Pontificia Universidad Católica de Valparaíso
Facultad de Ingeniería
Escuela de Ingeniería Informática

DESARROLLO SISTEMA DE CONTROL ROBÓTICO MEDIANTE KINECT

AGUSTÍN NICOLÁS ROMERO SALAZAR

INFORME FINAL DE PROYECTO
PARA OPTAR AL TÍTULO PROFESIONAL DE
INGENIERO DE EJECUCIÓN EN INFORMÁTICA

Marzo 2013

Pontificia Universidad Católica de Valparaíso
Facultad de Ingeniería
Escuela de Ingeniería Informática

DESARROLLO SISTEMA DE CONTROL ROBÓTICO MEDIANTE KINECT

AGUSTÍN NICOLÁS ROMERO SALAZAR

Iván Mercado Bermúdez
Profesor guía

Wenceslao Palma Muñoz
Profesor Co-referente

Marzo 2013

Dedicatoria

A mi familia, amigos y en especial a los que siempre creyeron…

Agradecimientos

A mi familia, por ser el pilar fundamental para desarrollarme como persona y profesional a lo largo de mi existencia, gracias por todo el apoyo brindado a lo largo de este largo camino universitario, sin Uds. ninguna de mis metas serían cumplidas.

A mis amigos, por entregarme ese amor incondicional día a día. Gracias por su apoyo y por levantarme tantas veces cuando lo necesite, sin Uds. es imposible obtener este logro.

A mis profesores , que a lo largo de la carrera me han entregado todas las herramientas necesarias para ser un profesional, agradecer los valores inculcados durante este largo proceso y el apoyo brindado en momentos difíciles.

Resumen

Kinect en la actualidad es una de las tecnologías más revolucionarias e innovadoras del último tiempo. Este controlador nació para la manipulación de manera natural sobre los videojuegos de la consola Xbox 360. Hoy, esta tecnología, no solo es utilizada para la industria de los videojuegos, su uso se ha expandido a diversas ramas como la educación, medicina, sistemas de seguridad, sistemas informáticos, transporte y más allá.

De esta forma se plantea la creación de un prototipo de interacción natural entre una persona y un robot, donde éste imite los movimientos que realiza la persona, todo esto será posible gracias al controlador Kinect mediante un código fuente, que se realizará para el éxito de este estudio.

Abstract

Nowadays Kinect is one of the most revolutionary and innovative technologies in recent times. This driver was born to handle naturally on the video games Xbox 360 console. Now this technology not only used to the videogames industry, its use has spread to others branches, such as education, medicine, security systems, computer systems, transport and beyond.

This would result in the creation of a prototype natural interaction between a person and a humanoid robot, where the robot mimics the movements done by one person.

All these will be possible thanks to driver source kinect to be held for the success of this study.

Índice

Índice de figuras y tablas

Capítulo I: Introducción

1.1 Motivación.

En el año 2009 Microsoft Research [Jarret Webb, 2011] anuncia que luego de 20 años de estudio finalmente se lanzará al mercado una nueva forma de interactuar con los videojuegos, una manera distinta a lo que se venía utilizando hasta ese entonces, donde la persona interactuaba con el videojuego por medio de un control. Esta nueva forma de interactuar con un videojuego corresponde a un sensor llamado Kinect que según sus creadores Microsoft, corresponde a "*Un controlador de juego libre y de entretenimiento desarrollado por Microsoft para la videoconsola Xbox 360. Kinect permite a los usuarios controlar e interactuar con la consola sin necesidad de tener un contacto físico con un controlador de videojuegos tradicional, mediante una interfaz natural de usuario que reconoce gestos, comandos de voz, y objetos e imágenes*" [Microsoft, 2012].

Sin embargo el uso de este controlador puede ir más allá y no solo encasillarse en un género como el de los videojuegos, sino que, ser un real aporte para la ciencia y las diversas ramas que esta posee.

Para efectos de este proyecto se centrará en lo que respecta a la Robótica .Ésta es una disciplina que consta de 2 vertientes: teórica y práctica. En el aspecto teórico se aúnan la automática, informática e inteligencia artificial. En el lado práctico o tecnológico hay aspectos de construcción (mecánica, electrónica).

Según el Checo ´Robota´ significa servidumbre o trabajador forzado , en el momento cuando se tradujo al inglés se convirtió en el término "robot", como se trata de que el robot cumpla labores es necesario que estos robots imiten las articulaciones del cuerpo humano para realizarlas de manera más natural.

Este estudio está enfocado principalmente en que la robótica manipulada, especialmente por interacción natural. Puede aplicarse a un sin fin de objetivos y distintos campos, ya sea como estudio o ser una ayuda para estos. Así por ejemplo , gracias a un robot humanoide más el sensor Kinect de la videoconsola Xbox360, se puede crear un estudio de reconocimiento de los movimientos que puedan realizar personas discapacitadas, a través del robot y de los resultados que arroja el software de Kinect, que nos muestra detalladamente el movimiento de las articulaciones de la persona, es decir, la persona que se encuentra en tratamiento, se le indica una rutina de ejercicios a realizar, el sensor kinect capturará los movimientos y manipulará el robot , permitiendo al discapacitado realizar dichos trabajos de una manera mucho más didáctica y a la vez permitir a los expertos analizar los datos que arroja el sensor

por medio del software instalado en el equipo. Esto mismo se puede llevar a cabo en el tratamiento médico de personas de la tercera edad, se pueden detectar avances, retrocesos e incluso caídas en el tratamiento.

Por, último, y en otra rama de estas tecnologías pueden ser aplicadas perfectamente para ámbitos militares, donde por medio de interacción natural, es decir, sin la necesidad de algún control remoto, se pueden controlar los robots para el uso que se estime conveniente dentro de este campo.

1.2 Objetivo General

El objetivo principal del proyecto consiste en implementar una solución que permita controlar un robot mediante el sensor Kinect de Microsoft.

1.3 Objetivos Específicos

Una vez definido el objetivo general del proyecto, éste se descompone de los siguientes objetivos específicos:

- Prototipo de captura de datos de las articulaciones del usuario: Es necesario representar los datos para poder controlar los datos con precisión.
- Prototipo de interpretación de datos para los movimientos del robot: Es necesario comprobar en una primera instancia los movimientos del robot y que coincidan con el simulador del esqueleto.
- Aplicación final que utilice las funciones de los prototipos para que cumpla el objetivo general del estudio.
- Validación y verificación del prototipo final de la aplicación desarrollada.

1.4 Metodología.

El concepto de metodología hace referencia al plan de investigación que se llevará a cabo para finalmente cumplir los objetivos planteados .Esta es una herramienta que permitirá Identificar, analizar, recopilar la información. Para efectos de este estudio se pretende utilizar la metodología de investigación científica clásica. Esta metodología tiene 3 características principales. Es sistemática, es decir, planificada, metódica, no hay nada al azar; empírica, se recolectan y se analizan los datos de la investigación y además crítica donde constantemente se está evaluando y mejorando.

La metodología científica consta principalmente de las siguientes etapas:

- En una primera etapa se debe tener una concepción de la idea central del tema a investigar, luego se debe plantear el problema a desarrollar por medio de los objetivos, cuestionamientos asociados y justificaciones referentes a la viabilidad del problema. En el apartado anterior se identifican los objetivos para este estudio.

- Una vez planteado el problema es necesario la construcción del marco teórico, donde este contextualiza el problema y lo sustenta teóricamente por medio de la recopilación de información donde se recopilan antecedentes válidos, teorías, enfoques teóricos, investigaciones previas, experimentos y más. Esto se asocia directamente con las referencias utilizadas en la investigación e investigaciones anteriores se verán reflejadas en apartados del estudio. Una vez construido el marco teórico se debe revisar, refinar o en casos redefinir las preguntas de investigación.

- Se desarrolla una hipótesis de trabajo o un planteamiento de trabajo sobre la problemática. Una hipótesis es una explicación tentativa del problema observado donde aporta con evidencias a favor de la teoría planteada. Se deben especificar las variables del problema de modo conceptual y operacional, además se establece si el problema se abordará experimentalmente. Por lo tanto, una hipótesis surge del desarrollo del marco teórico.

- Finalmente se propone una metodología para desarrollar una solución al problema planteado, donde se debe considerar el procesamiento de los datos, la definición y recolección de datos de prueba para validar la hipótesis propuesta.

- La metodología debe considerar, para todo el proceso de investigación la elaboración y presentación de informes de investigación adecuados.

1.5 Plan de trabajo.

En consecuencia a los objetivos planteados en los apartados 1.2 y 1.3 la planificación del estudio consta de las siguientes etapas:

1.-**Etapa de recolección de información y análisis**: En esta primera etapa en un principio se recolecta la información de las fuentes seleccionadas, luego se realiza un análisis de esta información viendo cada una de las referencias investigadas, luego se selecciona cuál de estas referencias son las de más relevancia y las que se utilizarán en este estudio. De estas

referencias se pretende llegar a obtener el marco teórico y comprender la problemática del tema a estudiar.

Los hitos de esta etapa serán la obtención del marco teórico y estado de arte , tanto de robótica como de Kinect , una vez obtenido este se podrá determinar finalmente cual es el prototipo de solución del estudio para luego pasar a la siguiente etapa.

Para esto es necesario cumplir las siguientes sub-etapas:

- Investigación: Tema a desarrollar, estado del arte, marco teórico, las tecnologías y metodologías a utilizar.
- Identificación de tópicos.
- Redacción del marco teórico y estado del arte.
- Identificar alcance y problemática del proyecto.

2.-**Análisis, desarrollo y construcción de modelos**: Con la información obtenida y analizada en la etapa anterior, se procede al análisis, desarrollo y construcción de modelos. Para lograr un buen modelo es necesario en una primera instancia una buena captura de requerimientos del problema planteado, una vez realizado esto, se procede a la construcción del modelo de la solución, haciendo los respectivos análisis u observaciones que surjan en el proceso.

La segunda etapa se divide en las siguientes sub-etapas:

- Análisis de requerimientos.
- Formalización de los requerimientos.
- Modelado del sistema (diagramas).
- Desarrollo en SDK Kinect.
- Utilización de la librería para el robot LEGO.
- Análisis de factibilidad del proyecto.

3.- **Implementación formal de los modelos y etapas de pruebas finales:** En base a los modelos y prototipos obtenidos se desarrolla un software (Firmware en este caso) para obtener la solución al problema planteado .Una vez desarrollado el código fuentes se realizarán sus posteriores pruebas en base a las distintas ejecuciones del middleware que se realicen y se compararán los resultados de este hasta ver cuál es la solución más óptima.

La tercera etapa se divide en las siguientes sub-etapas:

- Desarrollo del prototipo de captura del esqueleto para el sensor kinect.
- Desarrollo de prototipo de aplicación para el robot a utilizar.
- Reconocimiento de gestos en una aplicación.
- Unión de los prototipos señalados.
- Desarrollo de la aplicación final.
- Pruebas
- Detección de errores.
- mejoras

1.6 Estructura del documento.

El documento está dividido en diferentes capítulos, cada uno de ellos tocan temáticas diferentes pero con un cierto grado de cohesión como lo veremos en algunos capítulos que trata sobre el marco teórico de la solución al problema. Para ello se puede apreciar en este tópico un resumen de cada uno de ellos:

Capítulo 1: Es un capítulo introductorio al tema a investigar, en él se encuentran la definición del problema y los objetivos tanto el general como específico de este. También se compone de la planificación del estudio y la metodología de investigación más adecuada a seguir.

Capítulo 2: En este capítulo se aprecia todo el marco teórico relacionado con la robótica, en una primera instancia se define lo que es el concepto para dar paso luego a la historia de la Robótica , donde se enfoca principalmente al origen de la palabra y las 3 leyes de esta. Además se distinguen los componentes de un robot y la clasificación de estos. En una ultima instancia se dedica un apartado para los tipos de robot a utilizar en el proyecto, donde básicamente son corresponden a 2 que son el roboPhilo y el robot Lego NXT, en este último se describe el modelo a utilizar en el proyecto luego de acoplar las piezas.-

Capítulo 3: En este capítulo se aprecia todo el marco teórico con lo que respecta al medio de comunicación entre la persona y el robot, este medio es el sensor de Microsoft Kinect. En una primera instancia se define el concepto de interacción natural para dar paso a toda la investigación del sensor. Se dedica un apartado al funcionamiento de Kinect donde como información principal se aprecia el algoritmo de inteligencia artificial con el que se obtiene los datos del esqueleto humano y poder interpretarlos.

Capítulo 4: En este capítulo se explica detalladamente y en forma de un diagrama cuál el caso de estudio de la solución al problema. Para poder entender cómo será la solución, es

necesario, determinar de qué manera interactúan las tecnologías estudiadas en el marco teórico del estudio, para eso, se ha formulado un diagrama donde se muestran a los actores interactuando tanto con el sensor como con el robot y con las herramientas que estos poseen.

Capítulo 5: En este capítulo se realiza un estudio de factibilidad del proyecto el cuál se divide en un estudio de factibilidad económica, otro de factibilidad técnica y finalmente se realiza un estudio de análisis de riesgo para determinar qué riesgo pone más en peligro el correcto desarrollo del proyecto.

Capítulo 6: En este capítulo y una vez obtenido el producto final, se describen las distintas pruebas a las que fueron sometidas el software, para esto en una primera instancia es necesario dividir cada prueba en etapas para garantizar el éxito de estas y poder detectar de mejor forma los errores que se identifiquen.

Capítulo II Robótica

2.1 Introducción.

La robótica es una rama de la ciencia que abarca el diseño y la implementación de máquinas capaces de emular o imitar el comportamiento de los seres humanos u otro ser vivo , esta se complementa con otras ramas como la inteligencia artificial, la lógica, el álgebra , que ayudan a dar mejores soluciones a los problemas que surgen en esta rama.

Para lograr comprender de que se trata esta rama de la ciencia primero es necesario conocer algo de su historia donde la evolución de los robots han permitido que estas máquinas puedan ser de ayuda en la vida cotidiana de los seres humanos para realizar labores que estos no pueden realizar como enviar robots a la superficie lunar y planetas para un estudio detallado, robots manipuladores en la industria que transportan grandes pesos y los pueden transportar de un lugar a otro sin mayores inconvenientes.

En los siguientes apartados se profundizara más sobre esta importante rama de la ciencia, donde obteniendo el marco teórico de la robótica sustentará las bases para la solución del problema planteado anteriormente, donde también se dedica un apartado a las herramientas de entorno de desarrollo para generar el prototipo de código fuente y realizar las pruebas pertinentes para determinar la solución más óptima

2.2 Concepto

La robótica es una rama de la ciencia o de la tecnología, que estudia el diseño y construcción de máquinas que sean capaces de desempeñar tareas realizadas por el ser humano o que requieren el uso de inteligencia. De manera más global el concepto de robótica se entiende como "*el conjunto de conocimientos teóricos y prácticos que permiten concebir, realizar y automatizar sistemas basados en estructuras mecánicas poli articuladas, dotado de cierto grado de inteligencia y destinado a la producción industrial o a la sustitución del hombre en diversas tareas*" [Ricardo Lizcano, 2005].

Un sistema robótico puede describirse como *"aquel que es capaz de recibir información, de comprender su entorno a través del empleo de modelos , de formar y ejecutar planes y de controlarlo o supervisar su operación*" [Ricardo Lizcano, 2005]. La Robótica es esencialmente pluridisciplinaria ya que se encuentra en muchas disciplinas de la ciencia o en la vida cotidiana pero se apoya en los progresos de la microelectrónica y la informática, así de las nuevas disciplinas como el reconocimiento de patrones y la inteligencia artificial.

2.3 Historia de la Robótica

Durante siglos el hombre ha construido máquinas que imiten partes del cuerpo humano. Los egipcios unieron brazos mecánicos a las estatuas de los dioses. Estos brazos eran operados por los sacerdotes de la época quienes clamaban que el movimiento de estos se vieran representado el actuar de los dioses, a su vez los griegos construyeron estatuas que operaban con sistemas hidráulicos, las cuales se utilizaban para adorar a los dioses. El hombre desde ese entonces comenzó a creer que era importante dar vida a seres artificiales que realicen tareas repetitivas y ayuden en las tareas difíciles o pesadas a realizar por un ser humano [Ricardo Lizcano, 2005].

Durante los siglos XVII y XVII en Europa fueron construidos muñecos mecánicos que tenían algunas características de lo que hoy en día conocemos como robots. Jacques de Vauncansos construyó varios títeres de tamaño humano a mediados del siglo XVIII. Esencialmente se trataban de robots mecánicos diseñados para un propósito específico: la diversión. En 1805, Henri Maillardert construyó una muñeca mecánica que era capaz de hacer dibujos, en la cual, una serie de levas eran el sistema principal de control para que esta pudiese escribir y dibujar. Hubo otras invenciones mecánicas durante la revolución industrial, muchas de las cuales estaban dirigidas al sector de la producción textil, entre ellas se puede citar la hiladora giratoria de Hargreaves (1770), la hiladora mecánica de Crompton (1779), el telar mecánico de Cartwright (1785), el telar de Jacquard (1801), entre otros [Ricardo Lizcano, 2005].

El desarrollo en la tecnología, donde se incluyen las computadoras electrónicas, los actuadores de control retroalimentados, transmisión de potencia a través de engranajes, y la tecnología en sensores, han contribuido a facilitar la construcción de mecanismos autómatas para desempeñar tareas dentro de la industria. Entiéndase como autómata toda máquina que contiene un mecanismo que le permite realizar determinados movimientos. [RAE, 2012]. Son varios los factores que intervienen para que se desarrollaran los primeros robots en la década de los cincuenta; la investigación en inteligencia artificial desarrolló maneras de emular el procesamiento de información humana con computadoras electrónicas y proporciono una variedad de mecanismos para probar sus teorías.

Una obra Checoslovaca publicada en 1917 por Karel Kapek, denominada Rossum's Universal Robots, dio lugar al término robot. La palabra checa 'Robota' significa servidumbre o trabajador forzado, en el momento en que se tradujo al inglés se convirtió en el término "robot" [Roberto Alvarez, 2008].Entre los escritores de ciencia ficción, Isaac Asimov contribuyó con varias narraciones relativas a robots, comenzó en 1939, a él se atribuye el acuñamiento del término Robótica. La imagen de robot que aparece en su obra es el de una máquina bien diseñada y con una seguridad garantizada que actúa de acuerdo con tres principios:

- Un robot no puede actuar contra un ser humano o, mediante la inacción, que un ser humano sufra daños [Roberto Alvarez, 2008].

- Un robot debe obedecer las órdenes dadas por los seres humanos, salvo que estén en conflictos con la primera ley [Roberto Alvarez, 2008].

- Un robot debe proteger su propia existencia, a no ser que esté en conflicto con las dos primeras leyes [Roberto Alvarez, 2008].

Así, inicialmente, se definía un robot como un manipulador reprogramable y multifuncional diseñado para trasladar materiales, piezas, herramientas o aparatos a través de una serie de movimientos programados para llevar a cabo una variedad de tareas. El desarrollo en la tecnología, donde se incluyen las poderosas computadoras electrónicas, los actuadores de control retroalimentados, transmisión de potencia a través de engranes, y la tecnología en sensores han contribuido a flexibilizar los mecanismos autómatas para desempeñar tareas dentro de la industria. Son varios los factores que intervienen para que se desarrollaran los primeros robots en la década de los 50's. La investigación en inteligencia artificial permite el desarrollo de maneras de emular el procesamiento de información humana con computadoras electrónicas e invento una variedad de mecanismos para probar sus teorías [Roberto Alvarez, 2008].

Actualmente, el concepto de robótica ha evolucionado hacia los sistemas móviles autónomos, que son aquellos que son capaces de desenvolverse por sí mismos en entornos desconocidos y parcialmente cambiantes sin necesidad de supervisión. Es decir que no necesita control remoto o una persona que los manipule.

2.3 Clasificación de los robots

Existen muchos tipos de clasificación de los robots dependiendo de su grado de control, inteligencia, arquitectura, grados de libertad, forma, fin para el que son desarrollados, y aún más . Algunas clasificaciones son las siguientes: [Roberto Alvarez, 2008].

