulo de la articulación desde el eje $\mathbf{x_{i-1}}$ hasta el eje $\mathbf{x_i}$, medido respecto del eje $\mathbf{z_i\text{-}1}$, usando la regla de la mano derecha.

- $\mathbf{d_i}$: es la distancia medida desde el origen del sistema **i-1**, a lo largo del eje $\mathbf{z_i\text{-}1}$ hasta la intersección del eje $\mathbf{z_i\text{-}1}$ con el eje $\mathbf{x_i}$.

- $\mathbf{a_i}$: es la distancia de separación entre los orígenes de los sistemas de referencia **i-1** e **i**, medida a lo largo del eje xi hasta la intersección con el eje $\mathbf{z_i\text{-}1}$. (o la distancia más corta entre los ejes $\mathbf{z_i\text{-}1}$ y $\mathbf{z_i}$, cuando estos no se interceptan)

- $\boldsymbol{\alpha i}$: es el ángulo que separa los ejes $\mathbf{z_i}$ y $\mathbf{z_i\text{-}1}$, medido respecto del eje $\mathbf{x_i}$.

Tabla 4.
Modelo de cinemática directa-matriz Denavit Hartenberg

PARAMETROS D-H				
	ɵ	**D**	**A**	∝
1	θ_1	d_1	0	0
2	θ_2	0	a_1	0
3	0	0	a_2	0

Nota. Predicción de los parámetros D-H para el punto P, de la figura 10.
Elaborado por: Vladimir Araujo y Myriam Tipán

2.4.3.2 Resolución del problema cinemático inverso por métodos geométricos

Este método es muy adecuado para robots de pocos grados de libertad o para el caso donde solo se consideren los primeros grados de libertad para posicionar el extremo.
Este proceso básicamente se basa en encontrar suficiente número de relaciones geométricas en las que intervendrán las coordenadas del extremo del robot, sus coordenadas articulares y las dimensiones físicas.
A continuación se muestra el procedimiento a seguir para la resolución del problema cinemático inverso de un robot de 3GDL de rotación (estructura típica articular).

Robot articular 3GDL.

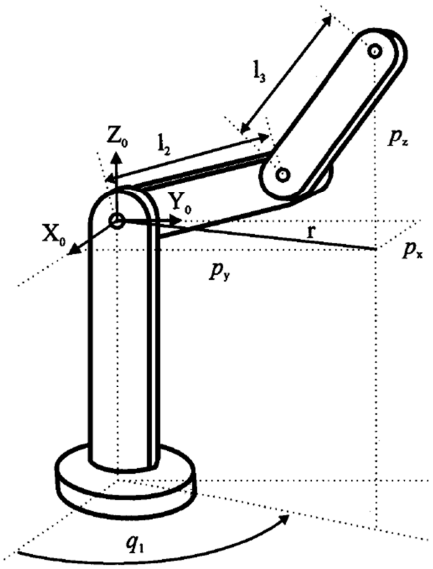


Figura 10. Robot articular 3GDL.
Fuente: (Barrientos, 2007)

La Figura 10 muestra la configuración del robot, los datos iniciales son las coordenadas $(\rho x, \rho y, \rho z)$ referenciadas a XYZ=0.

Este robot tiene una estructura planar, quedando el plano definido por el ángulo de la primera variable articular q1, el cual se obtiene mediante:

$$q1 = \tan^{-1}\frac{\rho y}{\rho x} \qquad [\,3\,]$$

Ahora tomando en cuenta solamente los elementos 2 y 3 que se ven en la Figura 11, y usando el teorema de coseno, se obtiene:

$$\left.\begin{array}{c} r^2 = \rho x^2 + \rho y^2 \\ r^2 + \rho z^2 = l2^2 + l3^2 + 2 * l2 * l3 * \cos q3 \end{array}\right\} \rightarrow$$

$$\cos q3 = \frac{\rho x^2 + \rho y^2 + \rho z^2 - l2^2 - l3^2}{2 * l2 * l3} \qquad [\ 4\ ]$$

Con la expresión obtenida permite calcular $q3$ en función del vector de posición del extremo ρ. Sin embargo es mejor usar la expresión arcotangente en lugar de arcoseno por motivos de ventajas computacionales.

Se conoce la ecuación trigonométrica:

$$\sin q3 = \pm\sqrt{1 - (\cos q3)^2} \qquad [\ 5\ ]$$

Entonces se tendrá

$$q3 = \tan^{-1}\left(\frac{\pm\sqrt{1 - (\cos q3)^2}}{\cos q3}\right) \qquad [\ 6\ ]$$

Es evidente que existen 2 posibles soluciones para $q3$ según sea el signo tomado en la raíz. Estas corresponden a las configuraciones de codo arriba (Figura 11-a) y codo abajo (Figura 11-b) del robot.

Codo Robot articular 3GDL.

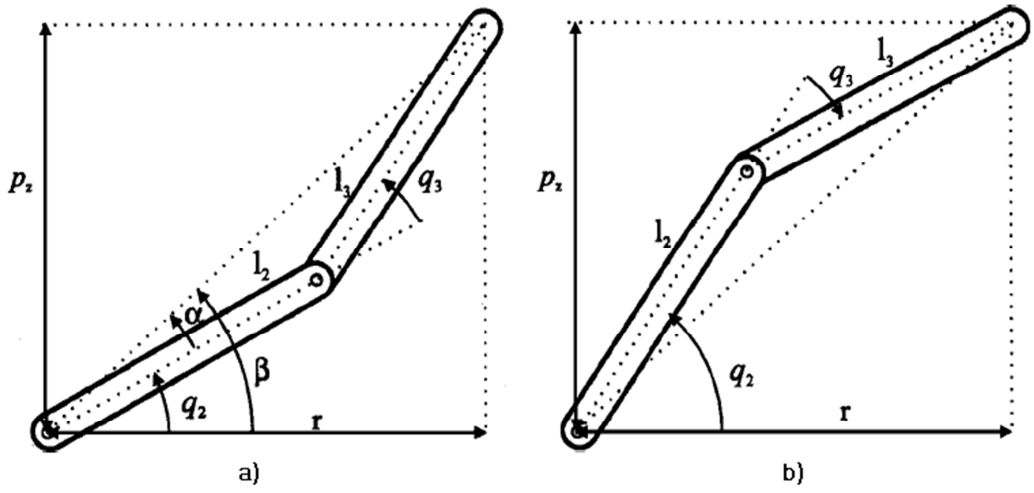


Figura 11. Elementos 2 y 3 del robot en a) configuración codo arriba, b) configuración codo abajo
Fuente: (Barrientos, 2007)

Para calcular $q2$ se obtiene la diferencia entre β y $\propto$:

$$q2 = \beta - \alpha \qquad [7]$$

Donde

$$\beta = \tan^{-1}\left(\frac{\rho z}{r}\right) = \tan^{-1}\left(\frac{\rho z}{\pm\sqrt{\rho x^2 + \rho y^2}}\right) \qquad [8]$$

$$\alpha = \tan^{-1}\left(\frac{l3 * \sin q3}{l2 + l3 * \cos q3}\right) \qquad [9]$$

Entonces la expresión final es

$$q2 = \tan^{-1}\left(\frac{\rho z}{\pm\sqrt{\rho x^2 + \rho y^2}}\right) - \tan^{-1}\left(\frac{l3 * \sin q3}{l2 + l3 * \cos q3}\right) \qquad [10]$$

Otra vez en esta expresión hay 2 posibles soluciones para $q2$ según el signo de la raíz que también corresponden a las configuraciones de codo arriba y abajo.

Con las expresiones [3], [6] y [10] halladas se resuelve el problema cinemático inverso para un robot de 3GDL. Hay que tomar en cuenta que este método no es sistemático y por lo tanto depende de la configuración del robot.

2.5 Interfaz natural de usuario (NUI)

La forma en la que interactuamos con las computadoras ha cambiado con el transcurso del tiempo comenzando desde las tarjetas perforadas hasta llegar a potentes sensores de la actualidad, que permiten la interacción sin necesidad de dispositivos externos con las que se realiza la comunicación.

Interfaces naturales de usuario

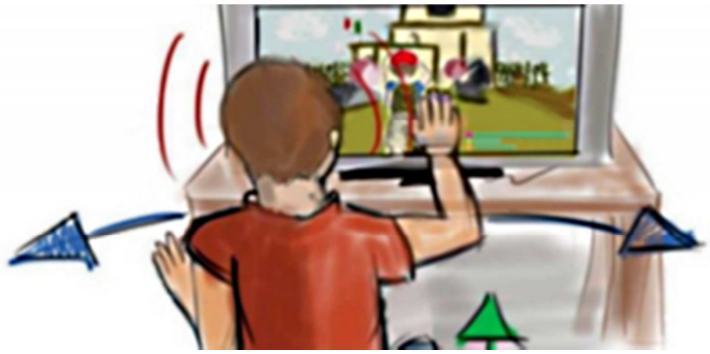


Figura 12. Tecnología de interfaces naturales de usuario.
Fuente: (Santana, 2014)

A este tipo de interacción se la conoce como interfaz natural de usuario (NUI, por sus siglas en inglés), la meta de esta es que el usuario tenga una experiencia de uso natural. Un NUI es activado con el tacto, los gestos, por sonido, por los sentidos en general.

Para que una interfaz sea considerada natural, debe cumplir con las siguientes consideraciones:

- Crear una experiencia, que dé la sensación de ser una extensión del cuerpo.
- Crear una experiencia, que le sea natural tanto a usuarios expertos, como a los usuarios nuevos.
- Crear una experiencia, que sea auténtica al medio.
- Crear una interfaz de usuario, que considere el contexto, tomando en cuenta las correctas metáforas, indicaciones visuales, realimentación y métodos de entrada y salida.
- Evitar caer en la trampa, de copiar los paradigmas de interfaz de usuario existentes.

La tecnología ha avanzado y la fusión de varias áreas permitió el desarrollo de varias tecnologías aplicables para las NUI. A continuación se muestran varios dispositivos con concepto NUI:

- Kinect
- Gafas interactivas de Google
- Leap Motion

2.6 Kinect

2.6.1 ¿Qué es Kinect?

El sensor Kinect mostrado en la Figura 13, fue anunciado bajo el nombre de Proyecto Natal el 1 de Junio del 2009 en el E3 2009. Al siguiente año fue lanzado el 4 de Noviembre y logro 8 millones de unidades vendidas en los primeros 60 días, entrando a los Guinness World Records como “el dispositivo de electrónica de consumo de más rápida venta en la historia”.

El sensor Kinect fue el primer dispositivo comercial para interactuar con una consola a través de Interfaz Natural de Usuario, usando movimientos del cuerpo, gestos y habla en vez de un mando. Suena a ciencia ficción pero ya es realidad, ahora el usuario reemplaza el mando del videojuego.

Sensor Kinect

Figura 13. Sensor Kinect diseñado por Microsoft.
Fuente: (Abrego, 2014)

2.6.2 Componentes del sensor Kinect

El Sensor Kinect es un dispositivo horizontal es como una cámara 3D, captura una secuencia de pixeles a color con datos de la profundidad de cada pixel. Además contiene un conjunto de micrófonos que permiten capturar sonido posicionado. Kinect tiene una sorprendente capacidad para obtener información 3D.

Desde el punto de vista de electrónica, Kinect tiene los siguientes componentes (Figura 14):

- Cámara de color
- Emisor infrarrojo (IR)
- Sensor de profundidad (IR)

- Motor de inclinación
- Conjunto de micrófonos
- LED indicador

Partes del Sensor Kinect

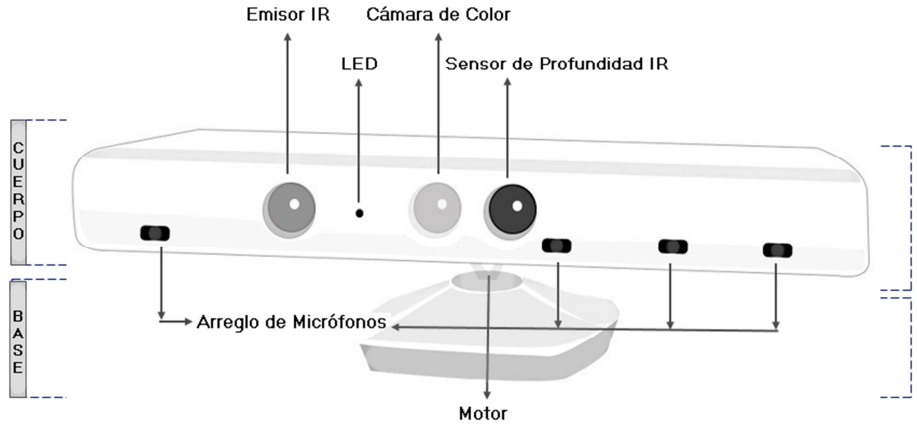


Figura 14. Partes del sensor Kinect.
Fuente: (Jana, 2012)
Modificado por: Vladimir Araujo

El sensor Kinect se comunica mediante puerto estándar USB 2.0, pero necesita una fuente de energía externa debido a que el puerto USB de un computador no provee la suficiente energía para el correcto funcionamiento.

2.6.3 ¿Cómo funciona Kinect?

Como se explicó anteriormente el sensor Kinect consta de un Emisor IR. La luz infrarroja tiene una longitud de onda más larga que de la luz visible por lo cual no se la puede ver a simple vista, el infrarrojo es inofensivo es más el ser humano está expuesto todos los días a esta, en forma de luz solar.

Emisor y sensor de IR en funcionamiento

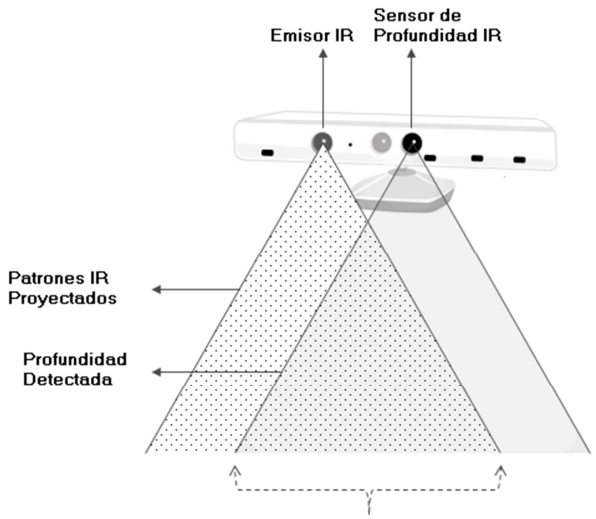


Figura 15. Emisor y sensor de IR en funcionamiento.
Fuente: (Jana, 2012)
Modificado por: Vladimir Araujo

El Proyector IR emite una cuadricula de puntos que son capturados por el Sensor de profundidad para generar una imagen de profundidad como la que se ve en la Figura 16-a.

Imágenes de la cámaras del Sensor Kinect

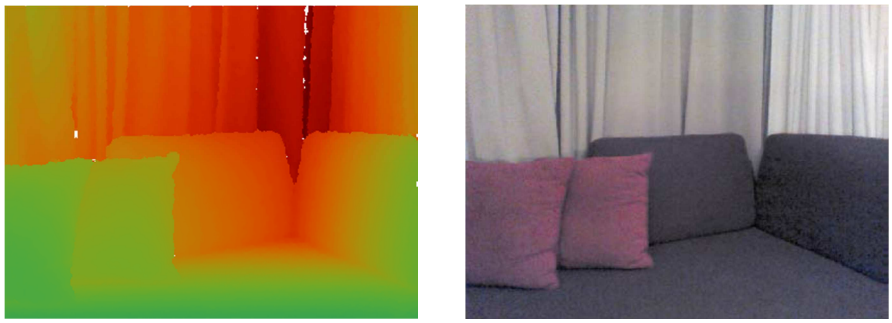


Figura 16. a) Imagen de Profundidad. b) Imagen de Cámara RGB
Fuente: (Jana, 2012)

El sensor Kinect también tiene una cámara RGB la cual es similar a una cámara web con una resolución de 640x480 pixels que obtiene una imagen como la Figura 16-b, la cámara en si no tiene nada especial pero al estar a la misma distancia del sensor de IR se puede alinear con la información de profundidad pudiendo desarrollar varias aplicaciones como por ejemplo esconder la imagen que este a cierta distancia.

Finalmente Kinect tiene 4 micrófonos que se distribuyen a lo largo del sensor, el arreglo de micrófonos no es solo para capturar sonido sino también localizar el

sonido dentro de una habitación. Por ejemplo el sensor será capaz de saber de dónde proviene el comando de voz.

Todas las señales que recepta Kinect son ingresadas en un Procesador Digital de Señales (DSP) el cual adecua las señales que posteriormente son enviadas por USB al computador para ser procesadas.

2.7 SDK de Kinect para Windows

El 16 de Junio del 2011 Microsoft anunció la liberación del Kit de Desarrollo de Kinect para Windows (SDK, por sus siglas en inglés). El cual es un conjunto de librerías que permiten programar en diferentes tipos de plataformas de Microsoft usando el sensor Kinect como entrada. Con estas librerías se pueden desarrollar aplicaciones WPF, aplicaciones WinForms e incluso aplicaciones XNA lo que permite generar juegos y cualquier otra idea del usuario.

El SDK de Kinect hace uso de la capacidad de reconocimiento de habla de un sistema operativo de Windows para interactuar con el sistema de audio de Kinect. Pero para esto se necesita Microsoft Speech Platform – Server Runtime, el Microsoft Speech Platform SDK, y paquete de lenguaje instalado en conjunto con el SDK de Kinect.

En los últimos años el SDK de Kinect ha sido mejorado desde la versión Beta 1 que no fue comercializada hasta la v1.8 que es la última versión comercial para el Kinect ahora conocido como Kinect 1, este documento hace uso de Kinect for Windows SDK v1.7.

El 21 de Mayo de 2013 Microsoft anunció su próxima consola Xbox One, y el Kinect 2.0 que incluirá detección de 6 personas y 1080 píxeles de resolución por consecuencia el 2014 Microsoft lanzo una nueva y mejorada SDK destinada para Kinect 2 con la cual se obtiene mayor precisión y capacidad de respuesta para el desarrollo de aplicaciones interactivas de nueva generación.

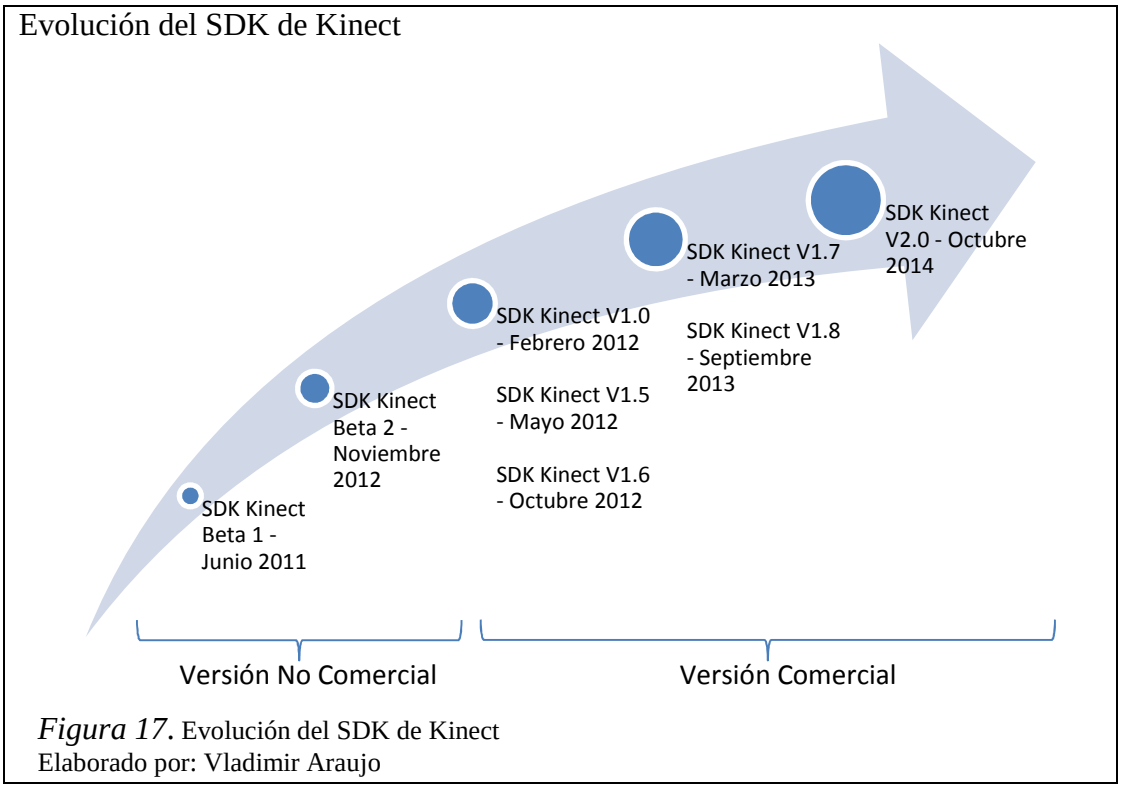


Figura 17. Evolución del SDK de Kinect
Elaborado por: Vladimir Araujo

2.7.1 Requerimientos

Como ya es de conocimiento Kinect es de Microsoft por lo cual el SDK de Kinect es funcional solamente en Windows ya sea x64 o x86, adicional al ser un dispositivo nativo de Xbox 360 el computador deberá tener una gran capacidad de procesamiento

A continuación se enlistan los requerimientos de hardware:

- Computador con Dual-Core a 2.66 GHz o superior
- 2 GB de RAM o superior
- Windows 7 o superior
- Adaptador de Poder para Kinect
- Sensor Kinect

Por el lado de software el SDK de Kinect necesita:

- Microsoft Visual Studio 2010 o superior
- Microsoft .NET Framework 4
- DirectX 9.0 (Opcional)

2.8 Microsoft Visual Studio

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas operativos Windows. Soporta múltiples lenguajes de programación tales como C#, C++, Basic, F# y más.

