UNIVERSIDAD TECNOLÓGICA EQUINOCCIAL

FACULTAD DE CIENCIAS DE LA INGENIERÍA
CARRERA DE INGENIERÍA MECATRÓNICA

DESARROLLO DE UNA APLICACIÓN REMOTA DE INTERACCIÓN ENTRE EL PROFESOR Y EL PC PARA AGILITAR EL PROCESO ENSEÑANZA APRENDIZAJE EN LAS AULAS DE LA UNIVERSIDAD TECNOLÓGICA EQUINOCCIAL.

TRABAJO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN MECATRÓNICA

JOSÉ EDUARDO CARVAJAL TERÁN

DIRECTOR: ING. JUAN CARLOS RIVERA

Quito, Julio, 2014

DECLARACIÓN

Yo **JOSÉ EDUARDO CARVAJAL TERÁN**, declaro que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

La Universidad Tecnológica Equinoccial puede hacer uso de los derechos correspondientes a este trabajo, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normativa institucional vigente.

_____

José Eduardo Carvajal Terán

C.I. 1716633498

CERTIFICACIÓN

Certifico que el presente trabajo que lleva por título "**Desarrollo de una aplicación remota de interacción entre el profesor y el PC para agilitar el proceso enseñanza aprendizaje en las aulas de la Universidad Tecnológica Equinoccial.**", que, para aspirar al título de **Ingeniero en Mecatrónica** fue desarrollado por **José Eduardo Carvajal Terán**, bajo mi dirección y supervisión, en la Facultad de Ciencias de la Ingeniería; y cumple con las condiciones requeridas por el reglamento de Trabajos de Titulación artículos 18 y 25.

_____

Ing. Juan Carlos Rivera

DIRECTOR DEL TRABAJO

C.I. 0501373823

DEDICATORIA

Quiero dar gracias a Dios en primer lugar por darme toda la fuerza y voluntad para terminar este trabajo y quiero agradecer a mis padres a mis tíos y a mis hermanas. A Eduardo Carvajal y Patricia Terán por darme todo el apoyo, amor y guía de padres, todo esto me ha ayudado en mi vida, así mismo agradezco de una forma infinita a mi querida Tía y segunda madre Yolanda Terán que al igual que mis padres inspiran mi vida para seguir adelante. A Diana y a Karina por su amor y preocupación de hermanas que siempre me ayudo y me motivo en todo aspecto de mi vida, a mí muy querido Tío Lenin por siempre estar ahí aun en las circunstancias más duras de mi vida. También quiero agradecer a todos los profesores que me ayudaron a hacer este trabajo, gracias por sus consejos y motivaciones. Y por último agradezco cada experiencia, cada consejo, cada ayuda que he vivido con mis amigos de la Universidad los cuales siempre los llevare en el corazón.

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

Cambiar entornos educativos convencionales a entornos de enseñanza interactivos donde se transformen las exposiciones comunes a una experiencia virtual tanto para el expositor como para el alumno, fue la meta. Esto se lo hizo posible gracias a la Interfaz Natural del Usuario (NUI) específicamente con el dispositivo Kinect. Este Hardware permitió realizar un seguimiento al cuerpo humano y reconoció gestos y movimientos; ya que tiene una cámara RGB, un sensor de profundidad, un sensor CMOS Infrarrojo, un micrófono de múltiples matrices, un acelerómetro de tres ejes y un microprocesador especializado que capta el cuerpo humano en 3 dimensiones. Se desarrolló la aplicación en el software de programación, simulación y control Simulink de Matlab donde se utilizó la librería NID de Kinect para formular algoritmos comparativos proporcionales del cuerpo, algoritmos de descripción unidimensionales y algoritmos de flujo óptico, los cuales permitieron diseñar un sistema de visión artificial con sus etapas de adquisición, segmentación, descripción y reconocimiento de movimientos que realiza el usuario. El control se realizó mediante una lógica de decisión combinatoria y secuencial basada en máquinas de estado y diagramas de flujo que identificaron los movimientos del usuario y los convirtieron en eventos del PC; estos eventos fueron: avanzar de diapositiva, retroceder de diapositiva, aumentar zoom y disminuir zoom.

Palabras claves: Kinect, Simulink, Matlab, aplicación, sensor, profundidad, acelerómetro, microprocesador, algoritmo, lógica, PC, evento, zoom.

ABSTRACT

Changing conventional education to an interactive education, which it can change the traditional expositions toward a virtual experience for teacher like to student. This it had been my goal. All this project was possible for Natural User Interface (NUI) speciality for a Kinect. This hardware allow make a skeleton tracking, gestures, movements and voice recognizer because it has a RGB camera, depth sensor, Infrared sensor, array microphone, 3 axes accelerometer and a specialized microprocessor that keep the human body in 3D. My app was developed in Kinect that it is software of programming, simulation and control from Matlab inside of this software was done many interactive models with the NID Kinect toolbox, all this models have algorithms comparative of body proportions that used one-dimensional and optic flow descriptors, them allow done an artificial vision system with stages like acquisition, segmentation, description and recognition afterwards all this stages PC interprets four specific movements made by the user later the control system was performed using a combinatorial logic and sequential decision based on state machines and flow that it joints the movements with events determined in the PC, these events were move slide, move back slide, increase zoom and zoom out.

Keywords: Kinect, Simulink, Matlab, app, sensor, depth, accelerometer, microprocessor, algorithm, descriptors, logical, PC, event, zoom.

1. INTRODUCCIÓN

Las Tecnologías de Información y Comunicación (TIC) permiten realizar investigaciones en campos científicos e industriales, pero son pocas las TIC que se desarrollan para el campo educativo. El desarrollo de TIC en ámbitos educativos dará origen a nuevos modelos y estrategias para el proceso enseñanza- aprendizaje.

Dentro del aula de clase son muchas las actividades que realiza el docente entre las cuales están: evaluar al estudiante, verificar la asistencia, impartir clases, calificar deberes, etc. En universidades ecuatorianas no existen aplicaciones tecnológicas que agiliten la gestión del proceso enseñanza-aprendizaje; debido a esto el Ecuador necesita desarrollar aplicaciones TIC para fomentar una mejor educación.

Con la nueva tecnología de interfaz natural del usuario se desarrolla una aplicación que permite interactuar de forma remota al docente con el computador personal (PC) mediante el reconocimiento de sus movimientos; esto facilita la gestión del docente en el aula de clases y se logra sistemas académicos más integradores con nuevos entornos y de fácil acceso.

Objetivo general

Desarrollar una aplicación remota de interacción que interprete movimientos específicos para agilitar la gestión del proceso enseñanza-aprendizaje en las aulas de la Universidad Tecnológica Equinoccial.

Objetivos específicos

- Analizar diferentes tecnologías de interpretación de gestos y movimientos aplicables a entornos educativos.
- Integrar una herramienta de interpretación de gestos y movimientos específicos a un PC.

- Controlar mediante movimientos específicos la secuencia de las presentaciones, el aumento y reducción del nivel de zoom.

La aplicación se desarrolló para el sistema operativo Windows versiones 7 y 8, esta aplicación identifica el movimiento que realiza el docente con su mano generando señales de reconocimiento a través del dispositivo Kinect; posteriormente con el programa de diseño y desarrollo de aplicaciones Matlab se analizó, filtro y escogió las señales más adecuadas para relacionarlas con acciones determinadas en el PC. El reconocimiento de cuatro (4) clases de movimientos realizados con la mano genera cuatro (4) señales que se procesan en Matlab para permitir el desplazamiento de presentaciones y el aumento o reducción de zoom.

El dispositivo Kinect debe ser ubicado debajo de un proyector de una aula de clases de la Universidad Tecnológica Equinoccial debido a dos razones: la primera es que el dispositivo Kinect tiene dos ángulos de apertura de visión: apertura horizontal y apertura vertical; la perfecta calibración de ambos ángulos permiten tener información sobre la altura máxima y mínima que puede sensar (altura necesaria del usuario para usar la aplicación), distancia máxima de movimiento horizontal del usuario (distancia horizontal que tiene el usuario para moverse) y un área disponible de trabajo (área útil del usuario para interpretación de todos sus movimientos). La segunda razón es que el uso de esta aplicación no incomode al usuario, ni lo prive de espontaneidad o cambio de comportamiento en sus presentaciones, charlas, conferencias o clases.

Debido a estas dos razones se ha optado por ubicar al Kinect debajo del proyector que es un lugar adecuado, que no incomoda al usuario, permite una la libre expresión y un correcto sensado.

Un requerimiento muy importante es encuadrar perfectamente el área normal que ocupa el usuario para sus presentaciones con el área disponible de trabajo que brinda el Kinect, para tener una sola referencia de control y mejorar la interacción de las presentaciones con los movimientos que realiza el usuario.

2. MARCO TEÓRICO

En este capítulo se presenta información detallada sobre el dispositivo Kinect, sus componentes, su funcionamiento, arquitectura, requerimientos, y se explica el tipo de tecnología del dispositivo.

2.1 INTERFAZ NATURAL DEL USUARIO (NUI)

La interfaz natural del usuario cuyas siglas en ingles son NUI (Natural User Interface) es la capacidad de interactuar con una máquina usando los movimientos del cuerpo humano.

Una interfaz es la conexión física y funcional entre dispositivos o sistemas independientes (Academia, 2013) que permite una comunicación entre distintos niveles de arquitectura. De una forma coloquial la interfaz es una extensión del cuerpo humano que permite realizar acciones en estos dispositivos o sistemas independientes.

Todos al momento de usar una máquina, un dispositivo o PC están obligados a usar una interfaz externa para interactuar con ellos, esta interfaz puede ser de fácil uso, pero la forma de comunicación entre la interfaz y el ser humano no es innata esto quiere decir que las personas tienen que aprender la manera de usar la interfaz.

Pero la Interfaz Natural del Usuario es controlada por la interacción natural humana, por los movimientos naturales de las personas, por los gestos que realizan y por la voz.

Desde la década de 1970 se desarrolla una serie de estrategias de interfaz para el usuario que utilizan la interacción natural con el mundo real, como una alternativa de la interfaz de línea de comandos (CLI) o de la interfaz gráfica de usuario (GUI). En la CLI, los usuarios tenían que aprender un medio artificial de entrada, el teclado, y una serie de insumos codificados, que tenían un rango limitado de respuestas, donde la sintaxis de los comandos era estricta. Luego, el ratón activó la interfaz gráfica de usuario, los usuarios pueden aprender fácilmente los movimientos y las acciones del ratón. La GUI se basó en metáforas para interactuar con el contenido o los

objetos en pantalla. La Interfaz Natural del Usuario (NUI por sus siglas en inglés) se basa en redes neuronales artificiales de algoritmos complejos que describen de forma acertada y muy exacta las medidas del cuerpo, color de la piel, etc., estas características permiten que a través de una cámara se pueda obtener información de las imágenes tomadas luego se procede a un entrenamiento neuronal de la red que puede llegar a identificar a los seres humanos, sus movimientos y sus gestos. Esta NUI es usada e investigada por desarrolladores y diseñadores de software para mejorar la experiencia al usuario (KinectForWindows, 2013). Entre los equipos que permiten realizar están:

- Kinect.

El sensor Kinect como lo muestra la Figura 2.1 fue desarrollado por Microsoft, y nació como un nuevo controlador para la videoconsola Xbox 360. Permite manejar la video consola sin necesidad de un controlador tradicional o mando, puede controlarla solamente con movimientos de su cuerpo y ordenes de voz (XBOX, 2013).

Figura 2.1. Fotografía Dispositivo Kinect XBOX 360
(XBOX, 2013)

- Gafas interactivas de Google

Google Glass es un ordenador portátil con una pantalla montada en la cabeza óptica que está siendo desarrollado por Google como lo muestra la Figura 2.2 sus funciones son las de un teléfono inteligente de manos libres,

se puede comunicar y conectar a internet a través del lenguaje natural comandos de voz. (Google, 2013).

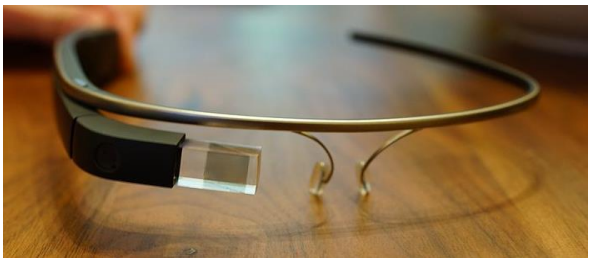

Figura 2.2. Fotografía Dispositivo Google Glass
(Google, 2013)

- Leap Motion

La Figura 2.3 muestra al Leap, que es una cámara de tiempo de vuelo creada para seguir los movimientos de las manos, cuenta con una resolución muy alta y un campo de visión limitado (LeapMotion, 2013).