2.3.1 Clasificación General

- **Manipuladores**: La mayor parte del robot industrial actual son esencialmente brazos articulados. De hecho según la definición del Robot Institute of America, un robot industrial es un manipulador programable multifuncional diseñado para mover materiales, piezas, herramientas o dispositivos especiales, mediante movimientos

variados, programados para la ejecución de distintas tareas. En la figura 2.1 se puede observar la forma de brazo mecánico para mover los materiales o piezas.

Figura 2. 1 Robot manipulador industrial [Ollero, 2001].

- **Robots de repetición**: Son manipuladores que se limitan a repetir una secuencia de movimientos, previamente ejecutada por un operador humano, haciendo uso de un controlador manual o un dispositivo auxiliar. El operario en la fase de enseñanza se vale de programación con diversos pulsadores o teclas , o bien de joysticks .En la figura 2.2 se aprecia el dispositivo con que se manipula el robot

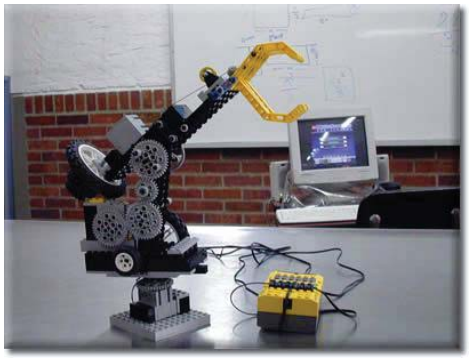

Figura 2. 2 Robot de repetición [Lerma, 2010]

- **Robots controlados por computadoras**: Son manipuladores, controlados por un computador, que habitualmente suele ser un microcomputador. En este tipo de robot , el programador no necesita mover realmente el elemento de la máquina, cuando la prepara para realizar el trabajo. El control por computador dispone de un lenguaje

específico, compuesto por varias instrucciones adaptadas al robot. En la figura 2.3 se precio varios robots manipuladores controlados por 2 computadores.

Figura 2. 3 Robot controlado por un computador en una cirugía [Arroyo, 2005]

- **Robots inteligentes**: Similares a los del grupo anterior, pero además son capaces de relacionarse con el mundo que les rodea a través de sensores y tomar decisiones en tiempo real (auto programable).De momento solo se encuentran en fase experimental y son muy poco conocidos en el mercado. En la figura 2.4 se aprecia un robot inteligente capaz de aprender a tocar un violín por medio de inteligencia artificial.

Figura 2. 4 Robot inteligente [Ramirez, 2011]

- **MicroRobots o de servicio:** Con fines educacionales, de entretenimiento o investigación, existen numerosos robots de formación o micro-robots a un precio muy asequible, y cuya estructura y funcionamiento son similares a los de aplicación industrial o manipuladores. En la figura 2.5 se aprecia un micro-robot donde se puede apreciar su increíble tamaño.

Figura 2. 5 Micro-Robot [Deyle, 2009]

Dentro de los robots de servicio se encuentran:

- **Robot estacionario**: Son robots que se fijan a su posición. Tiene las articulaciones móviles de muñeca, codo, cintura y hombro. Son también ampliamente utilizados en el sector de la industria donde son asignados para tareas específicas donde no requiera que este se movilice. En la imagen 2.6 se aprecia un robot estacionario en labores industriales.

Figura 2. 6 Robot estacionario

- **Telemanipuladores**: Robots dedicados a la manipulación de entornos inaccesibles. Se utilizan mayoritariamente en la medicina para cirugías complejas. Se caracterizan principalmente por la precisión que estos poseen y las distintas ramas donde pueden

cumplir su función. En la figura 2.7 se aprecia un robot tele manipulador utilizado en cirugías complejas.

Figura 2. 7 Robot telemanipulador [Arroyo, 2005]

- **Androides**: Son con forma humana, imitan el comportamiento de las personas, su utilidad hoy en día es de experimentación debido a que se busca que también puedan ser seres inteligentes , tarea muy compleja de realizar ya que son desarrollados con algoritmos de inteligencia artificial. En la figura 2.8 se aprecia un robot androide con rasgos idénticos a los de un ser humano.

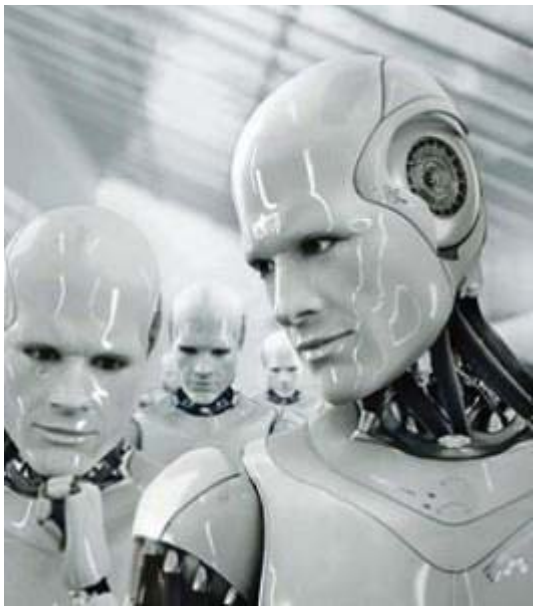

Figura 2. 8 Androide [Ramirez, 2011]

- **Humanoides**: Es un tipo de robot con apariencia similar a la del cuerpo humano, permitiendo la interacción con humanos, herramientas y/o entornos. La diferencia con los robots androides es que no buscan estéticamente parecerse a un ser humano. En la

figura 2.9 se aprecia un robot humanoide (Robo Philo) que es el que se utilizará en este estudio.

Figura 2. 9 Humanoide Philo [Robophilo, 2007]

- **Móviles**: Estos robots se desplazan mediante una plataforma rodante (ruedas); aseguran el transporte de piezas u otro elemento de un punto a otro ayudando en gran medida a las labores que se deben realizar sobre todo en el campo de la industria. Actualmente también son muy utilizados en estudios espaciales para recorre la superficie de planetas .En la figura 2.10 se aprecia un robot móvil utilizado en el campo de la industria

Figura 2. 10 Robot móvil [Ollero, 2001]

2.3.2 Clasificación según la Federación Internacional de Robótica.

Esta clasificación sale a la luz en el año 1988 enfocada principalmente para el sector manufacturero. Es de las clasificaciones más generales y una de las más utilizadas [Roberto Alvarez, 2008].

- Tipo A: Manipulador que se controla manualmente o por control remoto.
- Tipo B: Manipulador automático ajustado (accionamiento neumático, eléctrico o hidráulico).
- Tipo C: Robot programable con trayectoria punto a punto .

2.3.3 Clasificación según generación

- Primera Generación: Repite programas secuencialmente sin tomar en cuenta el entorno.
- Segunda Generación: Actúa en consecuencia de la información que recolecta de su entorno.
- Tercera Generación: Se programa a través de lenguaje natural.

2.4 Componentes de los robots

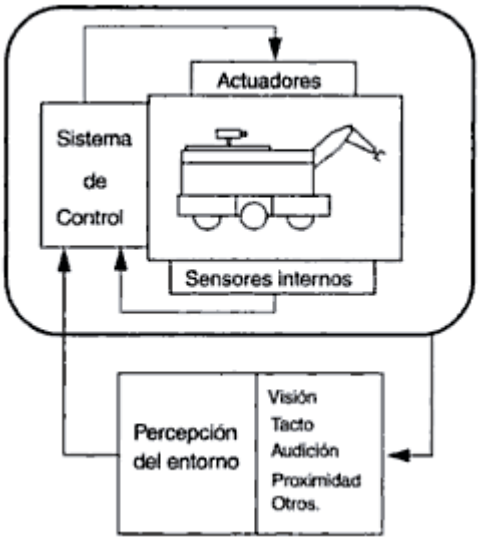

Figura 2. 11 Esquema general de un robot [Roberto Alvarez, 2008]

En la figura 2.11 se muestra el esquema básico de un robot. En ella se identifican un sistema mecánico, actuadores, sensores y el sistema de control como elemento básico necesario para cerrar la cadena actuación-medidas-actuación. En el sistema mecánico puede distinguirse entre el órgano terminal, el brazo articulado y un vehículo . En la mayor parte de los robots industriales no existe el vehículo, estando fija la base del brazo.

- **Control de movimiento**: Desde el punto de vista del procesamiento de la información, en robótica se involucran funciones de control de movimiento, percepción y planificación. Estas son coordenadas en las que el robot se va a desplazar, pueden ser cartesianas, cilíndricas o polares.
- **Sistema de Control:** El sistema de control involucra tanto bucles de retroalimentación de la información suministrada por los sensores internos como su entorno.
- **Sensores Internos:** Los sensores internos miden el estado de la estructura mecánica y, en particular, giros o desplazamientos relativos entre articulaciones, velocidades, fuerzas y pares. Estos sensores permiten cerrar bucles de control de las articulaciones de la estructura mecánica.
- **Sensores externos:** Los sensores externos permiten dotar de sentidos al robot. La información que suministran es utilizada por el sistema de percepción para aprender la realidad del entorno. Este sistema de percepción hace posible que un robot pueda adaptar automáticamente su comportamiento en función de las variaciones que se producen en su entorno, haciendo frente a situaciones imprevistas.
- **Sistemas de percepción**: El desarrollo de un sistema de percepción en los robots surge a partir de los procesos tecnológicos en sensores como visión , tacto , incluso audición . Sin embargo percepción no solo involucra captación de la información sensorial, sino que también su tratamiento e interpretación [Ollero, 2001].

2.5 Robophilo . Características y componentes.

La versión comercial de este robot fue lanzada en Febrero de 2007 , esta versión es lanzada por roboPhilo Company y busca tener las mismas funcionalidades de un robot humanoide a gran escala pero a un costo mucho menor .Este robot posee características que lo diferencian de los demás además de componente que serán descritos en los próximos apartados [Robophilo, 2007].

2.5.1 Componentes Kit comercial

El kit comercial de RoboPhilo se compone de lo siguiente [Robophilo, 2007]:

- Altura Robot: 33 cms.
- Peso: 1,2 Kg (1200 gramos) con batería.
- Unidad de PCB.
- Controlador de 36 KHZ.
- 36 KHZ del receptor.
- 6V batería Ni-MH.
- Cargador 7.2V 1000 ma.
- Software gráfico de movimiento (SDK).
- Soporte de suspensión.
- Calcomanías.

2.5.2 Características Robophilo.

Para efectos de este estudio roboPhilo será el robot a utilizar, debido a su bajo costo y que posee casi las mismas características a un robot humanoide de alto costo para poder realizar las pruebas pertinentes al conectarlo con el sensor Kinect.

Figura 2. 12 RoboPhilo [Robophilo, 2007]

Como se aprecia en la figura 2.12 roboPhilo se clasifica como un robot humanoide, intenta asemejar a lo que es un ser humano y las funciones que este realiza, pero si nos vamos a las clasificaciones globales realizadas en el estudio también puede corresponder a un robot manipulador o a un micro-robot ,debido a que este fué creado con fines de entretenimiento y estudio, este Robot fue lanzado el año 2007 revolucionado el mercado de robot humanoides debido a su bajo costo y lo intuitivo de este . Robophilo mide 33 cms, pesa 1,2 kilos y ofrece moverse como el mejor es una adición más el incipiente mercado de robots humanoides, su principal característica es que puede desarrollar muchas de las actividades que hoy en dia realizan robots del mismo tipo pero con presupuestos millonarios, Robophilo al contrario es de muy bajo costo y puede ser adquirido por el común de la personas y/ organizaciones que deseen investigar con este robot. Para mantener estos bajos costos Robophilo utiliza los marcos de plástico en lugar de aluminio, además posee servos aficionados y no digitales de alta precisión. Por supuesto algo de rendimiento es sacrificado, pero Robophilo lo compensa de muchas maneras. Para empezar es un robot muy bien articulado. Posee 20 servos, incluyendo la cadera y la rotación de cintura. Las articulaciones de la cintura y la cadera, están articuladas por los sistemas de tracción indirecta donde la carga se coloca directamente en los ejes de los servos. Además posee 24 canales para los servos y 8 interfaces de E/S. Un núcleo de interrupción especializado se utiliza para controlar el movimiento del robot además de un control remoto. Con el este control remoto a distancia conectado pro sensores infrarrojos al robot permite alrededor de 121 movimientos y se pueden controlar hasta 4 robots de forma independiente. Robophilo es fácilmente programable desde un computador que permitirá personalizar los movimientos como el usuario lo desee [Robophilo, 2007].

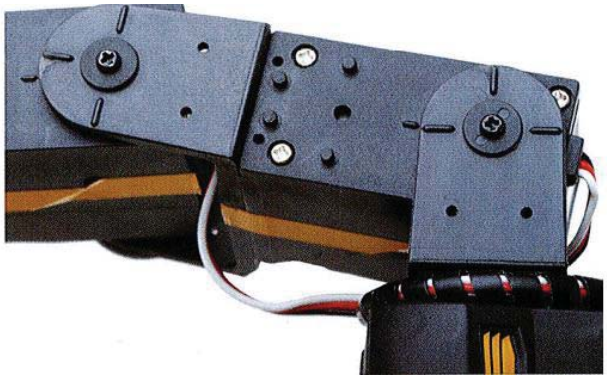

Figura 2. 13 Servomotor RoboPhilo [Robophilo, 2007]

La figura 2.13 muestra uno de los servomotores del roboPhilo, este permite que la articulación del brazo se pueda mover. Estos servomotores también se encuentran en las piernas, cabeza y en el tronco del robot.

2.6 LEGO Mindstorms NXT.

Lego Mindstorms es un juego de robótica para niños fabricado por la empresa Lego, el cuál posee elementos básicos de las teorías robóticas, como la unión de piezas y la programación de acciones, en forma interactiva. Este robot fue comercializado por primera vez en septiembre de 1998.

Lego Mindstorms puede ser usado para construir un modelo de sistema integrado con partes electromecánicas controladas por computador. Prácticamente todo puede ser representado con las piezas tal como en la vida real, como un elevador o robots industriales.

El bloque NXT es una versión mejorada a partir de LEGO Mindstorms RCX, que generalmente se considera la predecesora y precursora de los bloques programables de Lego.

Debido a la comercialización de los bloques programables, LEGO vendió la generación NXT en dos versiones: Retail Version y Education Base Set. Una ventaja de la versión Educacional es que se incluía las baterías recargables y el cargador, pero esta misma versión debía comprar el software según el tipo de licencia: Personal, Sala de clases, Sitio. [Beland, 2000].

2.7.1 Componentes kit Mindstorms NXT.

El juego tiene 4 partes importantes:

- El microprocesador (también llamado NTX).
- Los sensores (luz, sonido, tacto, color, rotación, ultrasonido).
- Los servomotores.
- Los elementos technic. (o sea las piezas).

- LEGO NXT: El ladrillo inteligente LEGO NXT, es el cerebro del robot, básicamente es un microprocesador, con puertas de entradas, de salida y memoria para almacenar los programas, se comunica con el computador a través de un puerto USB o Bluetooth. Es muy importante señalar para efectos de este estudio , los medios de comunicación que este posee , ya que ,el NXT se conectará con el sensor Kinect por medio de Bluetooth donde se le enviarán las órdenes desde el código fuente de las acciones que este debe realizar [Edubrick, 2006].

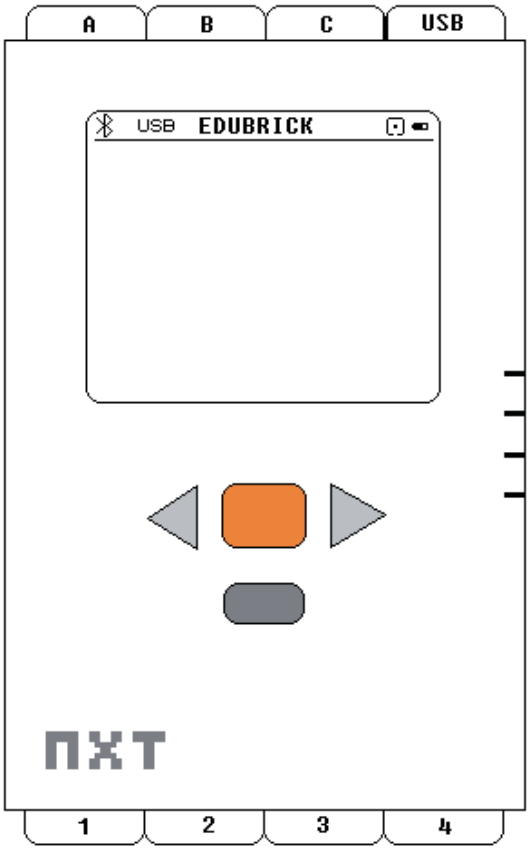

Figura 2. 14 Cerebro NXT [Edubrick, 2006]

❖ Tres puertas de salida: A (Motor para función extra),B(Motor Izquierdo), C (Motor derecho).

❖ Puerta de comunicación USB.

❖ Pantalla de Cristal Líquido.

❖ Miniparlantes.

❖ Botones de encendido, apagado, navegación.

❖ Cuatro puertas de salida,1(Sensor de tacto), 2(Sensor de Sonido), 3(Sensor de Luz) y 4(Sensor de ultrasonido).

❖ Sensor de tacto: Proporciona al robot el sentido del tacto, tanto cuando se presiona como cuando se suelta. El sensor de contacto permite detectar si el bloque que lo posee ha colisionado o no con algún objeto que se encuentre en su trayectoria inmediata. Al tocar una superficie, una pequeña cabeza externa se contrae, permitiendo que una pieza dentro del bloque cierre un circuito eléctrico comience a circular energía .También puede ser utilizado para ciclos completos de presionar y soltar [Edubrick, 2006].

Figura 2. 15 Sensor de tacto LEGO [Edubrick, 2006]

❖ Sensor de sonido: Le permite al robot escuchar. El sensor de sonido puede detectar tanto decibeles (dB), como decibeles ajustados (dBA). Decibel es una medida de la presión de sonido.

dBA: La sensibilidad del sensor es ajustada al oído humano.

db: La sensibilidad del sensor, no está ajustada y puede "escuchar" sonidos por debajo o por encima de la capacidad del oído humano [Edubrick, 2006].

Figura 2. 16 Sensor de sonido LEGO [Edubrick, 2006]

- Sensor de Luz: Le proporciona al robot el sentido de la visión. Le permite al robot distinguir entre luz y oscuridad. El sensor es monocromático, es decir puede distinguir entre el blanco y el negro pasando por una gama de grises, la lectura la entrega en porcentaje [Edubrick, 2006].

Figura 2. 17 Sensor de luz LEGO [Edubrick, 2006]

- Sensor ultrasónico: También le proporciona al robot un sentido de visión. El sensor emite un sonido y mide el tiempo que la señal tarda en regresar, para luego calcular la distancia, a la cual se encuentra el objeto u obstáculo.

  Es el mismo principio utilizado por los murciélagos y el sonar de las naves, tiene un rango de 0 a 255 cm con una precisión de +/- 3 cm [Edubrick, 2006].

Figura 2. 18 Sensor ultrasónico LEGO [Edubrick, 2006]

- Sensor de color: Le permite al robot “ver” colores. No sólo blanco y negro, este es un sensor capas de distinguir colores reales [Edubrick, 2006].

Figura 2. 19 Sensor de color LEGO [Edubrick, 2006]

- Sensor de inclinación y aceleración: Le permite al robot saber que inclinación tiene. Dispone de un acelerómetro en los tres ejes (X; Y y Z) con un rango entre –2g y +2g, lo cual le permite calcular la inclinación [Edubrick, 2006].

Figura 2. 20 Sensor de aceleración e inclinación LEGO [Edubrick, 2006]

- Servomotor: Los motores incluyen un sensor de rotación, con una precisión de +/- 1 grado, también se puede montar el motor en un eje y utilizarlo como sensor de rotación. Para el movimiento de un modelo motorizado el firmware (el sistema operativo interno del NXT), dispone de un sofisticado algoritmo PID, el cual permite que el modelo se desplace con precisión. También es posible utilizar los motores, estándar LEGO, mediante el cable adaptador [Edubrick, 2006].