Visual Studio permite a los desarrolladores crear aplicaciones, sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET (a partir de la versión .NET 2002). Así se pueden crear aplicaciones que se comuniquen entre estaciones de trabajo, páginas web, dispositivos móviles, dispositivos embebidos, consolas, etc.

2.8.1 Visual C#

Microsoft Visual C# es un potente lenguaje orientado a objetos que juega un importante rol en la arquitectura de Microsoft .NET Framework, muchos programadores comparan este rol con el de C en el desarrollo de UNIX.

C# tiene un parecido de sintaxis con respecto a C, C++ o Java, así que si se ha programado antes en alguno de esos lenguajes C# será muy familiar.

El nombre C Sharp fue inspirado por la notación musical, donde '#' (sostenido, en inglés sharp) indica que la nota (C es la nota do en inglés) es un semitono más alta, sugiriendo que C# es superior a C/C++. Además, el signo '#' se compone de cuatro signos '+' pegados.

2.8.2 Windows Presentation Foundation en Visual Studio

Windows Presentation Foundation (WPF, por sus siglas en inglés) es una tecnología de Microsoft, presentada como parte de Windows Vista. Permite el desarrollo de interfaces de interacción en Windows tomando características de aplicaciones Windows y de aplicaciones web.

WPF ofrece una amplia infraestructura y potencia gráfica con la que es posible desarrollar aplicaciones visualmente atractivas, con facilidades de interacción que incluyen animación, vídeo, audio, documentos, navegación o gráficos 3D. Separa, con el lenguaje declarativo XAML y los lenguajes de programación de .NET, la

interfaz de interacción de la lógica del negocio, propiciando una arquitectura Modelo Vista Controlador para el desarrollo de las aplicaciones.

Microsoft Silverlight es un subconjunto de WPF que permite crear aplicaciones similares a Flash para web y móviles, con el mismo modelo de programación que las aplicaciones .NET. No admite funciones 3D, pero soporta XPS y planos basados en vectores.

2.9 Robot humanoide RoboPhilo

El RoboPhilo es un robot humanoide de bajo coste diseñado en Estados Unidos por Robobrothers, Inc. y World Models Manufacturing Ltd. en 2007. PHILO es un acrónimo de "Programmable Humanoid In Lifelike Operation".

RoboPhilo tiene 20 motores que permiten el movimiento de cabeza, brazos, cintura, muslos y piernas, lo que permite variedad de movimientos.

Robot RoboPhilo

Figura 18. RoboPhilo
Fuente: (Royer, 2011)

Este robot es una plataforma fácil de usar y muy flexible. RoboPhilo fue diseñado para el uso de modelistas y aficionados no necesariamente conocedores de electrónica o robótica.

Actualmente se comercializan 2 versiones del robot:

- RoboPhilo Kit (Figura 19-a) es la versión de menor precio la cual incluye todos los materiales necesarios para su ensamblaje. Esta versión requiere de configuración de los motores mediante el software del fabricante.
- RoboPhilo RTW (Figura 19-b) es la versión lista para caminar (RTW, por sus siglas en inglés), con esta versión es solo necesario poner la batería y comenzar a usarlo pero su costo es mayor debido a que ya viene armado y configurado totalmente.

Versiones de RoboPhilo

Figura 19. Versiones: a) RoboPhilo Kit y b) RoboPhilo RTW
Elaborado por: Vladimir Araujo y Myriam Tipán

La versión usada en este proyecto es RoboPhilo RTW.

2.9.1 Características del robot RoboPhilo RTW

- Altura: 13 pulgadas (330.2 mm)
- Peso: 1.2 Kg (1200 g) con batería
- Controlador basado en Microcontrolador ATmega32-16PU
- Control Remoto 36 KHz
- Receptor 36 KHz
- Batería 6V Ni-MH
- Cargador 7.2V 1000mA

2.9.2 Philo gripper

Philo Gripper es un accesorio relativamente nuevo ofertado por el equipo de RoboBrothers, básicamente son pinzas que constan de 2 micro servomotores que permiten al RoboPhilo la manipulación de objetos.

RoboPhilo con Pinzas

Figura 20. RoboPhilo con Pinzas
Fuente: (RoboBrothers, RoboPhilo Gripper Manual, 2012)

Estas pinzas son vendidas por separado y deben ser instaladas y configuradas por el usuario. Como se mencionó anteriormente cada pinza tiene 2 motores dando un total de 4 por dos pinzas que son conectadas directamente a los 4 puertos restantes de controlador.

2.9.3 Controlador RoboPhilo

El controlador de RobotPhilo esta potenciado por un Microcontrolador AVR ATmega32-16PU el cual viene pre programado de fábrica.

El Firmware original permite el manejo del robot mediante un control remoto infrarrojo el cual interrumpe el núcleo, adicional 24 canales para el control de servomotores y hasta 8 puertos de entrada o salida para agregar hardware externo, y 8 Kb de flash para almacenar hasta 300 movimientos de usuario. Actualmente existe una actualización de Firmware la cual permite el control del robot mediante un control de Play Station en reemplazo del control remoto IR, dando mayor fiabilidad de control.

El controlador es fácilmente programable desde un PC a través de RS-232 y el software PHILO Motion Creator el cual permite modificar movimientos de fábrica y/o crear movimientos personalizados.

Para el movimiento de los servomotores el controlador genera ondas PWM, pero el AVR ATmega32 tiene únicamente 4 puertos que pueden generar PWM por hardware para los 20 puertos restantes se usa la generación de PWM por software. La técnica de generación de PWM por software presenta problemas como inestabilidad y ruido en los servomotores que se los conoce como Jitter esto debido a que la onda PWM se ve afectada por la variación de fuente o vibración mecánica en los motores.

Controlador con distribución de pines

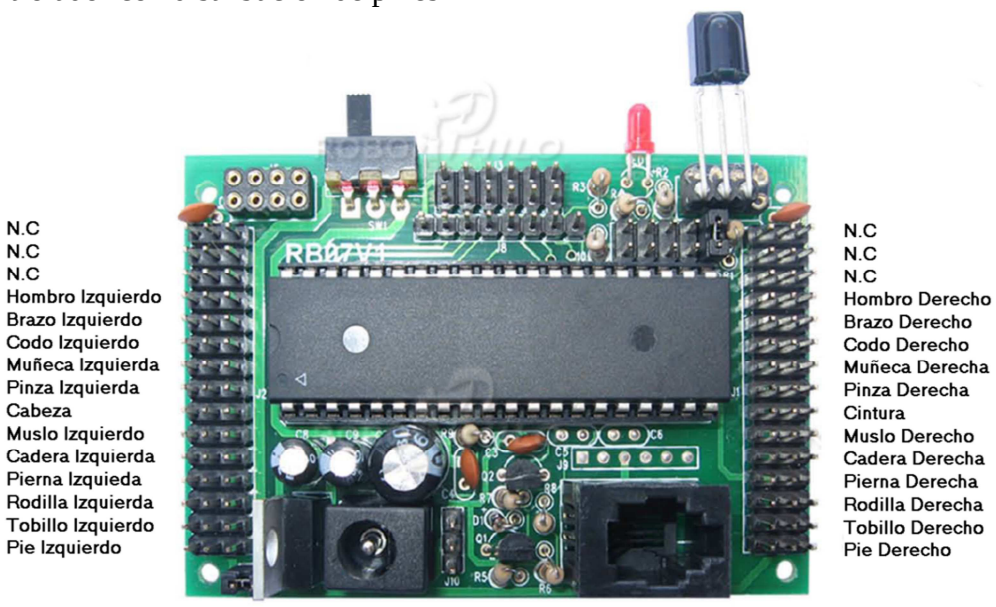


Figura 21. Controlador RoboPhilo con distribución de pines
Fuente: (RoboBrothers, RoboPhilo User Guide, 2007)
Elaborado por: Vladimir Araujo y Myriam Tipán

2.9.4 Servomotores

Un servo es un potente dispositivo cuyo eje es controlado, ya que en su interior dispone de un pequeño motor con reductor de velocidad y multiplicador de fuerza, también dispone de un circuito que controla el sistema. El eje puede ser llevado a posiciones angulares específicas, siempre que una señal codificada esté presente en la línea de entrada. El ángulo de giro del eje es de 180° en la mayoría de los casos. En la práctica el servo motor es el encargado de dar movilidad al robot y su forma física se puede apreciar en la figura 22.

Servo Motor

Figura 22. Servo Motor SV3031A
Fuente: (RoboBrothers, RoboPhilo User Guide, 2007)

Los servomotores se clasifican en servomotores analógicos también llamados estándar y los digitales a continuación se detallara su principal diferencia además se presentaran sus principales características.

Un servomotor digital incorpora un cristal de cuarzo y un microprocesador el cual analiza la señal que proviene del receptor a la vez que se encarga del control de funcionamiento del servomotor. En cuanto a hardware los servomotores analógicos son exactamente iguales a los servomotores digitales ya que incluso ambos poseen un potenciómetro para la realimentación de posición.

La principal diferencia entre ambos tipos de servomotores tiene lugar en la manera de procesar la señal recibida por el receptor, y en como controla el envío de potencia al servomotor de posicionamiento consiguiendo reducir la banda muerta, incrementando la resolución y generando unos valores de par estático y dinámico mucho más elevados.

2.9.4.1 Parámetros importantes de los servomotores

- Velocidad de posicionamiento

Al variar el ancho de pulso generado en la electrónica de potencia, varía proporcionalmente el ciclo de trabajo y por ende aumenta la velocidad.

El ciclo de trabajo es el resultado de dividir el tiempo durante el cual aplicamos tensión al motor entre el tiempo total del ciclo, es tomado en porcentaje [%] al multiplicarlo por 100

- Error de Posicionamiento

Para adquirir el error de posición es necesario comparar el ancho de pulso que define la posición deseada, con el ancho de pulso de la posición actual. A mayor error en la posición mayor ciclo de trabajo y por tanto mayor velocidad.

- Banda Muerta

Es el recorrido mínimo del eje para observar desplazamiento, ya que la tensión aplicada durante un lapso breve de tiempo no será capaz de vencer la fuerza contra electromotriz generando así un desplazamiento nulo.

- Resolución

Es la mínima variación de posición alcanzable por el servo, en donde la precisión del potenciómetro de realimentación de posición y la frecuencia de trabajo son factores primordiales.

En la tabla 5 Se puede observar la comparación de los principales parámetros que intervienen en un Servomotor analógico y digital.

Tabla 5.
Parámetros de un servo motor analógico y digital

Tipo de Servomotor	Velocidad de Posicionamiento	Error de Posicionamiento	Resolución de posicionamiento	Banda muerta
Analógico	Proporcional	Alto	Baja	Alta
Digital	Controlable	Controlable	Alta	Baja

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

2.9.4.2 Funcionamiento de un servomotor

Para controlar un servo se debe aplicar pulsos continuos a una frecuencia específica, los servos tienen 3 cables, dos para alimentación Vcc y Gnd, el tercero es usado para aplicar en el circuito de control un tren de pulsos, y de esta manera el eje llegará a la posición indicada. En la figura 23 se puede observar diferentes posicionamientos del eje del servo motor dependiendo del ancho de pulso, donde las posiciones de 0°, 90° y 180° son logradas con anchos de pulsos de 0.5, 1.5, y 2.5 [ms] respectivamente.

Posiciones del Servomotor

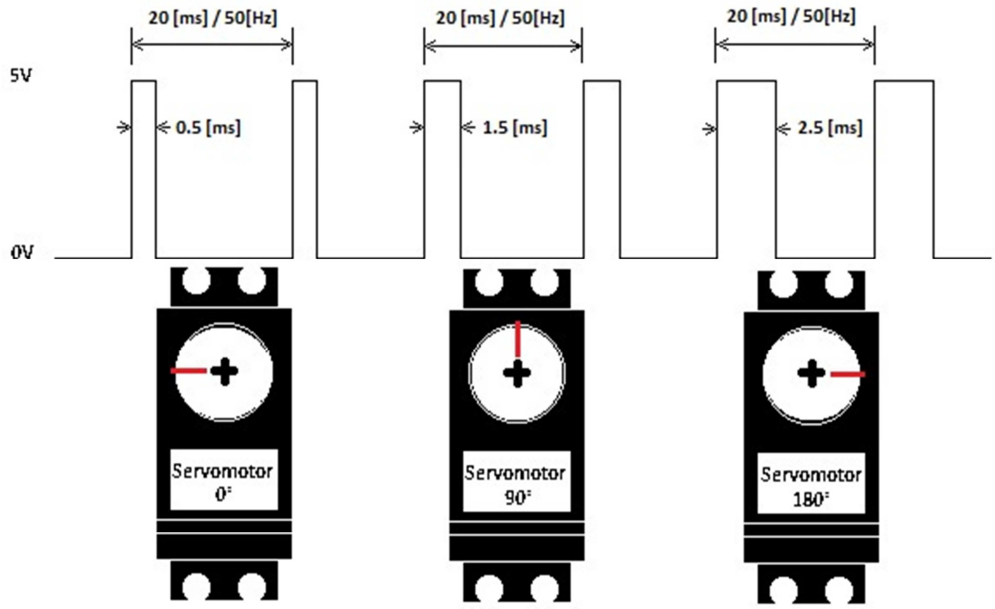


Figura 23. Posición del servomotor en función del ancho de pulso
Elaborado por : Vladimir Araujo y Myriam Tipán

Los servo motores usados por el robot RoboPhilo son diseñados y fabricados por World Models Manufacturing Ltd, Las características de los diferentes modelos usados se detallan en la tabla 5, tomando en cuenta que la frecuencia de estos es variable.

Tabla 6.
Características de servomotores

Modelo	Torque	Velocidad	Engranaje
High Torque Servo SV4140	6.5kg-cm	0.16 seg/60°	Plástico
Servo SV4032-A	4.1kg-cm	0.19 seg/60°	Plástico
Mini Servo SV2031-A	1.3kg-cm	0.12 seg/60°	Plástico

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

2.10 WiFi

Tecnología inalámbrica usada para establecer comunicación e intercambiar información entre dispositivos electrónicos sin utilizar cables físicos, mediante la emisión y recepción de ondas electromagnéticas que corresponden a una banda específica del espectro electromagnético, Wifi utiliza porciones del espectro que no requieren licencia dentro de las bandas ISM Y UNII.

Las bandas ISM (Industrial scientific and medical) permiten el uso de las porciones 2.4-2.5 GHz, 5.8 GHz, y muchas otras frecuencias (no utilizadas enWiFi).

Las bandas UNII (Unlicensed National Information Infrastructure) permiten el uso sin licencia de otras porciones del espectro de 5 GHz.

WiFi usa protocolos que establecen un conjunto de reglas y normas que permiten que dos o más sistemas se comuniquen entre ellos para transmitir información por medio de cualquier tipo de variación de magnitud física, en este caso la propagación de ondas electromagnéticas, la familia de protocolos que establecen la bace de WiFi es 802.11.

- 802.11a permite hasta 54 Mbps en las bandas no licenciada a 5 GHz
- 802.11b permite hasta 11 Mbps en las bandas no licenciada a 2.4 GHz
- 802.11g permite hasta 54 Mbps en la banda no licenciada a 2.4 GHz.
- 802.11n permite hasta 600 Mbps en las bandas no licenciadas a 2.4 GHz y 5 GHz.

El término Wi-Fi (wireless fidelity o fidelidad sin cables) es usado para distinguir todas las soluciones informáticas que utilizan el protcolo 802.11, además los dispositivos WiFi deben escoger ciertos parámetros como el nombre de la red, los canales utilizados y la seguridad implementada, para de esta forma poder establecer la comunicación. Como es característico de esta tecnología los parámetros ofrecen una conexión local (entre nodos de la misma red), hasta el nivel dos de la pila de protocolos TCP/IP.

Tabla 7.
Pila de protocolos TCP/IP

Pila de protcolos TCP/IP	
5	Aplicación
4	Transporte
3	Internet
2	Enlace de datos
1	Física

} WiFi

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

Las redes WiFi resultan útiles en situaciones en donde no se admiten el uso de cables; por ejemplo, son muy usadas en conferencias internacionales , y también son ideales para la comunicación entre maquina-humano, como es el caso de este proyecto en donde se requiere conectividad a largo alcance y alta fiabilidad teniendo bajo coste.

CAPÍTULO 3
ANÁLISIS Y ADECUACIÓN DEL ROBOT

3.1 USC servo controller

El Firmware del controlador oficial de RoboPhilo no es libre por lo cual no se podría realizar ciertas modificaciones que son necesarias para este proyecto, por tal razón se decide cambiarlo por otro controlador especializado para el manejo de servomotores y usado para el manejo de robots caseros llamado “USC servo controller”.

3.1.1 Características

El USC servo controller es desarrollado por una empresa China llamada Torobot, la cual comercializa versiones de 16, 24 y 32 canales, y adicional accesorios para comunicación inalámbrica de estos controladores.

En este proyecto se hace uso del controlador de 32 canales versión 2014 (USC-32 v4.7), el cual tiene varias mejoras como:

- Soporte en Windows, Linux, IOS y Android
- Soporta comunicación WIFI
- Compatibilidad con controles de Play Station
- 16Mb de Flash para grabar miles de movimientos
- Código reescrito (Mejorando estabilidad de hasta 10 veces)
- Rediseño de la placa, aislando las señales (Incrementando rendimiento hasta 10 veces)

USC Servo Controller

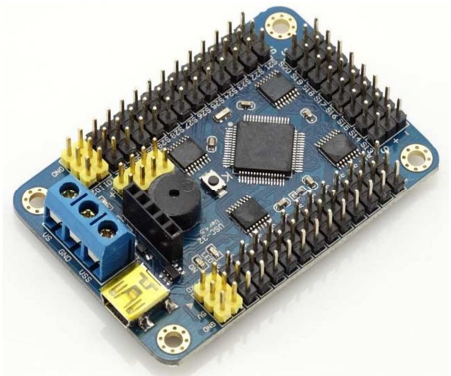


Figura 24. USC servo controller de 32 canales versión 2014
Fuente: (Torobot, 2014)

El controlador necesita 2 fuentes de alimentación, una para el chip “VSS” y otra para los motores “VS”. VSS requiere de 6.5 a 12V para el correcto funcionamiento del controlador, VS depende del voltaje al que trabajen los servomotores.

3.1.2 Montaje del controlador

El controlador USC-32 tiene pines con numeración desde S1 hasta S32 correspondiente a cada uno de los canales disponibles para la conexión de un servomotor, en estos se conectan los 24 motores que tiene el robot RoboPhilo. A continuación se muestra una imagen que detalla la distribución de Servomotores usada.