Figura 2.3. Fotografía del dispositivo Leap Motion
(LeapMotion, 2013)

2.2 DISPOSITIVO KINECT

Kinect es el dispositivo que permite realizar un seguimiento al cuerpo humano y reconoce gestos, movimientos y comandos de voz a través de la Interface Natural del Usuario. Kinect fue desarrollado por la empresa Microsoft en la línea de investigación de nuevas tecnologías de XBOX para una nueva consola de juegos.

Kinect se basa en la tecnología de software desarrollado por Rare , una empresa subsidiaria de Microsoft Game Studios propiedad de Microsoft, la cámara de alta tecnología es desarrollado por PrimeSense una empresa de tecnología israelita, que desarrolló un sistema que pueda interpretar gestos específicos, por lo que el control de Kinect es completamente a manos libres, esto es sin uso de cables o controles, para todo esto se usaron sensores infrarrojos, una cámara para la adquisición de datos y un microchip especial que permite seguir el movimiento de los objetos y de los individuos en tres dimensiones (XBOX, 2013).

Kinect es una barra horizontal conectada a una base pequeña con un pivote motorizado y está diseñado para ser colocado longitudinalmente por encima o por debajo de la pantalla de vídeo. El dispositivo cuenta con una cámara RGB, un sensor de profundidad y micrófono de múltiples matrices, la unión de estas tres clases de sensores es lo que proporciona una captura y seguimiento en 3D de todo el cuerpo, con reconocimiento de gestos, reconocimiento facial y reconocimiento de voz.

La empresa Microsoft determina como la principal innovación de Kinect, la tecnología del software avanzado que permite el reconocimiento de gestos , reconocimiento facial y reconocimiento de voz. Kinect es capaz de rastrear simultáneamente hasta seis personas, entre ellas dos jugadores activos para el análisis de movimiento con una extracción de características, de 20 articulaciones por jugador como lo muestra la Figura 2.4 sin embargo, PrimeSense ha señalado que el número de personas que el dispositivo puede identificar está limitado por la cantidad de personas que caben en el campo de visión de la cámara. Kinect tiene dos modos de sensado para el reconocimiento del cuerpo humano, el reconocimiento total del esqueleto el cual permite un seguimiento y mapeo del cuerpo humano con sus 20 articulaciones más importantes desde la cabeza hasta los pies, y el reconocimiento del cuerpo sentado el cual permite un seguimiento y mapeo solamente de la parte superior del cuerpo humano desde la cabeza hasta la cintura (Microsoft, 2013), ambos modos funcionan indistintamente de si una

persona se encuentra parada o sentada, esta información se la puede observar en la Figura 2.4.

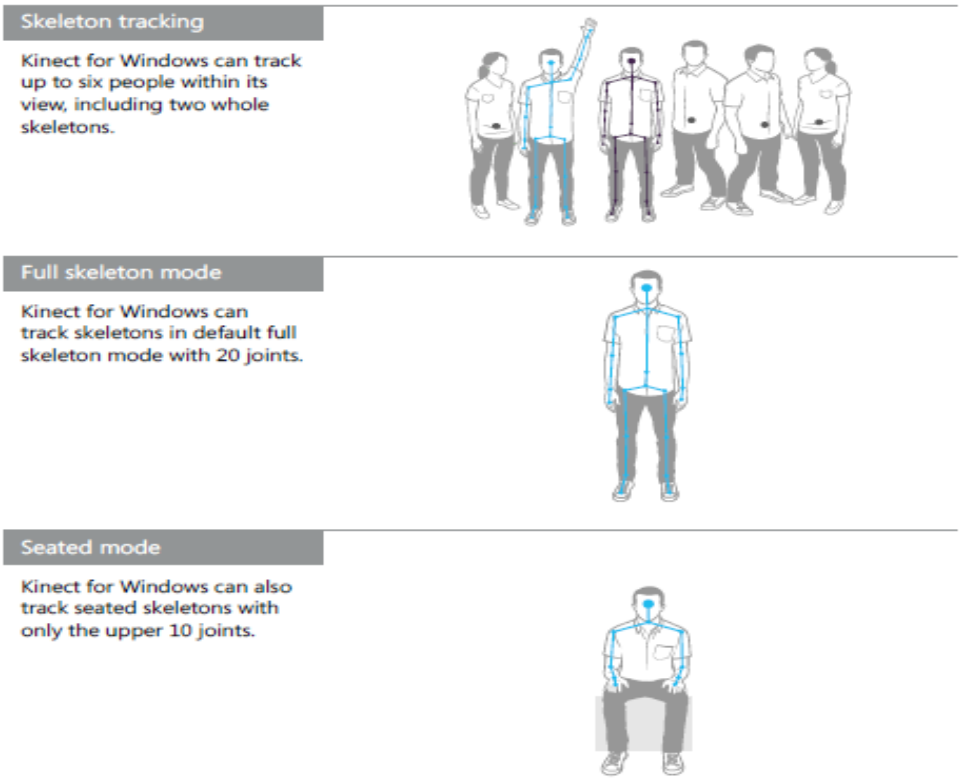

Figura 2.4. Opciones de seguimiento del cuerpo humano por Kinect (Microsoft, 2013)

La ingeniería inversa ha determinado que varios sensores de salida de video del Kinect tiene una velocidad de fotogramas de aproximadamente 9 Hz a 30 Hz dependiendo de la resolución. El flujo de video RGB predeterminado utiliza una resolución VGA de 8 bits por componente de color (640 × 480 píxeles) con un filtro de color Bayer[1] , pero el hardware es capaz de resoluciones de hasta 1280x1024 (a una velocidad de fotogramas más baja) y otros formatos de color, como UYVY . El flujo de video del sensor de

[1] Es el filtro de color que divide una imagen digital en matrices de colores rojos, verdes, y azules para que cada fotodiodo entregue información de la luminosidad correspondiente en esa sección, interpolando la información de los 3 fotodiodos se obtiene un pixel de color. Se llama así por su creador, Bryce Bayer, de la empresa Eastman Kodak.(Kodak,2007)

profundidad monocromo es en resolución VGA (640 × 480 píxeles) con profundidad de 11 bits, que ofrece 2.048 niveles de sensibilidad. El sensor tiene un campo de visión angular de 57.5° en horizontal y 43.5° en vertical, mientras que el pivote motorizado es capaz de inclinar el sensor hasta 27° ya sea hacia arriba o hacia abajo (Microsoft, 2013), como lo muestra la Figura 2.5; Kinect posee dos niveles de sensado o de visión por profundidad, el modo cercano que su campo de visión activa empieza desde los 0.8m hasta los 2.5m y el modo lejano que su campo de visión activa empieza desde los 1.2m hasta los 3.5 m de distancia esto se puede ver en la Figura 2.5. La matriz de micrófono cuenta con cuatro cápsulas de micrófonos y opera con cada canal de procesamiento de 16 bits de audio a una frecuencia de muestreo de 16 KHz .

Todas las ventajas para el seguimiento y reconocimiento de personas realizadas por el Kinect son gracias al SKD (Software Development Kit) o paquete de desarrollo de software. Actualmente existen cuatro versiones del SDK para Kinect y cada una presenta una mejora con respecto a la anterior, la primera versión fue lanzada para las personas interesadas en el desarrollo de aplicaciones Kinect y fue la versión 1.5 hoy en día se cuenta con la versión 1.8 del SDK.

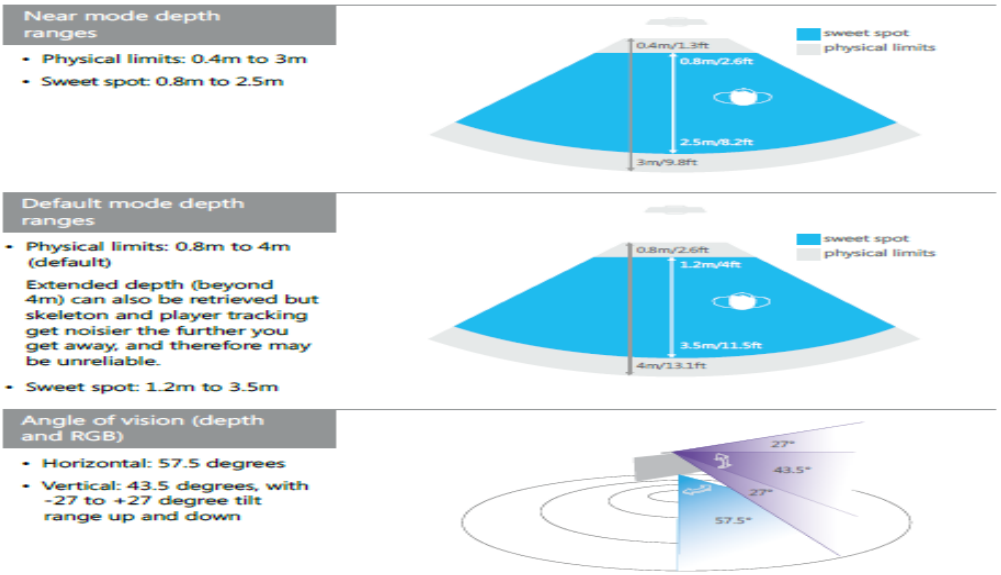

Figura 2.5. Rangos de Vision del Kinect
(Microsoft, 2013)

2.2.1 REFERENCIAS CONTEMPORÁNEAS PARA EL USO DEL KINECT

- Asistente para quirófano

En la Figura 2.6 se observa la aplicación basada en Kinect, que permite a los médicos que se encuentran interviniendo en una cirugía buscar en sus computadores toda la información del expediente del paciente, sin necesidad de tocar ningún objeto. Esta aplicación ganó el premio “Únete al movimiento Kinect” y fue uno de los diez proyectos que apoyo Wayra en España (Microsoft, 2013).

Figura 2.6. Consultas de imágenes médicas
(Microsoft, 2013)

- Asistente de compras

La aplicación tiene el nombre *Styku Apparel Shopping* fue creada para resolver el problema de encontrar ropa que le quede al consumidor. Se emparejó a Kinect para Windows con su software para dar a los consumidores la oportunidad de probar prácticamente la ropa antes de comprar como muestra la Figura 2.7. La tecnología tiene como objetivo mejorar la experiencia de compra, aumentar las ventas al por menor, reducir las devoluciones (Microsoft, 2013).

- Anuncios inteligentes

En la Figura 2.8 se muestra la aplicación de nombre *Advertising Framework* desarrollada por la empresa Intel, permite reconocer la edad y el género de la persona, con el fin de mostrarle de forma inteligente el anuncio publicitario que más se oriente a sus intereses (Microsoft, 2013).

Figura 2.7. Probador de ropa virtual
(Microsoft, 2013)

Figura 2.8. Anuncio de publicidad inteligente
(Microsoft, 2013)

- Asistente virtual de experiencias en automovilismo

Nissan Pathfinder Showroom Virtual como se muestra en la Figura 2.9 es la última herramienta de la industria del automovilismo. Las empresas Audi, Ford y Nissan están adoptando Kinect para Windows como la nueva herramienta para configuradores de autos en línea, con visores de 360 grados de imágenes que hacen que sea más fácil para los clientes visualizar el vehículo que quieren (Microsoft, 2013).

Figura 2.9. Asistente virtual de automovilismo
(Microsoft, 2013)

2.2.2 COMPONENTES DEL KINECT

Kinect proporciona ojos, oídos y cerebro a las computadoras, con Kinect las empresas y los desarrolladores están creando aplicaciones que permiten a sus clientes interactuar de forma natural con las computadoras simplemente gesticulando y hablando.

2.2.2.1 Componentes de hardware

El Kinect cuenta con una cámara RGB, un sensor de profundidad infrarrojo CMOS, un micrófono de múltiples matrices, una luz LED, un acelerómetro de tres ejes, un servomotor con movilidad en el eje vertical y en especial con un procesador especializado Prime Sensor que capta el entorno de tres dimensiones (Microsoft, 2013). La figura 2.10 muestra dichos componentes.