Figura 2. 21 Servomotor LEGO [Edubrick, 2006]

- Piezas: El Lego Mindstorms, a diferencia de algunas de los juegos que vende Lego, trae algunas piezas extras que permiten entregar flexibilidad y movimiento al robot que se esté construyendo.

Para clasificar las piezas, se sugiere una clasificación entre las piezas móviles, flexibles y de fijación, las cuales son las que incluye el Lego Mindstorms para desarrollar cualquier robot en especial.

Piezas móviles: Las piezas móviles que dispone Lego Mindstorms se centran principalmente en la rotación de bloque, para lograr que las ruedas se muevan en un movimiento circular con respecto al bloque completo. Estas piezas móviles se pueden clasificar en dos:

Pieza de rotación, permite rotar un bloque de Lego con respecto a otro, siendo hueco en el centro del mismo, y con la patas de conexión; lo cual permite añadir más piezas en la parte superior del bloque de rotación. Este bloque se usa fundamentalmente en los robots de movimiento o donde se realiza una cinta de transporte de materiales, y se conecta a uno de los motores para que provea el giro del bloque

Pieza de giro: A diferencia de la pieza de rotación, la pieza de giro permite girar un bloque en el espacio, permitiendo una simulación de ojos de un robot. Esta pieza no posee una utilidad real, pero sirve de adorno para el robot.

Piezas flexibles: Las piezas flexibles permiten recrear una articulación de un sistema articulado, donde se requiere que el robot deba realizar un movimiento no rígido en forma específica, como el brazo robot o el brazo clasificador de piezas. Las piezas flexibles por lo general son tubos de plástico capaces de conectarse con dos bloques que no se encuentren separados a una distancia mayor de 4 cm

- Piezas de fijación: Las piezas de fijación, son aquellas que sirven para fijar los ejes de rotación producidos por las piezas de rotación, lo cual implica que son usadas en el centro de las ruedas que posee el Lego. Por lo general, son tubos de 0.5 mm de diámetro el cual se puede poner en la punta de una barra que actúa como eje central de la rueda, fijando que la misma no se salga durante la ejecución de un programa [Edubrick, 2006].

2.7.2 Garra mecánica que simula el movimiento de una grúa.

El robot a utilizar simula los movimientos de una grúa con una garra en la parte superior de este, también se asimila al movimiento de un brazo mecánico. Los movimientos que el robot NXT realiza son sencillos, arriba, abajo, mover el brazo hacia la izquierda y también derecha , una vez ajustada la posición del brazo de puede realizar el movimiento de la garra, que puede tomar una pelota u otro utensilio que caiga dentro de esta y transportarla gracias a los movimiento antes descritos.

Las piezas han sido acopladas según manuales oficiales de la pagina Web de Lego, la estructura del robot consta principalmente de 3 servomotores, el controlador NXT y la garra en cuestión.

Luego del acoplamiento de las piezas, la apariencia del robot es la siguiente:

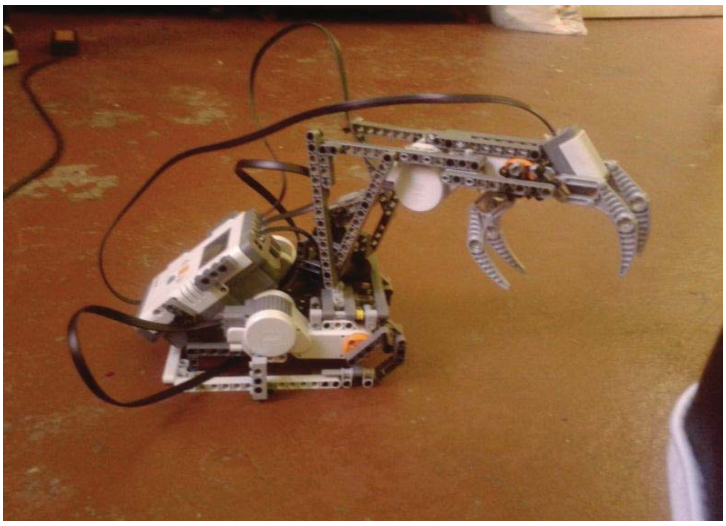

Figura 2. 22 Garra mecánica vista lateral

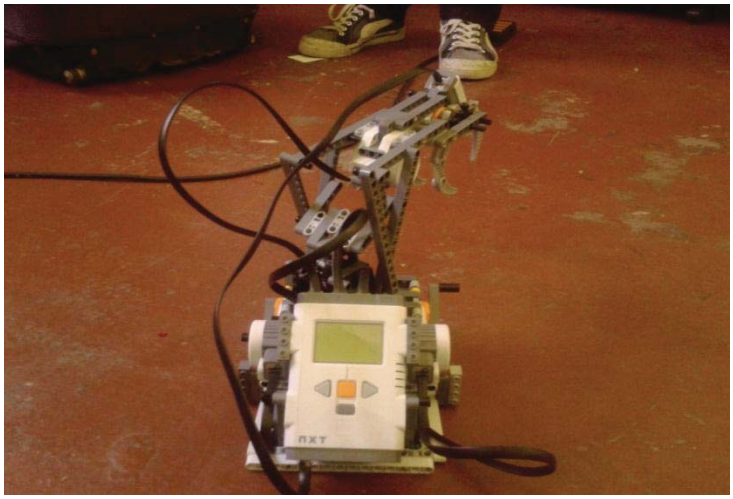

Figura 2. 23 Garra mecánica vista trasera

2.7.3 NxtNET (SDK LEGO NXT)

NXT.NET es una biblioteca. NET escrito en C # 3.0 que permite a las aplicaciones host para controlar el Lego Mindstorms NXT desde el código desarrollado en Visual Studio para el sensor Windows por medio de la conexión Bluetooth sin tener la necesidad de utilizar los sensores que posee el robot Lego Mindstorms. NXT.NET es creado por el Centro de Competencia de Hungría MSDN (y no por el Grupo LEGO), dentro de paquete de descarga de esta librería se encuentra la documentación correspondiente además de ejemplos de código fuente. [CodePlex,2008].

2.7.3.1 Detalles técnicos de la librería.

NXT.NET está compilado con Visual Studio 2008 para. NET 3.5 plataforma. El código fuente de la biblioteca está compilado como una biblioteca de escritorio. NET Framework 3.5 y también como una biblioteca móvil para. NET Compact Framework 3.5. La aplicación de escritorio es una aplicación de Windows Forms La documentación de ayuda HTML (CHM) se compila con NDoc 1,3 y el Taller de Ayuda HTML de Microsoft XML utilizando la documentación extensa en el código fuente de C #. NXT.NET, utiliza la clase NET Framework SerialPort para comunicarse con el Robot NXT a través de una conexión Bluetooth que se simula como un puerto COM en el PC o Pocket PC. El mando a distancia se lleva a cabo con los comandos directos del NXT y permite el movimiento del robot según el código fuente escrito en la aplicación a desarrollar.

2.7.3.2 Conexión con el robot mediante Bluethoot.

A través del código fuente de la aplicación en el proyecto WPF C# de Visual Studio, se puede realizar la conexión con el Bluethoot instalado en el computador para que este luego transmita los datos al robot LEGO por medio del siguiente código C#.

```
if(flagConexion){
                        this._nxt = new Nxt();
                        this._nxt.Connect("COM7");
                        msjConexion.Text = "Lego Conectado!";

                        flagConexion = false;
                        Motor = new Thread(new ThreadStart(detenerse));
                        Motor.Start();
                        Thread.Sleep(1000);
                    }
```

Para establecer la conexión entre el PC y el robot LEGO es recomendable utilizar un Bluetooth genérico 2.0 Plug and Play por conexión USB.

Una vez que ambos dispositivos (tanto como el del robot como el del computador) estén encendido y funcionando de manera correcta es necesario emparejarlos, es decir que ambos puertos COM coincidan, esto se realiza en la configuración de los dispositivos dentro del sistema operativo.

2.7.4 Mindsqualls Versión 2.0 C#

Es una biblioteca .NET para el control remoto del robot Lego Mindstorms NXT 2.0 desde el computador a través de una conexión bluetooth o USB.

Está escrito en el lenguaje de programación C#, pero se puede utilizar con cualquiera de los lenguajes .NET para Windows. Los códigos pueden ser compilados desde una PDA o un dispositivo móvil.

MindSqualls ofrece soporte completo para todos los "comandos directos". Soporte para el sensor de color de 2,0 NXT. Apoyo a la HiTechnic sensor Compas, el sensor de color HiTechnic, y el HiTechnic sensor NXT aceleración / Tilt [Handest., 2011].

Esta librería contiene la documentación correspondiente acerca de las funcionalidades que se pueden realizar desde el PC con el robot Lego, para efectos del estudio se dará mas importancia a la clase para controlar el motor del robot y además de la conexión por Bluetooth con el respectivo protocolo codificado dentro de un método de la clase a utilizar.

Las clases principales del proyecto de esta biblioteca se describen en el siguiente apartado.

2.7.3.1 Clases Librería Mindsqualls.

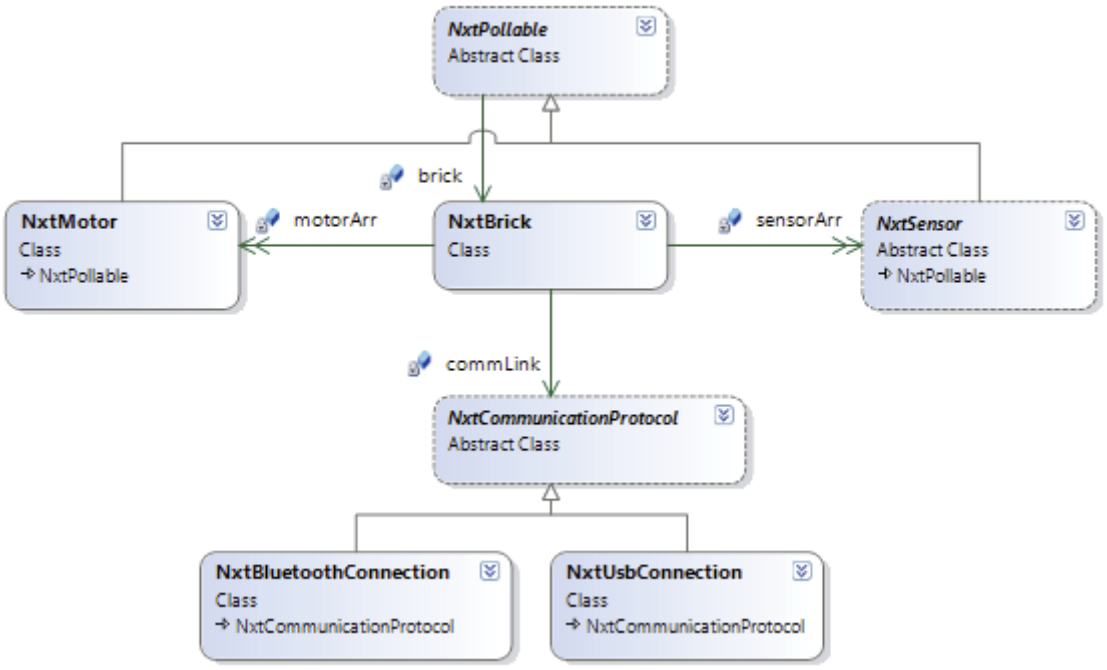

Figura 2. 24 Diagrama de clases librería Mindsqualls

El diagrama de clases, básicamente muestra como está desarrollada la librería a utilizar, la clase NxtPollable Implementa la funcionalidad común para el sondeo de un sensor o un motor para el sensor de entrada.

Luego 3 clases heredan los atributos y comportamientos de la clase principal, para el proyecto solo se utilizarán 2 de las 3, estas corresponden a la clase NxtMotor que permitirá manipular los motores A,B,C del robot NXT y la clase NxtBrick que básicamente identifica el cerebro NXT donde se deben enviar los datos. De esta clase también heredan 3 nuevas clases que son las que permiten la comunicación entre el dispositivo bluetooth o conexión USB y sus respectivos protocolos para el envió o recepción de datos. En el próximo apartado se profundiza acerca de estas clases.

2.7.3.2 Clase de protocolo de comunicación

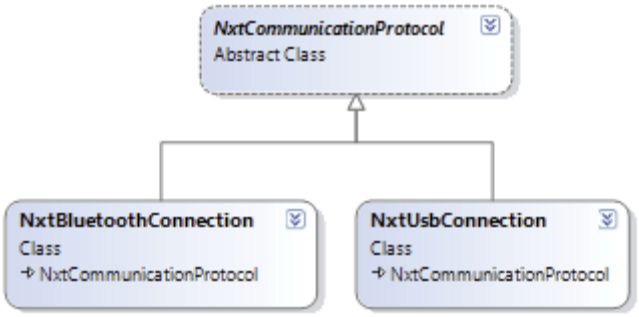

Figura 2. 25 Diagrama de clases protocolo de comunicación.

En esta clase, se aprecia la jerarquía entre las clases que permitirán la comunicación del PC con el robot por medio de conexión bluetooth o USB.

La conexión con el robot Lego se realizará por medio de Bluetooth, por lo tanto es necesario implementar la clase NxtBluetoothConecciton con su correspondiente comportamiento que se distingue en el diagrama de clases.

Antes de desarrollar algún tipo de código, es necesario, realizar algunas configuraciones de hardware en cuanto al dispositivo Bluetooth Plug and play 2.0. Una vez conectado el adaptador de Bluetooth al computador, se deben empareja el adaptador de Bluetooth con el del robot Lego, para esto, ambos dispositivos deben estar encendidos; éste emparejamiento, se consigue mediante la configuración del dispositivo en la pestaña de puertos COM de la configuración del adaptador, luego, sólo basta con comprobar los puertos de entrada y de salida, donde se debe visualizar el robot identificado con el nombre.

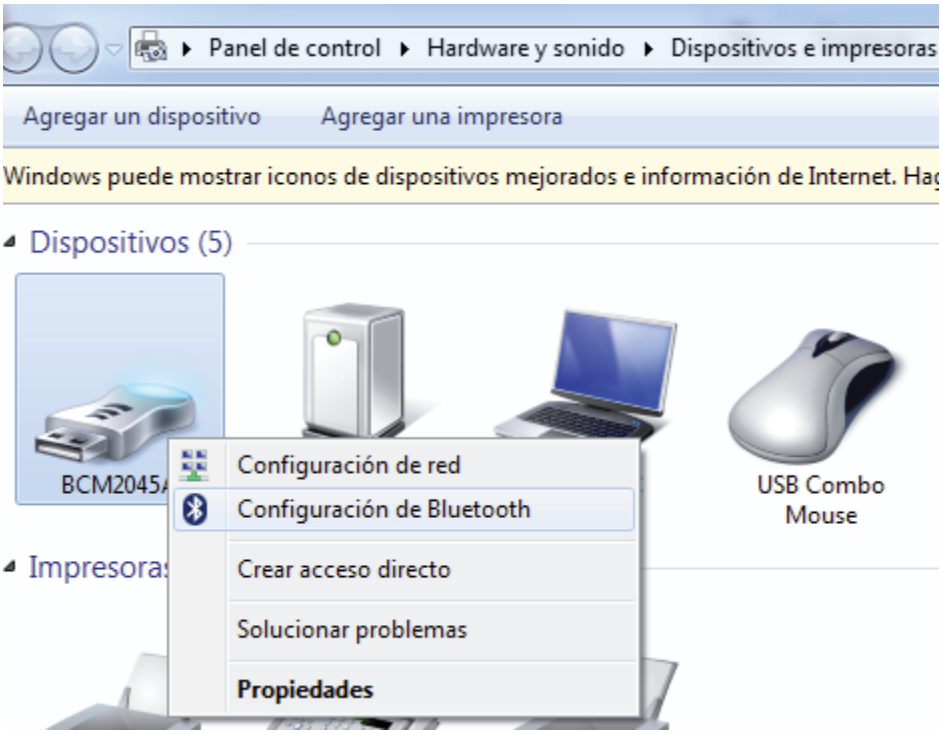

Figura 2. 26 Vista de configuración de dispositivos en Windows 7

Para efectos de programación solo basta en fijarse en el puerto de COM de salida de Bluetooth que será el encargado de enviar las ordenes al robot.

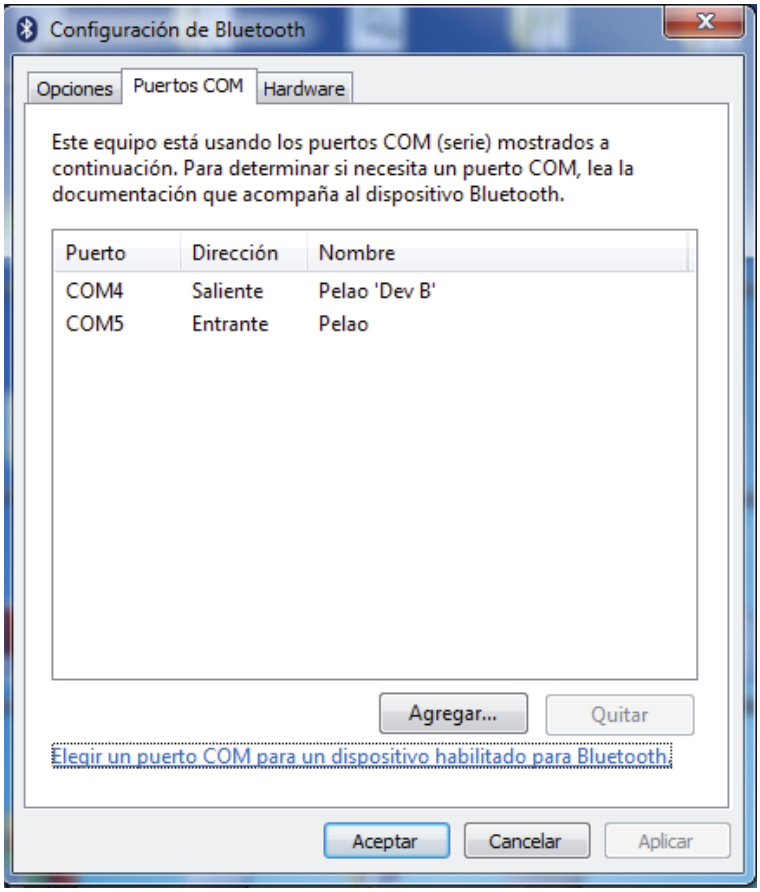

Figura 2. 27 Vista configuración dispositivo Bluetooth Plug and Play 2.0

Una vez identificado el puerto de salida y emparejados ambos hardware ya se puede codificar las instrucciones de conexión y luego la posterior manipulación del robot.

```
NxtBrick brick = new NxtBrick(4);
```

Con este método se puede enviar información al robot sol con señalar el puerto COM en que este se encuentra emparejado que en este caso es el puerto COM 4.

Todo lo señalado anteriormente se desarrolla en el lenguaje de programación C# con el entorno de desarrollo visual Studio 2010, sólo es necesario referenciar la librería correspondiente en el código de la kinect y se podrán utilizar las clases para la manipulación del robot desde el mismo código.

Capítulo III Interacción natural

3.1 Introducción

Entendamos en un principio interacción como la acción recíproca entre 2 o más objetos, sustancias, persona o agentes [RAE, 2012]**.** Si se habla de interacción natural para efecto de este estudio, nos referimos a la comunicación entre el robot y la persona sin la necesidad de manipular directamente el robot o controlar este por medio de un software. La interacción entre la persona y el robot se realiza por medio de sensores , donde la persona no manipula ningún tipo de control remoto , si no que con el movimiento de sus articulaciones el robot realizara las acciones que la persona haga. Kinect reconocerá estos movimientos gracias a un complejo software que este posee y arrojara los resultados pudiendo el programador adaptar los movimientos de la persona a los movimientos que desee que realice el robot a través de esa información [Jarret Webb, 2011]

Para lograr este tipo de interacción se utilizará una tecnología recientemente salida al mercado que ha revolucionado la industria desde su lanzamiento, ésta corresponde al controlador Kinect cuyo estudio del marco teórico se encuentra en el siguiente apartado.