Diagrama de distribución de servomotores

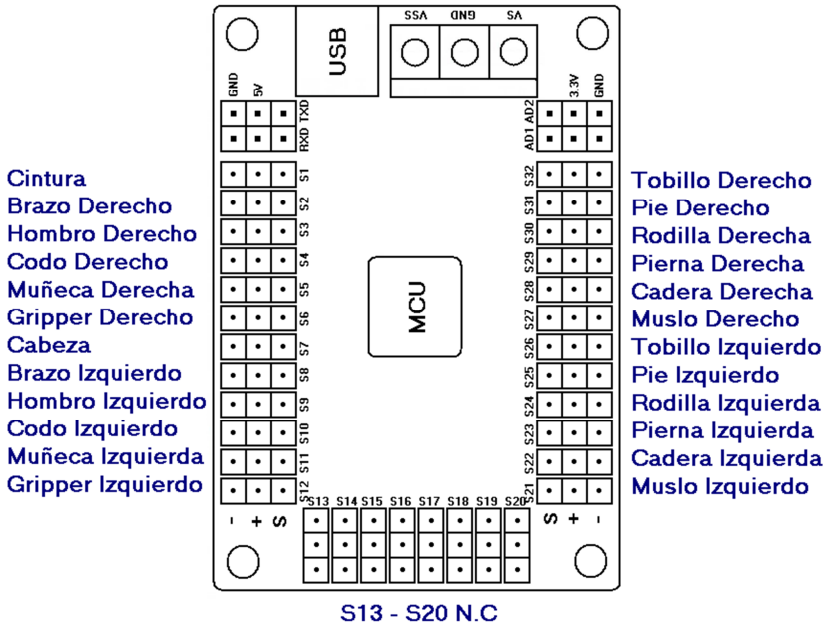


Figura 25. Diagrama de distribución de servomotores
Elaborado por: Vladimir Araujo y Myriam Tipán

Como se puede observar cada uno de los canales tiene la distribución de señal y alimentación que requiere un servomotor convencional, por lo cual la conexión es sencilla y directa.

Para la alimentación se optó por conectar directamente las baterías originales de RoboPhilo hacia VS y VSS esto debido a que el voltaje de 6V que proveen las baterías hace que el controlador funcione perfectamente y además es el voltaje óptimo de trabajo para los servomotores.

Los pines AD1 y AD2 son entradas analógicas que para esta aplicación no son usados por lo cual quedan sin conexión, los pines de Rx y Tx son usados para la conexión con el módulo WiFi que es explicado más adelante.

El controlador USC-32 es de menor tamaño en comparación con el controlador original de RoboPhilo por lo cual cabe perfectamente en el mismo sitio como se puede observar en la figura 26.

USC-32 en RoboPhilo

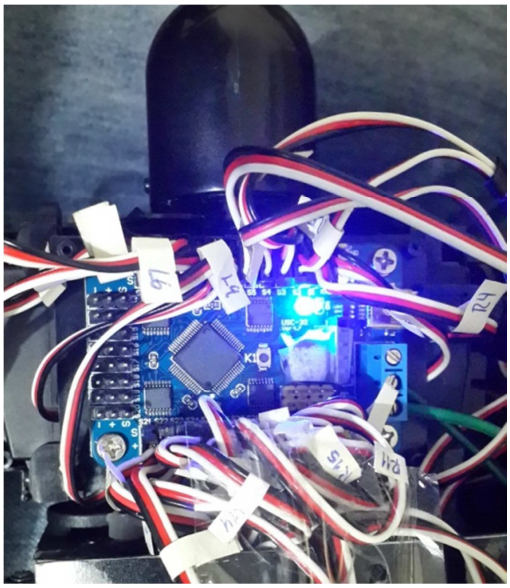


Figura 26. USC-32 en RoboPhilo.
Elaborado por: Vladimir Araujo y Myriam Tipán

El controlador USC-32 es un dispositivo esclavo es decir que debe recibir comandos de un superior (Computador o Microcontrolador) mediante comunicación serial para mover los motores o ejecutar secuencias guardadas en Flash. El formato de los comandos es predefinido por el fabricante y es el que se implementa en la aplicación de control.

Control servomotor

#<canal>P<posición>T<tiempo>\r\n

<canal> = Número de Servo, Rango 1~32

<posición> = Ancho de Pulso (Posición de Servo), Rango: 500~2500, Unidad: us (microsegundos)

<tiempo> = Tiempo de ejecución, Rango 100~9999

\r\n = Retorno de carro (CR, carriage return)

Como el controlador USC-32 no tiene ninguna relación con el controlador de RoboPhilo es necesario hallar las posiciones iniciales del robot y sus extremos, para esto se usa el programa TOROBOT RIOS USC el cual permite controlar los servos en tiempo real, crear acciones grupales y descargarlas en el controlador.

Interfaz TOROBOT RIOS USC

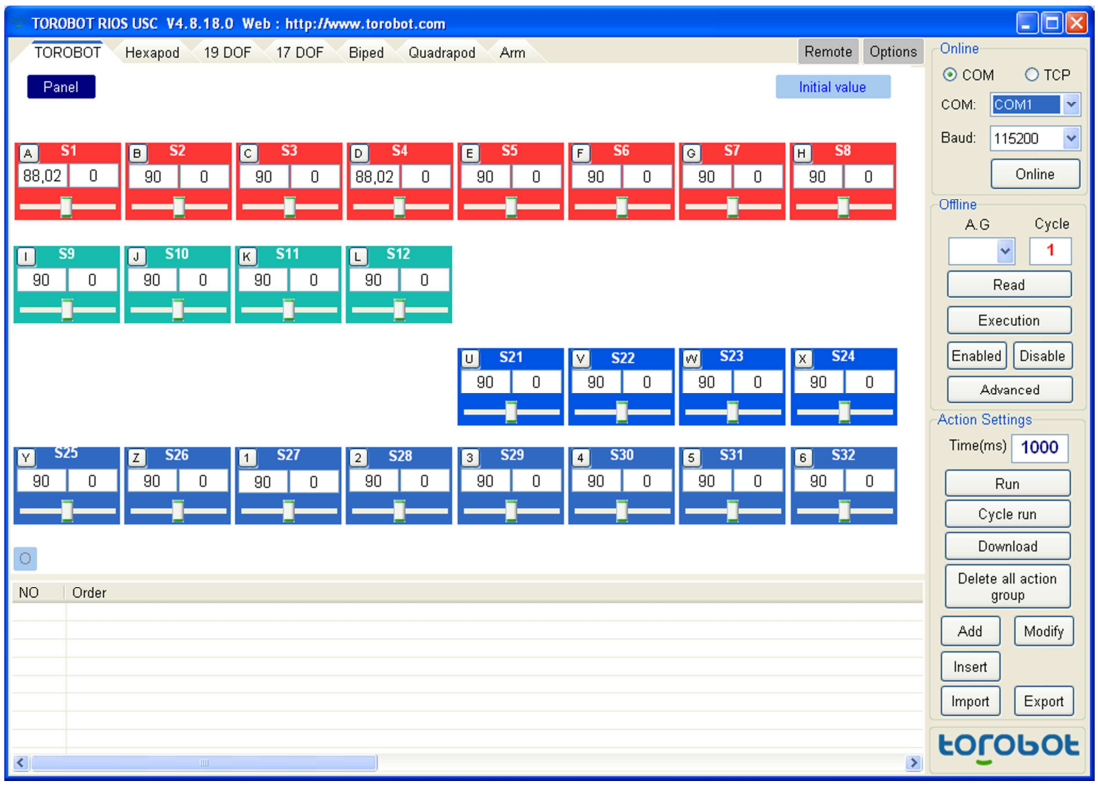


Figura 27. TOROBOT RIOS USC
Fuente: (Torobot, 2014)

Mediante el software del controlador USC-32 se puede variar los pulsos de PWM generados para los servomotores de manera entera desde 500 a 2500, con lo que obtiene los valores iniciales y extremos mediante prueba y error en tiempo real. También es posible obtener el valor equivalente en ángulo cambiando la configuración de PWM a Ángulo en el menú opciones.

A continuación se muestra una tabla que contiene los valores iniciales de ancho de pulso y ángulo equivalente para cada motor y además el rango en el cual se deberán mover. Se debe tomar en cuenta que cada RoboPhilo es ensamblado a mano y de diferente manera por lo cual estos valores serán válidos únicamente para el robot usado en este proyecto.

Tabla 8.
Valores de ancho de pulso y ángulo correspondiente para RoboPhilo

Nº Motor	**Descripción**	**PMW Min (us)**	**PMW Max (us)**	**PWM Inicio (us)**	**Ángulo Min (°)**	**Ángulo Max (°)**	**Ángulo Inicio (°)**
S1	Cintura	1256	1725	1522	68,04	110,25	91,98
S2	Brazo derecho	500	2200	500	0	153	0
S3	Hombro derecho	560	2200	560	5,4	153	5,4
S4	Codo derecho	500	2290	1430	0	161,1	8,3
S5	Muñeca derecha	850	1670	1670	31,5	105,3	105,3
S6	Pinza derecha	1010	1500	1100	45,9	90	54
S7	Cabeza	1576	2256	1856	96,84	158,04	22,04
S8	Brazo izquierdo	500	2230	2230	0	155,7	155,7
S9	Hombro izquierdo	500	2100	2100	0	144	144
S10	Codo izquierdo	500	2350	1380	0	166,5	79,2
S11	Muñeca izquierda	1060	1880	1060	50,4	124,2	50,4
S12	Pinza izquierda	1150	1600	1210	58,5	99	63,9
S13	N.C						
S14	N.C						
S15	N.C						
S16	N.C						
S17	N.C						
S18	N.C						
S19	N.C						
S20	N.C						
S21	Muslo izquierdo	1250	1650	1650	67,5	103,5	103,5
S22	Cadera Izquierda	740	1240	870	21,6	66,6	33,3
S23	Pierna izquierda	500	1960	1080	0	131,4	52,2
S24	Rodilla izquierda	650	2160	1880	13,5	149,4	124,2
S25	Pie izquierdo	600	2350	1500	9	166,5	90
S26	Tobillo izquierdo	1200	1700	1443	63	108	84,87
S27	Muslo derecho	1030	1430	1030	47,7	83,7	47,7
S28	Cadera derecha	1750	2250	2130	112,5	157,5	146,7
S29	Pierna derecha	730	2190	1590	20,7	152,1	98,1
S30	Rodilla derecha	500	2000	730	0	135	20,7
S31	Pie derecho	500	2250	1344	0	157,5	75,96
S32	Tobillo derecho	1200	1700	1446	63	108	85,14

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

3.2 Generación de secuencias de movimiento

El control del robot en este proyecto implica que pueda desplazarse en una superficie plana para esto se optó por generar secuencias de movimientos, esto significa generar cada una de las posiciones que debe tomar cada motor para que el robot pueda caminar.

Para tener un mejor control se decidió generar 8 secuencias de movimientos:

- Adelante: permite que el robot camine hacia el frente.
- Atrás: permite que el robot camine hacia atrás.
- Paso Izquierdo: hace que el robot de un paso lateral hacia la izquierda.
- Paso Derecho: hace que el robot de un paso lateral hacia la derecha.
- Giro Izquierdo: hace que el robot gire en su propio terreno hacia la izquierda.
- Giro Derecho: hace que el robot gire en su propio terreno hacia la derecha.
- Levantar Frente: permite al robot levantarse del suelo si cayó boca abajo.
- Levantar Espalda: permite al robot levantarse del suelo si cayó boca arriba.

Para esto se usa el programa TOROBOT RIOS USC, se debe posicionar cada uno de los motores con las barras de desplazamiento, asignar el tiempo de ejecución e ir agregando cada línea con el botón “Insert”, es recomendable usar “Run” para verificar el correcto funcionamiento de la secuencia generada. Si todo es correcto se procede a descargar la secuencia con el botón “Download”, se asignará un número a cada secuencia grabada.

Cada secuencia grabada en la memoria Flash del controlador USC-32 deberá ser llamada desde el programa de control, para eso se usa el siguiente comando que es definido por el fabricante.

Ejecución de grupo de acciones

#<grupo>GC<ciclos>\r\n

<grupo> = Número de acción grupal

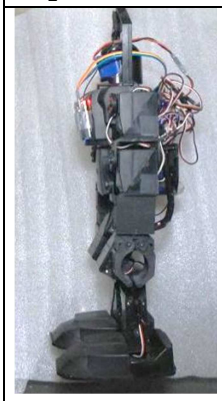
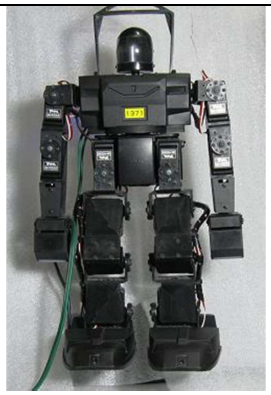
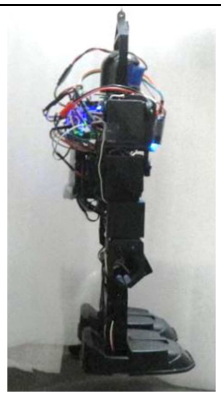
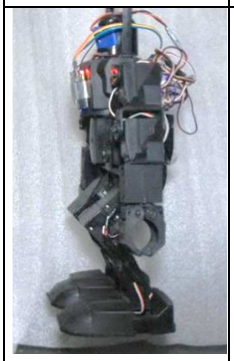
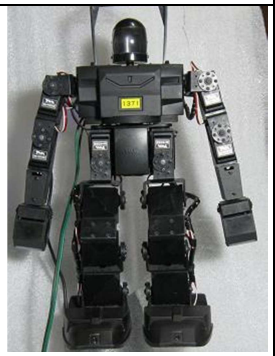
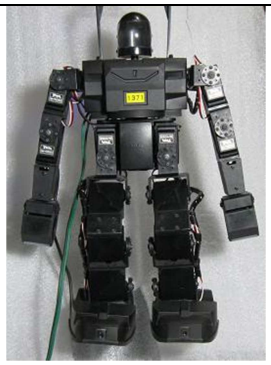
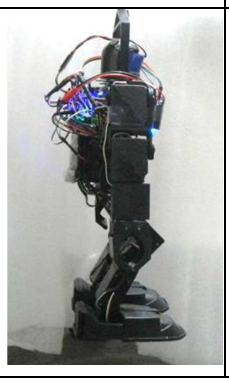
<ciclos> = Numero de repeticiones, Rango 1~999

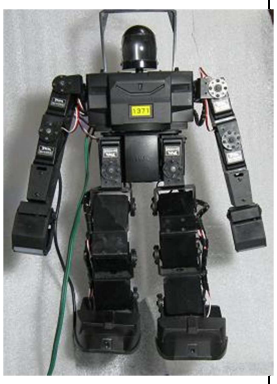
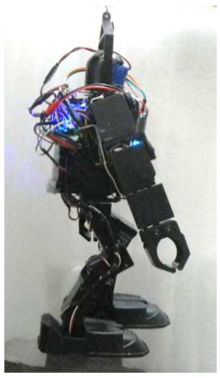
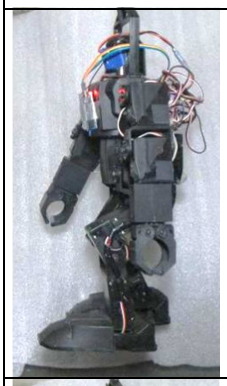
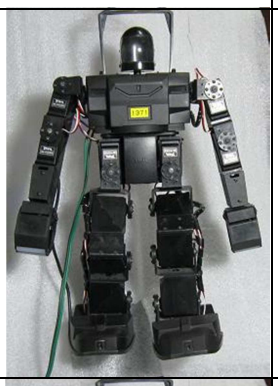
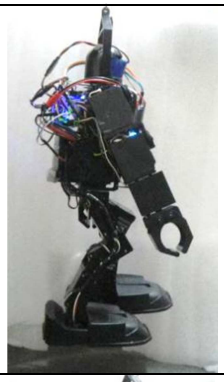
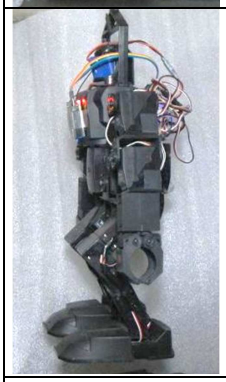
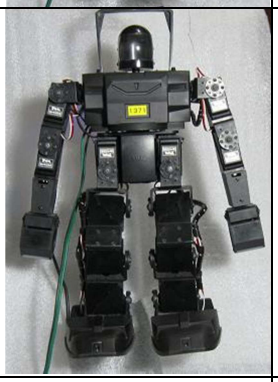
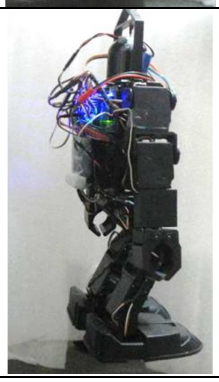
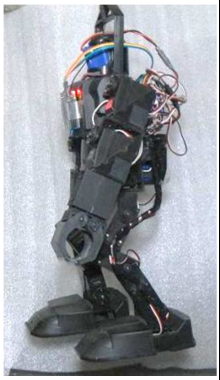
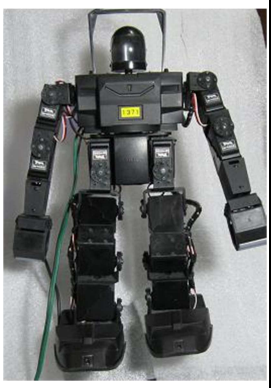
\r\n = Retorno de carro (CR, carriage return)

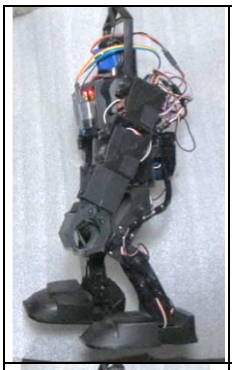
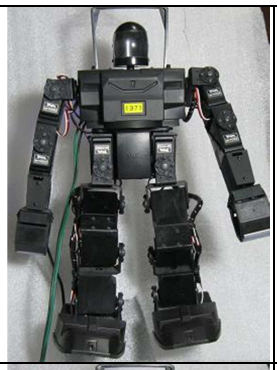
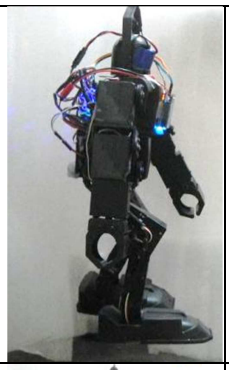
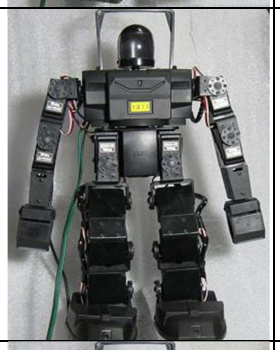
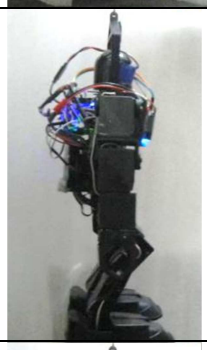
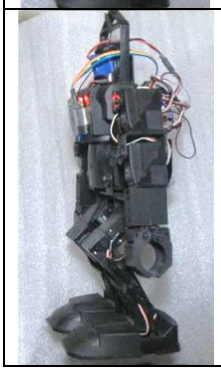
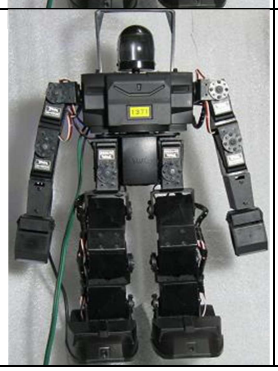
Todas las secuencias generadas fueron realizadas en base a prueba y error, y tomando como guía los movimientos originales del controlador de RoboPhilo.

A continuación se presenta una tabla de posición y descripción de la secuencia completa de caminar adelante, la secuencia de caminado hacia atrás es muy similar a esta.

Tabla 9.
Secuencia de caminado para RoboPhilo

Izquierda	Central	Derecha	Descripción
			**Posición inicial** La posición inicial configura a todos los motores de manera que el robot pueda estar de pie sin ayuda, tanto piernas como brazos en eje X, Y y Z son simétricos.
			**Posición inclinada** En esta posición el robot inclina ambas piernas, mientras que los brazos se abren en eje X.
			**Posición apoyo sobre pie derecho** El tobillo derecho se mueve en el eje X permitiendo inclinar al robot de lado derecho, apoyando el peso sobre la pierna derecha.