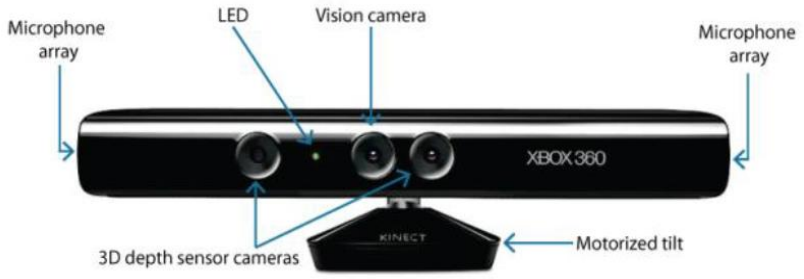

Figura 2.10. Componentes del Kinect
(Microsoft, 2013)

- Cámara RGB: Almacena los datos de tres canales en una resolución de 1280 x 960 a 12 cuadros por segundo, o una resolución de 640 x 480 a 30 cuadros por segundo.
- Micrófono de múltiples matrices: Contiene cuatro micrófonos para capturar sonido. Debido a que hay cuatro micrófonos, es posible grabar audio desde una dirección específica, así como encontrar la ubicación de la fuente de sonido y la dirección de la onda de audio.
- Luz LED: Es un indicador para saber si el Kinect se encuentra encendido.
- Servomotor (Pivote Motorizado): Permite el movimiento del dispositivo en el eje vertical con una inclinación de 27 grados por encima o por debajo de su posición inicial.
- Un emisor infrarrojo (IR) y un sensor de profundidad CMOS de IR: El emisor emite rayos de luz infrarroja y el sensor de profundidad CMOS lee los haces de IR reflejada de nuevo al sensor. Los haces reflejados

se convierten en la información de medición de profundidad de la distancia entre un objeto y el sensor. Esto hace posible la captura de una imagen de profundidad. La Figura 2.11 muestra la ubicación de estos sensores en el Kinect.

- Sensor de imagen CMOS para RGB: Captura la resolución espacial, es decir las coordenadas de los ejes X e Y. La Figura 2.11 muestra la ubicación del sensor en el Kinect.

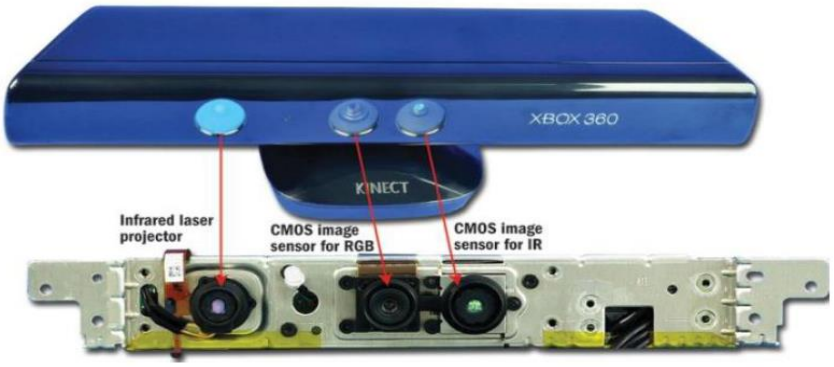

Figura 2.11. Sensores de imagen del Kinect
(Hernandez Toalla & Herrera Rodrigez, 2013)

- Chip PRIMESENSOR PS1080: Sirve para reconstruir una captura de movimiento 3D de la escena que esté ubicada al frente del Kinect ya que este chip captura su entorno en tres dimensiones y transforma esas capturas a imágenes en 3D. Este chip fue desarrollado por la empresa israelita PrimeSense. Los sensores de imágenes RGB y de CMOS de profundidad trabajan en conjunto con el chip como se muestra en la Figura 2.12.

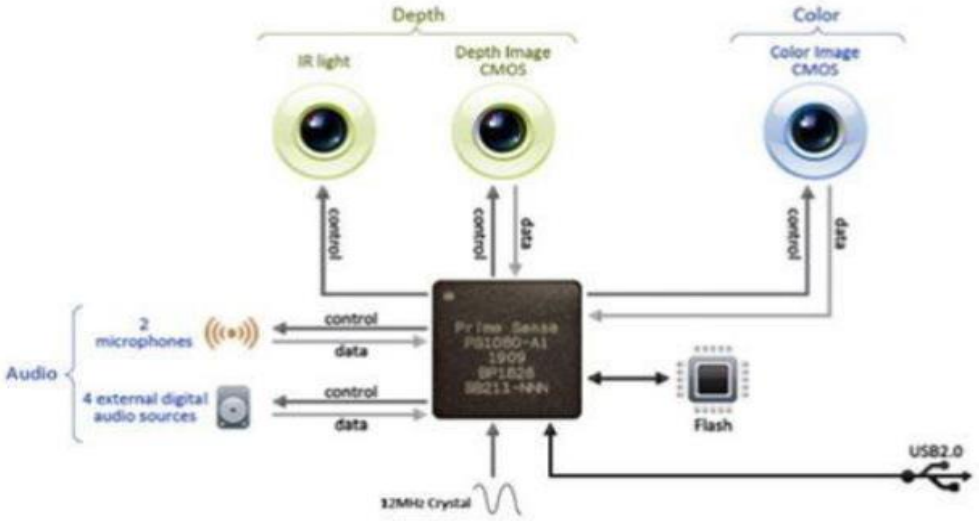

Figura 2.12. Sistema que conforma el Chip PrimeSense
(Hernandez Toalla & Herrera Rodrigez, 2013)

- Acelerómetro: configurado para tres ejes y para una gama 2G, donde ***G*** es la aceleración debida a la gravedad. Es posible utilizar el acelerómetro para determinar la orientación actual del Kinect.
- RAM 1GB
- Ventilador: Permite el enfriamiento del dispositivo.
- Adaptador: Kinect permite conectarlo a un computador sin necesidad de cables o adaptadores extras como lo muestra la Figura 2.13, éste contiene un conector para la fuente de alimentación de 110 voltios y un dispositivo USB dedicado para el ingreso de datos.

Figura 2.13. Adaptador USB para Kinect
(Microsoft, 2013)

2.2.3 FUNCIONAMIENTO DEL KINECT

El funcionamiento del Kinect se divide en las siguientes partes:

- Reconocimiento de imágenes RGB
- Reconocimiento de imágenes 3D
- Reconocimiento del esqueleto humano
- Reconocimiento de audio

2.2.3.1 Reconocimiento de imágenes RGB

El Kinect permite el reconocimiento de imágenes en tiempo real de una manera óptima gracias a la velocidad de fotogramas de aproximadamente 9 Hz a 30 Hz dependiendo de la resolución. El flujo de vídeo RGB predeterminado utiliza una resolución VGA de 8 bits por componente de color (640 × 480 píxeles) con un filtro de color Bayer , pero el hardware es capaz de resoluciones de hasta 1280x1024 (a una velocidad de fotogramas más baja) y otros formatos de color, como UYVY (Microsoft, 2013).

2.2.3.2 Reconocimiento de imágenes en 3D

El emisor de IR emite unos haces de láser, los cuales rebotan en toda el área de trabajo lo que permite que la cámara capte la profundidad de los diferentes objetos.

Al obtener estos datos el Kinect emplea una serie de filtros con la finalidad de determinar que es persona y que no lo es. El sistema emplea patrones característicos como tamaño de las extremidades, tiene cabeza, color de piel, etc., con la finalidad de determinar al ser humano de cualquier otra cosa.

Esta información es ordenada para convertirla en datos para la identificación de las partes del cuerpo humano, son ordenados en puntos de unión o de referencia absoluta para que el procesador Prime Sensor grafique el esqueleto humano (Microsoft, 2013).

2.2.3.3 Reconocimiento del Esqueleto Humano

Cuando la información tomada de la cámara RGB y de los sensores de profundidad es ordenada y reconocida a que parte del cuerpo humano corresponde, se crea un esqueleto. Kinect llena los espacios vacíos del esqueleto automáticamente en caso de que alguna acción haya tapado alguna parte del esqueleto y esto lo puede hacer ya que Kinect tiene por omisión cargado más de 200 posiciones y esqueletos comunes del ser humano (Microsoft, 2013).

2.2.3.4 Reconocimiento de audio

Kinect tiene ubicaciones específicas para los micrófonos, uno a la izquierda y tres a la derecha, ya que de esta manera se lograr un óptimo reconociendo de voz a la distancia, el ruido es anulado por la unidad de procesamiento y se utiliza su sistema de software para determinar de dónde proviene el sonido y de esta manera crear una especie de burbuja de sonido alrededor del usuario (Microsoft, 2013).

2.2.4 SDK DE KINECT

El SDK de Kinect presenta una infinidad de posibilidades a los usuarios, científicos, programadores, estudiantes y en fin a todo el público en general a desarrollar aplicaciones innovadoras aprovechando todo el potencial que este dispositivo presenta, además puede ser aplicado en infinitas áreas.

El SDK de Kinect proporciona el software sofisticado y las herramientas para ayudar a los desarrolladores a manejar el Kinect.

El SDK incluye:

- Controladores de Kinect
- Skeletal Tracking
- Procesamiento de audio

2.2.5 ARQUITECTURA DEL KINECT

La Figura 2.14 corresponde a la arquitectura del Kinect en donde los números corresponden a:

1. El hardware del Kinect
2. Los drivers de Kinect desarrollados para Windows 7, que se instalan al ejecutar el SDK y brindan soporte para:
   - El sistema de micrófonos como un dispositivo Kernel-mode al que se puede tener acceso con los API estándares de Windows
   - Streaming de datos e imágenes de profundidad
   - Funciones de enumeración de dispositivos que permitirán que una aplicación puede ocupar un Kinect conectado a la misma computadora
3. NUI API es el conjunto de API que recopilan los datos capturados por los sensores del Kinect y además controlan el dispositivo.
4. Kinect Audio DMO amplía el soporte de micrófonos de Windows 7 para exponer la localización de formación de la fuente acústica.
5. API estándares para Windows

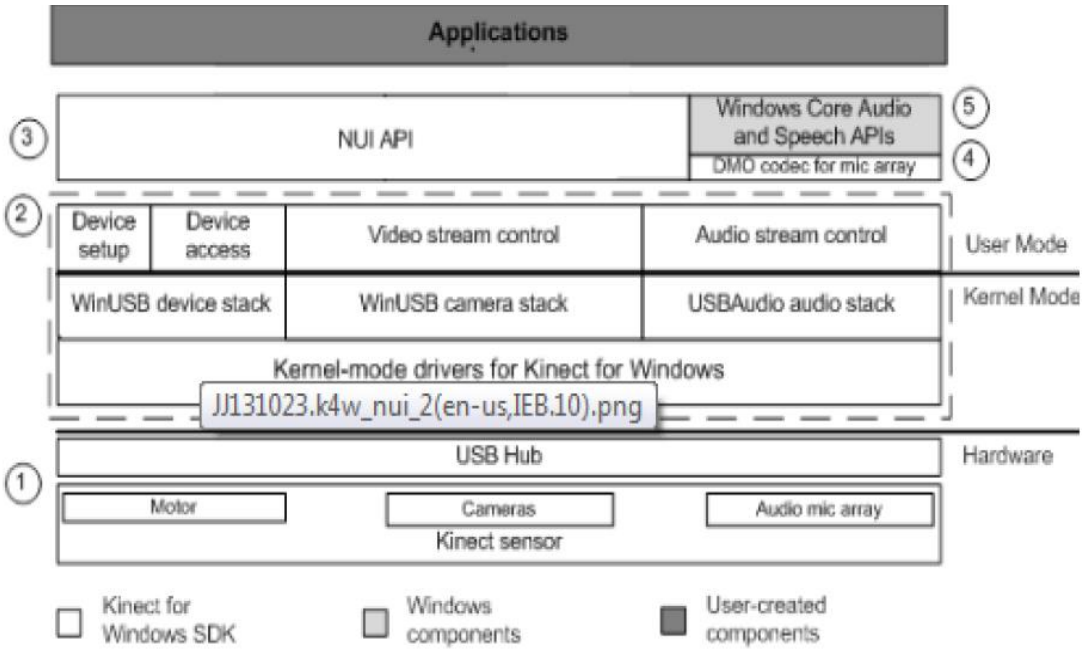

Figura 2.14. Arquitectura del Kinect
(Hernandez Toalla & Herrera Rodrigez, 2013)

2.2.6 NUI API

El NUI API (Natural User Interface o Interfaz Natural del Usuario) es el núcleo principal de Kinect para la API de Windows, que brinda soporte al manejo de datos y controla algunas de las propiedades del Kinect como:

- Brindar soporte al Skeletal Tracking por medio de las imágenes procesadas y datos de profundidad.
- Permite el acceso a las imágenes de profundidad captadas por el Kinect.
- Accesos a los distintos Kinect que se encuentran conectados al computador

El NUI API cuenta con software para reconocer y realizar el seguimiento de un cuerpo humano, permite que se reconozca hasta dos personas en frente de la cámara, la integración de API de Windows para permitir la ejecución de comandos de voz.