3.2 Kinect

Figura 3. 1 Kinect [Jarret Webb, 2011]

En la figura 3.1 se aprecia el sensor Kinect, que es una barra horizontal de aproximadamente 23 cms conectada a una pequeña base circular con un eje de articulación de rótula, y está diseñado para ser colocado longitudinalmente por encima o por debajo del televisor [Jarret Webb, 2011].

En el año 2010 sale al mercado este interesante sensor, comercialmente llamado Kinect , lanzado por Microsoft , que según la definición de estos mismos se trata de "*un*

controlador de juego libre y entretenimiento para la videoconsola Xbox 360 .Kinect permite a los usuarios controlar e interactuar con la consola sin necesidad de tener contacto físico con un controlador de videojuegos tradicional, mediante una interfaz natural de usuario que reconoce gestos, comandos de voz, y objetos e imágenes".1

En Julio de 2011 Microsoft libera de manera oficial el Kinect SDK , donde incluye libfreenect, una nueva librería que ayudaría a los hackers de la comunidad OpenKinect y OpenNI , desarrollada principalmente por PrimeSense, uno de los principales vendedores de esta tecnología [Jarret Webb, 2011]. Con la liberación del SDK para estas comunidades se da inicio al primer paso para el desarrollo de nuevas tecnologías pese a que personas en meses anteriores ya habían logrado "Hackear" Kinect, desarrollando controladores para poder ser utilizado en otros dispositivos como laptops, ipad, etc.

En febrero de 2012 Microsoft decide ampliar aún más el uso de este sensor y lanza al mercado Kinect for Windows, de iguales características para el de consolas pero con soporte para Windows 7 y la diferencia de poder manipular el controlador a 40 cms de distancia, a diferencia del sensor de Xbox 360 que para su funcionamiento correcto de este necesita una distancia de 2 mts y 30 cms.

3.2.1 Componentes de Kinect.

- Un emisor de infrarrojos.
- Una cámara tradicional (Resolución 640 X 480 RGB 30 fps VGA).
- Una cámara de infrarrojos.
- Un motor.
- 4 micrófonos.

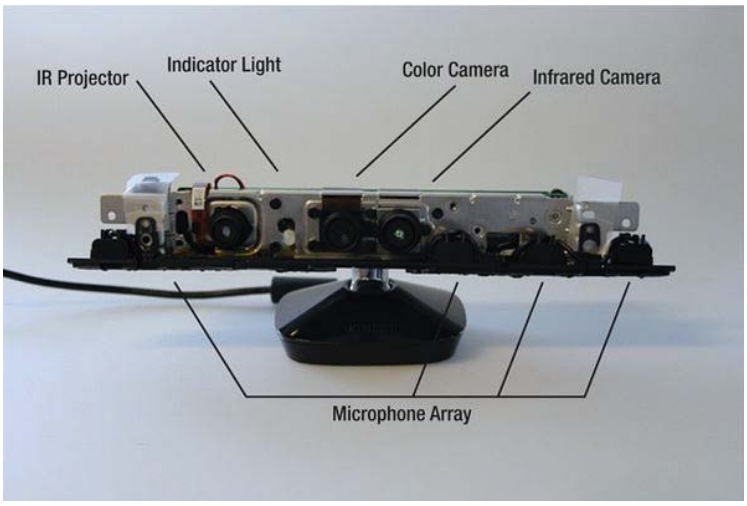

Figura 3. 2 Componentes Kinect [Jarret Webb, 2011]

1 Jarret Webb, J. A. (2011). *Beginning Kinect Programming with te Microsoft Kinect SDK.* apress

En la imagen 3.2 se identifica que en el centro se encuentra la cámara RGB que puede alcanzar hasta los 30 fps, en los costados se encuentran 2 sensores de profundidad, el de más a la izquierda proyecta un haz de luz infrarroja en la habitación donde se encuentra Kinect , luego al costado derecho se encuentra el sensor de profundidad que calcula la distancia de un punto determinado en función del tiempo que tarda la luz infrarroja en regresar al sensor. Luego en la parte inferior del sensor Kinect se encuentra la matriz de micrófonos. Uno está situado a la izquierda del sensor infrarrojo. Los otros tres están espaciados uniformemente a la derecha de la cámara de profundidad, estos cuatro micrófonos servirán para captar los sonidos del ambiente además del reconocimiento de voz del usuario. Finalmente en la parte posterior se encuentra el motor que se encarga de ajustar el ángulo de la cámara que puede llegar a los 27 ° hacia arriba y los 27 ° hacia abajo.

3.2.2 Funcionamiento Kinect

Kinect utiliza una complicada tecnología y un complejo sistema de reconocimiento corporal, donde este no sólo lee los movimientos corporales, si no que toda la posición del cuerpo humano en relación con el fondo y los objeto que lo rodean, además posee la capacidad de reconocer la voz del usuario y convertir sus comandos verbales en acciones dentro del videojuegos o en una interfaz.

Todo esto se genera en una cámara, un sensor de profundidad y un conjunto de micrófonos que trabajan sobre un software, desarrollado especialmente para este sistema, que reconoce los movimientos en 3D de un cuerpo completo y los comandos de voz. Con este "reconocimiento de voz" nos referimos a que Kinect puede localizar y ubicar fuentes sonoras y suprimir el ruido ambiental del lugar en el que se esté utilizando. Por otro lado el sensor de profundidad está equipado con un proyector infrarrojo y un sensor CMOS monocromático donde este detecta la luz permitiendo el control de luminosidad, conversor de contraste o un conversor analógico digital por lo que resulta importante en ocasiones el tipo de ropa que utiliza el usuario, ya que los colores claros se reconocen mucho mejor que los colores oscuros, además proporciona información sobre la altura y anchura del jugador en metros. Para poder utilizar de manera correcta Kinect se necesita de una distancia de 2,5 mts entre el sensor y la persona.

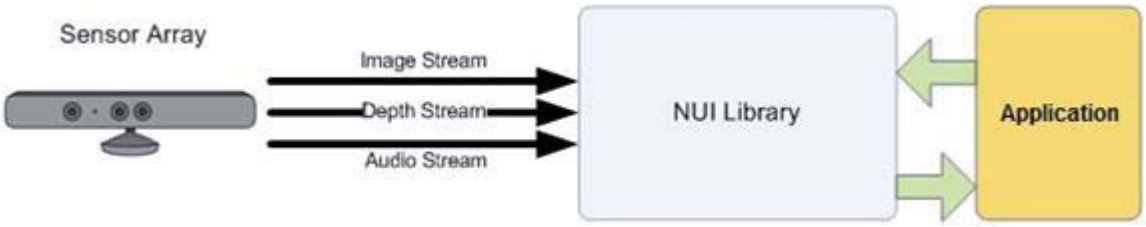

Figura 3. 3 Kinect para Windows Arquitectura

En la figura 3.3 se identifica básicamente los 3 tipos de funcionalidades que posee el sensor. En un primer lugar se encuentra la captura de imagen donde gracias a las librerías se puede obtener video por medio de la cámara RGB.

Luego se encuentra el sensor de profundidad que básicamente es una salida de video pero tomando tonalidades distintas para indicar cuando la persona se aleja o se acerca al sensor , mientras más cerca se encuentre la tonalidad será más oscura y a la inversa cuando la persona se aleja .Se pueden cambiar las tonalidades perfectamente desde el código fuente que implemente los métodos de profundidad.

Finalmente y el más importante para términos de este proyecto es el reconocimiento del esqueleto que permite a Kinect reconocer a las personas y seguir sus acciones. Más adelante se explicará en un apartado de este capítulo el detalle de cómo funciona y de que está constituido Skeleton Tracking.

Luego de las 3 principales funcionalidades del sensor se encuentra la API conocida como NUI (Natural User Interface, está diseñada para que sea compatible esencialmente con Windows. A través de ella se puede acceder a los datos de los sensores y crear las distintas aplicaciones con los datos que entrega el sensor Kinect. Tiene soporte para los lenguajes C++, C# y Visual Basic, los 2 anteriores exclusivos de Windows.

Todo este conjunto de componentes por separados, al integrarlos uno a uno se puede crear diversas aplicaciones con el sensor, donde este a través del código fuente desarrollado en el entorno de programación contendrá las instrucciones de la aplicación gracias a la API NUI del sensor con las que se pueden programar las 3 funcionalidades principales del sensor e innovar en nuevas ideas y llevarlas a una aplicación.

Pero la verdadera magia ocurre en el software que posee Kinect, este está desarrollado en cierta tecnología de inteligencia artificial conocida como Exemplar System , diseñada para procesar, analizar y aprender por sí misma la manera en que está constituido el cuerpo humano y la forma en que este puede moverse, no es de extrañarse que Microsoft por muchos años estuvo desarrollando este software observando múltiples videos de personas comunes y corrientes para así enseñar al software las distintas posturas que estas realizaban.

Todo este proceso le permite al desarrollador enseñar al software identificar las rodillas, los codos, manos, cara y otras 44 partes del cuerpo y llevarlo a aprender la manera en la que cada una de las articulaciones se mueve, con ello se crea una base de estadísticas y probabilidades con respecto a lo que el cuerpo humano puede hacer. Esto ayuda para que cuando Kinect obtiene la imagen del usuario este posee la base estadísticas que le informa que cierto pixel tiene 50% de probabilidades de pertenecer al brazo y 40% de pertenecer a la mano, y basado a esto , saber la posición en que se encuentra la persona , no puede cruzarse con el pixel que tiene el 70% de posibilidades de pertenecer al tobillo .

Para conocer la distancia que se encuentra cada pixel de la imagen de profundidad se emite una constelación de puntos con el emisor infrarrojo:

Figura 3. 4Haz de luz infrarroja que emite Kinect

Como muestra la figura 3.3 cuando la persona se para frente al controlador Kinect , este proyecta millones de pequeños haces de luz sobre todo lo que se encuentra frente a él y separa las formas humanoides del fondo , tras lo cual el software mide y juzga las distancias que hay entre los diferentes puntos del cuerpo y crea una especie de contorno rudimentario que representa una superficie en 3D; posteriormente , identifica las partes del cuerpo de la persona que está frente al sensor y basado en la experiencia con la forma humana y el comportamiento habitual que tiene el usuario crea la imagen en el controlador con las partes del cuerpo que reconoció.

3.2.2.1 Eskeleton Tracking

Traducido al español es, seguimiento del cuerpo (esqueleto), del usuario en tiempo real. Una vez que Kinect sabe dónde está la rodilla y los demás puntos de articulación , busca un esqueleto con 20 de estos puntos , que se adapte a la postura del usuario y lo sobrepone al contorno antes creado para cubrirlo con textura , ropa , cabello y todo lo demás que el juego determine lo que se tiene que ver en pantalla , solo para volver a realizar este algoritmo por más de 29 veces en el mismo segundo , de manera que los movimientos sean transmitidos a la consola Xbox 360 o al computador y esta los use para controlar el juego y tener una experiencia realmente fluida.

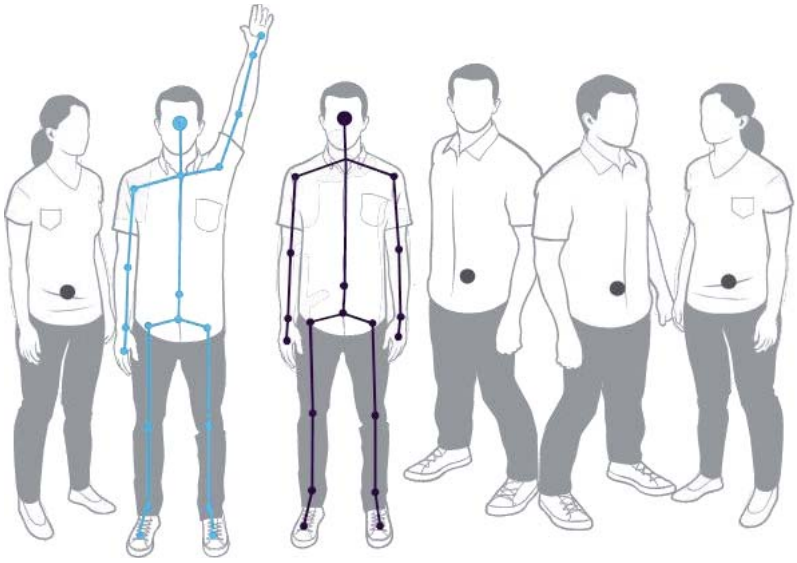

Figura 3. 5 Funcionamiento 3D de Skeleton Tracking

En la figura 3.4 se ve el tratamiento de los datos de imagen de profundidad para establecer las diferentes posiciones de las articulaciones del esqueleto en una forma humana. Por ejemplo el Skeleton Tracking determina el lugar donde se encuentra la cabeza del usuario, además de sus manos y su centro de masa. Skeleton Tracking proporciona X, Y y Z coordenadas para cada uno de estos puntos del esqueleto.

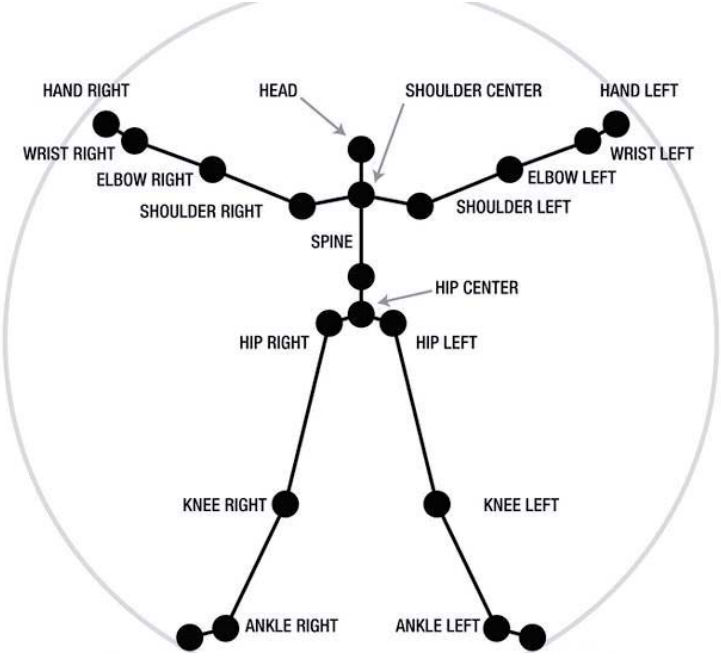

Figura 3. 6 Puntos del cuerpo que reconoce Kinect

La figura 3.5 [Jarret Webb, 2011] muestra los 20 puntos del cuerpo que el sensor Kinect reconoce del cuerpo humano gracias a su algoritmo Skeleton Tracking, se aprecia que el funcionamiento del sensor se basa en las principales articulaciones del cuerpo

Actualmente con el lanzamiento al mercado de Kinect for Windows viene incorporado el SDK propio para utilizar Kinect donde viene incluido skeleton tracking , con lo cual se ha creado una serie movimientos organizacionales para la creación de nuevos drivers y aplicaciones para Kinect opensource.

3.2.3 SDK Kinect

El SDK de Kinect para Windows es el conjunto de bibliotecas que permite programar aplicaciones en una variedad de plataformas de desarrollo para Microsoft utilizando el sensor Kinect como entrada. Con él es posible programar aplicaciones de WPF, Winforms , aplicaciones de XNA y con un poco más de trabajo incluso aplicaciones basadas en navegadores, pero aunque parezca extraño no se pueden crear juegos de Xbox 360 con el SDK [Jarret Webb, 2011].Con el SDK se puede sacar el máximo provecho a las capacidades de audio de Kinect, además posee aplicaciones adicionales de reconocimiento de voz con la plataforma Speech API. Al instalar SDK automáticamente se instalarán estos componentes adicionales. Es importante mencionar que las características de Kinect como reconocimiento de profundidad y seguimiento del esqueleto son componentes funcionales ya están incluidos en el SDK.

Tras dos versiones betas, es liberada la versión 1.0 , esta versión incluye algunas mejoras en comparación a sus predecesoras , con este SDK será posible conectar hasta 4 dispositivos Kinect en un mismo computador.

En Mayo de 2012 es lanzada la última versión 1.5 del SDK , ésta versión será la utilizar en el proyecto y presenta nuevas mejoras con respecto a sus predecesoras facilitando el proceso de desarrollo para las aplicaciones que se pretendan desarrollar: Las mejoras importantes en esta versión son las siguientes:

- Kinect Studio: Esta nueva herramienta permite a los desarrolladores grabar y reproducir datos del Kinect, acortando y simplificando el ciclo de vida del desarrollo de una aplicación de Kinect. Ahora un desarrollador puede grabar clips de los usuarios en el entorno de destino de la aplicación y luego reproducir los clips de película en un momento posterior para pruebas y desarrollo [Kinect, 2012].

- Seguimiento Facial: ofrece una malla 3D en tiempo real de características faciales, el seguimiento de la posición de la cabeza, la ubicación de las cejas, la forma de la boca, etc. Significativas adiciones y mejoras de código de ejemplo. Hay muchas nuevas

muestras, tanto en C + + y C #, además de un "básico" serie de muestras con la cobertura de idiomas en C + +, C # y Visual Basic [Kinect, 2012].

- Seguimiento del esqueleto sentado : Realiza un seguimiento de la cabeza además de 10 articulaciones , los hombros o los brazos del esqueleto, haciendo caso omiso la pierna y articulaciones de la cadera. No se limita a las posiciones de sentado, sino que también realiza un seguimiento de cabeza / hombros y los brazos cuando una persona está de pie. Esto hace posible la creación de aplicaciones que están optimizadas para los escenarios sentados (como el trabajo de oficina con el software de productividad o de interactuar con los datos en 3D) o escenarios de pie en el que la parte inferior del cuerpo no es visible para el sensor (por ejemplo, la interacción con un kiosco o cuando se navega a través de datos de resonancia magnética en una sala de operaciones) [Kinect, 2012].

Finalmente en esta nueva versión, se ha dividido en 2 herramientas fundamentales el SDK de Kinect . La primera es Kinect Toolkit, herramienta que permitirá a los desarrolladores ingresar a la documentación oficial y códigos de ejemplo para el posterior desarrollo de aplicaciones personalizadas por parte del desarrollador. Además posee las herramientas como Kinect Studio con su respectiva documentación y esta podrá ser instalada en el computador de manera fácil y práctica.

3.2.3.1 Requerimientos de Hardware

- Computador con procesador dual-core , 2.66 GHZ o superior.
- Windows 7 compatible con tarjetas gráficas y soporte de Microsoft DirectX 9.0
- 2GB de RAM (4GB es lo recomendado)
- Kinect for Windows o Kinect Xbox360.
- Adaptador USB (solo en caso que se utilice Kinect pra XBOX 360) [Kinect, 2012]..

Para desarrollar se necesita el entorno de desarrollo visual studio 2010 Express u otras ediciones 2010. Además se necesita tener DirectX 9.0 instalado en el computador. Las versiones posteriores de directX posteriores no son compatibles. También por supuesto se necesita la versión 1.0 del SDK de Kinect, el instalador del SDK instalará en el computador los controladores de Kinect, así como ejemplos de código que servirán de ayuda para los primeros pasos en el desarrollo de nuevas aplicaciones.

3.2.4. Osceleton, Skeleton Tracking

Osceleton Tracking es un pequeño programa que hace uso de los dato del esqueleto proporcionados por el OpenNiFramework para enviar por protocolo OSC las coordenadas correspondientes.

Un protocolo OSC (Open Sound Control) es un protocolo pensado para comunicar instrumentos musicales, dispositivos multimedia y ordenadores en tiempo real. Comúnmente es transportado por UDP.

Las características de este protocolo son:

- Ampliable, dinámico. Esquema de nombres simbólicos tipo URL.
- Datos numéricos simbólicos y de alta resolución.
- Lenguaje de coincidencia de patrones para especificar múltiples receptores de un único mensaje.
- Marcas de tiempo (time tags) de alta resolución.
- Mensajes empaquetados para aquellos eventos que deben ocurrir simultáneamente
- Sistema de interrogación para encontrar dinámicamente las capacidades de un servidor OSC y obtener documentación.