			**Posición avance pie izquierdo** El motor de cadera izquierda se mueve ligeramente hacia afuera permitiendo que el movimiento de pierna, rodilla y pie izquierdo en eje Z se realice sin rosar con la pierna derecha. Adicional el brazo derecho se mueve también hacia adelante en eje Z para un mejor balance.
			**Posición apoyo en marcha sobre pie izquierda** En esta posición ya con la pierna izquierda adelantada, se inclina el tobillo izquierdo en eje X para así lograr mantener el peso del robot en la pierna izquierda.
			**Posición recoger pierna derecha** Ya que el peso del robot esta sobre la pierna izquierda la pierna derecha se mueve hacia adelante en eje Z, para regresar a una posición similar a la de inclinación.
			**Posición avance pie derecho** En la anterior posición el tobillo izquierdo se mantuvo en la misma posición por lo cual el peso seguía sobre la pierna derecha, se aprovecha para mover la pierna derecha hacia adelante en eje Z. De manera equivalente en esta posición se mueve el brazo izquierdo para mejor balance del robot.

			**Posición apoyo en marcha sobre pie derecho** En esta posición ya con la pierna derecha adelantada, se inclina el tobillo derecho en eje X para lograr mantener el peso del robot en la pierna izquierda.
			**Posición recoger pierna izquierda** Ya que el peso del robot esta sobre la pierna derecha la pierna izquierda se mueve hacia adelante en eje Z, para regresar a una posición similar a la de inclinación.
			**Posición Final** En esta posición el tobillo derecho vuelve a posición inicial, básicamente esta posición es la misma que la posición inclinada.

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

3.3 Módulo WiFi HLK-RM04

El módulo HLK-RM04, permite dotar de conectividad Wifi a toda placa basada en microcontrolador, el módulo es configurable a través de una interface Web en forma inalámbrica, como se verá mas adelante.

Para la transmisión de información el módulo HLK-RM04, gracias a su UART puede comunicarse facilmente con el controlador de Servo motores, ya que todo lo que recibe es transmitido mediante su puerto serial. El UART ubicado en la PC toma bytes de datos y transmite los bits individuales de forma secuencial. En el destino es decir el driver de motor, un segundo UART re-ensambla los bits en bytes completos, como se puede observar en la figura 28.

WiFi HLK-RM04 – PC

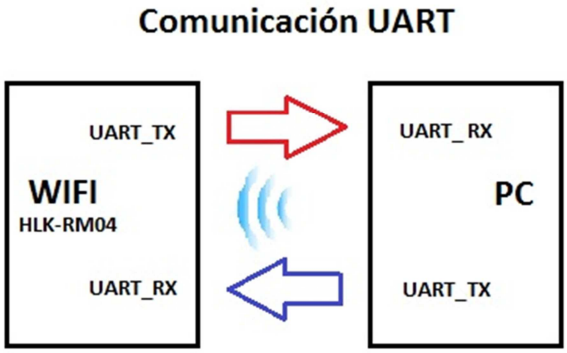


Figura 28. Comunicación WiFi HLK-RM04 – PC
Elaborado por: Vladimir Araujo y Myriam Tipán

El dispositivo UART tiene como función controlar la interfaz de comunicaciones seriales, a partir de importantes parámetros como son :

- Velocidad
- Bits de datos
- Bit de paradidad
- Bit de paridad

Otra de las características a destacar, es que el módulo funciona sobre una distribución Linux, con lo que se puede incluso compilar propios firmwares. Además se puede utilizar el puerto Ethernet (disponibles en los pines de conexión) que permite también conectividad cableada por LAN y funciones de router entre las interfaces (Ethernet-Serial-Wifi), asímismo puede funcionar como cliente o como AP.

A continucación se detalla las principales especificaciones del módulo HLK-RM04:

- Estándares soportados: 802.11n, 802.11g, 802.11b (inalámbrico) y 802.3, 802.3u (cableado)
- Data rate inalámbrico: 11n/b/g
- Potencia de transmisión: 12-15dBm
- Antena: onboard/externa
- Modo de funcionamiento Wifi: cliente/AP/router
- Seguridad Wireless: filtrado MAC/WEP/WPA/WPA2.
- Baud Rate Serial: 50-230400bps.

- Voltaje de operación: 5V.
- Voltage GPIO: 3V.

3.3.1 Configuración módulo WiFi HLK-RM04

Una vez conectado el dispositvo WIFI al controlador de servos como se observa en la figura 29, es necesario ingresar a la interfaz gráfica del módulo HLK-RM04 donde se debe configurar los parámetros necesarios para el establecimiento de conexión y envío de tramas.

Diagrama de conexión

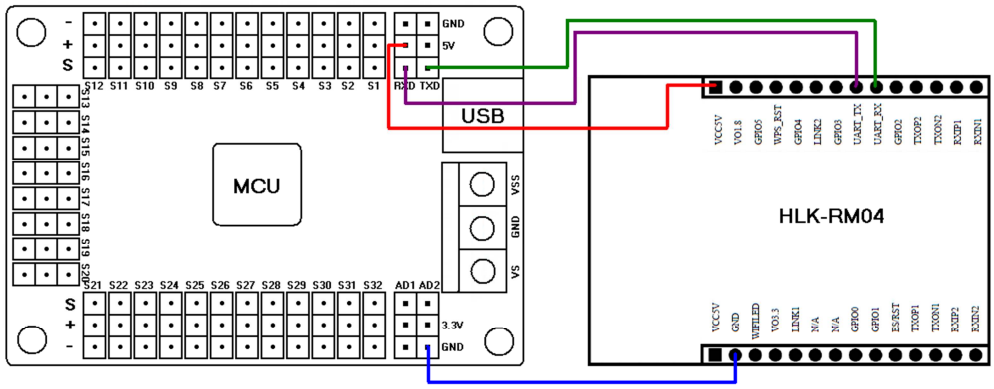


Figura 29. Diagrama conexión módulo WiFi y controlador de servos.
Elaborado por: Vladimir Araujo y Myriam Tipán

Basicamente la conexión va desde los pines UART Rx y Tx del módulo WiFi a los pines Rx y Tx del servo controlador, sin olvidar la alimentacion Vcc y Gnd.

3.3.1.1 Establecimiento de conexión

El dispositivo WIFI irradia una red con SSID (Service Set IDentifier) TOLNK, Para el establecimiento de conexión se debe seleccionar la red TOLNK , la cual no tiene contraseña. Como se puede apreciar en la Figura 30.

Conexión red TOLNK

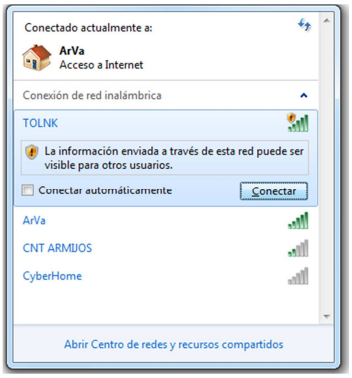


Figura 30. Establecimiento de conexión a traves de la red TOLNK
Elaborado por: Vladimir Araujo y Myriam Tipán

3.3.1.2 Conexión y autenticación con el portal.

Como se observa en la figura 31 ,se procede a ingresar la direccion IP 192.168.88.1 en cualquier explorador (Mozilla, Chrome…etc) , para luego realizar el proceso de autenticación.

Parametros de autenticación:

- Nombre de usuario: admin
- Contraseña: admin

Portal de conexión

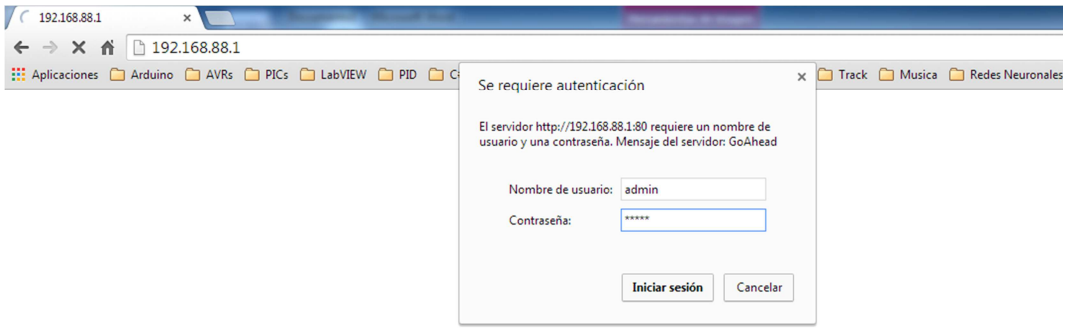


Figura 31. Portal de conexión para el dispositivo WiFi
Elaborado por: Vladimir Araujo y Myriam Tipán

Una vez en el portal de configuración, se tendrá la siguiente pantalla, correspondiente a la presentación, donde señala las utilidades del dispositivo y se puede observar un menú en la parte superior.

Portal de configuración

Figura 32. Portal de conexión para el dispositivo WiFi
Elaborado por: Vladimir Araujo y Myriam Tipán

3.3.1.3 Configuración de parámetros

Para tener comunicación entre la PC y el controlador de servos es necesario configurar ciertos parámetros dentro de la interfaz web, para esto se debe ingresar en la opción “Serial Settings”.

Configuración “Serial Settings”

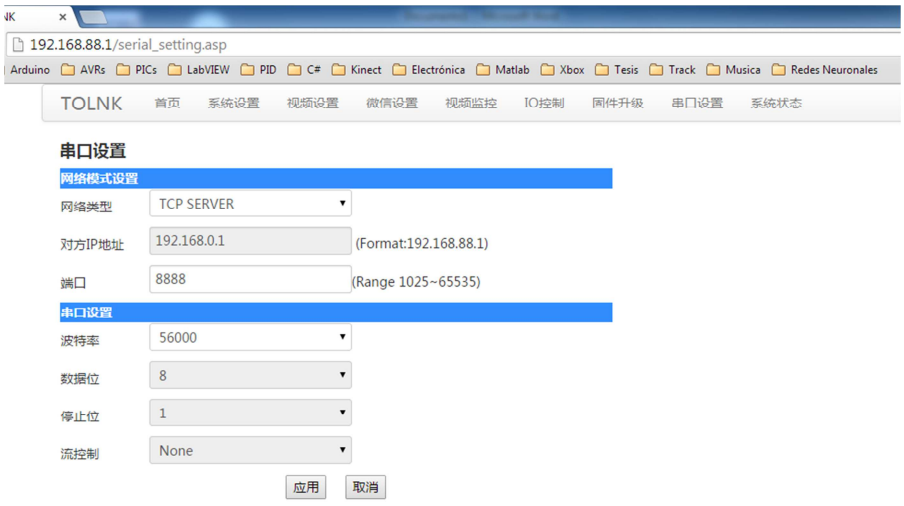


Figura 33. Configuración en menú “Serial Settings”
Elaborado por: Vladimir Araujo y Myriam Tipán

En la primera área se configura el modo de red, en este caso se puede escoger entre configurar el dispositivo como Cliente TCP o Servidor TCP.

Durante una comunicación usando el protocolo TCP, los dos dispositivos deben establecer una conexión. El dispositivo emisor (el que solicita la conexión) se llama

cliente, y el dispositivo receptor se llama servidor. Por lo cual se selecciona la opción "TCP SERVER", creando así un entorno Cliente-Servidor, donde el controlador de servos será el servidor y la computadora el cliente.

Al haber seleccionado la opción de Servidor TCP automáticamente se configura la IP 192.168.88.1 para el dispositivo, el puerto puede ser cualquiera en el rango de 1025 a 65535 pero en este caso se lo configuró con 8888.

En la segunda área se realizan las configuraciones del puerto serie, para esto se debe tomar en cuenta las características del servo controlador USC. El antes mencionado soporta varias velocidades de transmisión por lo que la configuración elegida es:

- Velocidad de transmisión: 56000
- Bits de datos: 8
- Bits de parada: 1
- Control de flujo: ninguno

3.4 Análisis cinemático del robot humanoide: RoboPhilo

En los robots móviles, la resolución de la cinemática directa no tiene una gran utilidad, sin embargo esta cinemática es usada para calcular la cinemática inversa del robot, utilizada para la elaboración de algoritmos para el caminar del robot.

3.4.1 Cinemática inversa

Existen varios procedimientos genéricos capaces de ser programados, de modo que un computador pueda obtener los valores articulares que posicionan y orientan el extremo del robot. El inconveniente de estos procedimientos es que se trata de métodos numéricos iterativos, cuya velocidad de convergencia e incluso su convergencia en si no está siempre garantizada.

En este proyecto es necesaria una solución cerrada debido a que es una aplicación en tiempo real, un método sistemático para resolver este problema es el método geométrico explicado anteriormente.

El robot RoboPhilo tiene 4 extremidades principales las cuáles serán las encargadas de replicar el movimiento del usuario frente al sensor Kinect, cada una de ellas posee 3 grados de libertad lo que facilita en cierta medida la resolución de su cinemática

inversa. Cada extremidad será contenida por un plano independiente y cada extremidad tendrá su análisis individual.

Para iniciar la resolución se tomará el brazo derecho del robot para el análisis, donde la muñeca será el extremo $(\rho x, \rho y, \rho z)$, mientras que el hombro será XYZ=0 y adicional el brazo tiene 3 ángulos de movimiento que se llamarán:

- Brazo derecho (brd): es el movimiento en el plano YZ del brazo
- Hombro derecho (hod): es el movimiento en el plano XY del brazo
- Codo derecho (cod): es el movimiento en el plano XY del codo

Básicamente el análisis corresponde al del ejemplo del capítulo 2, por lo cual las ecuaciones [3], [6] y [10] halladas son válidas para el RoboPhilo. Entonces para encontrar el ángulo de la variable articular brd se hará uso de la ecuación [3].

$$brd = \tan^{-1}\frac{\rho y}{\rho z} \qquad [\ 11\ ]$$

Como se puede observar la ecuación no es completamente la misma, esto debido a que el plano coordenado está orientado de diferente manera.

Ángulo brazo derecho de RoboPhilo

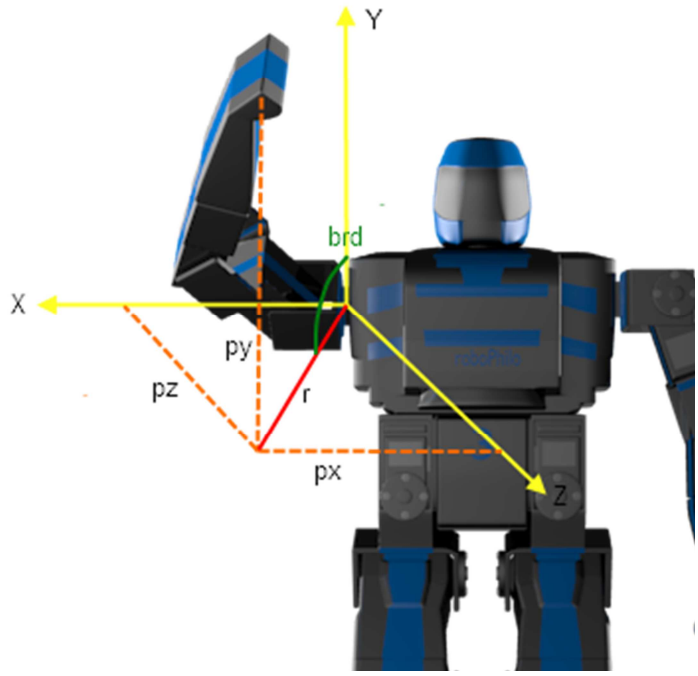


Figura 34. Ángulo brazo derecho de RoboPhilo
Elaborado por: Vladimir Araujo y Myriam Tipán

Ahora para hallar cod se debe usar la ecuación [6], en este caso no habrá cambio en la ecuación debido a la modificación del plano. Entonces se tendrá:

$$cod = \tan^{-1}\left(\frac{\pm\sqrt{1-(\cos cod)^2}}{\cos cod}\right) \qquad [\ 12\ ]$$

con

$$\cos cod = \frac{\rho x^2 + \rho y^2 + \rho z^2 - l2^2 - l3^2}{2 * l2 * l3}$$

Finalmente para calcular hod la expresión es:

$$hod = \tan^{-1}\left(\frac{\rho z}{\pm\sqrt{\rho x^2 + \rho y^2}}\right) - \tan^{-1}\left(\frac{l3 * \sin cod}{l2 + l3 * \cos cod}\right) \qquad [\ 13\ ]$$

Todas estas expresiones deberán ser implementadas en C# tomando en cuenta que se debe controlar la obtención de solución dependiente de la configuración del robot para las ecuaciones [12] y [13].

Ángulo codo y hombro derecho de RoboPhilo

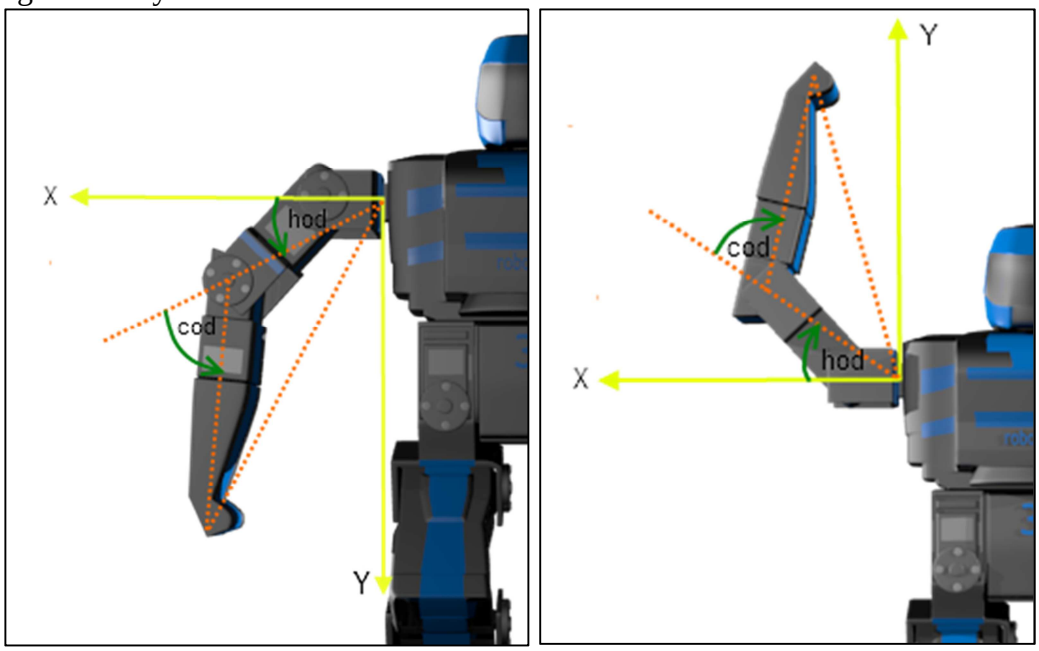


Figura 35. Ángulo codo y hombro derecho de RoboPhilo
Elaborado por: Vladimir Araujo y Myriam Tipán

El resto de extremidades usan el mismo análisis resultando las mismas expresiones para todos los casos; por ejemplo para la pierna derecha los ángulos de movimientos correspondientes son:

- Pierna derecha (pid) corresponde a la ecuación de Brazo Derecho (brd).
- Cadera derecha (cad) corresponde a la ecuación de Hombro Derecho (hod).
- Rodilla derecho (rod) corresponde a la ecuación de Codo Derecho (cod).

CAPÍTULO 4
DISEÑO E IMPLEMENTACIÓN DE LA APLICACIÓN DE CONTROL

4.1 Procesamiento de señales

En el sensor Kinect se puede controlar la cámara infrarroja, de profundidad y arreglo de micrófonos de audio. Los datos pasan entre el sensor y la aplicación en la forma de flujos de datos de los siguientes tipos:

• Flujo de datos en color

• Flujo de datos en Profundidad

• Flujo de datos de audio

En este proyecto se hace uso de los datos de profundidad, para esto es necesario verificar la conexión del sensor kinect, en el código que se muestra a continuación, la aplicación reconoce el sensor que primero se haya conectado, y lo define como el sensor potencial que será usado en todo el proceso.

```
if (potentialSensor.Status == KinectStatus.Connected)
  {
   this.sensor = potentialSensor;
   break;
  }
```

Luego, se habilita el método `SkeletonStream`, para recibir tramas del esqueleto, el SkeletonStream puede realizar un seguimiento de hasta seis esqueletos al mismo tiempo. El sensor transmite los datos del esqueleto al ordenador mediante el uso de objetos del evento `SkeletonFrame`, evento que se activa cuando nuevos datos del esqueleto están disponibles.

```
this.sensor.SkeletonStream.Enable();
this.sensor.SkeletonFrameReady += this.SensorSkeletonFrameReady;
```

4.1.1 Detección de esqueleto

El proyecto tiene como objeto la réplica de movimientos del usuario, esto es posible gracias al SDK de kinect, el cual permite obtener hasta 20 puntos de las articulaciones del usuario y manipularlos.