Además cuenta con una amplia integración con el SDK para realizar el seguimiento del rostro humano (Hernandez Toalla & Herrera Rodrigez, 2013).

El NUI API se divide en los siguientes subsistemas:

2.2.6.1 Data Stream

Permite la captura de datos como color, audio, datos de profundidad y permite procesar los datos de profundidad para generar datos del esqueleto.

Datos de la imagen a color

Los datos de la imagen a color tienen dos niveles de calidad y dos formatos diferentes. El nivel de calidad determina la velocidad con la que los datos son transferidos desde el Kinect y con el formato se determina si los datos de la imagen de color se codifican como RGB o UYVY.

- Formato RGB: Proporciona mapas de bits a color de 32 bits lineares con formato X8R8G8B8, dentro del espacio de colores RGB.
- Formato YUV: Proporciona mapas de bits a color de 16 bits con corrección gamma. Como este flujo usa 16 bits por cada pixel, este formato usa menos memoria y asigna menos buffer de memoria cuando se abre el flujo de datos. Para trabajar con YUV es necesario especificar YUV COLOR cuando se abre el flujo de datos.

La imagen que es captada por el sensor 1280 x 1024 es comprimida y transformada a RGB antes de la transmisión al Runtime. El Runtime descomprime estos datos antes de entregarlos a la aplicación. El uso de la compresión permite él envió de datos a una tasa de 30 FPS (Hernandez Toalla & Herrera Rodrigez, 2013).

Funcionamiento de la cámara RGB

Cada imagen está formada por un conjunto de pixeles. Cada pixel de la imagen está compuesto por cuatro componentes que representan el valor del rojo, verde, azul y el último componente que es el valor de transparencia (alfa), como lo muestra la Figura 2.15.

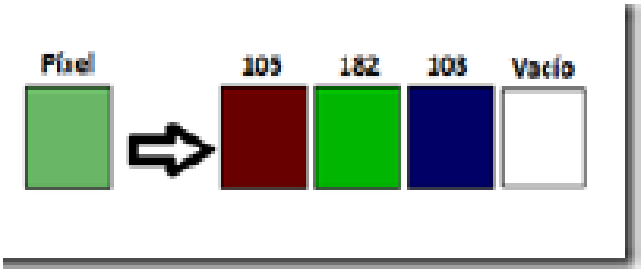

Figura 2.15. Representación de un Pixel
(Hernandez Toalla & Herrera Rodrigez, 2013)

Cada componente del pixel tiene un valor entre 0 y 255, que corresponde a un byte. El Kinect codifica las imágenes que obtiene en un vector de bytes, donde cada byte representa un componente de cada pixel.

La organización de pixeles es de arriba hacia abajo y de izquierda a derecha donde los cuatro primeros elementos del vector son los valores de rojo, verde, azul (Hernandez Toalla & Herrera Rodrigez, 2013). Como lo muestra la Figura 2.16.

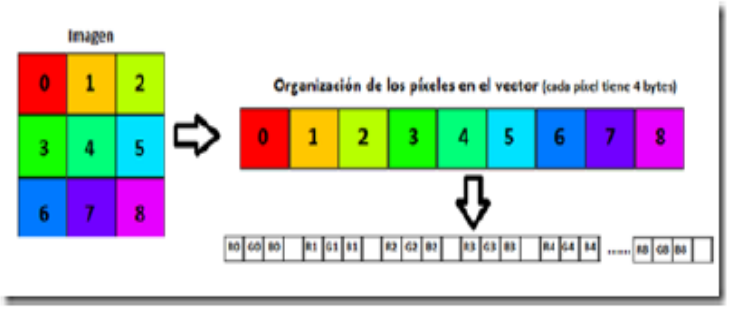

Figura 2.16. Disposición de los pixeles en una imagen RGB
(Hernandez Toalla & Herrera Rodrigez, 2013)

Datos de la cámara de profundidad

La cámara de profundidad en si se encarga de generar un vector de bytes que representa la distancia del objeto hacia el Kinect. Como lo se observa en la Figura 2.17.

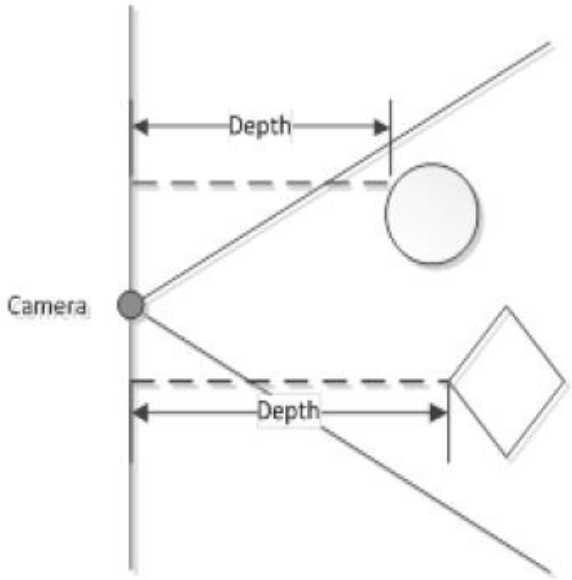

Figura 2.17. Campo de visión en profundidad del Kinect
(Hernandez Toalla & Herrera Rodrigez, 2013)

El sensor de profundidad almacena una imagen en escala de grises de todo el campo visible del sensor de profundidad, como lo muestra la Figura 2.18, en donde cada pixel representa la distancia cartesiana más cercana desde la cámara al objeto. Las coordenadas cartesianas del sensor de profundidad solo representa la ubicación de un pixel en el marco de profundidad (Hernandez Toalla & Herrera Rodrigez, 2013).

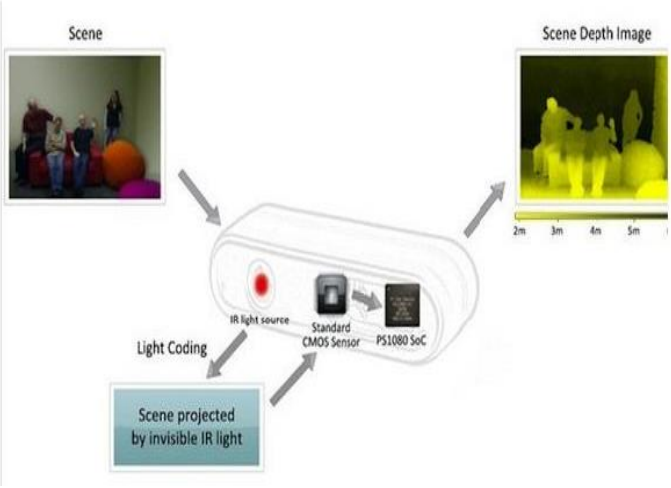

Figura 2.18. Imagen de profundidad en escala de grises
(Hernandez Toalla & Herrera Rodrigez, 2013)

Cada pixel es guardado en dos bytes del vector, como lo muestra la Figura 2.19, esto se debe a que como Kinect tiene dos cámaras de infrarrojo cada pixel guarda la distancia de cada cámara.

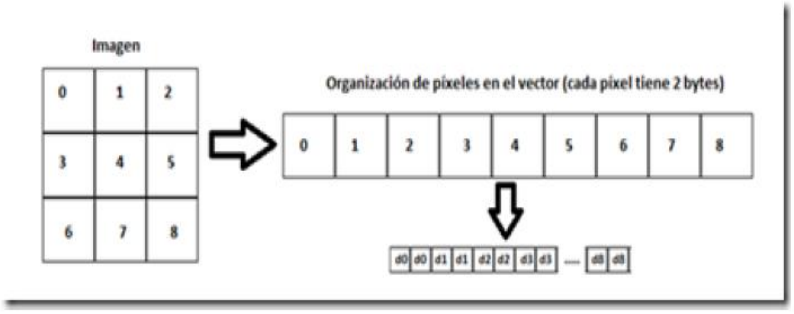

Figura2.19. Disposición de los pixeles en una imagen de profundidad
(Hernandez Toalla & Herrera Rodrigez, 2013)

Un valor de profundidad igual a cero indica que no hay datos de profundidad disponibles para esa posición, puede ser debido a que todos los objetos están muy cerca de la cámara o muy lejos.

La figura 2.20 muestra como el campo de visión del Kinect es almacenado en dos mapas uno de color y el otro de profundidad y, con el empleo del emisor IR se logra eliminar la luz ambiente, ya que no registra la luz visible, por su longitud de onda.

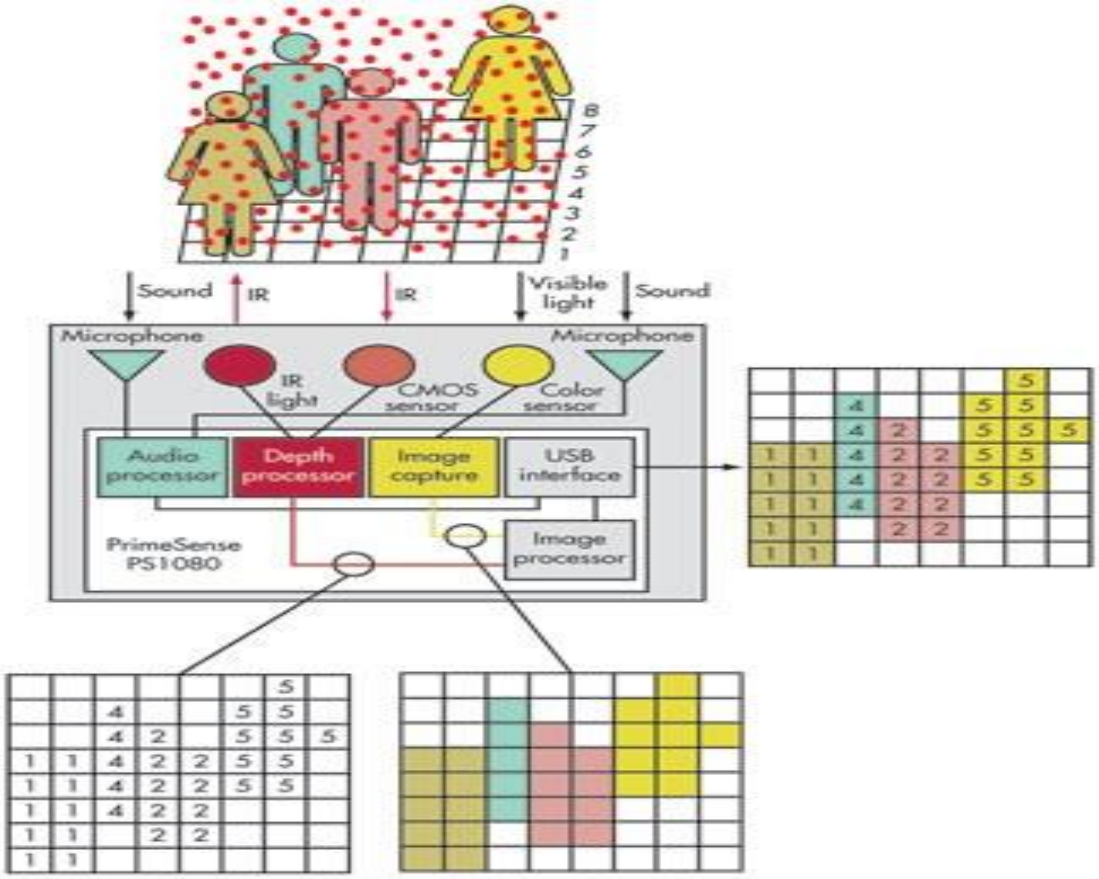

Figura 2.20. Proceso de adquisición de datos de la cámara RGB y de los sensores de profundidad
(Hernandez Toalla & Herrera Rodrigez, 2013)

2.2.6.2 Skeletal Tracking

Es la función más prometedora del Kinect, permite detectar la figura humana (esqueleto) cuando se está moviendo, estos movimientos son asociados a través de unos puntos o joints los cuales grafican el esqueleto humano. Para

la creación del cuerpo humano Kinect coge las siguientes articulaciones que se puede observar en la Figura 2.21 (Microsoft, 2013).