3.3 Guiado gestual de un robot humanoide mediante sensor Kinect

Estudio realizado por un alumno memorista [Pfeiffer, 2011] de la Facultad de Informática de la Universidad de Barcelona. El objetivo principal del estudio es básicamente controlar los brazos de un robot humanoide (Bioloid) a través del sensor Kinect. Para esto, logró implementar un firmware para poder controlar los actuadores del robot.

La comunicación entre el ordenador y el robot se realiza por serie (usando un adaptador USBserie en este caso). Con lo cual cuanto más cortos y sencillos sean los mensajes mejor reaccionaran las articulaciones.

En la etapa de implementación, se programa el firmware del robot biolid para el acceso a los dynamixel, esto quiere decir, que se capturan las coordenadas que entrega el sensor y luego estas son traspasadas al robot e interpretadas en el código fuente de este sabiendo que movimientos debe realizar según el formato de coordenadas XX YYYY que arroja el sensor.

Donde XX es el número identificador del Dynamixel que debe recibir la orden e YYYY es la cantidad de movimiento o el ángulo en que se quiere que vaya el motor.

Esta cantidad de movimiento esta codificada en una escala de 0 a 1023 representando los 360grados [Pfeiffer, 2011].

3.3.1 Programación del simulador de coordenadas de Kinect.

Se necesitarán las coordenadas de las manos en el mismo sistema de coordenadas en el que se encuentra la simulación. Para ello el primer paso es conseguir las coordenadas de las manos en referencia al pecho del usuario del Kinect. En este caso se toma el origen de coordenadas en el hombro izquierdo por comodidad. Luego solo hace falta hacer una traslación de coordenadas para situarlo en el origen del simulador (el centro del cubo que representa el torso [Pfeiffer, 2011].

Procedimiento:

Sean (x,y,z) las coordenadas del punto respecto a los ejes de coordenadas X-Y-Z.
Sean (x0, y0, z0) las coordenadas del origen de coordenadas de los ejes X-Y-Z respecto al nuevo sistema de coordenadas X'-Y'-Z'.

Sean a,b,c el ángulo que se gira el eje X' respecto a los ejes X-Y-Z, sean d, e, f el ángulo que se gira el eje Y' respecto a los ejes X-Y-Z, y sean g, h, i el ángulo que se gira el eje Z' respecto a los ejes XY-Z.

En la figura 3.6 [Pfeiffer, 2011] se observa el sistema de coordenadas obtenidos a través de los datos que entrega el sensor por medio de su algoritmo Skeleton Tracking

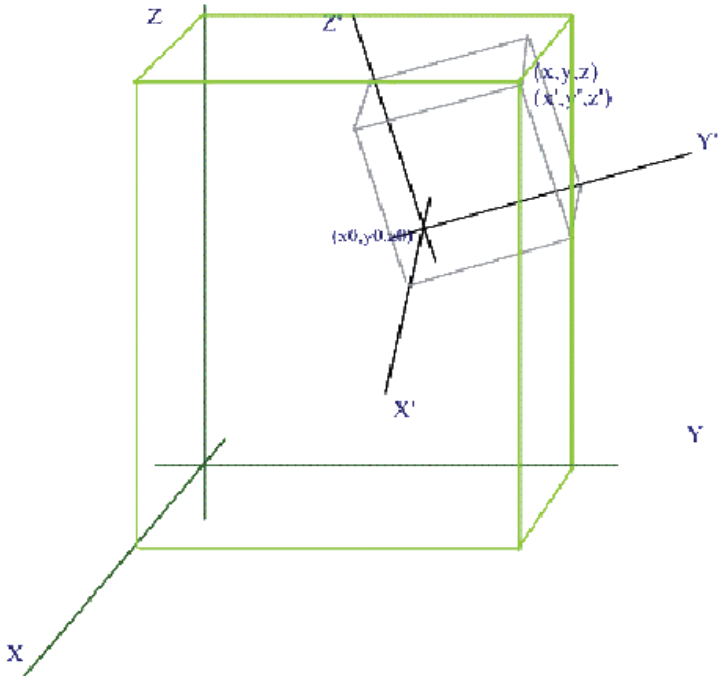

Figura 3. 7 Coordenadas que entrega el sensor

$$X = Xo + X' \cos(a) + Y' \cos(d) + Z' \cos(g)$$

$$Y = Yo + X' \cos(b) + Y' \cos(e) + Z' \cos(h)$$

$$Z = Zo + X' \cos(c) + Y' \cos(f) + Z' \cos(i)$$

Sera el sistema de ecuaciones que nos dará las coordenadas de nuestro punto en referencia al nuevo sistema de coordenadas.

3.3.2 Programación del escalado de las coordenadas obtenida tras el cambio se sistema de referencia

Dado que el cuerpo humano es distinto al cuerpo del Bioloid (principalmente el antebrazo sumado a la mano, es más largo que el brazo, mientras en el Bioloid pasa al contrario) se necesita hacer un escalado de las coordenadas del punto teniendo esto en cuenta.
Para el brazo izquierdo (el proceso es muy similar para el derecho):

La figura 3.7 [Pfeiffer, 2011] muestra el sistema de coordenadas interpretado por el sensor Kinect y el robot Bioloid.

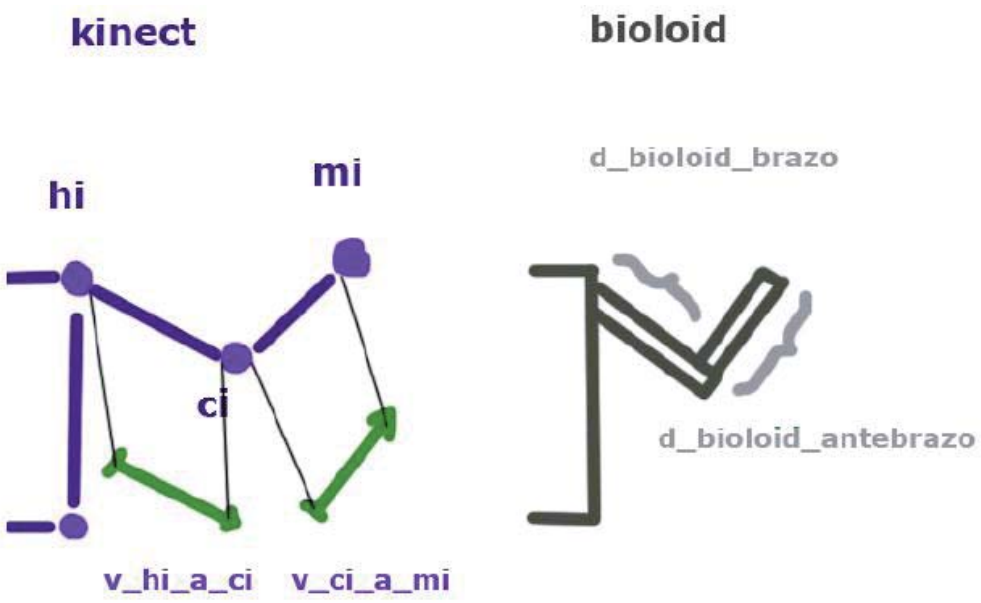

Figura 3. 8 Coordenadas traspasadas de Kinect al Robot

Hombro izquierdo, codo izquierdo, mano izquierda se refiere a las coordenadas de estos puntos:

d_brazo = distancia (hombro izquierdo, codo izquierdo)
d_antebrazo = distancia (codo izquierdo, mano izquierda)
d_bioloid_brazo = 0.09
d_bioloid_antebrazo = 0.043
d_todo_kinect = d_brazo + d_antebrazo
d_todo_bioloid = d_bioloid_brazo + d_bioloid_antebrazo
v_hi_a_ci = (codo izquierdo – hombro izquierdo)
v_hi_a_ci = (codo izquierdo / modulo (codo izquierdo)) * d_bioloid_brazo
v_ci_a_mi = (mano izquierda – codo izquierdo)
v_ci_a_mi = (v_ci_a_mi /modulo (v_ci_a_mi)) * d_bioloid_antebrazo
punto mano izquierda = v_hi_a_ci + v_ci_a_mi + (0.05, 0.05, 0.025)

Donde (0.05, 0.05, 0.025) es la traslación al origen de coordenadas.

Se utilizan los vectores proporcionales al tamaño del brazo y del antebrazo para situar Correctamente y en escala la posición de la mano.

3.3.3 Comparación con el proyecto.

Una vez estudiado el estado del arte, es necesario realizar una comparación entre el proyecto a desarrollar y el proyecto estudiado en este apartado. Si bien ambos proyectos son distintos, se asemejan en la captura de los datos del esqueleto por parte del usuario, para esto es necesario comparar los métodos utilizados en el código fuente para poder concluir algunas similitudes.

En el siguiente apartado se identifican los métodos utilizados en el proyecto a desarrollar para la captura de los datos de las articulaciones de la persona que se encuentra frente al sensor.

```
SetEllipsePosition(headEllipse, skeleton.Joints[JointType.Head]);

SetEllipsePosition(leftEllipse, skeleton.Joints[JointType.HandLeft]);

SetEllipsePosition(rightEllipse, skeleton.Joints[JointType.HandRight]);

SetEllipsePosition(elbowlefEllipse, skeleton.Joints[JointType.ElbowLeft]);

SetEllipsePosition(elbowrigthEllipse, skeleton.Joints[JointType.ElbowRight]);

SetEllipsePosition(hip, skeleton.Joints[JointType.HipCenter]);
```

```
GestosKinect(skeleton.Joints[JointType.Head], skeleton.Joints[JointType.HandRight],
skeleton.Joints[JointType.HandLeft], skeleton.Joints[JointType.ElbowLeft],
skeleton.Joints[JointType.ElbowRight],
skeleton.Joints[JointType.HipCenter],skeleton.Joints[JointType.ShoulderRight]);

GestosKinect(Joint cabeza, Joint manoDerecha, Joint manoIzquierda, Joint
codo_izquierdo, Joint codo_derecho, Joint tronco,Joint hombro)


 if (manoDerecha.Position.Y > cabeza.Position.Y){

// Movimiento hacia arriba del robot

}

else if (manoIzquierda.Position.Y > cabeza.Position.Y){

//Movimiento de la garra hacia abajo

}

else if(manoIzquierda.Position.Z<codo_izquierdo.Position.Z &&
manoIzquierda.Position.Y<tronco.Position.Y){

//Movimiento hacia la derecha del robot, se comparan las coordenadas y,z

}
```

En comparación al código del apartado anterior, este se diferencia bastante, debido a que la aplicación se ha desarrollado en la versión 1.5 del SDK Kinect, permitiendo casi una captura automática de los datos de las articulaciones. Una vez capturados las coordenadas, éstas se evalúan en condicionales y se identifica el gesto que el usuario está realizando, comprobada la condicional, posteriormente el robot realiza el movimiento que se le ha señalado.

Existe un punto en común entre ambas aplicaciones en lo que respecta al eje de coordenadas, ambas aplicaciones realizan la captura de coordenadas de las articulaciones en un plano cartesiano de 3 dimensiones (x, y, z).

Capítulo IV Análisis y diseño de la solución.

4.1 Introducción

En los capítulos anteriores se ha obtenido el marco teórico tanto de Robótica como del sensor con que el robot va a poder interactuar con la persona que es Kinect. Ahora en este capítulo se observará de qué manera se complementarán ambas tecnologías para la creación de la aplicación, cuál será la solución y descripción de esta, y las tecnologías (herramientas) que se utilizarán durante el proyecto para la creación de la aplicación que cumpla los objetivos mencionados en los apartados anteriores.

Para el éxito del estudio es necesario tener en consideración 2 casos de estudio en caso de que alguno de los 2 falle, existiendo una probabilidad que existan problemas de comunicación sobre todo con sensores como el infrarrojo, donde será necesaria la adquisición o fabricación de un adaptador.

4.2 Caso de Estudio o de Prueba. Kinect y Robophilo.

Como se mencionó anteriormente el principal objetivo del estudio es llegar a manipular un robot por medio del sensor Kinect donde la creación de un firmware permitirá que esto sea posible

Se puede identificar 2 tipos de actores principales del sistema, donde según el uso que se le dé a la aplicación, dependerá si es necesario que interactúen los 2 actores con el sistema o solo uno. El primer tipo de usuario es aquel que interactúa con el sensor y el robot, este deberá simplemente situarse frente al sensor y con el movimiento de sus brazos manipulará los movimientos del robot, es decir si el usuario realiza un movimiento con el brazo derecho, el robot también lo realizará. El proceso para el usuario parece ser sencillo , donde el sólo deberá realizar movimientos con su cuerpo para que los imite el robot. En una primera instancia del estudio, solo se implementará el movimiento de los brazos y no de otras articulaciones.

Si el uso de la aplicación es para fines médicos u otros, existe un segundo usuario que interactúa con el sistema , donde este recibirá los datos que entrega Skeleton Tracking de Kinect y analizará el sistema de coordenadas que este entrega, además del procesamiento de imágenes de cómo está realizando los movimientos la persona para realizar el estudio correspondiente y sacar conclusiones como por ejemplo, que un discapacitado ha

evolucionado en su tratamiento kinesiológico porque este presenta una evolución en sus movimientos desde que comenzó el tratamiento.

Ambas tecnologías (Kinect y Robot) deben conectarse y comunicarse para el traspaso de datos por medio de un sensor infrarrojo. RoboPhilo al funcionar con un control remoto ,envía los datos al receptor IRDA que posee el robot e interpreta las instrucciones para que el robot funcione. Es por que el sensor Kinect actuará como el control remoto, emulándolo para poder enviar los datos a través del adaptador infrarrojo y manipular el robot por medio de las instrucciones que le sean enviadas.

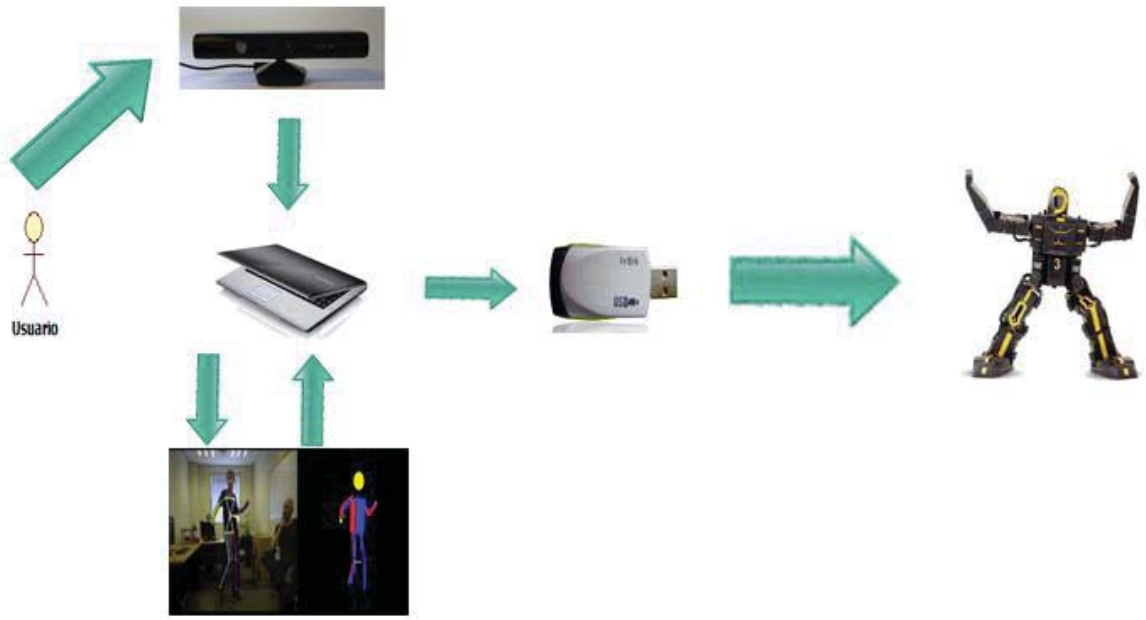

Figura 4. 1 Esquema funcionamiento sistema de control robótico mediante Kinect y roboPhilo

En la figura 4.1 se aprecia el funcionamiento del sistema donde interactúan los 2 actores identificados. El primer actor corresponde al usuario en que interactúa con el sensor Kinect , donde Kinect a través del sensor de movimiento y el procesamiento de imagen envía los datos capturados al software Skeleton Tracking, este los procesa, se capturan y son traspasados a un sistema de coordenadas a través del SDK donde luego de procesar los datos estos se interpretan en el código fuente del robot y permite el movimiento de este a partir de los datos que envía Kinect de los movimientos que realice el usuario que está situado frente al sensor.

En el caso del segundo usuario no será necesario que este se situé frente al sensor, este interpretara los movimientos de la persona que si está utilizando el sensor y por medio de skeleton tracking analizará los movimientos que le permita sacar conclusiones acerca del

contexto en que se está aplicando el sistema. Luego el funcionamiento principalmente es el mismo que el usuario anterior.

De esta manera ambos usuarios interactúan con el sistema dando funcionalidades distintas a la aplicación. Es importante señalar que gracias a los respectivos SDKS y los sensores infrarrojos que poseen ambas tecnologías se puede lograr el envío de datos entre ellos.

4.3 Caso de Estudio o de Prueba. Kinect y Lego Mindstorm NXT.

El principal objetivo del estudio es la manipulación directa de un robot por medio del sensor Kinect. En el caso de estudio número 1 se ha especificado el funcionamiento del sistema y la forma en que se comunica el sensor con el roboPhilo. Sin embargo en este caso de estudio, es compleja la comunicación entre el robot y el sensor, ya que se realizada por medio de un sensor infrarrojo, tecnología obsoleta en la actualidad siendo esta desplazada por Bluethoot. Para esto será necesario adquirir o construir un adaptador USB que actué como emisor y receptor de la luz infrarroja y así poder traspasar los datos emulando al control remoto que manipula el robot por medio del sensor Kinect.

Especificadas algunas de las complejidades que puede presentar la comunicación entre ambas tecnología es necesario planificar un segundo caso de estudio en caso de que falle la realización del primero manipulando el robophilo.

Considerando que pueden existir varias formas de manipular algún tipo de robot por medio del sensor Kinect se ha seleccionado el robot Lego Mindstorm NXT como segundo caso de estudio. Este robot se comunica por medio de Bluethoot con otros hardwares, por lo tanto será necesario un adaptador de Bluethoot para la comunicación entre el robot y el sensor Kinect.

El usuario realizará los movimientos con sus brazos frente al sensor Kinect, éste interpretará el movimiento de la persona en coordenadas X,Y,Z gracias a Skeleton Tracking estos datos serán traducidos y transmitidos al robot a través de la librería NxtNET, adaptada para soportar códigos fuentes C# para la manipulación del robot Lego y en donde se puede realizar la conexión con el Bluethoot en el mismo código donde se realiza el seguimiento del esqueleto con Kinect , además de indicar que servomotor es el que se quiere activar para desplazamiento del robot y la potencia de este.

Básicamente el objetivo al igual que el caso de estudio anterior es el mismo, la diferencia está en que el robot Lego al ser más limitado en los movimientos no tendrá la misma funcionalidad que el robophilo, qué si busca imitar los movimientos de un ser humano

al poseer 23 servomotores dando un mayor grado de movilidad. Por lo tanto el robot Lego se limitará a realizar solo una serie de movimientos según su capacidad, un ejemplo de aplicación puede ser la simulación de un automóvil donde el robot según los movimientos de la persona puede realizar los movimientos de avanzar, girar, retroceder, frenar, etc. De esta manera ambos usuarios interactúan con el sistema dando funcionalidades distintas a la aplicación. Es importante señalar que gracias a los respectivos SDKS y al sensor Bluethoot se puede lograr el envío de datos entre ellos.