Los componentes de kinect como son la cámara infrarroja y de profundidad, son activados en la aplicación llamando a la siguiente librería:

```
using Microsoft.Kinect;
```

Para poder ubicar la silueta del cuerpo del usuario y poder extraer la ubicación tridimensional de diversos puntos del cuerpo es necesario localizar el primer esqueleto reconocido mediante el código.

```
Skeleton[] skeletons = new Skeleton[0];
```

El sensor Kinect puede identificar el rango de píxeles de un usuario a través de los datos de profundidad. Cada pixel dentro de los datos representa la distancia entre el objeto y el sensor, con estos valores y una vez rastreado el esqueleto, se puede graficar los huesos y articulaciones.

```
if (skel.TrackingState == SkeletonTrackingState.Tracked)
                    {
                        this.DrawBonesAndJoints(skel, dc);
if ((mimitar == true || mcmaniobrar == true) && tcpclnt.Connected)
                            this.ControlarRobot(skel);
                    }
```

Como se puede observar existe un condicionante referente al control del robot, en donde únicamente si el ordenador está conectado a la red TCP y se haya seleccionado una de las opciones de imitar del menú, la aplicación procederá con el cálculo de cinemática inversa, en el siguiente código se muestra cómo crear una variable de tipo *joint* la cual almacena los datos de una articulación especifica del esqueleto.

```
//Brazo Derecho
Joint MunecaDer = skel.Joints[JointType.WristRight];
Joint HombroDer = skel.Joints[JointType.ShoulderRight];
Joint CodoDer = skel.Joints[JointType.ElbowRight];
```

Con estas líneas se obtienen los valores x, y, z de las articulaciones del brazo derecho, de la misma forma se procede con el resto de las extremidades, para finalmente obtener 12 variables tipo *joint* que almacenaran los datos, de esta manera se podrá hallar los ángulos por medio de cinemática inversa, calcular los valores usados para los servos en "PWM", el despliegue de los ángulos y posición de los servos para enviar al robot.

Cuando un usuario se mueve hacia el borde del campo de vista, el sistema de seguimiento de la aplicación del esqueleto advierte que el esqueleto del usuario se recorta. Esta información puede ser utilizada por la aplicación para avisar al usuario que se debería mover hacia el centro de la escena para permitir un mejor seguimiento. Mediante el siguiente código se podrá observar en la aplicación unas líneas de advertencia de color rojo referidas a los bordes; top, right, left, bottom,

estas le indicaran al usuario cuando se haya sobrepasado el campo de visión de la cámara.

```
private static void RenderClippedEdges(Skeleton skeleton, DrawingContext
drawingContext)
        {
            if (skeleton.ClippedEdges.HasFlag(FrameEdges.Bottom))
            {
                drawingContext.DrawRectangle(
                    Brushes.Red,
                    null,
                    new Rect(0, RenderHeight - ClipBoundsThickness,
RenderWidth, ClipBoundsThickness));
            }

            if (skeleton.ClippedEdges.HasFlag(FrameEdges.Top))
            {
                drawingContext.DrawRectangle(
                    Brushes.Red,
                    null,
                    new Rect(0, 0, RenderWidth, ClipBoundsThickness));
            }

            if (skeleton.ClippedEdges.HasFlag(FrameEdges.Left))
            {
                drawingContext.DrawRectangle(
                    Brushes.Red,
                    null,
                    new Rect(0, 0, ClipBoundsThickness, RenderHeight));
            }

            if (skeleton.ClippedEdges.HasFlag(FrameEdges.Right))
            {
                drawingContext.DrawRectangle(
                    Brushes.Red,
                    null,
                    new Rect(RenderWidth - ClipBoundsThickness, 0,
ClipBoundsThickness, RenderHeight));
            }
        }
```

4.1.2 Reconocimiento de voz

Una parte importante del proyecto es el control del robot, mediante comandos de voz, para lo cual es necesario incluir las librerías de reconocimiento de voz a la aplicación, con el siguiente código.

```
    using Microsoft.Speech.AudioFormat;
    using Microsoft.Speech.Recognition;
```

La librería Microsoft.Speech.AudioFormat establece la información sobre el formato del audio a través de su única clase llamada SpeechAudioFormatInfo, por otro lado la librería Microsoft.Speech.Recognition ofrece la funcionalidad con la que se puede adquirir y controlar la entrada de voz.

Para que el sensor Kinect sea capaz de reconocer la voz del usuario, se debe configurar diversos parámetros como el formato de codificación, las muestras por segundo, bits por muestra, el número de canales, media de bytes por segundo, valor de la alineación de bloques, y una matriz que contiene los datos específicos del formato, en el presente código se muestra los valores asignados a los parámetros respectivamente.

```
new SpeechAudioFormatInfo(EncodingFormat.Pcm, 16000, 16, 1, 32000, 2, null);
```

El formato de audio escogido es el PCM (Pulse code modulation), que es la digitalización de la señal de voz en valores discretos y su transformación en forma de señal binaria a través del filtrado de la señal, muestreo, cuantificación y codificación.

Para determinar la frecuencia de muestreo se tomó en consideración el criterio de Nyquist el cual establece que la frecuencia de muestreo debe ser mayor o igual al doble de la frecuencia de la señal de entrada, es conocido que la voz trabaja a una frecuencia de 4 kHz, el sistema requiere de una reconstrucción total de la señal por ese motivo se ocupa 16000 Hz como frecuencia de muestreo y 16 bits por muestra que equivale a 2 bytes por muestra, el tipo de canal es mono por lo cual tiene el valor de 1,

La velocidad de transmisión es de 32000 bits por segundo, y se halla mediante la expresión:

$$Vtx = fm * Nm \qquad [\ 14\ ]$$

Donde:

Vtx= Velocidad de Transmisión
fm= frecuencia de muestreo
Nm= Número de bits por muestra

El siguiente parámetro es el número de bloques de alineación en donde se establece que:

$$Nblo = Nbm * Nc \quad [15]$$

Donde:

Nblo= Número de bloques

Nbm= Número de bytes por muestra

Nc= Número de canales

Entonces el número de bloques alineados para un valor de 16 bits por muestra en formato PCM, en un canal mono, es de 2.

El sensor Kinect recepta las muestras de sonido captadas por el arreglo de micrófonos, para extraer y comparar las palabras que contienen dichas muestras, se emplean los métodos speechRecognizer y SreSpeechRecognized, los cuales hallan coincidencias con las palabras almacenadas en el archivo auxiliar XML, que va agregado a la librería y las palabras que han sido agregadas como opciones para el modo controlar.

4.2 Comunicación serial

Como fue explicado anteriormente el sistema de control, es conectado a una red WLAN (wireless local area network), para controlar el acceso a la red, es necesario el uso de las siguientes librerías.

```
using System.Net;
using System.Net.Sockets;
```

System.Net proporciona una interfaz de programación sencilla para muchos de los protocolos utilizados en las redes de hoy en día, sin embargo la librería System.Net.Sockets es la que permite la administración de la interfaz de Windows Sockets (Winsock) para los desarrolladores que necesitan controlar estrictamente el acceso a la red.

Con el siguiente código se crea un objeto de tipo cliente TCP, en la segunda línea se establece la conexión con el servidor usando el método connect, donde se indica la dirección IP, y el número de puerto, ambos datos tomados de los textbox de la aplicación.

```
TcpClient tcpclnt = new TcpClient();
```

```
tcpclnt.Connect(txtIP.Text, Int32.Parse(txtPort.Text));
```

Luego, se utiliza el método tcpclnt.GetStream para enviar y recibir datos por la red, además se crea un objeto de tipo ASCIIEncoding para representar una codificación en caracteres ASCII

```
Stream stm = tcpclnt.GetStream();
ASCIIEncoding asen = new ASCIIEncoding();
```

Para enviar los datos se creó un método que recibe una la trama de tipo string, luego esta es codificada en una secuencia bytes y almacenada en un vector, para finalmente ser enviada al robot.

```
public void EnviarDatos(String stg)
        {
            if (tcpclnt.Connected)
            {
                byte[] ba = asen.GetBytes(stg + "\r\n");
                stm.Write(ba, 0, ba.Length);
            }
        }
```

4.2 Diseño de la aplicación

La aplicación fue desarrollada completamente en Visual Studio 2012 en lenguaje C# y con el uso de WPF para una mejor presentación gráfica y más amigable al usuario.

En el proyecto se hace uso del Kinect SDK y Speech SDK para el uso de todos los componentes que tiene el sensor Kinect.

4.2.1 Ventana splash

Una ventana splash es la ventana de presentación que aparece al principio de la aplicación; sin los botones clásicos y que va a servir para darle una mejor presentación al proyecto.

Esta ventana tiene como objetivo ser la portada, mostrar el título del proyecto y sus desarrolladores, ver Figura 36:

Ventana Splash

Figura 36. Ventana splash de aplicación de control
Elaborado por: Vladimir Araujo y Myriam Tipán

4.2.2 Ventana principal

La ventana principal es donde se obtiene las señales del sensor Kinect que luego son procesadas para obtener los puntos del esqueleto, graficarlo y resolver la cinemática inversa la cual es usada para enviar al robot vía WiFi y posicionar sus motores.

Esta ventana contiene los siguientes elementos:

- Un panel de dibujo donde se grafica el esqueleto haciendo uso de los puntos obtenidos con el SDK de Kinect.
- Un menú animado el cual es manejado por medio de comandos de voz.
- Una casilla de verificación que permite seleccionar el tipo de seguimiento del esqueleto, los cuales son por defecto y sentado.
- Un par de cuadros de texto y botones para realizar la conexión WiFi TCP con el Robot
- Un cuadro de texto que muestra el estado de reconocimiento de comandos de voz.

Ventana principal

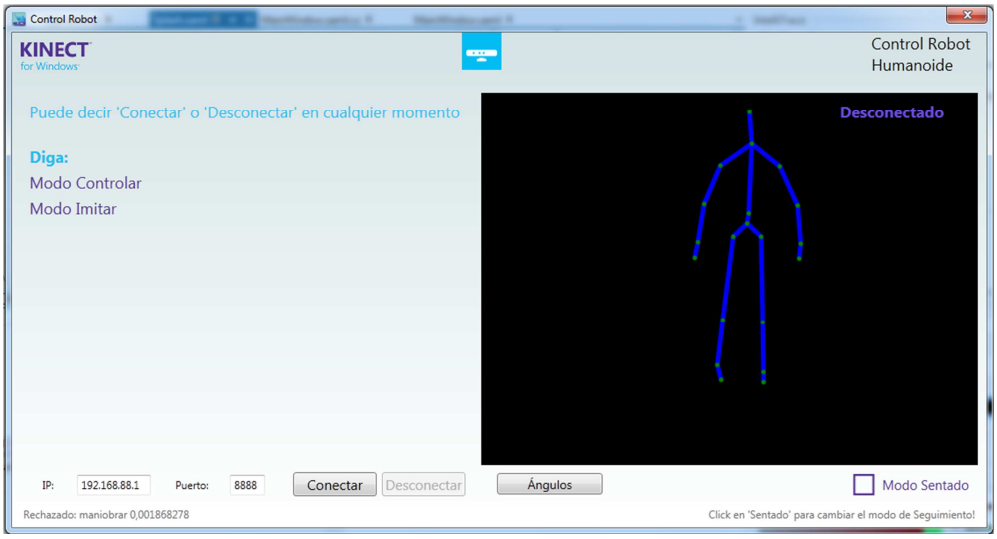


Figura 37. Ventana principal de aplicación de control
Elaborado por: Vladimir Araujo y Myriam Tipán

4.2.3 Ventana de ángulos

Esta ventana sirve para observar las variables articulares es decir los ángulos resultantes del cálculo cinemático inverso, y adicional el valor PWM correspondiente a cada uno de ellos que es mandado al controlador de servomotores.

La transferencia de datos desde la ventana principal a esta ventana se realizó con la ayuda de delegados y eventos.

Ventana de ángulos

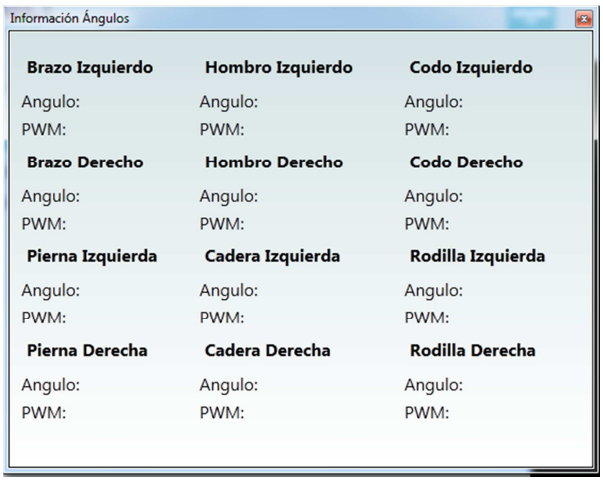


Figura 38. Ventana de ángulos de aplicación de control
Imagen: Vladimir Araujo y Myriam Tipán

4.3 Diagramas de flujo

A continuación se describe el proceso del sistema mediante el uso de diagramas de flujo, estos diagramas muestran el ciclo de la aplicación de control que se desarrolló para el control de movimiento del robot, adquisición de señales de los sensores del kinect y para transmisión y recepción de datos por comunicación serial inalámbrica entre el robot y el computador.

4.3.1 Diagrama principal

Diagrama de flujo principal

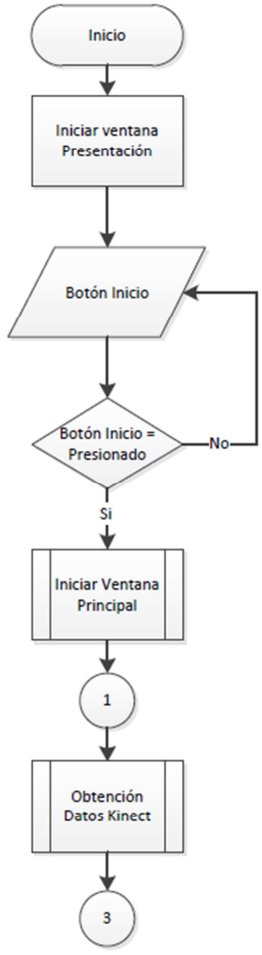


Figura 39. Diagrama de flujo principal
Elaborado por: Vladimir Araujo y Myriam Tipán

El diagrama principal (Figura 39 y 40) presenta la estructura más simple del proyecto, en la Figura 39 se puede observar la inicialización de la ventana de presentación y de la ventana principal, adicional un bloque de subproceso de

obtención de datos del Kinect y procesamiento de los mismos. El bloque de ventana principal también es un subproceso que será explicado más adelante a detalle. Se puede también observar dos conectores con referencia 1 y 2; el conector 1 es de retorno, y el conector 2 es el que conecta al resto del diagrama principal.

Diagrama de flujo principal

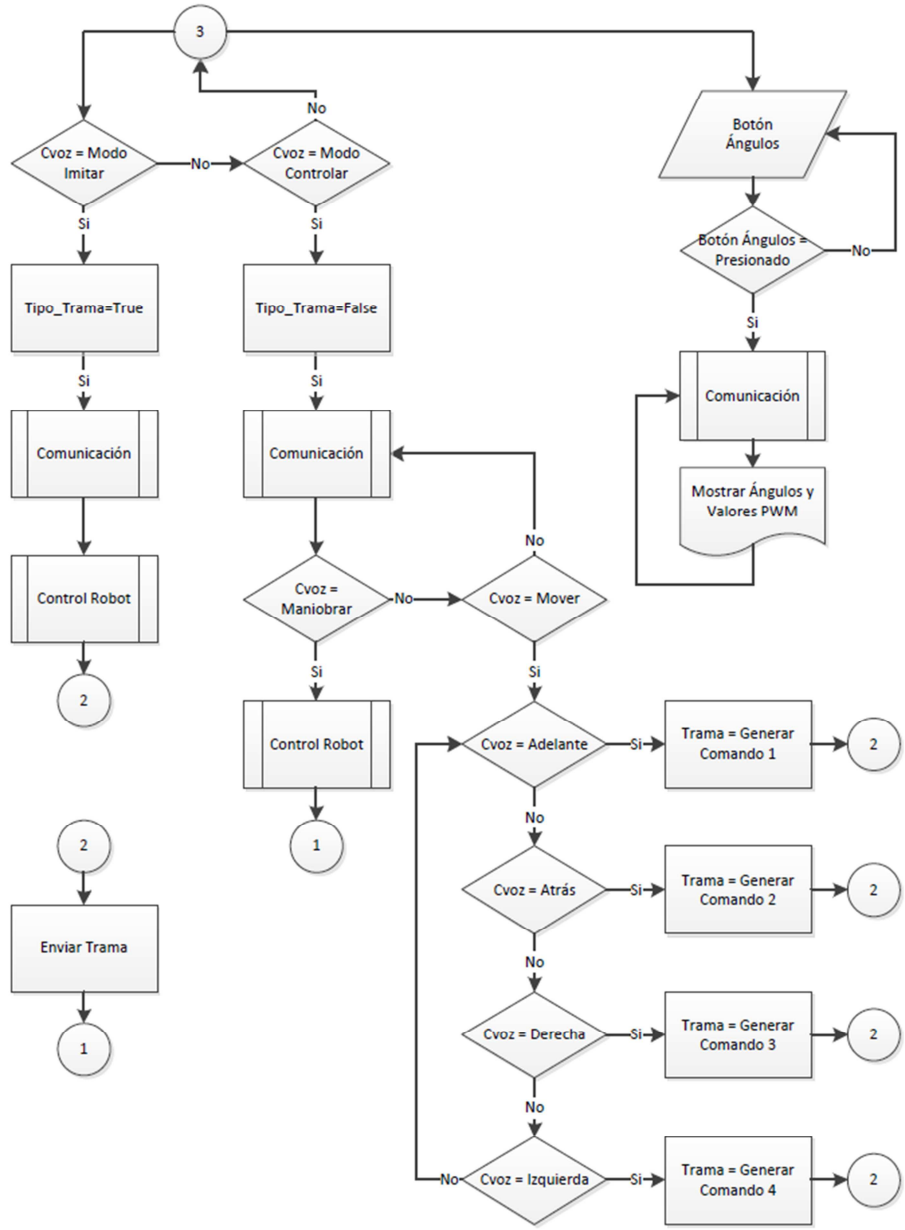


Figura 40. Diagrama de flujo principal (continuación)
Elaborado por: Vladimir Araujo y Myriam Tipán

En esta figura se puede observar el menú generado y que es controlado con el comando de voz obtenido desde el SDK de Kinect, dependiendo del menú que se elija se genera la trama la cual es finalmente enviada por WiFi.

4.3.2 Subproceso iniciar ventana principal

Diagrama de flujo de subproceso iniciar ventana principal

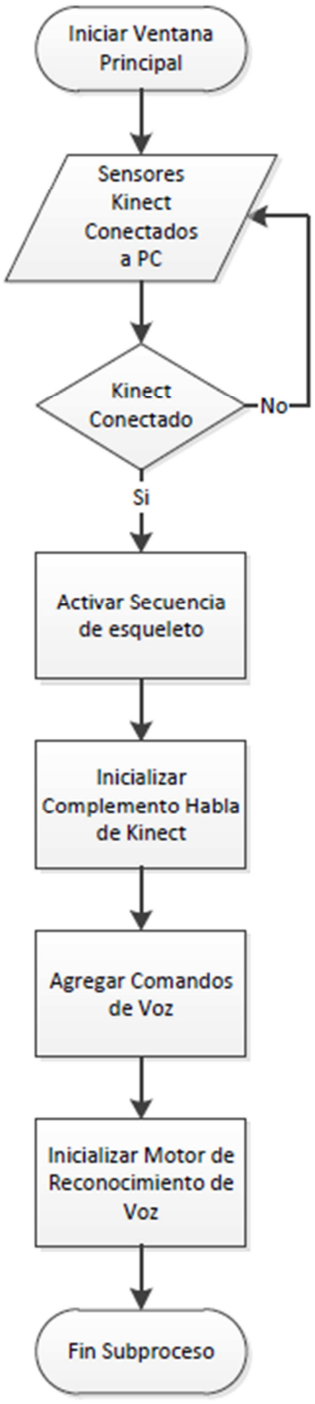


Figura 41. Diagrama de flujo de subproceso iniciar ventana principal
Elaborado por: Vladimir Araujo y Myriam Tipán

La figura 41 muestra el subproceso iniciar ventana principal, el cual detecta el Kinect conectado y realiza configuraciones iniciales para el uso de audio y video.