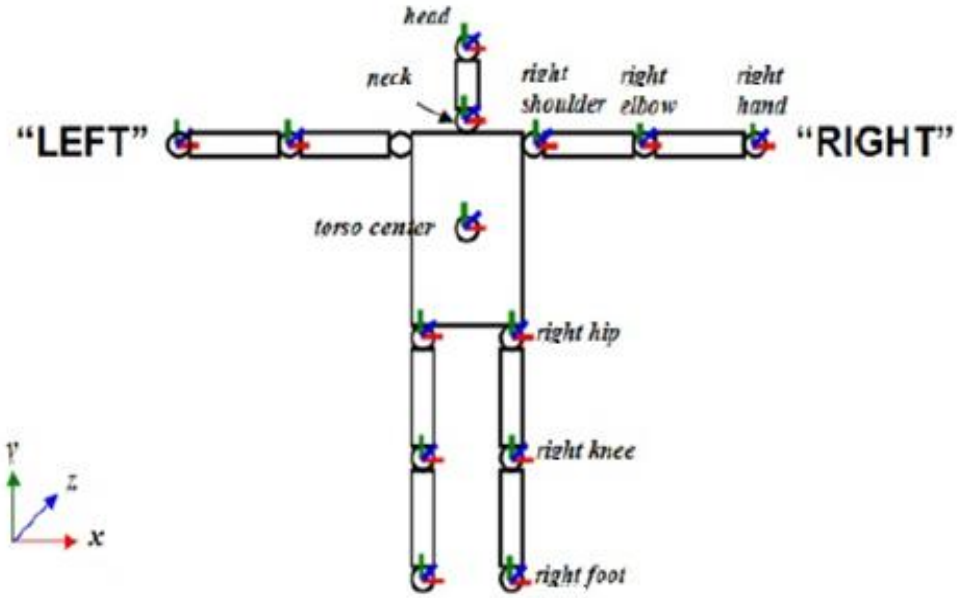

Figura 2.21. Joints del cuerpo humano que reconoce el Kinect (Matlab-Simulink, 2013)

Todo esto se logra debido a que el Skeletal Tracking tiene almacenado los datos de la imagen a color y los datos de la cámara de profundidad, lo que permite reconocer entre objetos y seres humanos ya que asocia parámetros del cuerpo humano como extremidades superiores, articulaciones e incluso gestos, para que de esta forma el cuerpo humano sea reconocido y detecte el movimiento de todas las partes que conforman el esqueleto humano (brazos, manos, muñeca, bazos, rodillas, codos, cadera, etc.).

La Figura 2.22 muestra el esquema del Skeletal Tracking que consta de un Skeleton Frame y este a su vez está compuesto por un vector (array) de SkeletonData (clase que contiene los datos de cada esqueleto), no todos los Skeleton Frame contienen SkeletonData (Hernandez Toalla & Herrera Rodrigez, 2013).

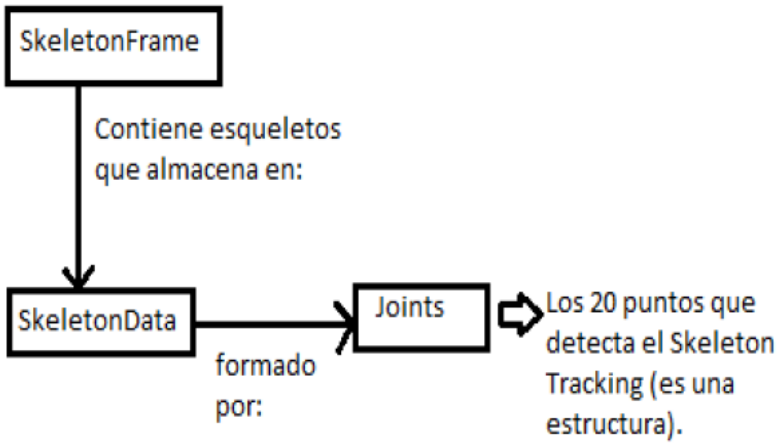

Figura 2.22. Esquema Skeletal Tracking
(Hernandez Toalla & Herrera Rodrigez, 2013)

De esta manera cuando una persona realice un movimiento el Kinect lo asocia con la información que contiene el Skeleton Tracking y lo muestra en la aplicación.

El proceso para que el Kinect sea capaz de detectar el esqueleto humano consta de seis pasos, como muestra la Figura 2.23.

1. El emisor IR lanza haces de rayos infrarrojos que son recibidos y reconocidos.
2. Kinect crea el mapa de profundidad a partir de los puntos reconocidos.
3. Detecta el suelo y separa los objetos del fondo para encontrar el contorno humano.
4. Asocia las partes detectadas para hacer una clase de calificación de las partes humanas.
5. Identifica las articulaciones.
6. Simula un esqueleto.

2.2.6.3 Speach (procesamiento de audio)

El sistema de micrófonos consiste en un conjunto de cuatro micrófonos dispuestos en forma de L, el tener el conjunto de micrófonos en esta posición da diversas ventajas respecto a un micrófono solo.

- Mejora la calidad del audio.
- Como el audio llega en diferente tiempo a cada micrófono, se puede determinar la dirección de donde está el origen del sonido y usar el conjunto de micrófonos como un micrófono direccional orientable.
- Las aplicaciones que usen el SDK podrán utilizar los micrófonos para lo siguiente:
    - Captura de audio de alta calidad
    - Formación de haz y localización de la fuente
    - Reconocimiento de voz

2.2.7 REQUERIMIENTOS DE HARDWARE Y SOFTWARE

2.2.7.1 Requerimientos de hardware

- Procesador de 32 bits(x86) ó 64 bits(x64)
- Procesador de doble núcleo de 2.66 GHz o más rápido
- Tarjeta gráfica compatible con Windows 7 que soporte Directx 9.0
- 2GB de RAM o más recomendado
- Bus USB 2.0 dedicado

2.2.7.2 Requerimientos de software

- Sistema Microsoft Windows 7 o Microsoft Embedded Standard 7
- Visual Studio 2010, cualquier edición
- Microsoft. NET Framework 4.0

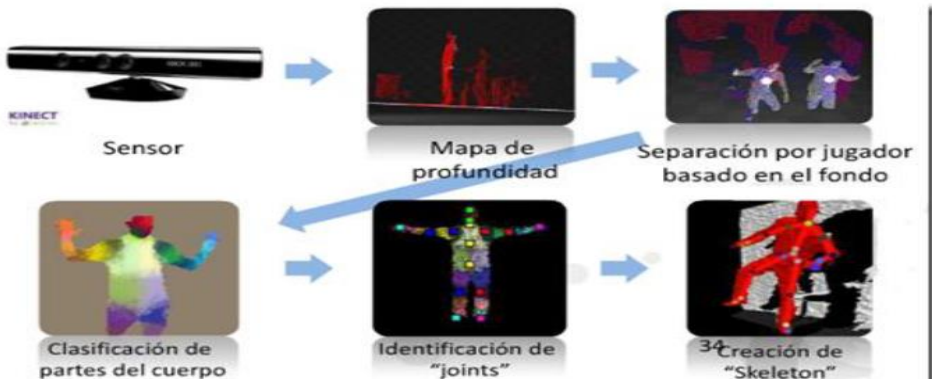

Figura 2.23. Etapas del reconocimiento del Skeleton Tracking (Microsoft, 2013)

2.3 TIC EN LA EDUCACIÓN

La sociedad de la información, la Internet, la globalización y la movilidad estudiantil nos hacen cambiar e innovar constantemente los modelos educativos tradicionales donde un profesor era casi el dueño del conocimiento y su labor consistía en transmitir ese conocimiento a quienes eran sus alumnos. Ahora se debe establecer nuevos paradigmas educativos y proponer modelos que permitan mejorar la calidad de la educación superior, utilizando otros recursos y nuevas herramientas. Por lo que es de vital importancia contar con modelos pedagógicos que den respuesta a las demandas sociales caracterizadas por la influencia y el uso de las Tecnologías de la Información y Comunicación (TIC). (Negrete, 2013)

Las TIC se están convirtiendo en un instrumento indispensable en las instituciones educativas, ya que facilitan la toma de decisiones en grupo intercambiando ideas de forma ágil entre los integrantes.

Con la llegada de la tecnología, el entorno educativo está cambiando de un enfoque centrado en el docente basado en clases magistrales, hacia un enfoque centrado en el estudiante basado en clases integradoras; donde el estudiante interactúa adquiriendo nuevos conocimientos a través de una

búsqueda continua de contenidos y procedimientos viéndose obligado a tomar decisiones.

Desde diversas instancias se pide a las instituciones de educación superior que flexibilicen sus procedimientos y su estructura administrativa para adaptarse a nuevas modalidades de formación más acordes con las necesidades que la nueva sociedad presenta.

2.3.1 TIC EN EL PROCESO ENSEÑANZA-APRENDIZAJE

La sociedad actual, la sociedad llamada de la información demanda cambios en los sistemas educativos de forma que éstos se tornen más flexibles, accesibles y menos costosos. Las instituciones ecuatorianas de formación superior para responder a estos desafíos deben revisar sus referentes actuales y promover experiencias innovadoras en los procesos de enseñanza-aprendizaje apoyados en las Tecnologías de la Información y la Comunicación (TIC).

Existen muchas dimensiones para entender las TIC en educación, el conjunto de dimensiones más usado técnicamente se llama LA PIOLA, la cual es una manera de entender las oportunidades tanto en el proceso educativo como en la vida personal. A través de ella se puede acceder y difundir la información interactuando con otras personas. (Saenz, 2010)

Se analiza las cinco dimensiones para entender las TIC en educación a través de la Piola.

P. Para mejorar la **productividad** individual, simplificando actividades y ampliando la capacidad personal.

Con relación a ésta las TIC presentan algunas herramientas como: comunicación basada en textos, comunicación escrita, procesamiento de datos, cálculo y análisis de datos numéricos, análisis estadísticos de datos, expresión gráfica, etc. Una de las herramientas más utilizada por el docente en el área de bioquímica es la presentación de multimedia como Power point, videos y páginas web.

I. Interacción con otros individuos o grupos, dialogando de forma sincronizada por medio de herramientas como: Skype y MSN o asincrónicamente por medio de herramientas como: foros en la red.

Este tipo de herramienta es muy utilizada pero hay que tener cuidado, porque hay mucha comunicación errónea por este medio.

O. Objetos de Estudio. En esta herramienta los estudiantes se ven en la necesidad de apropiarse de mucho conocimiento cultural, científico y tecnológico.

Estas pueden contribuir a entender lo que se estudia, pero suelen no ser suficiente para lograr un dominio conceptual de lo aprendido. Por ejemplo, en el área del campo de la salud se sabe que existen los laboratorios de simulación, pero estos no son suficientes para que un estudiante en este campo adquiera un total conocimiento porque su disponibilidad es limitada.

L. Labores educativos. La construcción de mapas conceptuales por parte del estudiante se convierte en una forma de expresar su entendimiento de lo que ha estudiado, pero también puede ser usado por el docente para crear sistemas de navegación visual por conceptos relacionados, como apoyo al proceso de aprendizaje.

A. Ampliar nuestro acervo cultural, científico y tecnológico, manteniéndonos actualizados en lo que nos interesa, a partir de consultas a fuentes directas.

La Internet y otras redes son medios que el docente puede utilizar para mantenerse actualizado en diferentes temas relacionados con la asignatura que se encuentre dictando y puede usar con sus estudiantes herramientas como Yahoo, Google académico entre otros.

LA PIOLA permite entender las TIC en educación y genera estrategias para su integración al proceso de formación en el desarrollo profesional docente.

En lo pedagógico se busca que cada docente pueda vivir el ciclo de vida de un proceso educativo centrado en el estudiante valiéndose de las TIC.

2.3.1.1 Experiencias del uso del Kinect en el proceso enseñanza aprendizaje

En la Educación Kinect ha demostrado su utilidad como herramienta para facilitar el aprendizaje y hacer la experiencia en clase más interactiva y enriquecedora “Poner un Kinect en el aula es transformarla en un espacio de aprendizaje interactivo” (Lopez, 2013)

Kinect mantiene la atención permanente en el aula de clase, además crea un ambiente con más motivación. “En nuestro proyecto con niños, nos hemos dado cuenta de que tras las ocho semanas de duración del proyecto, los niños han mostrado una mejor integración en su entorno escolar, además de una mejora en su equilibrio, coordinación y estado anímico." (Rosa, 2013)

Kinect ha logrado una nueva forma de enseñanza que motiva, divierte y facilita el aprendizaje significativo.

3. METODOLOGÍA

La metodología mecatrónica como lo muestra la Figura 3.1 (NationalInstruments, 2010) parte de un concepto único de especificaciones y necesidades del sistema, para luego desarrollar paralelamente los diseños mecánicos, diseños electrónicos/eléctricos, diseños de hardware y software embebido y un diseño de control acorde a las necesidades. Luego todos esos diseños pasan a un prototipo virtual donde se simula el funcionamiento del prototipo, se corrigen fallas y errores; finalmente pasa a su última etapa de diseño que es la elaboración, manufactura y diseño físico del producto. A esto se le suma la parte de servicios que se le ofrece como soporte y mantenimiento. Toda esta metodología se la conoce con el nombre de Model based in Design según (Mathworks, 2010) y tiene como objetivo fundamental no depender de prototipos físicos y propone el diseño basado en modelos de sistemas especializados y al detalle, totalmente ejecutables durante todo el desarrollo. Esto permite una compatibilidad completa que abarca el diseño, componentes, simulación, generación automática de código, pruebas y verificaciones continuas en el sistema.