El envió de datos desde el computador al robot se realiza mediante las librerías anteriormente descritas, dentro de la misma aplicación Kinect se invocan a los métodos de las clases del robot Lego y se manipulan sus motores y/o sensores si así se desea. La ventaja de utilizar estas librerías es que se puede desarrollar toda la aplicación en un mismo código, en este caso en el lenguaje de programación C# a través del entorno de desarrollo Visual Studio 2010.

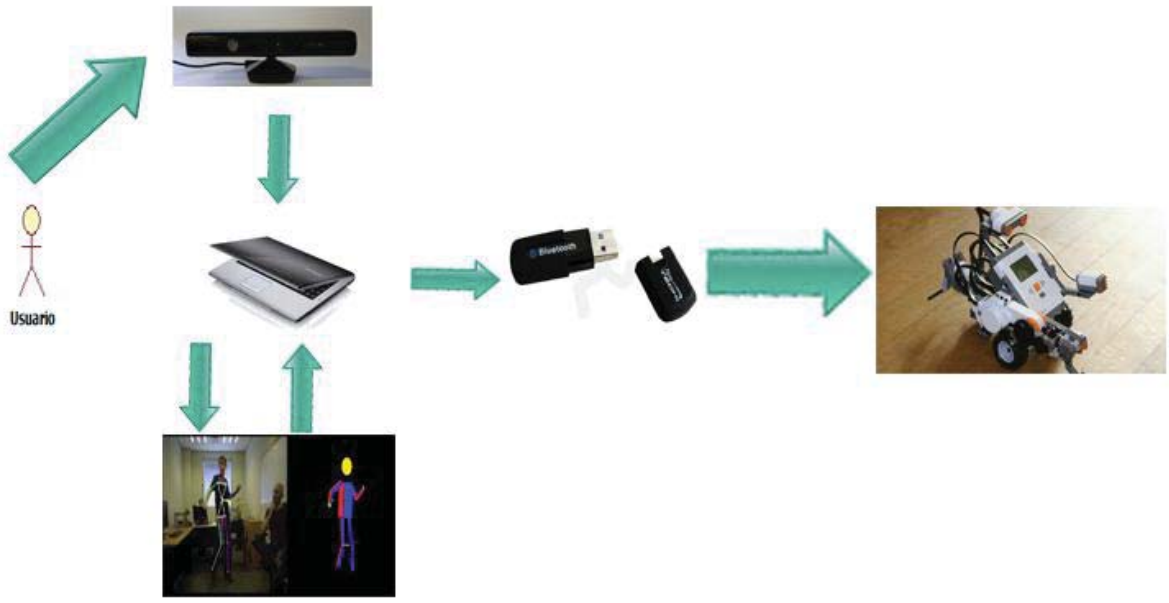

Figura 4. 2 Esquema funcionamiento sistema de control robótico mediante Kinect y LEGO NXT

La Figura 4.2 detalla el funcionamiento del sistema de control robótico entre Kinect y el robot Lego NXT. Este corresponde al segundo caso de estudio pero básicamente tiene el mismo modo de interactuar de las partes, donde se diferencia al anterior en que utiliza un

adaptador USB Bluetooth que comunicará directamente al computador con el Robot Lego NXT.

4.3.1 Diagramas UML de la aplicación

Para este proyecto es necesario realizar algunos diagramas de suma importancia para comprender como interactúa el sistema con el usuario, e identificar distintos roles y actividades que debe cumplir cada uno de los componentes que son parte del sistema .

Para lograr dicho objetico es necesario modelar un caso de uso para identificar las distintas actividades que deberá realizar al usuario al interactuar con la interfaz del sistema y poder mover el robot , por otra parte , es necesaria también la creación de un diagrama de actividades que tal como lo dice su nombre identifica cada una de las actividades del sistema , pero esta vez involucrando a todos los componentes del software a desarrollar.

4.3.1.1 Diagrama de caso de uso (Escenario: Usuario frente al sensor).

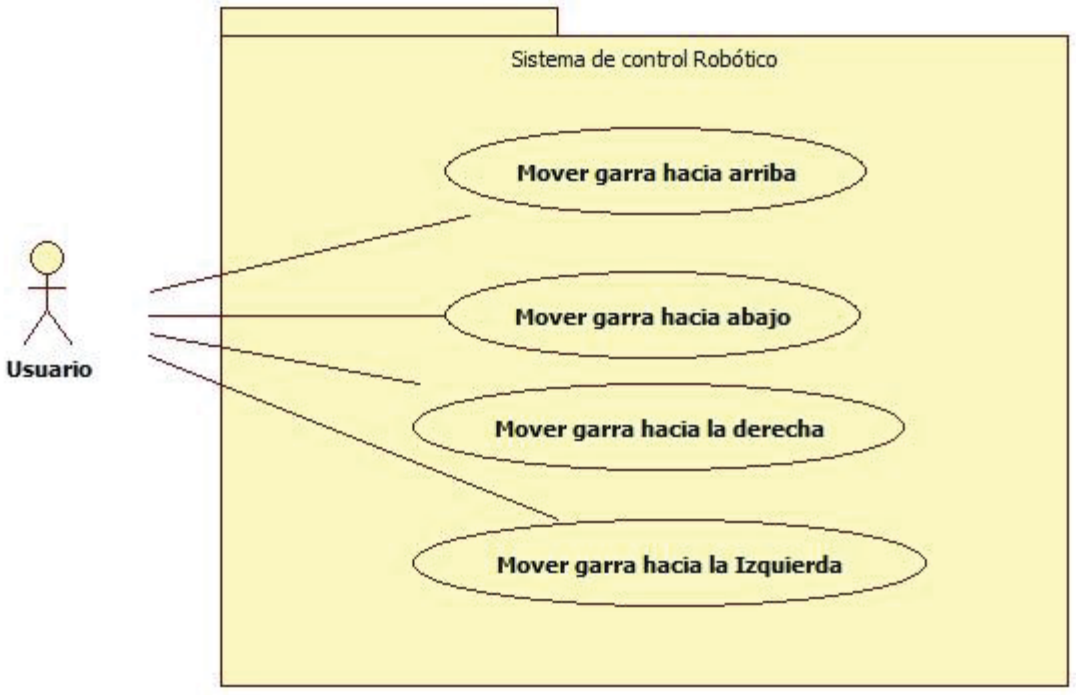

Figura 4. 3 Diagrama de caso de uso del sistema

En el diagrama de caso de uso se puede identificar el único actor que tiene el sistema, que corresponde a la persona que se sitúa frente al sensor, la aplicación al dibujar el esqueleto de la persona, ya está apta para poder interpretar las coordenadas que le envíe el usuario a través de los gestos que esta realice, para poder así enviarlas al robot y que esta imite los movimientos que realiza el usuario.

Además se puede identificar las 4 principales acciones que puede realizar el usuario para poder manipular el robot a través del intérprete que es la aplicación, estas corresponden a mover garra arriba, abajo, mover hacia la izquierda y mover hacia la derecha.

4.3.1.2 Diagrama de actividades.

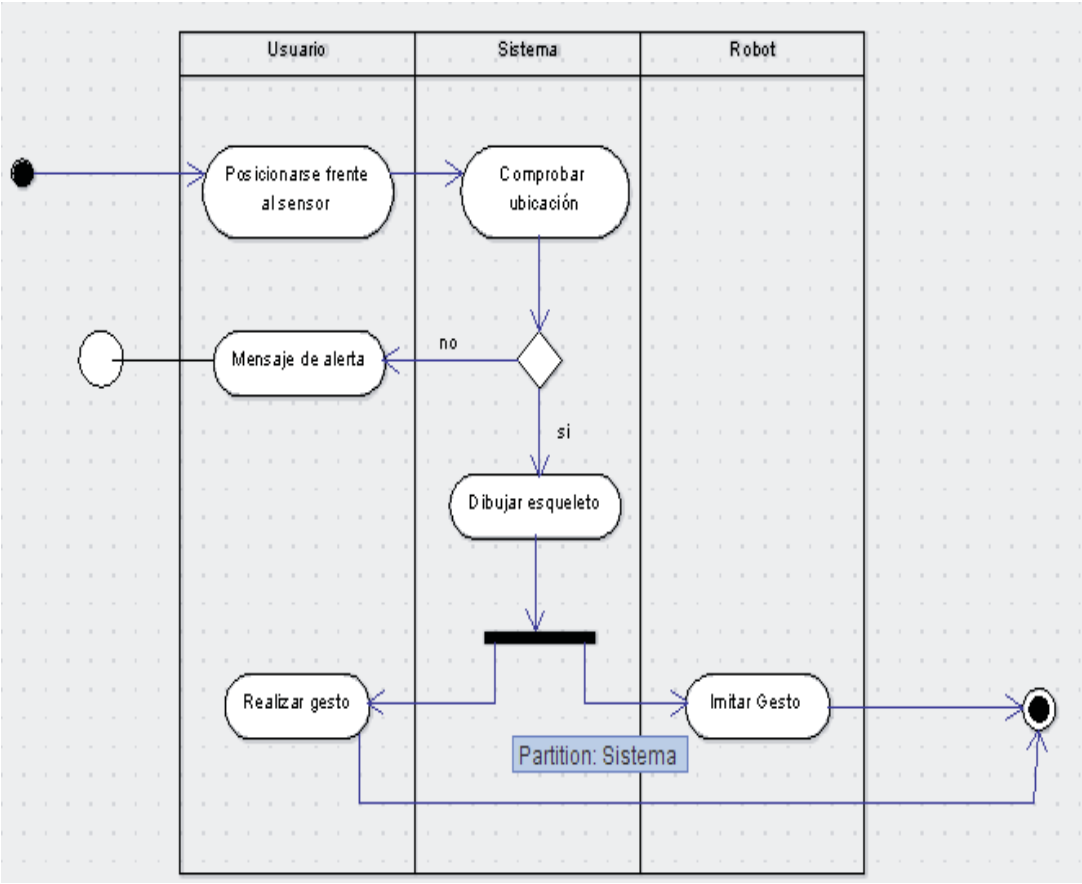

Figura 4. 4 Diagrama de actividades

Este diagrama principalmente, muestra el flujo de actividades que se realiza con cada uno de los componentes del sistema, como interactúan entre sí para que la aplicación funcione en forma correcta y óptima para lograr los objetivos planteados del proyecto.

4.3.2 Descripción de la Aplicación.

La aplicación es muy intuitiva para el usuario, básicamente lo único que debe realizar es situarse frente al sensor Kinect, esperar a que las bolas de colores dibujen la parte de la persona y luego realizar los movimientos que permitan mover al robot . La aplicación de forma interna moverá el robot, sólo se permitirán una cantidad limitadas de gestos, ya que el robot posee 3 servomotores, por lo tanto, los movimiento serán, mano derecha o izquierda arriba (el robot desplaza el brazo hacia arriba),mano derecha o izquierda bajo abajo (el brazo del robot se desplaza hacia abajo), mano deslizando hacia la derecha el robot mueve su eje hacia esa dirección , mano izquierda el robot mueve su eje en esa dirección y finalmente ambas manos juntas permitieran activara el servomotor de la garra que permitirá manipular distintos utensilios y moverlos del lugar.

La aplicación en forma dinámica informará por pantalla los gestos que realiza el usuario, el número de gestos estará limitados al movimiento de las manos, en caso de que el usuario realice un gesto que no esté dentro de la aplicación , ésta le informa al usuario a través por pantalla que el gesto no existe y que debe volver a realizarlo, esto se debe a las limitaciones del robot de realizar a lo más 5 movimientos, si se tratase de otro robot como el Philo aumenta considerablemente la cantidad de movimientos que se pueden realizar, sin embargo en esta versión del software solo se limita a manipular el robot Mindstorms NXT.

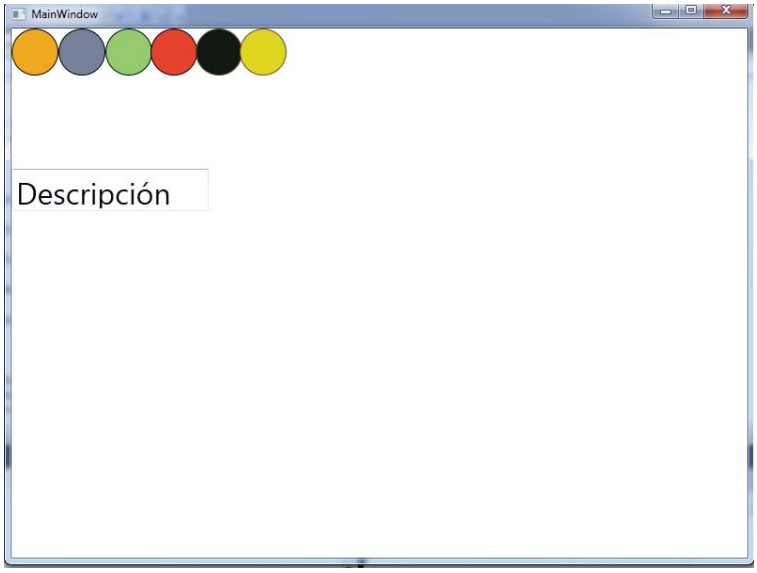

Figura 4. 5 Inicio del simulador a través del ejecutable

En la figura 4.3 se aprecia el simulador ya en funcionamiento, se pueden distinguir distintas elipses que representan a las partes del cuerpo que reconocerá Kinect a través de

Skeletan Tracking. Estas partes del cuerpo representadas en distintos colores corresponde a las articulaciones de la cabeza (elipse naranja) , manos (elipse verde y elipse gris), codos (elipse roja, elipse negra) y la cintura (elipse amarilla).

En la parte inferior de las elipses se encuentra la descripción del movimiento que realizará la persona que se sitúe frente al sensor Kinect. Una vez que la persona esté frente al sensor automáticamente el simulador dibujara las partes del cuerpo que este reconocen y seguirá los movimientos de la persona.

Posteriormente la persona puede realizar distintos gestos, cuando el simulador determine un gesto que este dentro de los predeterminados para manipular el robot en la descripción del programa indicará que gesto es el que ha realizado y automáticamente el robot se moverá hacia la dirección que le indique el usuario. De esta forma el simulador luego la unión de los 2 hardware (Kinect y robot) y los hace interactuar entre sí.

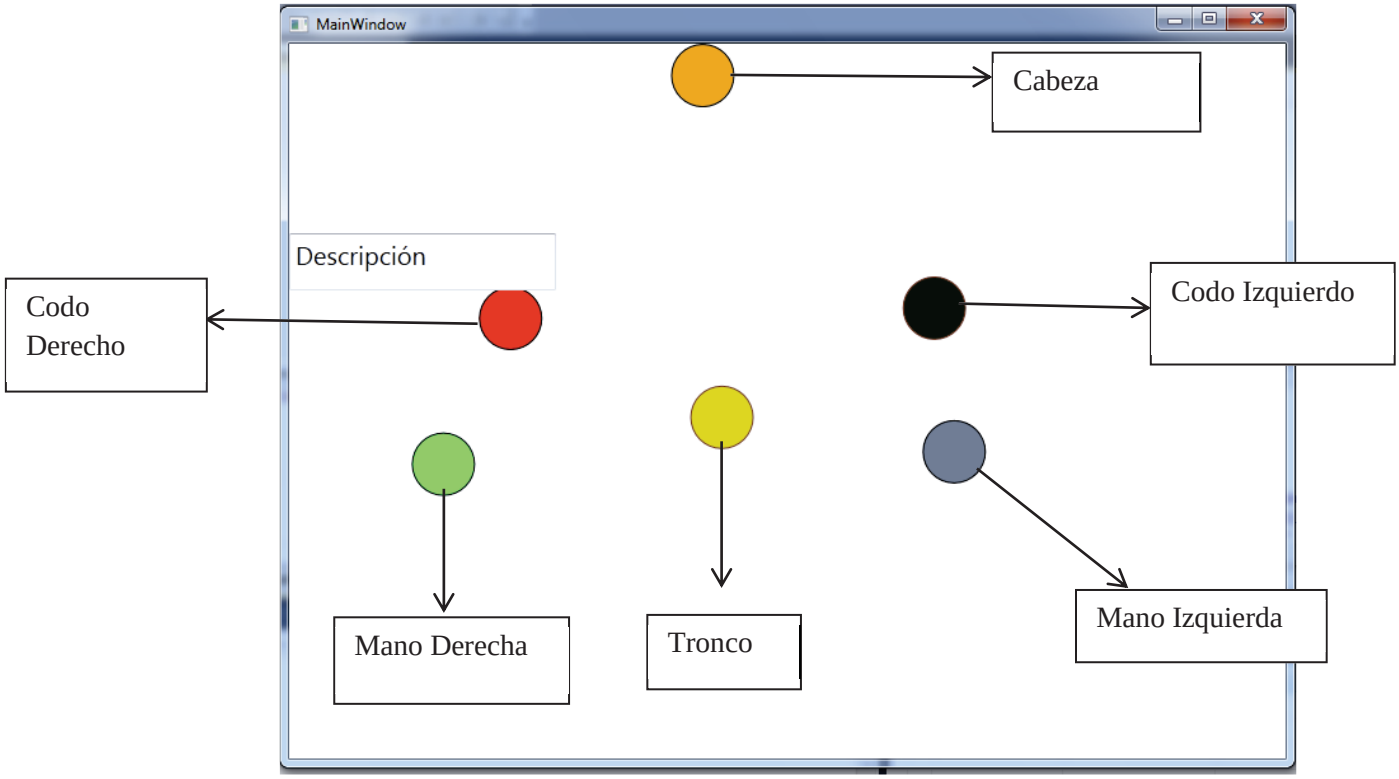

Figura 4. 6 Reconocimiento de articulaciones

En la figura 4.4 se observa como el simulador reconoce las articulaciones de la persona y las dibuja en pantalla. Una vez reconocidas las articulaciones están seguirán todos los

movimientos que la persona realice, es importante señalar que cada elipse representa una articulación distinta como se explicó anteriormente.

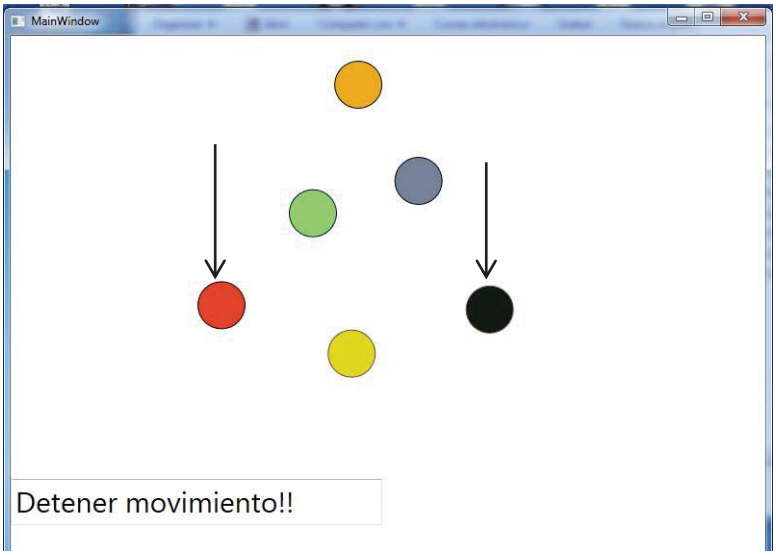

Figura 4. 7 Reconocimiento de gesto bajar la mano

En la figura se observa como el simulador identifica el gesto de bajar la mano izquierda o derecha bajo el tronco. Una vez que el simulador reconoce el gesto, informa al usuario a través de la descripción y muestra por pantalla el gesto que la persona ha realizado frente al sensor. A su vez y en tiempo real el robot desactiva los servomotores por lo que quedará completamente inactivo hasta que se le de una nueva instrucción, es decir, la aplicación reconoce el gesto que ha realizado el usuario y por medio de ésta se le informa al robot que acción es la que debe seguir.