4.3.3 Subproceso obtención datos Kinect

Diagrama de flujo de subproceso obtención datos Kinect

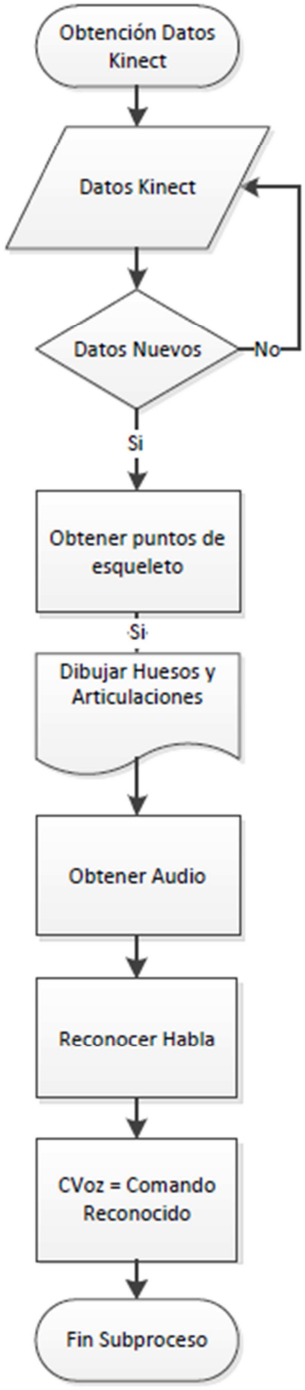


Figura 42. Diagrama de flujo de subproceso obtención datos Kinect
Elaborado por: Vladimir Araujo y Myriam Tipán

La figura 42 presenta el sub proceso obtención datos, esta se encarga de actualizar los datos que capta el sensor Kinect para posteriormente procesarlos y obtener los puntos del esqueleto y comandos de voz.

4.3.4 Subproceso control robot

Diagrama de flujo de subproceso control robot

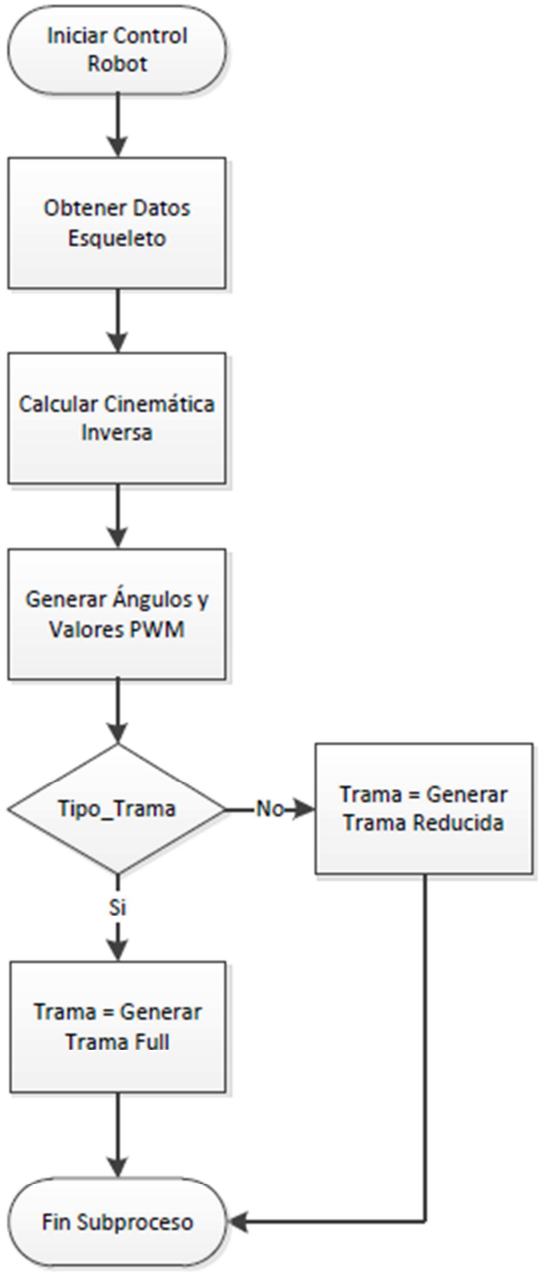


Figura 43. Diagrama de flujo de subproceso control robot
Elaborado por: Vladimir Araujo y Myriam Tipán

El subproceso control robot (Figura 43) obtiene los datos del esqueleto y calcula la cinemática inversa para así obtener los ángulos que serán enviados al robot, también se puede observar que se ha optado por generar dos tipos de tramas una completa que manda datos para los 24 motores que tiene el robot y una reducida donde solo manda a brazos y piernas.

4.3.5 Subproceso comunicación

Diagrama de flujo de subproceso comunicación

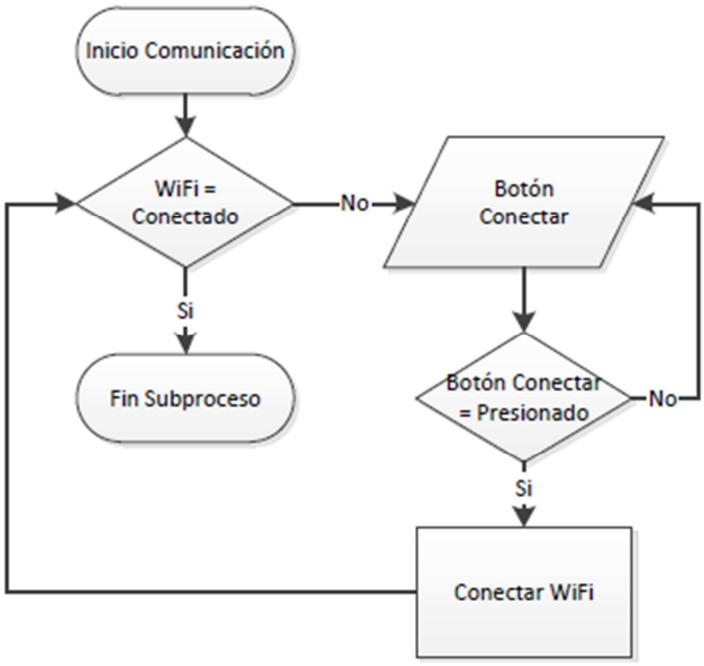


Figura 44. Diagrama de flujo de subproceso comunicación
Elaborado por: Vladimir Araujo y Myriam Tipán

El subproceso comunicación se usa para verificar el estado de la conexión con el robot mediante WiFi, solo si esta activa la conexión podrá enviar la trama hacia el robot.

CAPÍTULO 5

PRUEBAS Y RESULTADOS

En este capítulo se presentan los resultados de las pruebas realizadas sobre la plataforma RoboPhilo, a continuación se presenta un esquema general del funcionamiento del sistema Figura 45.

Diagrama de funcionamiento

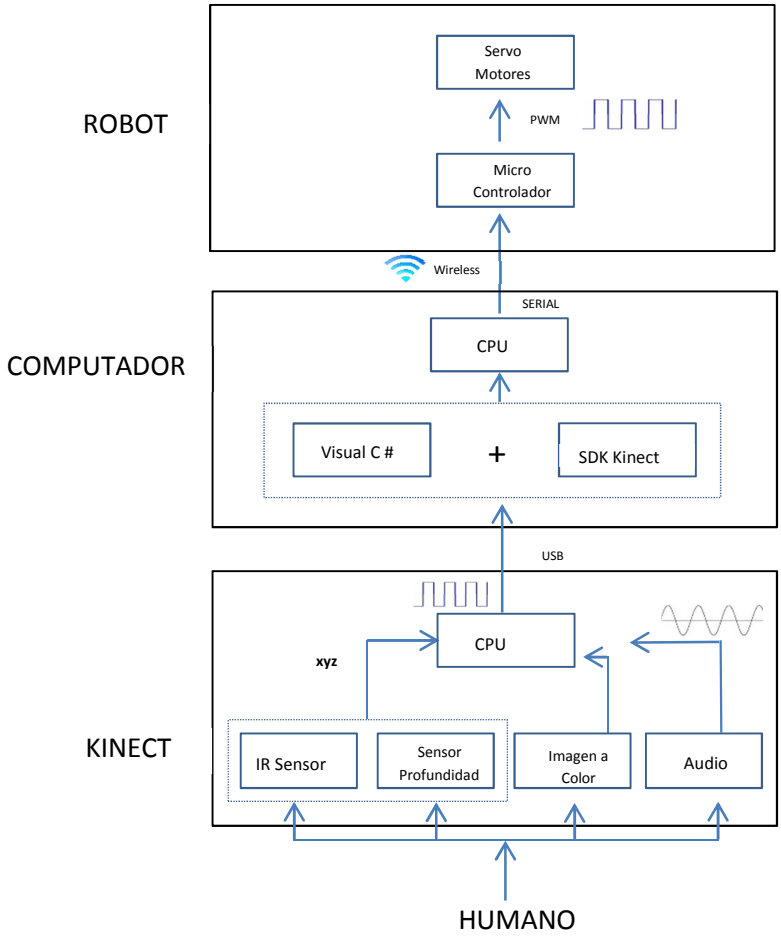


Figura 45. Diagrama de funcionamiento del sistema completo
Elaborado por: Vladimir Araujo y Myriam Tipán

El sensor Kinect obtiene todos los datos del entorno en video y audio, estos ingresan a un procesador digital de señales el cual manda toda la información vía USB al computador. El computador en conjunto con el SDK de Kinect procesa los datos, para poder así obtener información del esqueleto la cual entra en un proceso de cálculo de cinemática inversa y obtener ángulos de movimiento los cuales son enviados en una trama serial vía WiFi hacia el robot. El robot lee la trama con el microcontrolador y mueve los servomotores posicionándolos al igual que el usuario.

5.1 Pruebas en modo imitar

Para probar el modo imitar se optó por recolectar valores de ángulos correspondientes a las extremidades del cuerpo humano; para medir los ángulos del robot y del usuario se usó un graduador y los valores correspondientes al cálculo de la cinemática inversa se la obtiene directamente del computador en tiempo real. El rango de profundidad del sensor de Kinect está entre 0.4 y 4 metros por lo cual es recomendable ubicarse dentro de ese rango para las pruebas.

La tabla 9, 10, 11 y 12 muestra ángulos promedio obtenidos del usuario, robot y computador; se toma el ángulo de usuario como principal. La variación 1 pertenece a la diferencia entre los ángulos de usuario y robot mientras que la variación 2 entre los ángulos de usuario y computador.

Medición 1 (brazo izquierdo)

La primera medición se la obtuvo del brazo izquierdo, el cual se lo colocó hacia arriba y el puño un poco más adelante del hombro. Ver figura 46.

Posición para medición del brazo izquierdo

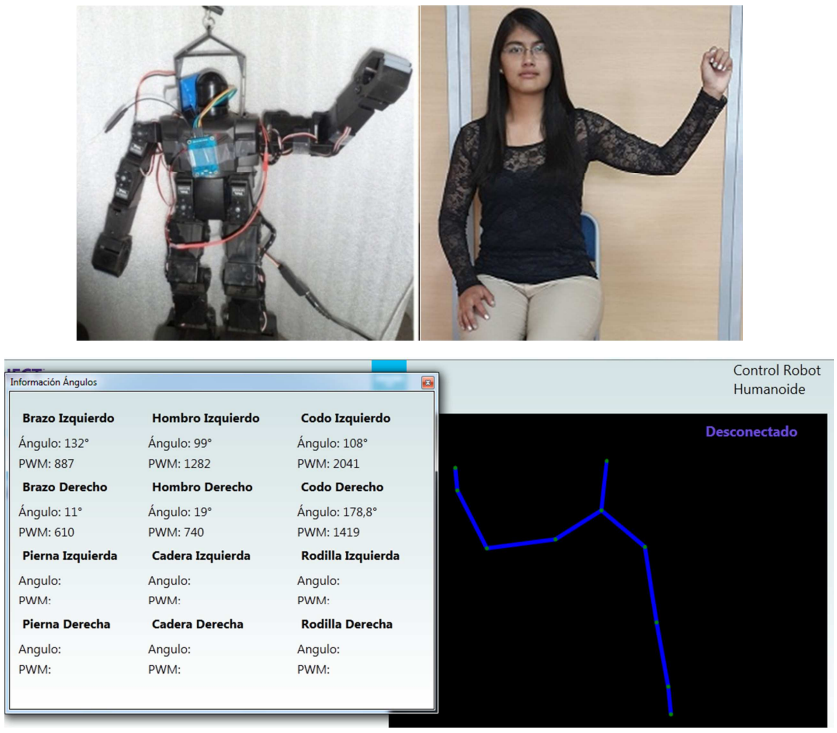


Figura 46. Posición para medición del brazo izquierdo
Elaborado por: Vladimir Araujo y Myriam Tipán

Tabla 10.
Ángulos Brazo Izquierdo

Articulación	Ángulo Usuario [°]	Ángulo Robot [°]	Ángulo Computador[°]	Variación 1 [°]	Variación 2 [°]
Brazo izquierdo	139,9	130,8	132,5	9,1	7,4
Hombro izquierdo	105,1	94,9	99,7	10,2	5,4
Codo izquierdo	91,9	103	108,2	11,1	5,2

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

Se puede observar que la mayor variación es de 11,1 grados entre usuario y robot, mientras que la mínima variación es de 5,2 grados y se da entre usuario y computador.

Medición 2 (brazo derecho)

La segunda medición se la obtuvo del brazo derecho, el cual se lo colocó hacia abajo y con el puño un poco más adelante del hombro. Ver figura 47.

Posición para medición del brazo derecho

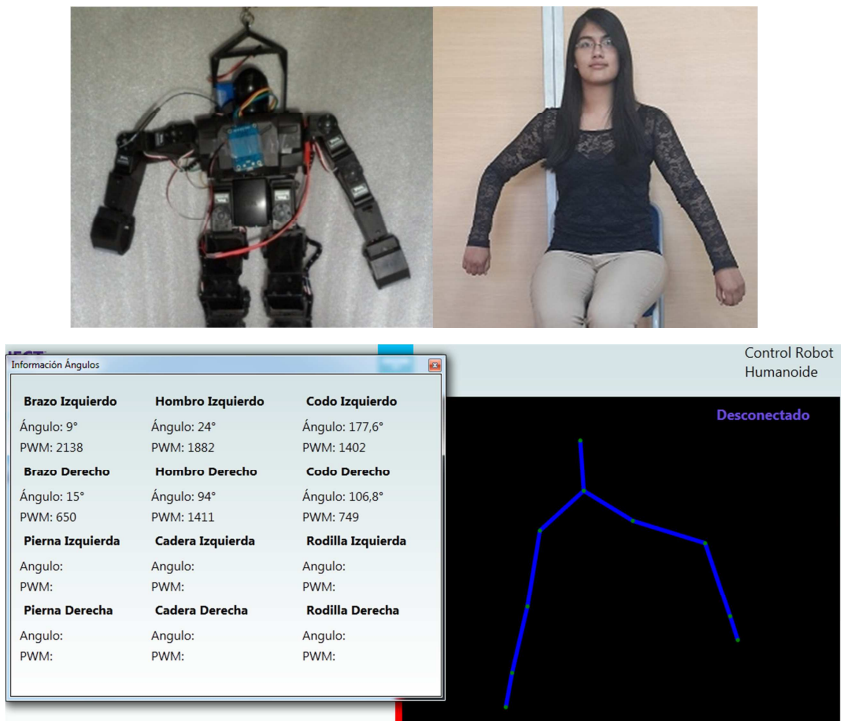


Figura 47. Posición para medición del brazo derecho
Imagen: Vladimir Araujo y Myriam Tipán

Tabla 11.
Ángulos brazo derecho

Articulación	Ángulo Usuario [°]	Ángulo Robot [°]	Ángulo Computador[°]	Variación 1 [°]	Variación 2 [°]
Brazo derecho	19,4	14,2	15,1	5,2	4,3
Hombro derecho	95,5	91,9	94	3,6	2,1
Codo derecho	115	109,6	106,8	5,4	2,8

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

En este caso la mayor variación es de 5,4 grados entre usuario y robot, mientras que la mínima variación es de 2,1 grados entre usuario y computador.

Medición 3 (pierna izquierda)

La tercera medición se la obtuvo de la pierna izquierda, abierta y con unos pocos grados hacia el frente. Ver figura 48.

Posición para medición de la pierna izquierda

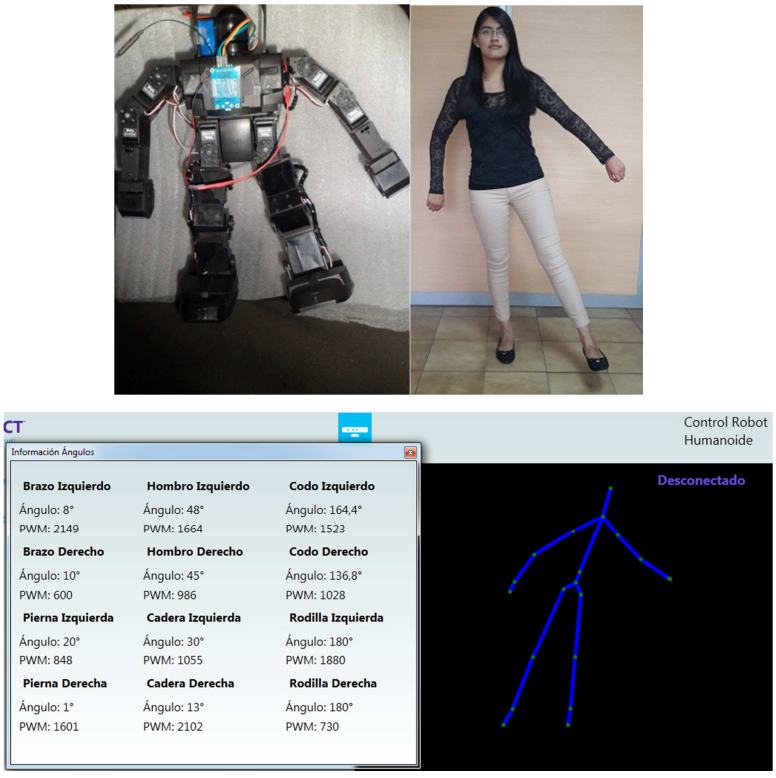


Figura 48. Posición para medición de la pierna izquierda
Imagen: Vladimir Araujo y Myriam Tipán

Tabla 12.
Ángulos pierna izquierda

Articulación	Ángulo Usuario [°]	Ángulo Robot [°]	Ángulo Computador[°]	Variación 1 [°]	Variación 2 [°]
Pierna izquierda	25,5	30	20,9	4,8	4,6
Cadera izquierda	35,1	40,7	30,2	5,6	4,9
Rodilla izquierda	170,2	165,9	180	4,3	9,8

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

Para la tercera medición la mayor variación es de 9,8 grados entre usuario y computador, y la mínima variación es de 4,3 grados entre usuario y robot.

Medición 4 (pierna derecha)

La última medición se la obtuvo de la pierna derecha, al igual que en la medición 3, abierta pero esta vez la rodilla fue recogida hacia atrás. Ver figura 49.