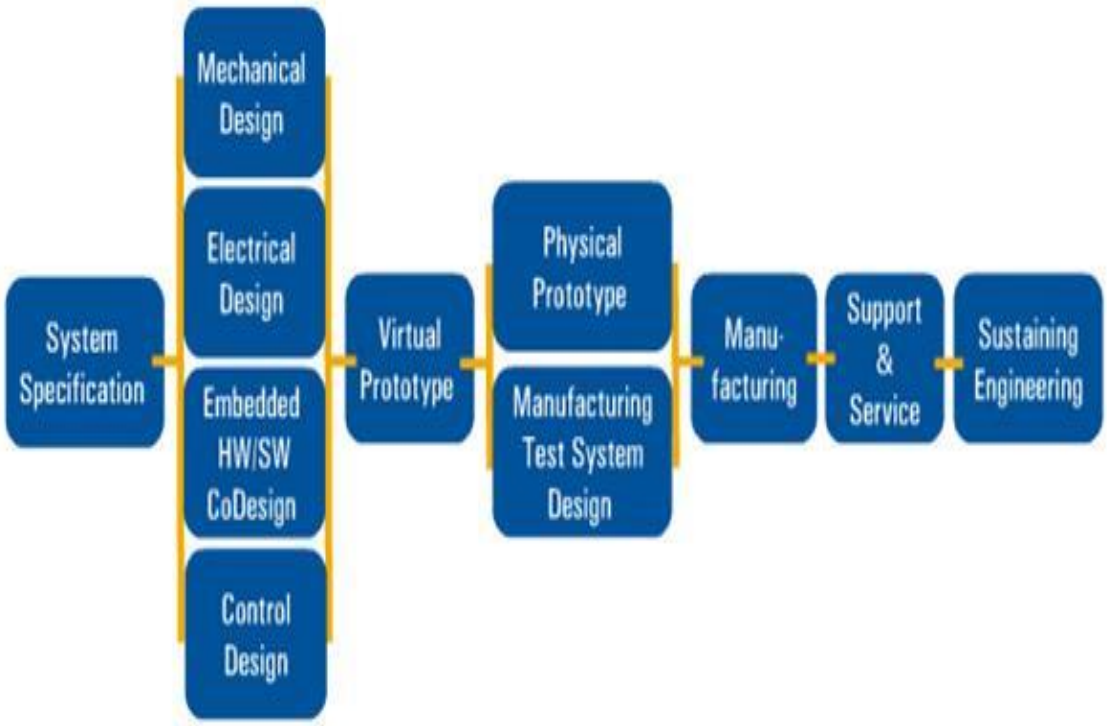

Figura 3.1. Metodología Mecatrónica
(NationalInstruments, 2010)

El desarrollo de aplicaciones se encuentra dentro del campo de la mecatrónica como una herramienta de interacción entre el sistema mecatrónico con el usuario, y como una herramienta de control para ciertas tareas donde se involucra la parte física (hardware) y el sistema de control (software).

Cuando se diseñó la aplicación remota de interacción entre el docente y un PC se tomó en cuenta que la aplicación formaba parte de dos grupos de metodologías: la metodología mecatrónica por usar un dispositivo sensorial mecatrónico para controlar un PC y la metodología estructurada por ser un programa diseñado como herramienta de control que permite al usuario realizar ciertas tareas.

Se ha enfocado el desarrollo de la aplicación con la metodología estructurada, ya que esta permite explicar de una manera más detallada las diferentes etapas que tuvo la aplicación. La metodología estructurada tiene un modelo de desarrollo tipo cascada, que empieza desde una visión general del problema y va descendiendo a niveles de abstracción más sencillos; estos niveles están definidos por procesos especializados en cada nivel. Esto permite que el producto evolucione a través de una secuencia de etapas ordenadas en forma lineal, permitiendo iteraciones al estado anterior. Es muy importante que se cumplan todas sus etapas de la metodología como lo muestra la Figura 3.2.

En este capítulo se realiza una investigación preliminar sobre la aplicación y se analiza todos los requerimientos tanto del sistema como de la aplicación. En el capítulo 4 se explica el diseño de la aplicación, por último en el capítulo 5 se efectúa las pertinentes pruebas y se explica los resultados.

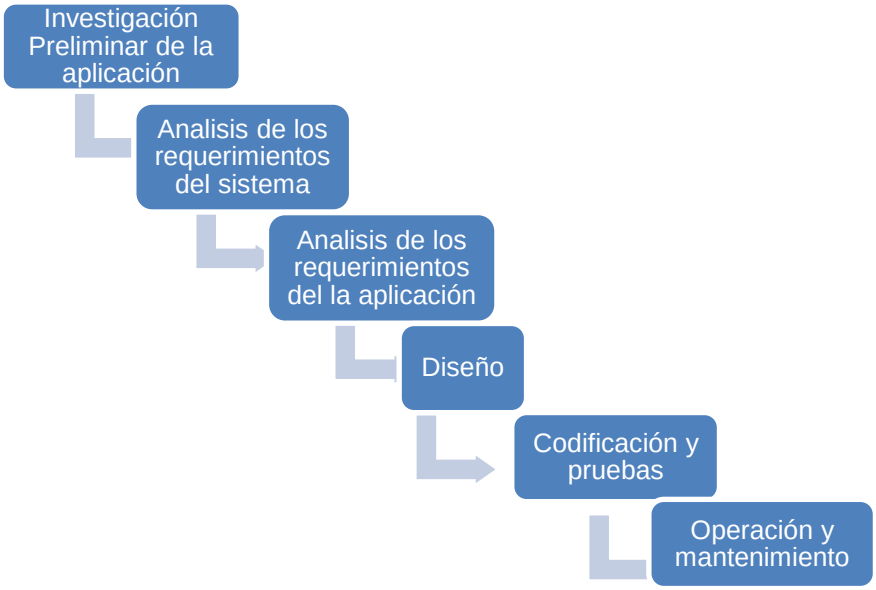

Figura 3.2. Metodología estructurada cascada para desarrollo de software (Sigwart, 1990)

3.1 INVESTIGACIÓN PRELIMINAR

Las entidades educativas para estar acorde al desarrollo tecnológico ven la necesidad de adquirir nuevos dispositivos y nuevas aplicaciones que permitan una mejor interacción en clases para agilitar el proceso enseñanza-aprendizaje entre el docente y el estudiante. Los dispositivos más usados en las entidades educativas en el Ecuador son las pizarras inteligentes (Smart Boards) que tienen un costo elevado.

El desarrollo de la aplicación de interacción remota entre el docente y el PC nació de la necesidad de crear nuevos entornos y ambientes educativos para el alumno y el docente que sean viables, económicos y desarrollados dentro de la Universidad.

El desarrollo de esta aplicación que puede cumplir con las misma funciones de una pizarra inteligente tiene un costo más bajo con relación a las pizarras inteligentes. El punto trascendental es que esta aplicación se la puede realizar en la Universidad Tecnológica Equinoccial debido a que las licencias de los programas que se usan para el desarrollo de la aplicación la

Universidad ya las posee; además los estudiantes están capacitados técnica y teóricamente para desarrollar cualquier aplicación. Con esto los costos ya están cubiertos y el único gasto seria en comprar el dispositivo Kinect.

En la siguiente tabla se especifican los costos de la aplicación y los costos de los dispositivos tecnológicos de funcionalidad parecida.

Tabla 3.1. Costos de dispositivos tecnológicos educativos

Dispositivo	**Marca**	**Costo($)**
Pizarra Inteligente	Smart Board 885ix2	7999
Pizarra Inteligente	Smart Board	2600
Pizarra Inteligente	Mimio	650
Pizarra Inteligente	Pizarra Wii	500

(Mecadolibre, 2014)

En la Tabla 3.1 se puede observar la variabilidad de los costos de los dispositivos tecnológicos educativos (pizarras inteligentes), la variabilidad de sus costos depende de la tecnología que utilicen y de las dimensiones de la pizarra.

Tabla 3.2. Rubros del desarrollo de la aplicación

Rubros	**Costo($)**
Kinect	220
LEDS Indicadores	10
Horas de Trabajo y Desarrollo	200
TOTAL	**430**

En la Tabla 3.2 se puede observar los rubros del desarrollo de la aplicación, los rubros más importantes son el Kinect y las horas de trabajo y desarrollo.

El desarrollo de esta aplicación es viable por su menor costo en comparación a dispositivos tecnológicos usados para el mismo fin, existe el personal capacitado para el desarrollo e investigación de la aplicación, existen las herramientas para realizarla, se cuentan con todas las licencias de los programas de desarrollo para la aplicación.

3.2 REQUERIMIENTOS DEL SISTEMA

Es el conjunto de necesidades exclusivas tanto de hardware como de software que permiten la funcionalidad del sistema. Estas necesidades se las debe suplir de una forma precisa y detallada para evitar fallos y averías del mismo sistema.

3.2.1 REQUERIMIENTOS HARDWARE

Se necesita un dispositivo sensorial que permita la detección de personas y que realice un seguimiento al cuerpo humano con la posibilidad de identificar cada parte del cuerpo.

3.2.2 REQUERIMIENTOS SOFTWARE

Se necesita un software de control y programación que permita el desarrollo de aplicaciones y que tenga la conectividad con diferentes plataformas tecnológicas como Arduino y Microsoft Kinect. Debido a que la aplicación se la realiza para la Universidad Tecnológica Equinoccial es muy importante que la UTE tenga o adquiera el licenciamiento de este software o que se desarrolle la aplicación en software libre.

Cumpliendo todas estas premisas se escogió como requerimiento de software el programa Matlab-Simulink.

3.2.2.1 Matlab

Es un programa que permite realizar cálculos técnicos y científicos para diferentes disciplinas, se ha convertido prácticamente en un estándar de programación y de desarrollo rápido de aplicaciones. La unidad de procesamiento de Matlab es la matriz, por lo que se lo utiliza en el procesamiento de imágenes. El lenguaje de programación de Matlab es el código M.

3.2.2.2 Simulink

Simulink puede ser considerado como un ambiente de simulación de sistemas incorporado al programa Matlab, el cual permite analizar la respuesta en tiempo de sistemas dinámicos complejos. Con la incorporación de la herramienta llamada Video and Image Processing Blockset, Mathworks logró generar un ambiente de implementación de algoritmos de procesamiento de imágenes en tiempo real.

Librería Computer Vision System Toolbox

Esta librería provee de algoritmos y herramientas para el diseño y simulación de visión artificial y procesamiento digital de video. La librería incluye algoritmos de extracción de características, detección de objetos, seguimiento de objetos y análisis de video.

Librería NID en Simulink

La librería NID que su acrónimo en inglés significa Natural Interface Device es un conjunto de bloques desarrollados con S-Function en código M de Matlab para la simulación de estados discretos en Simulink. Esta librería permite la interacción entre la interfaz de Simulink y la interfaz natural del usuario (NUI) de Microsoft Kinect a través del reconocimiento del cuerpo, sus movimientos y gestos.

Requerimientos de la librería NID

- Matlab R2013b ultima versión recomendada
- Simulink
- Computer Vision System Toolbox de Mathworks última versión recomendada
- Microsoft Visual Studio 2010
- Microsoft Kinect for Windows SDK ultima version

Microsoft Visual Studio 2010 Profesional

Microsoft Visual Studio se usó en Simulink como compilador para generar código C de un entorno de control de aplicaciones llamado StateFlow.

3.2.2.3 Microsoft Kinect for Windows SDK

Es el software de desarrollo de Kinect que permite realizar aplicaciones en Windows y posee todos los archivos de cabecera, bibliotecas, muestras, documentación, herramientas y API necesarias para el desarrollo de aplicaciones.

3.3 REQUERIMIENTOS DE LA APLICACIÓN

Se necesita desarrollar una aplicación de reconocimiento e interpretación de los movimientos y gestos del cuerpo humano para controlar eventos en un PC. Estos eventos son: secuencia de la presentación y el nivel de zoom.

3.3.1 REQUERIMIENTOS DEL CONTROL DE EVENTOS

En la Figura 3.3 se observa el diagrama de flujo de la aplicación, esta es la pieza clave que permite entender de una forma global su funcionamiento sin entrar en detalle de las especificaciones de control, solo con la información de los movimientos que se realizan para controlar la secuencia de presentaciones y el nivel de zoom.