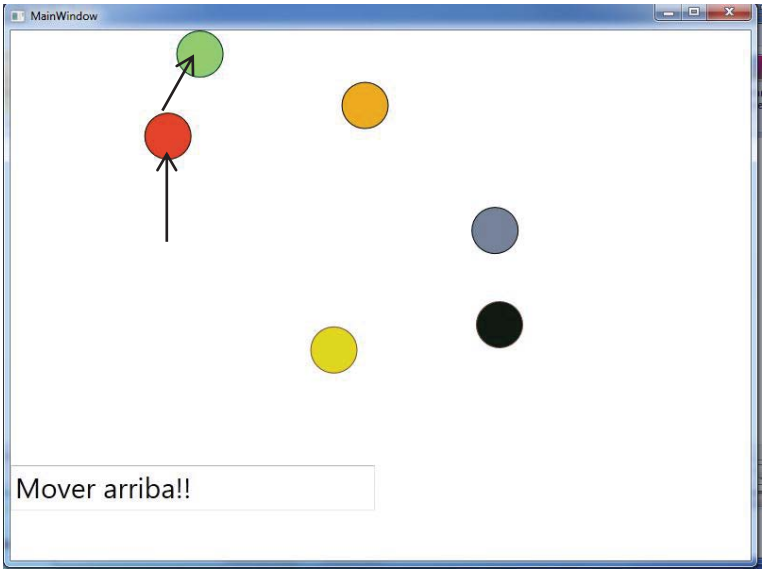

Figura 4. 8 Reconocimiento de gesto levantar mano derecha

En la figura se observa como el simulador identifica el gesto de levantar la mano derecha. Una vez que el simulador reconoce el gesto, informa al usuario a través de la descripción y muestra por pantalla el gesto que la persona ha realizado frente al sensor. A su vez y en tiempo real el robot activa el servomotor que lo hará moverse hacia arriba, es decir, la aplicación reconoce el gesto que ha realizado el usuario y por medio de ésta se le informa al robot que acción es la que debe seguir.

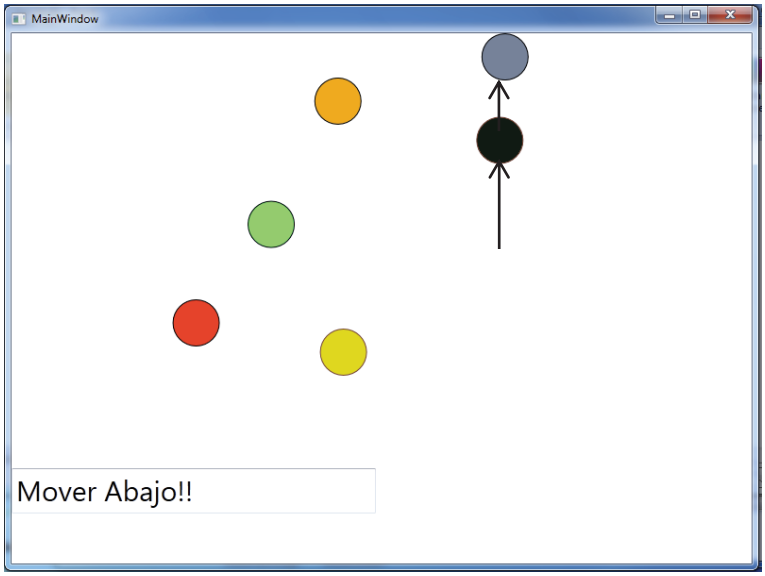

Figura 4. 9 Reconocimiento levantar mano iquierda.

En la figura se observa como el simulador identifica el gesto de levantar la mano izquierda. Una vez que el simulador reconoce el gesto, informa al usuario a través de la descripción y muestra por pantalla el gesto que la persona ha realizado frente al sensor. A su vez y en tiempo real el robot activa el servomotor que lo hará moverse hacia abajo, es decir, la aplicación reconoce el gesto que ha realizado el usuario y por medio de ésta se le informa al robot que acción es la que debe seguir.

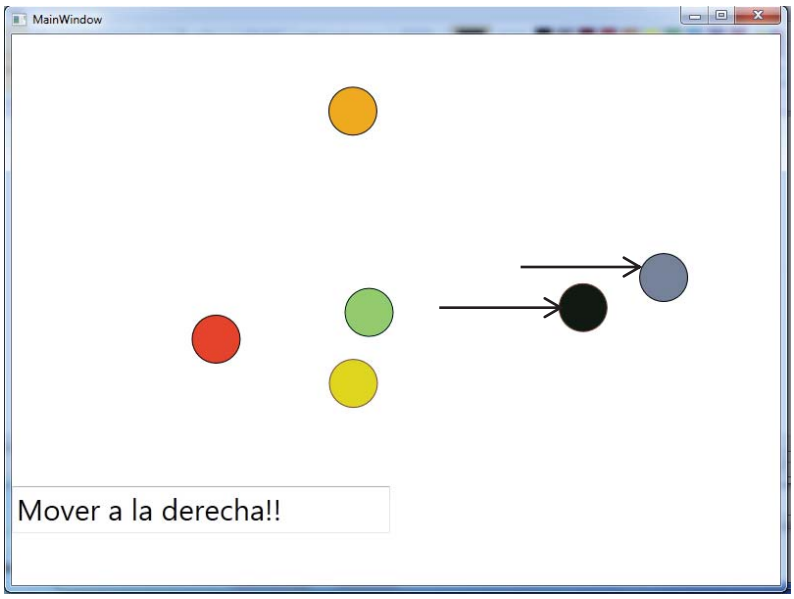

Figura 4. 10 Reconocimiento de gesto mover hacia la derecha.

En la figura se observa como el simulador identifica el gesto de desplazar la mano derecha. Una vez que el simulador reconoce el gesto, informa al usuario a través de la descripción y muestra por pantalla el gesto que la persona ha realizado frente al sensor. A su vez y en tiempo real el robot activa el servomotor que lo hará moverse hacia la derecha, es decir, la aplicación reconoce el gesto que ha realizado el usuario y por medio de ésta se le informa al robot que acción es la que debe seguir.

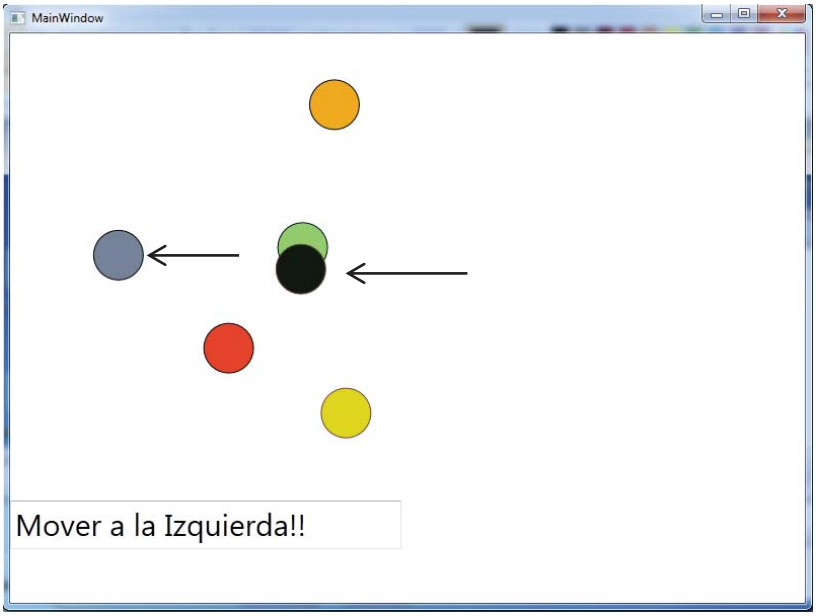

Figura 4. 11 Reconocimiento de gesto mover hacia la iquierda

En la figura se observa como el simulador identifica el gesto de desplazar la mano derecha pero en esta oportunidad hacia el sentido izquierdo. Una vez que el simulador reconoce el gesto, informa al usuario a través de la descripción y muestra por pantalla el gesto que la persona ha realizado frente al sensor. A su vez y en tiempo real el robot activa el servomotor que lo hará moverse hacia la izquierda, es decir, la aplicación reconoce el gesto que ha realizado el usuario y por medio de ésta se le informa al robot que acción es la que debe seguir.

4.4 Entorno de desarrollo para la aplicación

En el siguiente apartado se explicará un breve pero efectivo tutorial para comenzar a desarrollar con el SDK de kinect y comprender como ha sido el desarrollo del software.

En una primera instancia se crea crear un proyecto apto para la programación con el SDK de Kinect para Windows y que servirá como proyecto base para cualquier aplicación que se desee desarrollar.

Abriremos Visual Studio 2010 y nos crearemos un proyecto *WPF* (Windows Presentation Foundation).Le damos un nombre y aceptamos.

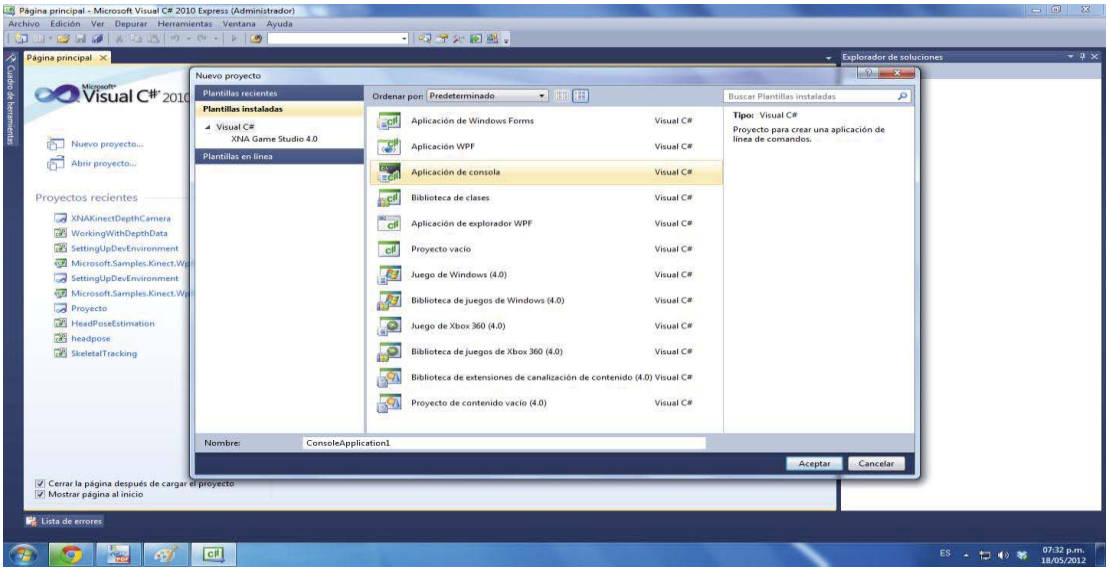

Figura 4. 12 Vista de la creación de un proyecto WPF en Visual studio 2010

Cuando finalice el proceso de creación y preparación de la solución, se abrirá el panel de diseño con la ventana principal (MainWindows.xaml) de la aplicación y el explorador de proyectos a la derecha. Desplegaremos el nodo de MainWindow.xaml para acceder al fichero MainWindow.xaml.cs (el *code-behind* de la ventana) para empezar a programa en él.

Para poder programar con el SDK de Kinect deberemos agregar una referencia a nuestro proyecto. Presionamos con el botón derecho del ratón encima de "*References*" elegimos "añadir referencia". Nos aparecerá un cuadro de diálogo en el que seleccionaremos la referencia al SDK "Microsoft.Research.Kinect" y aceptamos. Después de aceptar veremos como se ha agregado una nueva referencia a nuestro proyecto que corresponde al SDK.

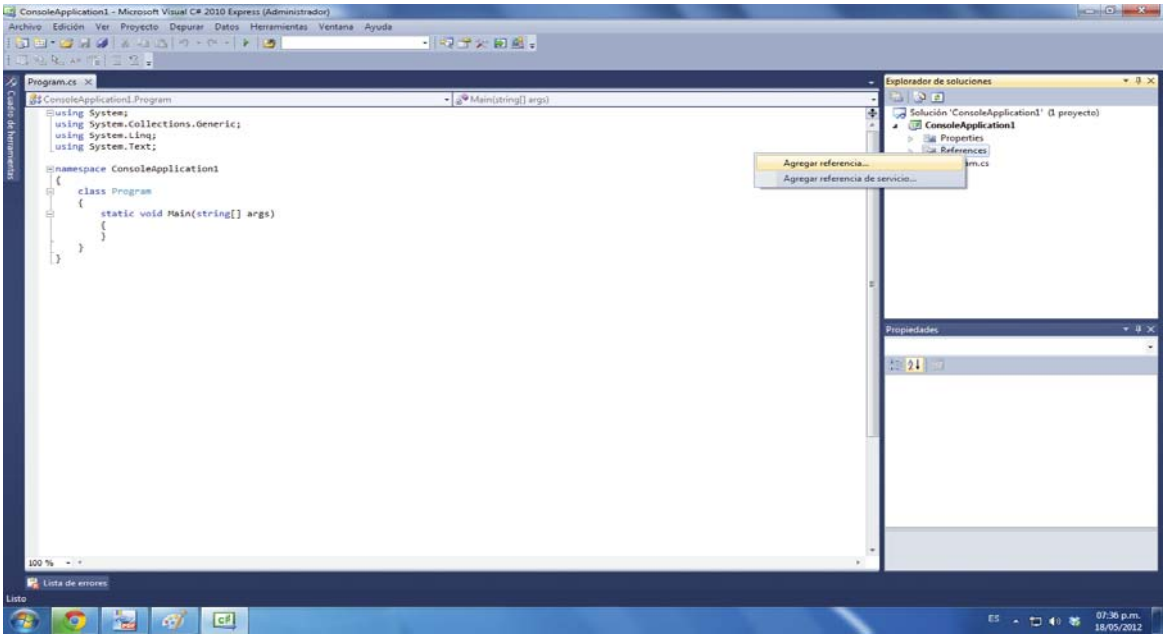

Figura 4. 13Vista de realizar una referencia a la biblioteca de Kinect

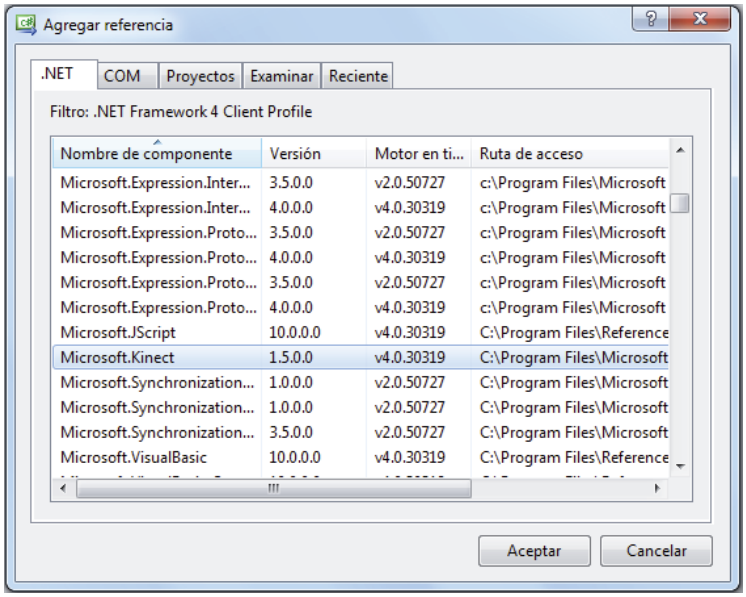

Figura 4. 14 Referencia a la biblioteca del SDK de Kinect 1.5

Con la referencia agregada añadiremos el espacio de nombres a junto con los demás *usings* del archivo para poder utilizar las clases y métodos del SDK.

Con estos pasos ya es posible realizar la aplicación que se desee en el SDK.

Capítulo V Estudio de factibilidad.

5.2 Factibilidad Económica

Este estudio tiene como objetivo comprobar que tan factible es la realización del proyecto por medio de los costos de producción y operación del proyecto.

El proyecto será esta evaluado durante 3 meses desde su implementación.

Fuente de financiamiento para el proyecto: Escuela de Informática de la Pontificia Universidad Católica de Valparaíso.

5.2.1 Costos de producción.

Kinect	$125.000
Robot Lego	$225.000
Notebook Samsung r430	$350.000
Monitor LCD	$90.000
Adaptador RS232	$3.500
Adaptador Bluetooth	$7.000
TOTAL	$ 805.000

Tabla 5. 1 Costos de producción del proyecto

La tabla 5.1 muestra los costos asociados a la producción del proyecto donde indica todas las tecnologías que implica la realización del proyecto.

5.2.1 Costos de operación

Integrantes : 1 persona	$0
Horas semanales : 15 hrs	$0
Costo de envío Kinect	$20.000
Total:	$20.000

Tabla 5. 2 Costos de operación del proyecto

La tabla 5.2 muestran los costos de producción del proyecto donde no se consideran las horas hombres ya que este es un estudio con fines educativos.

5.2.3 Costo total del proyecto.

Costos de producción + Costos de operación = $825.000

Por lo tanto el desarrollo del sistema de control robótico mediante Kinect tiene un costo total de $825.000 pesos , donde al ser un proyecto con fines educativo no tendrá gastos en lo que respecta a horas hombres pero sí a lo que respecta a las 2 principales tecnologías que interactúan que son el sensor Kinect y el robot

5.3 Factibilidad Técnica.

La factibilidad técnica, consistió en realizar una evaluación de la tecnología requerida para la implementación del sistema propuesto y de ser necesarios los requerimientos tecnológicos que deben ser adquiridos para el correcto funcionamiento de la misma.

Principalmente en lo que respecta a tecnología se han dividido en 2 tipos para el desarrollo del sistema, estas son las siguientes:

5.3.1 Hardware

En lo que implica al Hardware del proyecto en sí, corresponde al computador a utilizar, este debe ser lo suficientemente potente para poder compilar los código que se desarrollen en los SDK de Kinect y de RoboPhilo , en consecuencia la potencia que necesita el computador depende de los requerimientos de hardware que posean los SDK correspondientes.

Además se especifica los adaptadores que se necesitan para el éxito del proyecto, ya que las entradas en el computador en el mercado hoy se encuentran obsoletas y sin estos adaptadores sería mucho más compleja la comunicación entre las tecnologías.

5.3.1.1 Requerimientos de Hardware SDK Kinect

Los requerimientos de hardware para el SDK Kinect for Windows son los siguientes [Microsoft, 2012]:

- Procesador 32bit(X86) o 64 bit (X64)
- Procesador dual-core 2.66 GHz o procesador faster.
- Bus USB 2.0 Dedicado.
- 2 GB RAM

5.3.1.2 Requerimientos de Hardware SDK RoboPhilo

Los requerimientos de Hardware para el SDK de Kinect son los siguientes [Robophilo, 2007] :

- Memoria RAM 512 MGBytes o más
- Interfaz: Puerto serie (RS-232) o en su defecto un adaptador USB.

5.3.1.3 Adaptador infrarrojo con puerto USB.

El Adaptador USB a Infrarrojos, USB2IR3 permite convertir un puerto USB en un puerto infrarrojo compatible con IrDA proporcionando una comunicación con una amplia variedad de dispositivos compatibles con IrDA, como PDAs, teléfonos celulares, equipos de prueba e instrumental médico.

Un diseño compacto y de bajo consumo que funciona en un rango de hasta 1 metro con enlaces en línea de vista y soporte para modos FIR (4 Mbps), MIR (1,15 Mbps), SIR (115,6 Kbps) y ASKIR (56 Kbps) que proporcionan compatibilidad con una amplia gama de dispositivos infrarrojos.

Figura 5. 1 Adaptador infrarrojo USB

5.3.1.4 Adaptador USB cable serial RS232.

Son adquiridos fácilmente en el mercado, Robophilo viene con un cable serial RS232 ,donde ya se encuentra obsoleta este tipo de entrada para los computadores en la actualidad. Por lo tanto es necesario adquirir un adaptador para transformar esta entrada en una USB y poder realizar sin inconvenientes la descarga del código fuente del robot a este para que este interprete las instrucciones que se han codificado en el SDK del roboPhilo.