Posición para medición de la pierna derecha

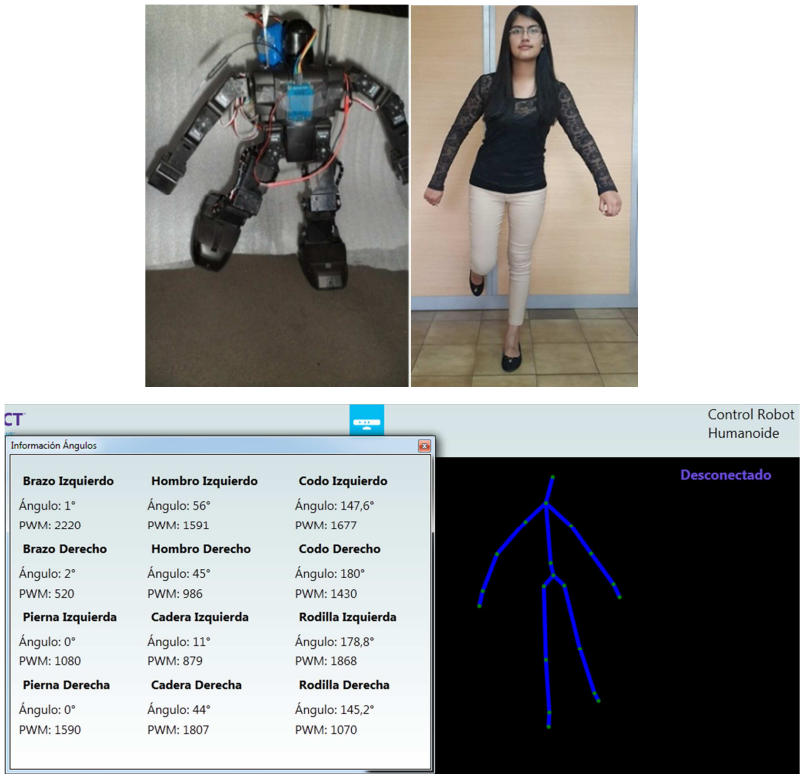


Figura 49. Posición para medición de la pierna derecha
Imagen: Vladimir Araujo y Myriam Tipán

Tabla 13.
Ángulos pierna derecha

Articulación	Ángulo Usuario [°]	Ángulo Robot [°]	Ángulo Computador[°]	Variación 1 [°]	Variación 2 [°]
Pierna derecha	8,9	5,2	0	3,7	8,9
Cadera derecha	25,9	30,1	34	4,2	8,1
Rodilla derecha	125,9	130	135,2	4,1	9,3

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

Finalmente en la cuarta medición la mayor variación es de 9,3 grados entre usuario y computador, y la mínima variación es de 3,7 grados entre usuario y robot.

En la tabla 9 y 10 se obtuvo las mayores variaciones entre los valores de usuario y robot y las menores entre usuario y computador, esto debido a que las posiciones extremas del robot no llegan a ser cien por ciento correctas y se producen una variación mínima.

Con respecto a la tabla 11 y 12, en estas se obtuvo una variación mayor entre valores de usuario y computador, esto se debe a que en algunas posiciones de la pierna el Kinect pierde de vista puntos pero realiza una estimación de la ubicación dando un valor no válido.

5.2 Pruebas en modo controlar

En el modo controlar se optó por tomar mediciones del tiempo que el robot demora en responder al comando de voz, además se toma porcentajes de confianza respecto a la coincidencia que tienen los comandos dichos por el usuario y los ya establecidos en la aplicación, por último se agrega una trayectoria predefinida para comprobar el desempeño del robot bípedo humanoide.

5.2.1 Medición modo mover

En los resultados que se muestra en la tabla 14 se observa cinco tomas de muestras de tiempo para cada respuesta a un determinado comando de voz, los resultados muestran que el menor valor promedio de tiempo se presenta en el comando cerrar mano izquierda con un valor de 1,24 segundos, esto se debe a que el grado de

confiabilidad de las palabras dichas por el usuario y los ya establecidos en la aplicación fueron altos , sin embargo para el comando abrir mano derecha el valor promedio de tiempo es de 1,32 segundos, siendo así el mayor tiempo obtenido, teniendo como variación máxima 1,4 segundos, esto se debe a que se generan retardos debido al reconocimiento, proceso y envió de la trama.

Se puede decir de la tabla 14, que la respuesta que el robot tiene a los demás comandos de voz varía entre un rango de 1,24 a 1,32 segundos.

Tabla 14.
Valores de tiempo en modo mover

Comando	T1 [s]	T2 [s]	T3 [s]	T4 [s]	T5 [s]	Promedio
Abrir mano derecha	1,3	1,34	1,3	1,26	1,4	1,32
Abrir mano izquierda	1,35	1,3	1,22	1,3	1,26	1,29
Cerrar mano derecha	1,31	1,23	1,2	1,35	1,39	1,30
Cerrar mano izquierda	1,17	1,39	1,34	1,14	1,18	1,24
Girar mano izquierda	1,27	1,3	1,3	1,3	1,26	1,29
Girar mano derecha	1,23	1,18	1,3	1,22	1,32	1,25

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

El nivel de confianza se refiere a la probabilidad de que una frase reconocida por el método `speechRecognizer` coincida con un comando ya establecido en la aplicación, este nivel de confianza toma un valor aceptable a partir de 0.7, el rango de probabilidad va de 0.0 a 1.0, de menor a mayor confianza, respectivamente.

Los resultados de la tabla 15 tienen datos referidos al valor de confianza, en donde se observa que el comando girar mano izquierda posee la mayor probabilidad y es de 0,98732, esto significa que la frase es más probable que sea la coincidencia correcta para el comando ya establecido que con otros resultados que tienen valores de confianza de menos de 0, 98732. Además se tiene que todos los demás comandos ya establecidos en la aplicación tienen un grado de confianza no menos al 0,7, por lo tanto el grado de confianza es adecuado.

Tabla 15.
Valores de confianza en modo mover

Comando	Valor 1	Valor 2	Valor 3	Valor 4	Valor 5	Promedio
Abrir mano derecha	0,9698	0,9818	0,9448	0,9691	0,9891	0,97092
Abrir mano izquierda	0,9721	0,9697	0,9006	0,9982	0,9897	0,96606
Cerrar mano derecha	0,9911	0,8465	0,9191	0,8704	0,9756	0,92054
Cerrar mano izquierda	0,9901	0,9667	0,9435	0,9921	0,9984	0,97816
Girar mano izquierda	0,9967	0,9541	0,9955	0,9921	0,9982	0,98732
Girar mano derecha	0,9977	0,9912	0,7345	0,9924	0,9804	0,93924

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

5.2.2 Medición modo maniobrar

A partir de los resultados que se observan en la tabla 16, se analiza que de las cinco pruebas realizadas, el comando giro izquierda tiene un valor de tiempo promedio de 2,34 segundos siendo así el mayor tiempo obtenido, teniendo como variación máxima de tiempo 2,41 segundos, por otro lado, el comando atrás tuvo un promedio de tiempo menor de 2,21 segundos.

Los datos de la tabla 16 en comparación con los datos de la tabla 14 tienen un valor de tiempo mayor, debido a que en este modo la aplicación realiza más procesos, como son la obtención de los puntos del esqueleto a través del sensor Kinect necesarios para el cálculo de cinemática inversa.

Tabla 16.
Valores de confianza en modo maniobrar

Comando	T1 [s]	T2 [s]	T3 [s]	T4 [s]	T5 [s]	Promedio
Adelante	2,11	2,3	2,12	2,33	2,25	2,22
Atrás	2,33	2,12	2,4	2,08	2,13	2,21
Paso Derecha	2,43	2,3	2,22	2,29	2,21	2,29
Paso Izquierda	2,12	2,43	2,08	2,21	2,26	2,22
Giro Derecha	2,04	2,3	2,27	2,16	2,32	2,22
Giro Izquierda	2,31	2,3	2,41	2,34	2,34	2,34

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

Los valores promedios de probabilidad referente al grado de confianza que se muestran en la tabla 17 fluctúan en un rango de 0,82032 a 0.99836, esto significa que la frase dicha por el usuario es más probable que sea la coincidencia correcta para el comando ya establecido en la aplicación.

Tabla 17.
Valores de confianza en modo maniobrar

Comando	Valor 1	Valor 2	Valor 3	Valor 4	Valor 5	Promedio
Adelante	0,8798	0,8384	0,7584	0,8212	0,8038	0,82032
Atrás	0,9272	0,9745	0,9317	0,9888	0,7445	0,91334
Paso derecha	0,9896	0,9628	0,9988	0,9953	0,9928	0,98786
Paso izquierda	0,9986	0,9973	0,9992	0,9984	0,9983	0,99836
Giro derecha	0,9598	0,9809	0,9974	0,9932	0,9988	0,98602
Giro izquierda	0,9817	0,9988	0,976	0,9991	0,9685	0,98482

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

De los resultados de confianza que se muestran en la tabla 14 y 16 se tiene que los valores de probabilidad promedio tanto para el control en modo maniobrar y mover son altos, esto se debe a que el control de voz realizado en la aplicación es independiente del envió de tramas usado para el control del robot.

5.2.3 Prueba de desempeño del robot

Para determinar el desempeño del robot al ser controlado por voz se optó por realizar una trayectoria la cual culmina cuando el robot se encuentra en posición de manipular un objeto, una vez en posición el usuario procederá a maniobrar el robot para tomar el objeto, luego el usuario deberá arrojar el objeto a un recipiente y de esta manera se completa la prueba.

Iniciada la conexión entre la aplicación y el kinect, el usuario procede a seleccionar mediante comandos de voz el modo mover, desplegando las siguientes opciones:

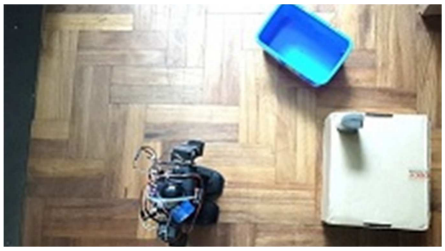
- Adelante
- Atrás
- Derecha
- Izquierda

En la figura 50-a el robot se encuentra en posición incial, luego el usuario pronuncia la opción adelante 5, esto quiere decir que el robot dara 5 movimientos hacia adelante, cada movimiento equivale a 2 pasos, por lo tanto al robot le tomo 10 pasos para llegar a la posición que se observa en la figura 50-b, ademas se observa una ligera desviación en la trayectoria seguida por el robot, para esto el usuario expresa la opción giro izquierda corriguiendo asi la trayectoria esto se puede observar en la figura 50-c, acontinuación el usario pronuncia el comando adelante 2 seguido del

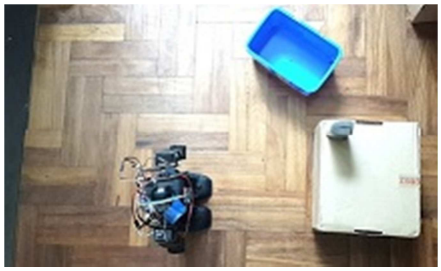
comando adelante 1, como se observa en las figuras 50-d y 50-e, en este punto se aprecia que los comandos son almacenados en el bufer del controlador realizando todo el proceso de los 3 movimientos hacia adelante, es decir el robot se desplaza hacia adelante mediante 6 pasos, finalmente para lograr poner al robot en posición de manipular el objeto, el usario escoge el comando paso izquierda de esta manera el robot queda en la posición deseada como se aprecia en la figura 50-f.

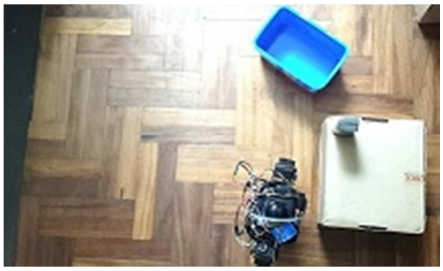
Robot en modo controlar – mover

a) b)

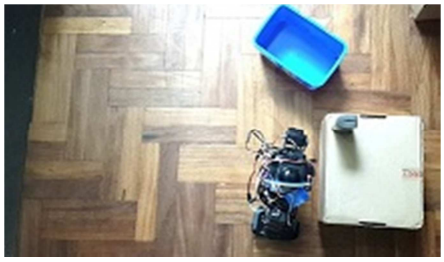


c) d)





e) f)



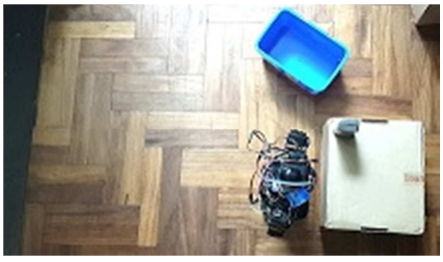


Figura 50. Control por voz – trayectoria de prueba
Elaborado por: Vladimir Araujo y Myriam Tipán

Una vez que el robot está en posición, el usuario mediante comandos de voz escoge el modo maniobrar, este modo utiliza la cinemática inversa para determinar los valores que deben adoptar las coordenadas articulares del robot, dependiendo del movimiento que el usuario realice, en la figura 51-a y 51-b se aprecia que el usuario maniobra correctamente al robot para colocar el brazo izquierdo sobre el objeto, en ese momento el usuario realiza la petición del comando de voz girar mano izquierda esto se observa en la figura 51-c, una vez que robot se encuentra en posición de tomar el objeto el usuario expresa abrir mano izquierda, seguido del comando cerrar mano izquierda, esto se aprecia en figura 51-d y 51-e, finalmente el objeto es colocado en el recipiente bajo el comando abrir mano izquierda como se observa en

la figura 51-f y 51-g , además se observa que el robot replica los movimientos que el usuario realiza manteniendo fija su posición desde el dorso hasta los pies , en conclusión se puede decir que se cumple el objetivo de la prueba de desempeño mediante el uso de los comandos de voz para lograr el desplazamiento por una superficie plana y replicar los movimientos del usuario para colocar el objeto definido en el recipiente.

Robot en modo controlar - maniobrar

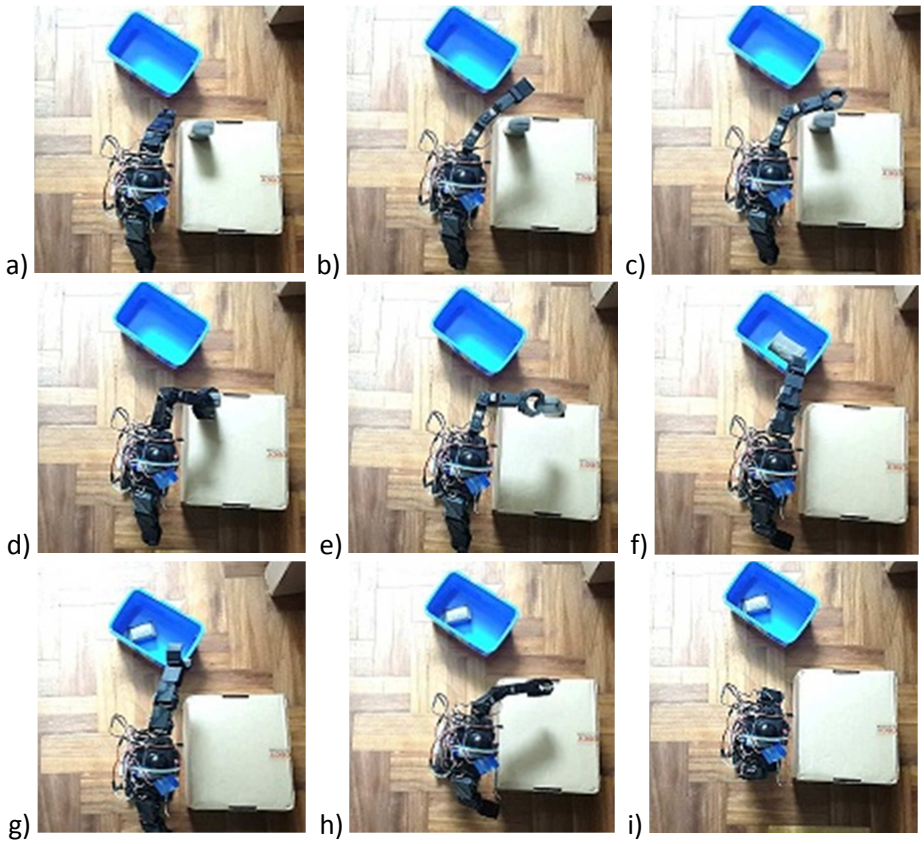


Figura 51. Modo maniobrar – traslado del objeto
Elaborado por: Vladimir Araujo y Myriam Tipán

5.3 Análisis financiero

5.3.1 Costo de materiales

Tabla 18.
Listado de costo de materiales

Cantidad	Descripción	Precio Unitario	Precio Total
1	Plataforma RoboPhilo	600	600
2	Grippers	75	150
1	Controlador de Servos	80	80
3	Baterías	30	90
1	Módulo WiFi	50	50
		TOTAL	970

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

5.3.2 Costo de ingeniería

Para el presente proyecto se asume un costo de mano de obra de $ 400 mensuales por cada autor, el total del proyecto demoró 8 meses de trabajo continuo de 10 horas semanales, por lo tanto el valor total de mano de obra es de $ 6.400.

5.3.3 Costo total

Tabla 19.
Listado total de costos

Rubro	Valor
Costo de Materiales	970
Costo de Ingeniería	6.400
TOTAL	7.370

Nota. Elaborado por: Vladimir Araujo y Myriam Tipán

CONCLUSIONES

- El sensor Kinect además de ser un buen complemento para los videojuegos ha demostrado ser un dispositivo con grandes características al adquirir señales de su entorno de manera integral, lo que permite el desarrollo de nuevas aplicaciones en la robótica como es la asistencia a personas.

- Los datos obtenidos del esqueleto desde el sensor Kinect tienen un porcentaje muy alto de fiabilidad por lo cual la réplica de movimientos en el robot logra ser bastante precisa con una variación máxima de 11,1 grados que representa el 6,16% del movimiento total de los motores del robot; estos valores de error podrían ser mejorados usando un robot más especializado.

- La resolución de cinemática inversa por medio de métodos geométricos es la más adecuada para esta aplicación debido a que se puede obtener una respuesta en tiempo real, se acerca mucho a la deseada y no requiere de mucho procesamiento del computador en comparación con métodos iterativos o el uso de matrices.

- El controlador original del robot posee tecnología cerrada por lo que fue reemplazado por el controlador USC-32 que es la mejor opción debido al costo y características, al ser un dispositivo esclavo pre programado fue posible controlar el robot y generar secuencias de movimiento de manera sencilla.

- El sensor Kinect en conjunto con el SDK de reconocimiento de voz de Microsoft permitieron crear una interacción mucho más amigable entre el usuario y el sistema, ya que todas las funciones son controladas mediante comandos de voz, permitiendo a los usuarios con discapacidad en sus extremidades inferiores controlar el sistema sin necesidad de estar de pie.

- Para la comunicación inalámbrica se escogió como mejor opción al módulo WiFi HLK-RM04 debido a su costo, velocidad y alcance de hasta 100 metros, a diferencia de otros dispositivos como el bluetooth que tienen poco alcance de transmisión y el módulo Xbee que es más costoso ya que se necesita adquirir un transmisor y receptor.

- De las pruebas realizadas en modo controlar, se puede concluir que la implementación de reconocimiento de voz para mover al robot es satisfactoria puesto que el robot se desplazó en una trayectoria recta y a pesar de las pequeñas desviaciones en la trayectoria, esta fue corregida por el usuario enunciando otros movimientos para de esta manera completar la actividad.

- Se concluye que el sistema implementado permite al usuario ser asistido por el robot, sin embargo existen limitaciones debido a que la plataforma usada no cuenta con sensores que proveen de autonomía, también se puede decir que su tamaño es pequeño, y los movimientos no son finos.

- De los valores de confianza que se muestra en la tabla 15 y 17 se puede concluir que el arreglo de micrófonos en conjunto con el SDK de reconocimiento de voz son de alto rendimiento debido a que los valores de confianza fluctúan en un rango de 0,82032 a 0.99836, esto significa que la frase dicha por el usuario es más probable que sea la coincidencia correcta para el comando ya establecido en la aplicación.

- En la tabla 14 y 16 se tiene que los valores de probabilidad promedio tanto para el control en modo maniobrar y mover son altos, esto se debe a que el control de voz realizado en la aplicación es independiente del envió de tramas usado para el control del robot.

- Visual Studio y el SDK de Kinect son propiedad de Microsoft por lo que se logra una excelente integración del sensor con el computador, y adicional el uso del lenguaje C# permitió crear una aplicación amigable y un buen enlace con el robot.

RECOMENDACIONES

- Es recomendable mantener la red WiFi del robot conectada únicamente al computador cliente ya si se conecta a otro dispositivo el robot dejaría de ser controlable.

- Se recomienda colocarse con la mirada frente al sensor Kinect para que este pueda captar correctamente los comandos de voz, adicional evitar ruido externo ya que esto afecta al reconocimiento de los comandos.

- Es necesario que ningún otro usuario se cruce frente al sensor kinect durante la manipulación del robot ya que esto confunde al sensor solapando puntos del esqueleto y haciendo que el robot tenga inestabilidad.