De acuerdo a la Figura 3.3 se determinaron los posibles parámetros del cuerpo que deben ser identificados y procesados para control de la aplicación y se limitó la zona de análisis a la parte superior del cuerpo donde se encuentran las manos; por esta razón el resto del cuerpo que se encuentra de la cintura para abajo no es necesario analizarlo.

En el próximo capítulo se analiza de una forma más detallada como el programa identifica y procesa toda la información entregada por el Kinect.

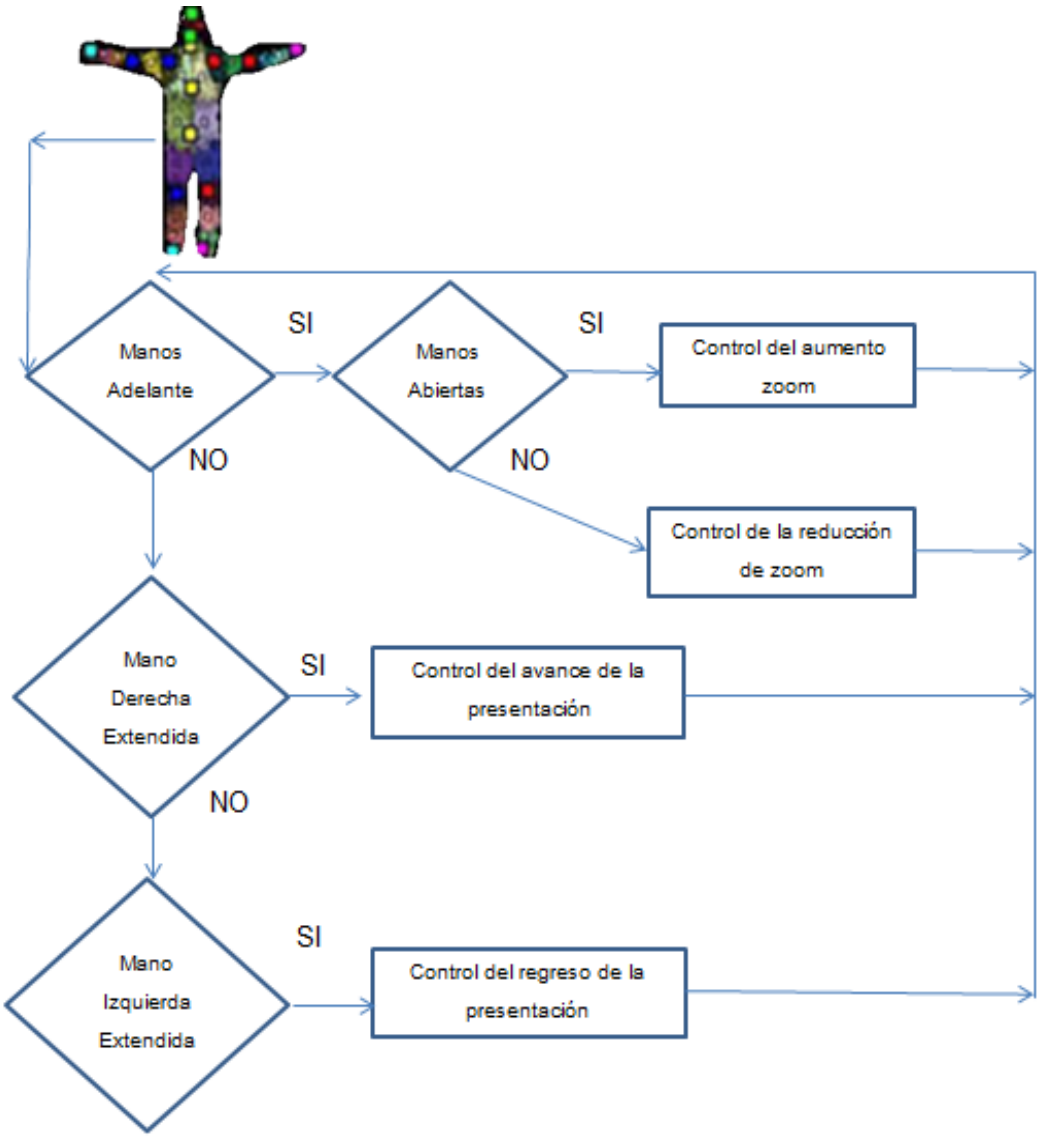

Figura 3.3. Diagrama de flujo del funcionamiento general de la aplicación

3.3.2 REQUERIMIENTOS DE LOS DATOS CARACTERÍSTICOS DE LA APLICACIÓN

Los datos característicos son la información que describe a cada uno de los patrones que se necesita para el reconocimiento y la interpretación de imágenes. En la tesis de maestría en control y automatización (Sobrado, 2003) se describe cuáles son los requerimientos de los datos característicos:

- Discriminación: Los valores numéricos deben ser diferentes para objetos de clases distintas.
- Fiabilidad: La diferencia de valores numéricos en una misma clase debe ser pequeño, los objetos de una misma clase deben representar la menor dispersión.
- Incorrelación: Las características son independientes para cada clase, nunca debe utilizarse características que dependan fuertemente entre clases.
- Dimensionalidad: El tamaño del vector característico debe ser menor que el vector patrón original.

3.3.3 REQUERIMIENTOS DIMENSIONALES

Hace mención a la ubicación espacial del Kinect en referencia con el usuario. Se debe tomar en cuenta que el Kinect tiene un rango de visión en el cual permite la detección de la persona, de sus movimientos y gestos. El rango de visión está vinculado con el ángulo de apertura horizontal y el ángulo de apertura vertical de su cámara. La calibración de estos ángulos se la debe hacer en simultáneo con el tamaño de la proyección de la presentación o diapositiva, ya que encuadrar a ambos permite un mejor funcionamiento de la aplicación. Ubicar al Kinect debajo del proyector facilita esta calibración.

Se debe tomar en cuenta los siguientes parámetros de:

- Distancia entre el usuario y el Kinect
- Estatura del usuario
- Área óptima de trabajo

El usuario siempre debe estar de frente al Kinect por su función principal que es controlar la secuencia de las presentaciones; además debe existir una distancia determinada entre los dos para una explicación adecuada del contenido. Estos parámetros deben acoplarse a cualquier persona dentro de un rango de alturas de 1.50 a 2.0 m.

4. DISEÑO DE LA APLICACION

Para la aplicación se analizó los diferentes dispositivos que pueden reconocer los gestos y movimientos del usuario. Se escogió el dispositivo Kinect por ser el dispositivo más usado en video juegos con NUI (Natural User Interface) que reconoce todo el cuerpo y posee un SDK (Software Development Kit) libre al público para desarrollar aplicaciones con sus librerías en el sistema operativo Windows.

El software de desarrollo matemático de aplicaciones Matlab y su ambiente de simulación de modelos dinámicos complejos Simulink permitieron el desarrollo de la aplicación en esta plataforma. Se usó la librería NID de Kinect en Simulink y se realizaron diferentes modelos de interacción con la librería NID en Simulink para controlar diferentes sensores del Kinect, adquiriendo datos como: imágenes RGB, imágenes de profundidad, imágenes en movimiento, coordenadas del esqueleto y el seguimiento del esqueleto.

A los modelos de interacción con la librería NID se los modificó, reestructuró y ensambló con otras librerías de Computer Vision System Toolbox en Simulink, llegando a obtener datos característicos esenciales que se necesitan para el control de esta aplicación. Para poder obtener estos datos se cumplieron las etapas de la visión artificial (Sobrado, 2003) censado, segmentación, descripción y reconocimiento.

Para la etapa de interpretación se usó un State Flow en Simulink, que es un entorno para el modelado y simulación de lógica de decisión combinatoria y secuencial basado en máquinas de estado y diagramas de flujo; con la unión de la lógica de decisión combinatoria y los datos característicos provenientes del reconocimiento en las anteriores etapas se desarrolló un control de eventos accionados por transiciones. Las transiciones permiten el cambio de un estado a otro y los estados son los bloques que contienen los eventos.

Los eventos obtenidos en cada estado del StateFlow entran en un bloque de control construido a través de una S-Function en el cual los eventos son la condición de verdad en una secuencia lógica de código con estructuras de

selección. Si el evento ocurre, la condición es verdadera por lo cual se realiza una acción o una operación; esta acción es la que permite controlar la secuencia de presentaciones y el aumento o reducción del zoom en el PC.

4.1 BLOQUES NID DE KINECT EN SIMULINK

En la Figura 4.1 se observa los bloques que posee la Librería NID en Simulink para la adquisición de imágenes RGB, de profundidad, con movimiento, IR e imágenes con detección del esqueleto humano usando el dispositivo Kinect.

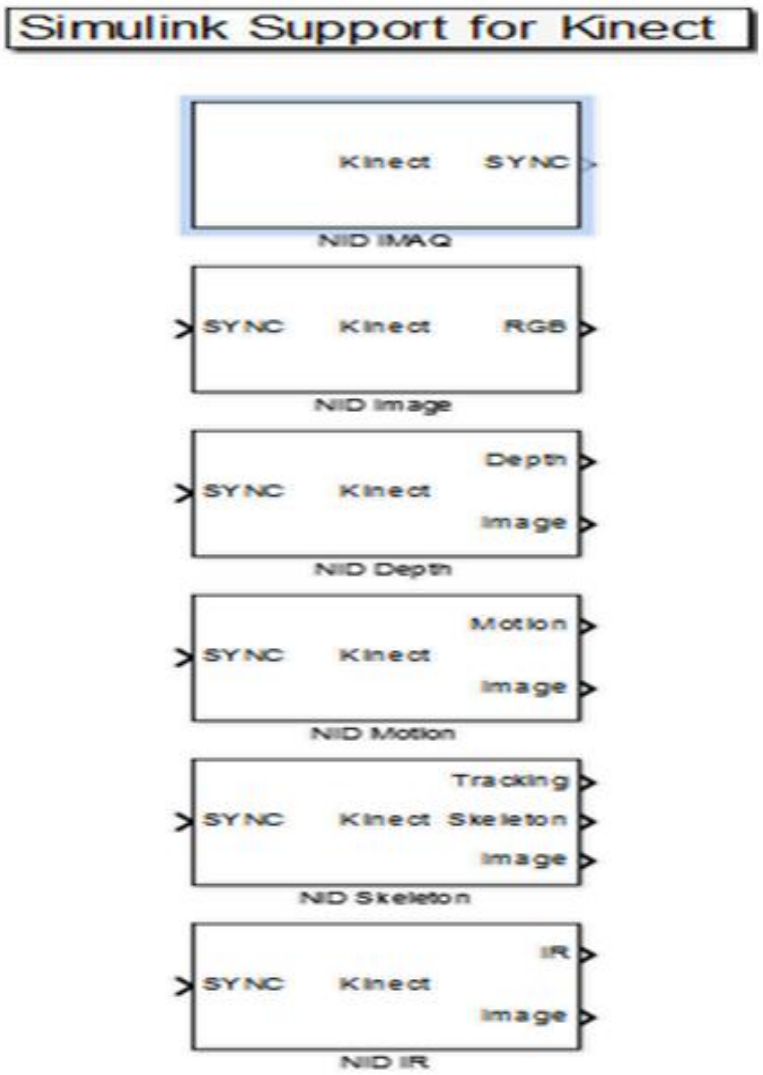

Figura 4.1. Bloques NID para Simulink
(Matlab-Simulink, 2013)

4.1.1 BLOQUE IMAQ NID

El bloque IMAQ mostrado en la Figura 4.2 es el bloque principal de la librería NID, adquiere todas las señales del dispositivo Kinect. Este bloque necesita una simulación en tiempo fijo; el tipo de simulación se puede configurar en la opción simulación en la barra de tareas de Simulink y escoger la opción configuración de parámetros de simulación.

Figura 4.2. Bloque IMAQ de la Librería NID en Simulink
(Matlab-Simulink, 2013)

El bloque IMAQ es el enlace entre el Kinect y Simulink, este permite adquirir todos los data stream provenientes de la NUI API y usar todas las funciones del SDK del Kinect, como lo muestra la Tabla 4.1.

Tabla 4.1. Descripción de las señalas de entrada/salida del bloque IMAQ en Simulink

Tipo de Señal	**Nombre de la Señal**	**Tipo de Dato de la Señal**	**Descripción de la Señal**
Entrada	Angulo	Int32	Controla el ángulo de elevación con respecto a la gravedad del Kinect en grados. El valor del ángulo se encuentra entre -27º por debajo del dispositivo y 27º por encima del dispositivo.
Salida	SYNC	Uint32	Entrega los datos de adquisición del Kinect en ms para sincronizar con los bloques de Depth/Motion/Image/Skeleton/IR La señal SYNC tiene que alimentar directamente con los bloques Depth/Motion/Image/Skeleton/IR.
Entrada	Angulo	Int32	Entrega la información del ángulo de elevación con respecto a la gravedad del Kinect.