Figura 5. 2 Adaptador USB Serial RS232

5.3.1.5 Adaptador Bluethoot USB Plug And Play 2.0

Adaptador USB Bluetooth que permite a su ordenador de escritorio o portátil conectarse sin cables a cualquier dispositivo Bluetooth. Permite a los equipos y dispositivos un rango de trabajo de más de 10 metros con una transferencia de 2.1 Kbps en velocidad de datos. Cumple con Bluetooth 2.0 y especificaciones 2.0. Con este adaptador será posible conectar el robot Lego con el PC que contendrá las instrucciones desde el sensor Kinect.

Figura 5. 3 Adaptador USB Bluethoot

5.3.2 Software

En el apartado anterior se ha descrito los requerimientos que debe tener el computador para poder compilar los códigos fuentes tanto para Kinect como para RoboPhilo. Además de los adaptadores necesarios para la correcta comunicación entre las tecnologías para ambos casos de estudio. Ahora a continuación se describe todo lo necesario con respecto a los software utilizados para el desarrollo de la aplicación.

5.3.2.1 SDK RoboPhilo.

Estudiado más en profundidad en el capítulo 2.El software es muy intuitivo, ofrece una estructura de datos orientada a objeto. Con esta estructura jerárquica se pueden crear una amplia gama de bibliotecas de secuencias de movimiento reutilizables. Estas se desglosan en 6 pestañas distintas, que ayuda a que sea más fácil de entender y trabajar con ellas. Estas pestañas corresponden a configuración, pose de rutina, secuencia, clave y puesta a punto. Soporta como lenguaje de programación C y C++.

5.3.2.2 SDK Kinect

También estudiado más en profundidad en el capítulo 3. El SDK de Kinect para Windows es el conjunto de bibliotecas que permite programar aplicaciones en una variedad de plataformas de desarrollo para Microsoft utilizando el sensor Kinect como entrada. Al instalar SDK automáticamente se instalarán estos componentes adicionales. Es importante mencionar que las características de Kinect como reconocimiento de profundidad y seguimiento del esqueleto son componentes funcionales ya están incluidos en el SDK.

5.3.2.3 C++

Diseñado a mediados de 1990, C++ es un lenguaje de programación de propósito general y de orientación a objetos basado en el lenguaje C. Además de los recursos de C , C++ proporciona clases, funciones en línea, sobrecarga de operadores , sobrecarga de nombre de funciones , tipos constantes , referencias , verificación de argumentos de funciones y conversiones de tipo [Ellis-Stroupstrut, 1990].

5.3.2.4 C#

Es un elegante lenguaje de programación orientado a objetos que permite a los desarrolladores crear una variedad de aplicaciones seguras y robustas que se ejecutan en. NET Framework.

La sintaxis de este lenguaje es muy expresiva, es un lenguaje fácil de aprender por el desarrollador que esta familiarizado con JAVA o C++. La sintaxis de C# simplifica muchas de las complejidades de C + + y proporciona potentes funciones, tales como tipos de valor que aceptan valores NULL, enumeraciones, delegados, expresiones lambda y el acceso directo a la memoria, que no se encuentran en Java.

Como es un lenguaje orientado a objetos, C # admite los conceptos de encapsulación, herencia y polimorfismo. Todas las variables y métodos, incluyendo el método principal, punto de entrada de la aplicación, se encapsula dentro de las definiciones de la clase [Microsoft, 2012].

5.3.2.5 Microsoft Visual C++ 2010

Visual C++ Proporciona un entorno de desarrollo potente y flexible de aplicaciones basadas en Microsoft Windows y Microsoft .NET. Se puede utilizar en un sistema integrado de desarrollo o puede utilizar las herramientas de forma individual. Visual C++ se compone de los siguientes componentes [Microsoft, 2012]:

- Herramientas de Compilación.
- Bibliotecas.
- Desarrollo para el medio ambiente (código fuente).

5.4 Análisis de riesgo.

Para conseguir un buen análisis de riesgo es necesario identificar los posibles riesgos que puede tener el desarrollo del sistema. Una vez identificado los posibles riesgos se seleccionan los 5 principales o más importantes y se ordenan en orden de prioridad para luego calcular el factor de que tanto impacto podría generar ese riesgo en el proyecto.

5.4.1 Orden de prioridad riesgos más importantes.-

- Que las tecnologías no sean compatibles y no se puedan comunicar.
- Que el sistema no cumpla con los objetivos planteados del estudio.
- No adquisición del Hardware utilizado en el proyecto.

- Obtención del marco teórico y estado del arte incorrecto.
- Mala Programación de las actividades.

Riesgo	Probabilidad	Impacto	Exposición	Prioridad
Las tecnologías no son compatibles	3	4	12	1
El sistema no cumpla con los objetivos.	2	4	8	2
No adquisición del Hardware utilizado en el proyecto	2	3	6	3
Obtención del marco teórico y estado del arte incorrectos	1	2	2	4
Mala programación de las actividades.	1	2	2	5

Tabla 5. 3Tabla de análisis de riegos y grado de exposición

En la figura 5.3 se puede observar que el riesgo con mayor a exposición y que puede afectar el normal desarrollo del proyecto corresponde al riesgo número 1, donde con un factor de exposición de 12 es el mayor indicador de la tabla. Para calcular el factor de exposición solo basta con multiplicar la probabilidad de que ese riesgo ocurra por el impacto que este pueda tener. Importante mencionar que el rango de los factores es de 1 hasta 5 .

Posteriormente se crea un plan de mitigación de riesgos donde se crean las estrategias necesarias y la forma de actuar del equipo de trabajo para que estos no ocurran y tengan un impacto sobre el proyecto. Por ejemplo un buen plan de mitigación es hacer una prueba antes de comenzar el proceso de desarrollo de las 2 tecnologías, obtener toda la información necesaria y comprobar si son compatibles o no. En caso de no serlo aún se puede estar a tiempo de cambiar el robot e intentar el proyecto con otro tipo , esto causaría un menor daño en comparación con el proyecto en plena etapa de desarrollo . Para esto es esencial obtener el marco teórico de manera correcta para el éxito del proyecto.

Riesgo	Exposición	Medidas de Mitigación
Las tecnologías no son compatibles	12	Investigar ambas tecnologías, adquirir los adaptadores necesarios ,tener un segundo caso de estudio.
El sistema no cumple con los objetivos especificados.	8	Seguir la planificación , estar preparado a los cambios en los requerimientos según como se vaya desarrollando el proyecto
No adquisición del Hardware utilizado en el proyecto	6	Comprarlos en distribuidoras especializadas, segur la planificación para evitar problemas de tiempo.
Obtención del marco teórico y estado del arte incorrectos	2	Seguir la planificación del proyecto, consultar a fuentes confiables y oficiales, Búsqueda de información en la literatura disponible.
Mala programación de las actividades.	2	Construcción de una carta Gantt, Registrar los hitos del proyecto.

Tabla 5. 4 Plan de mitigación de riesgos

En la Tabla se muestra los 5 riesgos principales del proyecto, además del factor de exposición calculado en la tabla anterior ordenado de mayor a menor y posteriormente una especificación de que medidas de mitigación se pueden realizar para evitar que ocurra los riesgos señalados.

Capítulo VI Plan de pruebas y resultados

Una vez realizado la codificación de cada una de las aplicaciones desarrolladas por separado y su posterior unión para el producto final de software, es necesario fabricar un plan de pruebas para así, asegurar el correcto uso de la aplicación final para el usuario.

Un buen plan de pruebas permitirá identificar posibles errores, planificar algunos escenarios no estudiados anteriormente, revisar estándares de usabilidad, como también identificar las diferentes herramientas, controladores y frameworks para el correcto funcionamiento de la aplicación en el computador que sea instalada.

6.1 Etapas plan de prueba

El plan de pruebas se dividió básicamente en 3 etapas, cada una encargada de comprobar el correcto funcionamiento de cada uno de los hardwares y softwares desarrollados por separado y la última la unión de estas como una aplicación única y final.

6.1.1 Implementación de aplicación como controlador de los servomotores del robot NXT Lego

En una primera instancia se desarrolló un software que controlaba los 3 servomotores del robot, a través de la librería NxtNET, que permite el control de lo servomotores a través del lenguaje de programación C#.

Gracias a esta aplicación se comprobó el correcto uso de la librería, además de realizar pruebas acerca de las potencias de los 3 servomotores para realizar los ajustes correspondientes para la posterior manipulación del robot.

6.1.1.1 Resultados

Implementada la aplicación no se observaron mayores inconvenientes. Los 3 servomotores mostraron una buena capacidad de repuesta y en tiempo real, es decir, no existe un retardo en el momento del envió de la orden vía bluetooth desde el PC al brick del robot.

Además la librería NxtNET se comportó de manera perfecta, los métodos principales funcionan sin inconvenientes y permiten tener un control total sobre los servomotores del robot.

6.1.2 Implementación librería skeleton tracking a través de SDK Kinect.

Se comprobó que los datos sobre el esqueleto fueran un flujo constante contando el número

de actualizaciones de cada articulación por segundo. También se comprobó que las

Coordenadas no sufrieran saltos bruscos de posición analizando las coordenadas de varias

escenas manualmente.

Para ver el representar el seguimiento del esqueleto de manera fabrica se dibujaron 6 elipses representando las articulaciones del cuerpo que eran necesarias para la manipulación de robot, estas fueron ambos codos, ambas manos, cabeza, además del tronco de la persona

Una vez que la persona se sitúa frente al sensor kinect, automáticamente la aplicación comienza reconocer el esqueleto y lo grafica, permitiendo al usuario observar que precisamente está siguiendo sus movimientos en tiempo real y que la aplicación funciona como corresponde.

6.1.2.1 Resultados

No se observaron mayores inconvenientes, salvo algunos problemas de ruido que en ocasiones movían a la elipse que grafica la articulación sin que lo hiciera el usuario. Sin embargo esto es ajustable desde el código fuente encontrando la mejor configuración de frames por minuto según el entorno.

La aplicación funciona de manera correcta con un solo usuario frente al sensor, al situarse 2 personas, el sensor del esqueleto detecta al que esté mejor posicionada y reconocerá las articulaciones de su cuerpo.

6.1.3 Implementación de la unión de ambas aplicaciones en un simulador final de interacción natural

Una vez realizada las pruebas para ambas aplicaciones por separado se realizan las pruebas finales con la unión de ambas aplicaciones en una solo que corresponde al producto final que manipulara al robot a través de interacción natural.

Básicamente La aplicación es la encargada de reconocer a la persona que se sitúa frente al sensor, reconocer las articulaciones de ésta, interpretar los datos y luego comprobar si existen los gestos desarrollados para poder manipular el robot. Una vez que reconoce los gestos ya pre definidos el robot se mueve según los gestos que realice la persona, además le

informa por medio de un texto en pantalla cual ha sido el gesto que ha realizado, logrando finalmente el objetivo del proyecto que es manipular el robot por medio de interacción natural conectando ambos hardwares (Kinect y robot).

6.1.3.1 Resultados

Realizada las pruebas correspondientes la aplicación se comporto bastante estable, alrededor de 20 usuarios probaron la aplicación, situándose frente al sensor y manipulando el robot sin inconvenientes cumpliendo el objetivo de desarrollar una aplicación estable y sin errores graves que interrumpan el correcto funcionamiento del programa.

Algunos problemas que se presentaron fueron, que luego de varias horas de uso de la aplicación al aire libre, el sensor de kinect comenzó a presentar fallas, no reconociendo el esqueleto de la persona, por lo tanto, no permitía la manipulación del robot, sin embargo este es un problema no controlable en cuanto a desarrollo de software y es necesario realizar una investigación en cuanto a como se comporta el sensor kinect al aire libre.

Los tiempos de repuesta siguen comportándose de manera óptima, tanto como el reconocimiento de gestos como la activación de los servomotores del robot, estos responden casi al instante pasando desapercibido algún retardo en caso que este existiera permitiendo de manera casi perfecta la interacción natural que es el objetivo final del proyecto.

Para ver en funcionamiento de un plan de pruebas y además la unión de ambas aplicaciones revisar el siguiente video:

http://www.youtube.com/watch?v=pnRJqLZUNcs&feature=g-all-esi

Capítulo VII Conclusiones

Durante este estudio se ha observado la importancia que tiene en la actualidad el sensor Kinect, creado en un principio para jugar en consolas de videojuegos sin la intervención de un control, pero que con el tiempo y la intervención al código fuente de este fue evolucionando en útiles e innovadoras aplicaciones. Es por esto que se ha decidido llevar a cabo este proyecto, donde se busca la creación de una aplicación intuitiva, innovadora e ingeniosa, para ampliar aún más el campo de esta nueva tecnología que es Kinect y comprobar que tan poderosa es esta nueva herramienta que día a día abarca más ramas de la ciencia.

Para experimentar en dicha tecnología fue necesario implementarla en algunas de las ramas de la ciencia, es por esto, que se decidió implementar el sensor dentro de la robótica ,donde a través de la obtención del marco teórico se observó cómo hoy en día resulta casi indispensable el uso de los robots ya sea para fines de investigación o simplemente para ayudar al hombre en tareas que él no puede realizar , como el transporte de grandes pesos en la industria o enviar robots a superficies de la luna o planetas y así obtener datos que para el hombre sin el uso de estas hoy en día prácticamente sería inalcanzable. También el uso de los robots en la actualidad ad ha evolucionado de forma significativa, gracias a la inteligencia artificial, hoy existen los denominados robots inteligentes, donde estos realizan tareas igual de complejas que realizan los humanos gracias al aprendizaje que estos van obteniendo según las tareas que estos realizan, algo impensado en el pasado donde los robots simplemente intentaban semejarse a un ser humano y ayudar al hombre en ciertas tareas. Es en este punto donde se investigó un robot muy amigable pero poderoso llamado RoboPhilo. Este robot es de tipo humanoide, es decir intenta asemejarse a la forma humana y gracias a sus 24 servomotores permite una gran cantidad de movimientos, es por esto, que este tipo de robot humanoide es ideal para un estudio como el realizado hasta ahora, donde su bajo costo, pero no menos poderoso, es un excelente complemento para interactuar con el sensor Kinect.

Una vez estudiado el marco teórico de ambas tecnologías , surgió la interrogante de como conectar el robot con el sensor y que función puede tener el crear un sistema de control robótico por medio del sensor, para esto ,tanto Kinect como RobopHilo cuentan con poderosas herramientas de desarrollo llamadas SDK, donde el programador puede desarrollar distintos códigos fuentes y crear nuevas ideas ,que pueden ser de gran ayuda en distintos ámbitos como la salud, fines militares, de seguridad o incluso hasta entretención .

A medida que se avanzó en el proyecto al realizar el estudio de factibilidad se identificaron distinta problemáticas que podían surgir acerca de la conexión que se realizaría entre las tecnologías , por ejemplo que los sensores infrarrojos prácticamente se encuentran obsoletos hoy en el mercado o la mayoría de los computadores no tienen entrada serial RS232,

por lo que fue necesario la adquisición o construcción de adaptadores para manipular las tecnología sin inconvenientes y estas se puedan comunicar, todo esto gracias a estudio de factibilidad técnica y al análisis de riesgo lo que permitió crear un segundo caso de estudio o de prueba en caso de que el primero y principal falle.

Finalmente el caso de estudio número uno presento problemas en lo que respecta a la configuración del robophilo, por lo tanto se debió continuar el proyecto con el segundo caso de estudio que es básicamente la misma idea del anterior con la diferencia que cambia el robot por un lego NXT, que son algo más limitados pero no menos poderosos en comparación al robophilo. Esto reafirma aún más la importancia de tener una planificación correctamente programada además de realizar un buen estudio de factibilidad técnica y económica, sin haber hecho estos estudios antes de la concepción del proyecto, este no hubiese tenido el éxito esperado y no se podría haber desarrollado la aplicación.

Para finalizar el proyecto, como conclusión final, es posible afirmar que es viable la realización de este, tanto en lo económico, como en la factibilidad técnica. Existen las herramientas necesarias para desarrollar una aplicación innovadora, de calidad, que cumpla con los estándares de usabilidad y por sobre todo cumplir el objetivo que es la manipulación de un robot (garra mecánica) por medio de interacción natural, todo esto gracias al estudio previo realizado y su posterior ejecución.

Luego del desarrollo de la aplicación quizás bien una de las etapas mas importante del proyecto que es el plan de prueba, es en esta etapa donde realmente se puede medir el proyecto en sí, tanto en lo técnico, usable e innovador. las pruebas realizadas no arrojaron mayores inconvenientes , solo pequeños detalles que son perfectamente corregibles para el futuro, el software tuvo una buena acogida en los usuarios que lo probaron y por sobre todo se trata de una aplicación innovadora, realizada con las nuevas tecnología el presente, tanto la robótica como el uso del sensor kinect siguen creciendo día a día y el proyecto buscó justamente eso , poder llegar a ser un nuevo aporte en lo que respecta a innovación y el hacer ciencia.

Sin lugar a dudas Kinect seguirá evolucionando con el tiempo y aportando mucho más a la ciencia, pero depende de cada desarrollador el dejar un legado con aplicaciones novedosas y que sean un real aporte a la sociedad. El campo donde puede ser aplicada esta tecnología parece no tener fin y en un futuro no muy lejano Kinect pasara a ser parte de nuestras, vidas siendo indispensable el uso de este debido a su alto potencial , bajo costo , y la facilidad de poder intervenir su código fuente.

Referencias

Arroyo, C. (2005). *Cirugía Robótica.* Puebla,Mexico.

Beland, C. (2000). *LEGO Mindstorms . The Structure of an Engineering (R)evolution.*

CodePlex. (s.f.). *codeplex*. Recuperado el 2 de 06 de 2012, de https://nxtnet.codeplex.com/

Deyle, T. (29 de Agosto de 2009). *hizook*. Recuperado el 18 de Abril de 2012, de http://www.hizook.com/blog/2009/08/29/i-swarm-micro-robots-realized-impressive-full-system-integration

Edubrick. (2006). *Guía de Robótica.* Santiago : Lego.

Handest., N. K. (2011). Recuperado el 5 de Septiembre de 2012, de mindsqualls: http://www.mindsqualls.net

Jarret Webb, J. A. (2011). *Beginning Kinect Programming with te Microsoft Kinect SDK.* apress.

Juan A. Alonso, S. B. (2004). *Tecnologías de la Información y de la Comunicación. Capítulo 6, Programación y control de procesos.* . Ra-ma.

Kinect, S. (24 de 05 de 2012). *Microsoft.* Recuperado el 6 de 06 de 2012, de http://go.microsoft.com/fwlink/?LinkID=251756

Lerma, A. (2005 de Noviembre de 201o). *arturolerma*. Recuperado el 17 de Abril de 2012, de http://arturolerma.blogspot.com/2010/11/programacion-de-robotica.html

Margaret Ellis, B. S. (1990). *Manual de referencia C++ con anotaciones.* Massachussets, Estados Unidos: Adisson Wesley Diaz de Santos.

Microsoft. (2012). *Miicrosoft Corporation*. Recuperado el 15 de Marzo de 2012, de http://www.microsoft.com

Ollero, A. (2001). *Robótica manipuladores y robots móviles.* Barcelona: marcombo-Alfaomega.

Pfeiffer, S. (2011). *Guiado gestual de un robot mediante un sensro Kinect.* Barcelona.

RAE. (2012). *rae*. Recuperado el 15 de Marzo de 2012, de www.rae.es

Ramirez, J. C. (17 de Noviembre de 2011). *pni*. Recuperado el 17 de Abril de 2012, de http://pni.senati.ws/comunidad/2011/11/17/robots-inteligentes/

Ricardo Lizcano, J. F. (2005). *Control para un robot articulado con 3 grados de libertad que simule el movimiento de pata.* Bogotá: Memoria de titulo Pntificia Universidad Javeriana.

Roberto Alvarez, R. C. (2008). *Robótica.* Chihuahua: anónimo.

Robophilo. (2007). *roboPhilo*. Recuperado el 31 de Marzo de 2012, de http://www.robophilo.com/live/en/index.php#

RoboPhilo, S. (2007). *User Guide.*

RoboPhilo, S. (2007). *User guide SDK RobopHilo.*

U de atacama. (2012). Recuperado el 29 de Marzo de 2012, de sitio web de industrias y negocios chile: http://www.industriaynegocios.cl/Academicos/AlexanderBorger/