- De preferencia se recomienda que se enfoque totalmente el cuerpo del usuario esto para evitar movimientos en falso que también se transmitirían al robot.

- En el modo controlar la réplica de movimientos es únicamente de extremidad superiores por lo que es recomendable usar el modo sentado ya que así se podrá maniobrar adecuadamente y el usuario podrá estar más cómodo.

- Se recomienda que el robot sea usado en superficies ligeramente lisas para que pueda desplazarse correctamente y es obligatorio que la superficie sea totalmente plana y sin inclinación.

- En cuanto a los requerimientos de hardware, se recomienda el uso de un computador de mínimo 1gb de RAM y procesador CORE 2 DUO.

LISTA DE REFERENCIAS

Telemanipuladores Argentinos para el mundo. (6 de Mayo de 2012). Recuperado el 1 de Octubre de 2014, de http://www.tomamateyavivate.com.ar/tecnologia-argentina/telemanipuladores-argentinos-para-el-mundo/

Abrego, M. (7 de Abril de 2014). *Usando la cámara del Kinect como cámara fotográfica*. Recuperado el 26 de Octubre de 2014, de https://malenyabrego.wordpress.com/2014/04/07/usando-la-camara-del-kinect-como-camara-fotografica/

Aldebaran. (s.f.). *Romeo*. Recuperado el 2014 de Octubre de 25, de http://www.aldebaran.com/en/robotics-company/projects

Aldebaran. (s.f.). *Who is NAO?* Recuperado el 12 de Octubre de 2014, de http://www.aldebaran.com/en/humanoid-robot/nao-robot

Aldebaran. (s.f.). *Who is Pepper?* Recuperado el 5 de Octubre de 2014, de http://www.aldebaran.com/en/a-robots/who-is-pepper

Barrientos, A. (2007). *Fundamentos de Robótica.* España: S.A. MCGRAW-HILL.

Borenstein, G. (2012). *Making Things See.* Canada: O'Reilly Media, Inc.

Catuhe, D. (2012). *Programming with the Kinect for Windows SDK.* Estados Unidos: Microsoft Press.

Craig, J. J. (2006). *Robótica.* México: Prentice Hall.

Fu, K. S., & González, R. (1988). *Robótica: control, detección, visión e inteligencia.* MCGRAW-HILL.

infobae. (21 de Julio de 2014). *"Asimo", la nueva versión del robot que toma decisiones*. Recuperado el 5 de Octubre de 2014, de http://www.infobae.com/2014/07/21/1582228-asimo-la-nueva-version-del-robot-que-toma-decisiones

Jana, A. (2012). *Kinect for Windows SDK Programming Guide.* United Kingdom: Packt Publishing.

Ollero, A. (2001). *Robótica Manipuladores y Robots Móviles.* España: S.A. MARCOMBO.

Ramos, E., Castro, C., & Jaworski, P. (2012). *Arduino and Kinect Projects.* Estados Unidos: Apress.

RoboBrothers. (2007). *RoboPhilo User Guide.* Estados Unidos.

RoboBrothers. (2012). *RoboPhilo Gripper Manual.* Estados Unidos.

Robotis. (s.f.). *DARwIn-OP*. Recuperado el 12 de Octubre de 2014, de http://www.robotis.com/xe/darwin_ko

Royer, E. (16 de Noviembre de 2011). *The World Models RoboPhilo*. Recuperado el 31 de Octubre de 2014, de http://www.botmag.com/the-world-models-RoboPhilo/

Santana, P. (Abril de 2014). *Interfaces Naturales de Usuario - La Experiencia de la Universidad de Colima*. Recuperado el 20 de Octubre de 2014, de http://sg.com.mx/revista/43/interfaces-naturales-usuario-la-experiencia-la-universidad-colima#.VE2RJiKG-WY

Sharp, J. (2010). *Microsoft® Visual C#® 2010.* Estados Unidos: Microsoft Press.

Torobot. (2014). *Servo Controller Instructions for Use.* China.

Webb, J., & Ashley, J. (2012). *Beginning Kinect Programming with the Microsoft Kinect SDK.* Estados Unidos: Apress.

ANEXOS

ANEXO 1. Datasheet módulo WiFi HLK-RM04

Pin Configurations

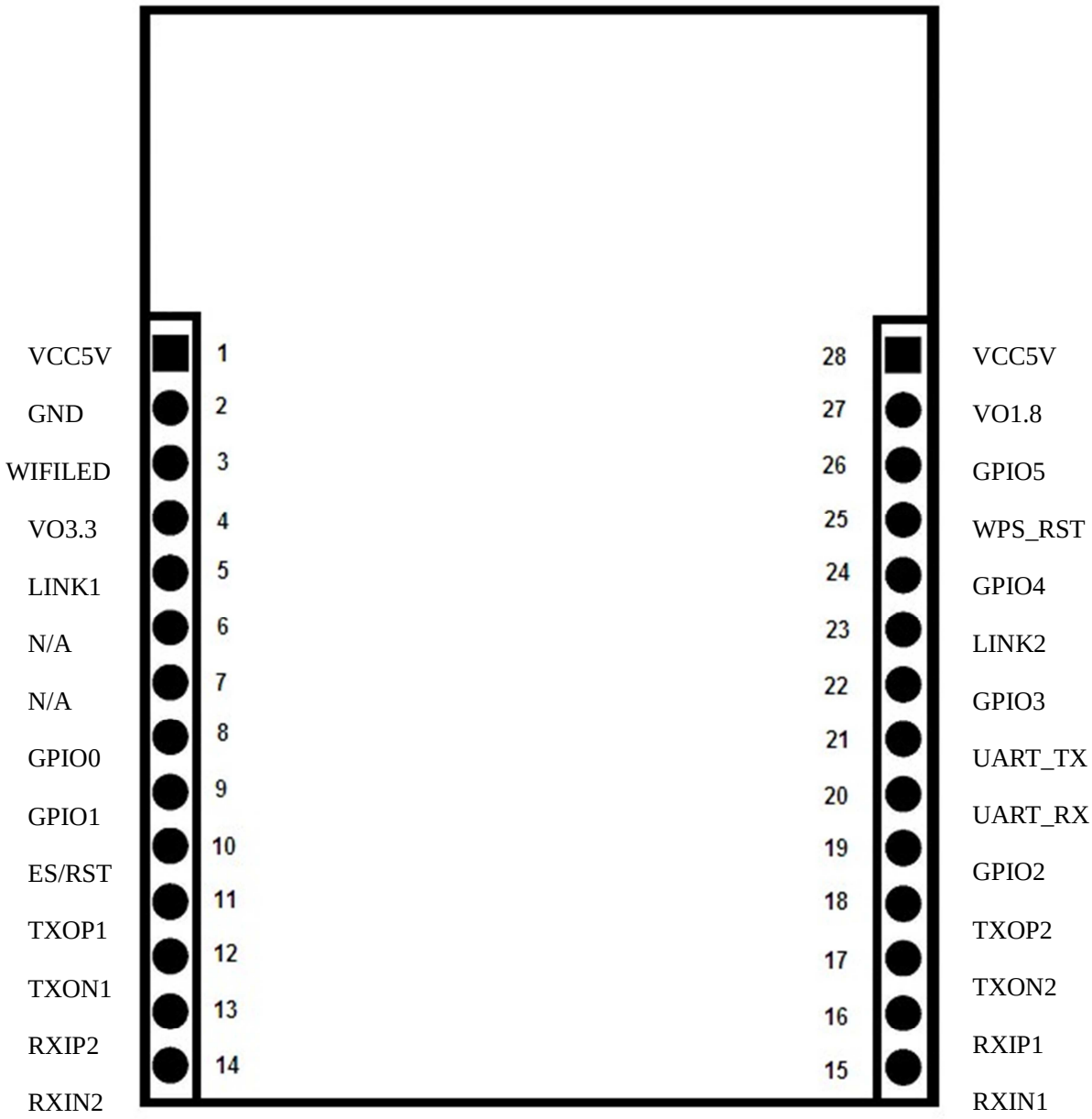


Figure 1: Pin Configuration (top view)

Pin Assignment

Pin No	Signal Type	Description
1	VCC5V	Supply Voltage, 5V+/-10%
2	GND	Analogue Ground
3	WIFILED	WLAN Activity LED
4	VO3.3	3.3V Output (Suuport Atmost 300mA)
5	LINK1	10/100 PHY Port #1 activity LED
6	N/A	Reserved
7	N/A	Reserved
8	GPIO0	General GPIO Reserved
9	GPIO1	General GPIO Reserved
10	ES/RST	Exit transparent transmission mode/Restore factory value
11	TXOP1	10/100 PHY Port #1 TXP
12	TXON1	10/100 PHY Port #1 TXN
13	RXIP2	10/100 PHY Port #2 TXP
14	RXIN2	10/100 PHY Port #2 TXN
15	RXIN1	10/100 PHY Port #1 RXN
16	RXIP1	10/100 PHY Port #1 RXP
17	TXON2	10/100 PHY Port #2 OXN
18	TXOP2	10/100 PHY Port #2 OXP
19	GPIO2	General GPIO Reserved
20	UART_RX	UART RXD.
21	UART_TX	UART TXD.
22	GPIO3	General GPIO Reserved
23	LINK2	10/100 PHY Port #2 activity LED
24	GPIO4	General GPIO Reserved
25	WPS/RST	WiFi Protected Setup /Restore factory value
26	GPIO5	General GPIO Reserved
27	VO1.8	1.8V Output (Suuport Atmost 300mA)
28	VCC5V	Supply Voltage, 5V+/-10%

Electrical Characteristics

Parameter	Min	Max
Module supply voltage VCC	3.9V	5.5V
Module Voltage Output	3.1V	3.5
Module Voltage Output	1.65V	1.9
GPIO Voltage	3.1V	3.5V
Storage temperature	-40ºC	95ºC

ANEXO 2. Datasheet plataforma RoboPhilo

RoboPHILO has been designed carefully for safety purposes. However, any electrical device, if used improperly, has the potential for causin fire, electrical shick or personal injury. TO help ensure safe operation, follow these guidelines:

- Observe all warnings, precautions and instructions.
- Regularly inspect the battery pack for damage
- Stop use, unplug the charger and disconnect the battery pack immediately if PHILO functions in an abnormal manner, produces unusual sounds or smells or becomes too hot to touch.

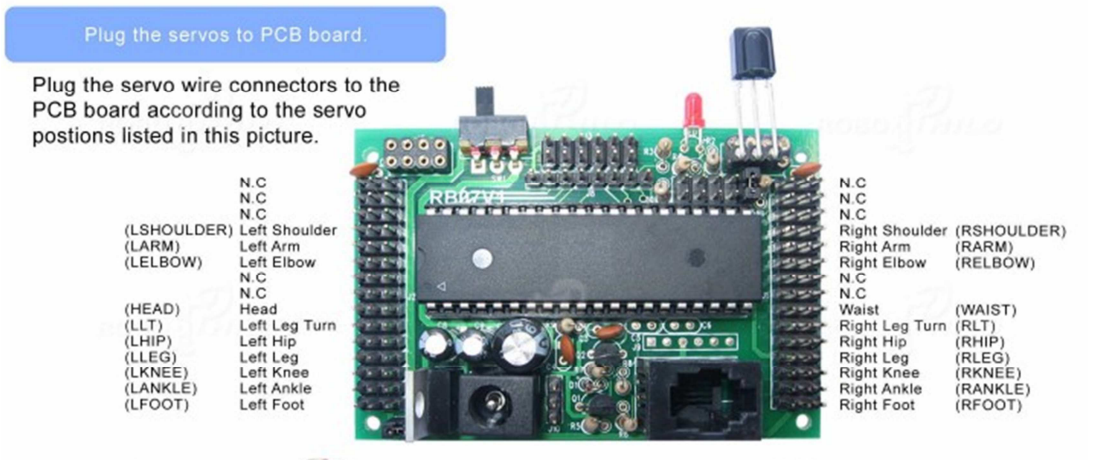
Features:

1. 20 Servos for 20 degree of freedom including head, waist, leg turning, arms and legs.
2. Each motion routine can have up to 30 sequences, and each sequence can have up to 15 poses.
3. Sequence and pose can be reused for other motions to save flash memories.
4. One motion routine can have up to 450 pose transitions.
5. RS232 serial connection to PC for motion programming and execution.
6. InfraRed Hand held remote executes the user created program motions.

Hardware Features

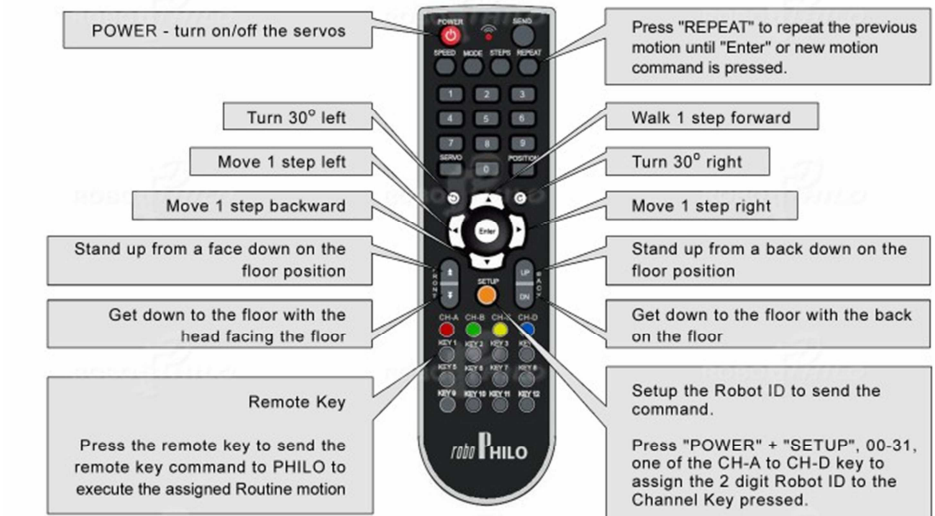
Controller:

- ATmega32-16PU
- Interrupt Driven kernel for InfraRed Remote and Servos handling
- 24 Servos Channels
- Up to 8 I/O interface fo add on hardware
- 8Kb flash for more than 300 user motions



InfraRed Remote:

- Can play 121 user defined motion routines
- Control up to 4 RoboPhilo independently



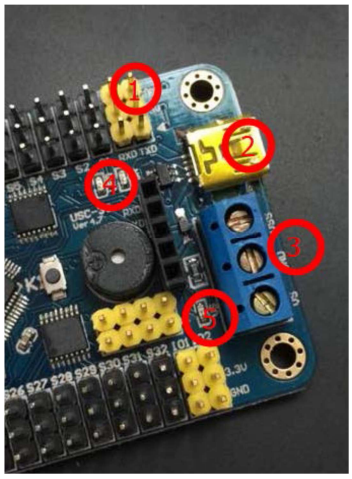
Power:

- Charger can be used for motion programming while charging the battery to extend the time of play
- Provide regulated 5VDC and unregulated 6V – 7,2V DC for add on hardware

ANEXO 3. Datasheet servo controller

Power Supply

The servo controller needs two power supplies: servo power supply and chip power supply.



- Servo power supply (+): VS (left of the blue connecting terminal at Position 3 in the figure)
- Servo power supply (-): GND (middle of the blue connecting terminal at Position 3 in the figure)
- Servo power supply's parameters depend on the parameters of the attached servo. For example, if the TR213 servo has a power supply of 4.8-7.2V, the servo power supply can use the power source of 4.8-7.2V.
- Chip power supply (+): VSS (right of the blue connecting terminal at Position 3 in the figure)
- Chip power supply (-): GND (middle of the blue connecting terminal at Position 3 in the figure)
- There is a VSS requirement of 6.5-12V. If the chip power is input through the VSS port, the power supply has to range from 6.5 to 12V.

Notes:

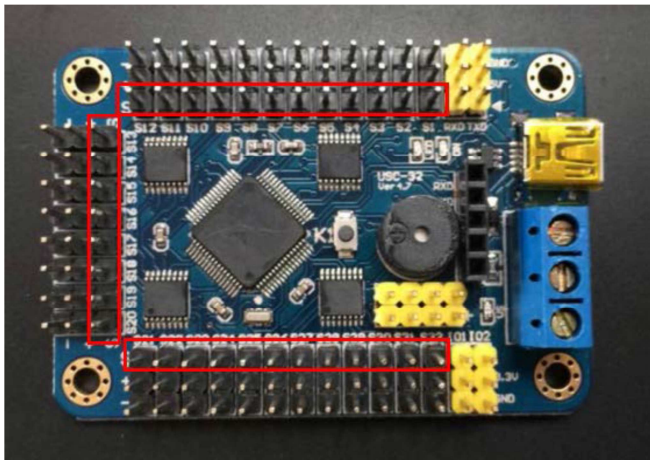
- 1. The USB port at Position 2 in the figure can supply power to the chip. So it is adequate to choose the USB port or alternatively the VSS port.

- 2. Position 1 in the figure can supply power to the chip as well, marked 5V and GND, where 5V is the anode and GND is the cathode. The power supply has to be 5V.

- 3. Positions 1, 2 and 3 can supply power to the chip. It is adequate to choose any of them.
- 4. The green LED light at Position 4 in the figure is the chip power indicator. If the green light is on, it indicates the chip power works correctly; if the light is off, it indicates the chip power malfunctions.

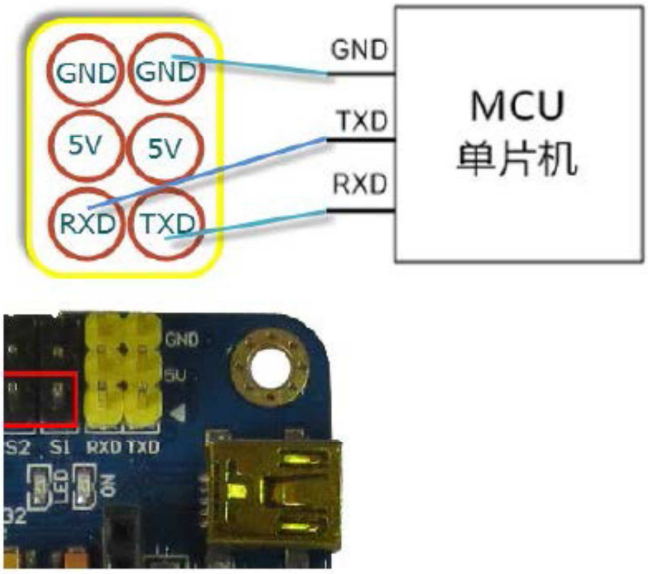
- 5. The green LED light at Position 5 in the figure is the servo power indicator. If the green light is on, it indicates the servo power works correctly; if the light is off, it indicates the servo power malfunctions.

Connect the servo to the servo controller



- What is marked red in the figure are the servo's connectors for signal wires (be careful about the direction when connecting to the servo).
- What is marked yellow in the figure are not the servo's connectors.
- Pay attention to the white textual symbols aside when connecting to the servo. For example, S1, S2,…, S32 refers to the servo channels that correspond to the computer software.

Connect to the MCU



Secondary development

- The servo controller is a slave device, meaning that it can either accept commands or execute preset commands. It cannot think at all.
- Communication protocol: serial communication (TTL level), baud rate 9600, no check bit, 8 data bits, 1 stop bit
- To control the servo through the servo controller, users can self-develop computer software or use the MCU to send commands to the servo controller.

Command format: Name	Command	Description
Control one single servo	#1P1500T100\r\n	Data 1 refers to the servo's channel Data 1500 refers to the servo' location, in the range 500-2500 Data 100 refers to the time of execution and represents the speed, in the range 100-9999
Control multiple servos	#1P600#2P900#8P2500T100\r\n	Data 1, 2, and 8 refer to the servo's channels Data 600, 900, and 2500 refer to the locations of the servos that correspond to three channels Data 100 refers to the time of execution and represents the speed of three servos. Regardless of the number of servos, there is only one time, or one T. The command is executed at the same time; that is, all servos operate simultaneously.
Execute one single action group	#1GC2\r\n	Data 1 refers to the serial number of the action group Data 2 refers to the number of cycles
Execute multiple action groups	#1G#3G#1GC2\r\n	Execute the first, third and first action group,The number of cycles is 2. One particular group of action can appear repeatedly. There can be only one number of cycles or C. The command is executed in sequence; that is, the action groups are executed in sequence.