4.1.1.1 Parámetros del bloque IMAQ

Parámetros principales del Bloque IMAQ

- Ajuste del punto de vista de la imagen: Ajusta la nitidez de la imagen para todos los bloques de la librería NID; si no se lo utiliza en el modelo, este parámetro no afecta.
- Imagen espejo: Genera datos de imagen de espejo; afecta a todos los datos del dispositivo.
- Tiempo de la muestra: Controla el tiempo de la muestra que es igual 1/(Frames por segundo) y este a su vez debe ser mayor al tiempo fijo de simulación debido a que el bloque tiene una espera interna que transcurre desde la última adquisición datos.

Parámetros SDK

- Entrada del ángulo: Controla el ángulo de elevación con respecto al centro de gravedad del Kinect.
  - ❖ Entrada por lectura: Lee el ángulo de elevación elegido desde la configuración del Kinect
  - ❖ Entrada por sensor: Establece el ángulo de elevación tomado desde un sensor externo del Kinect.
- Modo cercano: Establece la configuración del Kinect para medir la distancia del objeto sensado por la cámara de profundidad, el campo de visión del modo cercano va desde los 0.8m hasta los 2.5m al frente del Kinect.
- Modo sentado: Permite realizar un seguimiento de la parte superior del esqueleto humano.
- Parámetros de suavizado del esqueleto: Estos parámetros permiten controlar la estabilidad de la inferencia de las posiciones en el Kinect. Los parámetros son: Suavizado, Corrección, Predicción, Radio Jitter, Máxima desviación del radio.

4.1.2 BLOQUE IMAGE NID

El bloque IMAGE mostrado en la Figura 4.3 permite adquirir una imagen RGB24 desde el Kinect como lo muestra la Tabla 4.2.

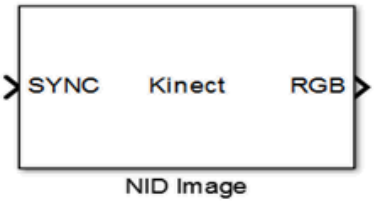

Figura 4.3. Bloque IMAGE de la Librería NID en Simulink
(Matlab-Simulink, 2013)

Tabla 4.2. Descripción de las señales de entrada/salida del bloque IMAGE en Simulink

Tipo de Señal	Nombre de la Señal	Tipo de Dato de la Señal	Descripción de la Señal
Entrada	SYNC	Uint32	Entrega los datos de adquisición del Kinect en ms para sincronizar con los bloques de Depth/Motion/Image/Skeleton/IR. La señal SYNC tiene que alimentarse directamente con los bloques Depth/Motion/Image/Skeleton/IR.
Salida	RGB	Uint8	Entrega una imagen RGB de 24 bits.
Salida	R/G/B	Uint8	Entrega una imagen separada RGB de 24 bits.

4.1.3 BLOQUE DEPTH NID

El bloque DEPTH mostrado en la Figura 4.4 permite adquirir una imagen de profundidad desde el Kinect como lo muestra la Tabla 4.3.

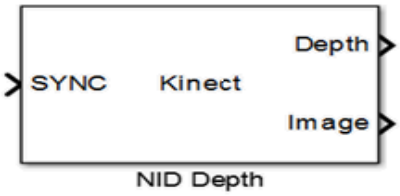

Figura 4.4. Bloque DEPTH de la Librería NID en Simulink
(Matlab-Simulink, 2013)

Tabla 4.3. Descripción de las señales de entrada/salida del bloque DEPTH en Simulink

Tipo de Señal	Nombre de la Señal	Tipo de Dato de la Señal	Descripción de la Señal
Entrada	SYNC	Uint32	Entrega los datos de adquisición del Kinect en ms para sincronizar con los bloques de Depth/Motion/Image/Skeleton/IR. La señal SYNC tiene que alimentarse directamente con los bloques Depth/Motion/Image/Skeleton/IR.
Salida	Profundidad	Double	Entrega información sobre la profundidad en m de cada pixel.
Salida	XYZ	Double	Entrega las coordenadas XYZ del mundo real o de la proyección, esto puede ser configurado.
Salida	X/Y/Z	Double	Entrega las coordenadas XYZ separadas del mundo real o de la proyección, esto puede ser configurado.
Salida	Image	Uint8	Entrega una imagen de profundidad RGB

Parámetros del bloque DEPTH

El bloque DEPTH NID permite adquirir dos tipos de coordenadas en profundidad: las coordenadas del mundo real las cuales se miden en metros su profundidad y que tiene como centro de coordenadas el Kinect y las coordenadas de proyección que incurren en un procesamiento previo de conversión de unidades, el cual convierte los metros en pixeles y toma como centro de coordenadas el extremo superior izquierdo de la pantalla del PC.

- XYZ Coordenadas del mundo real: Coordenadas del mundo real (X, Y, Z en metros) según la Figura 4.5.

- XYZ Coordenadas proyección: Coordenadas de la proyección en profundidad (X, Y en pixeles y Z en metros) según la Figura 4.6.

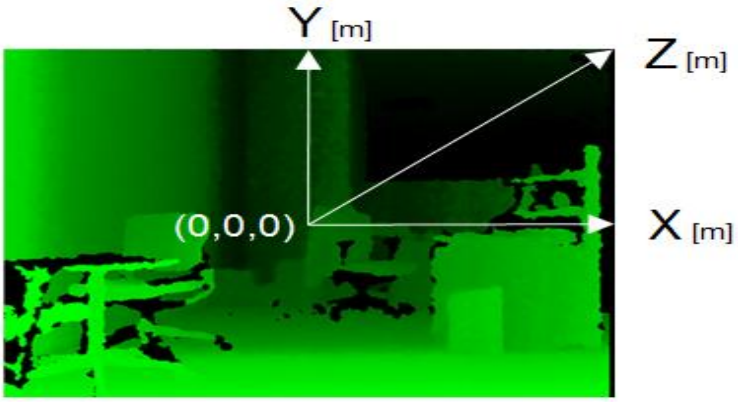

Figura 4.5. Coordenadas XYZ en mundo real
(Matlab-Simulink, 2013)

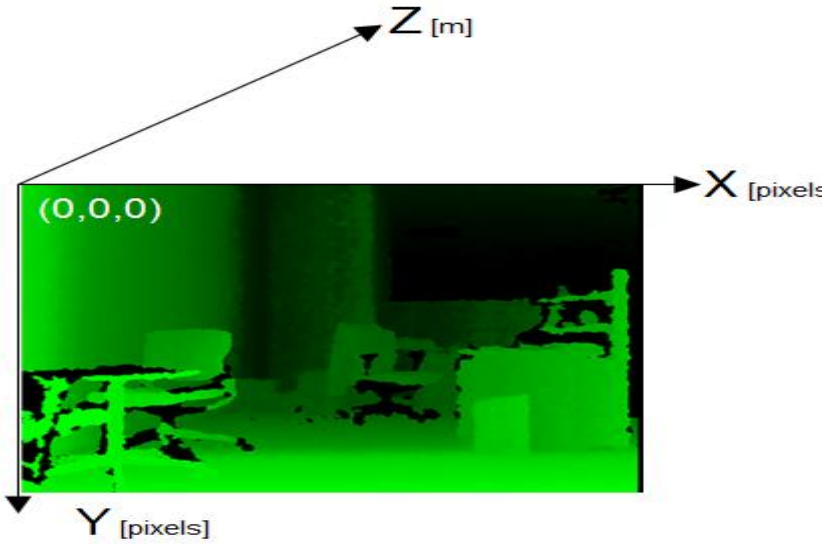

Figura 4.6. Coordenadas XYZ en proyección
(Matlab-Simulink, 2013)

4.1.4 BLOQUE IR NID

El bloque IR mostrado en la Figura 4.7 permite adquirir datos IR (Infrared) desde el Kinect como lo muestra la Tabla 4.4.

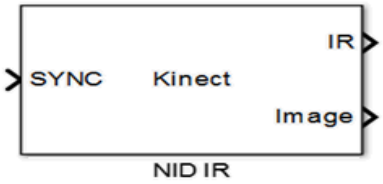

Figura 4.7. Bloque IR de la Librería NID en Simulink
(Matlab-Simulink, 2013)

Tabla 4.4. Descripción de las señales de entrada/salida del bloque IR en Simulink

Tipo de Señal	Nombre de la Señal	Tipo de Dato de la Señal	Descripción de la Señal
Entrada	SYNC	Unit32	Entrega los datos de adquisición del Kinect en ms para sincronizar con los bloques de Depth/Motion/Image/Skeleton/IR. La señal SYNC tiene que alimentarse directamente con los bloques Depth/Motion/Image/Skeleton/IR.
Salida	IR	Uint16	Entrega la información IR de cada pixel.
Salida	Image	Uint8	Entrega una imagen en escala de grises RGB

4.1.5 BLOQUE MOTION NID

El bloque MOTION mostrado en la Figura 4.8 detecta el movimiento de un objeto y lo identifica con un Id como lo muestra la Tabla 4.5.

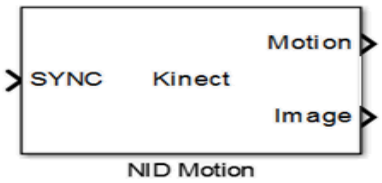

Figura 4.8. Bloque MOTION de la Librería NID en Simulink
(Matlab-Simulink, 2013)

Tabla 4.5. Descripción de las señales de entrada/salida del bloque MOTION en Simulink

Tipo de Señal	Nombre de la Señal	Tipo de Dato de la Señal	Descripción de la Señal
Entrada	SYNC	Unit32	Entrega los datos de adquisición del Kinect en ms para sincronizar con los bloques de Depth/Motion/Image/Skeleton/IR La señal SYNC tiene que alimentarse directamente con los bloques Depth/Motion/Image/Skeleton/IR.
Salida	Motion	Uint16	Detecta el movimiento de un objeto y lo identifica con un ID (0-15). ID =0 significa que no existe movimiento. Los ID objetos son: Red = 1, Green = 2, Blue = 3, Yellow = 4, Cyan = 5, Magenda = 6, Orange = 7, Dark 47reen = 8, Teal = 9, Purple = 10, Olive = 11, Maroon = 12, Silver = 13, Navy = 14, Gold = 15.

Tipo de Señal	Nombre de la Señal	Tipo de Dato de la Señal	Descripción de la Señal
Salida	Image	Uint8	Entrega una imagen de movimiento RGB

4.1.6 BLOQUE SKELETON NID

El bloque Skeleton mostrado en la Figura 4.9 detecta las 20 articulaciones del esqueleto humano como lo muestra la Tabla 4.6.

Figura 4.9. Bloque SKELETON de la Librería NID en Simulink
Fuente: (Matlab-Simulink, 2013)

Tabla 4.6. Descripción de las señales de entrada/salida del bloque SKELETON en Simulink

Tipo de Señal	Nombre de la Señal	Tipo de Dato de la Señal	Descripción de la Señal
Entrada	SYNC	Unit32	Entrega los datos de adquisición del Kinect en mseg para sincronizar con los bloques de Depth/Motion/Image/Skeleton/IR La señal SYNC tiene que alimentarse directamente con los bloques Depth/Motion/Image/Skeleton/IR.
Salida	Seguimiento	Int32	Entrega el número de personas a las que está realizando el seguimiento.
Salida	Esqueleto	Double	Entrega información sobre las coordenadas XYZ de las articulaciones del esqueleto en el mundo real o en proyección.

4.1.6.1 Parámetros del bloque SKELETON

El SDK para Windows permite acceder al Skeletal tracking entrega como información un seguimiento de las 20 articulaciones más importantes del cuerpo humano y los representa en un vector [x y z]. Si el vector es [0 0 0]

significa que el esqueleto no es detectado. La articulación tiene un marcador de vector específico representado por números en orden ascendente, los cuales se puede observar en la Figura 4.10 y los marcadores son:

Hip center = 1, Spine = 2, Shoulder center = 3, Head = 4, Shoulder left = 5, Elbow left = 6, Wrist left = 7, Hand left = 8, Shoulder right = 9, Elbow right = 10, Wrist right = 11, Hand right = 12, Hip left = 13, Knee left = 14, Ankle left = 15, Foot left = 16, Hip right = 17, Knee right = 18, Ankle right = 19, Foot right = 20.

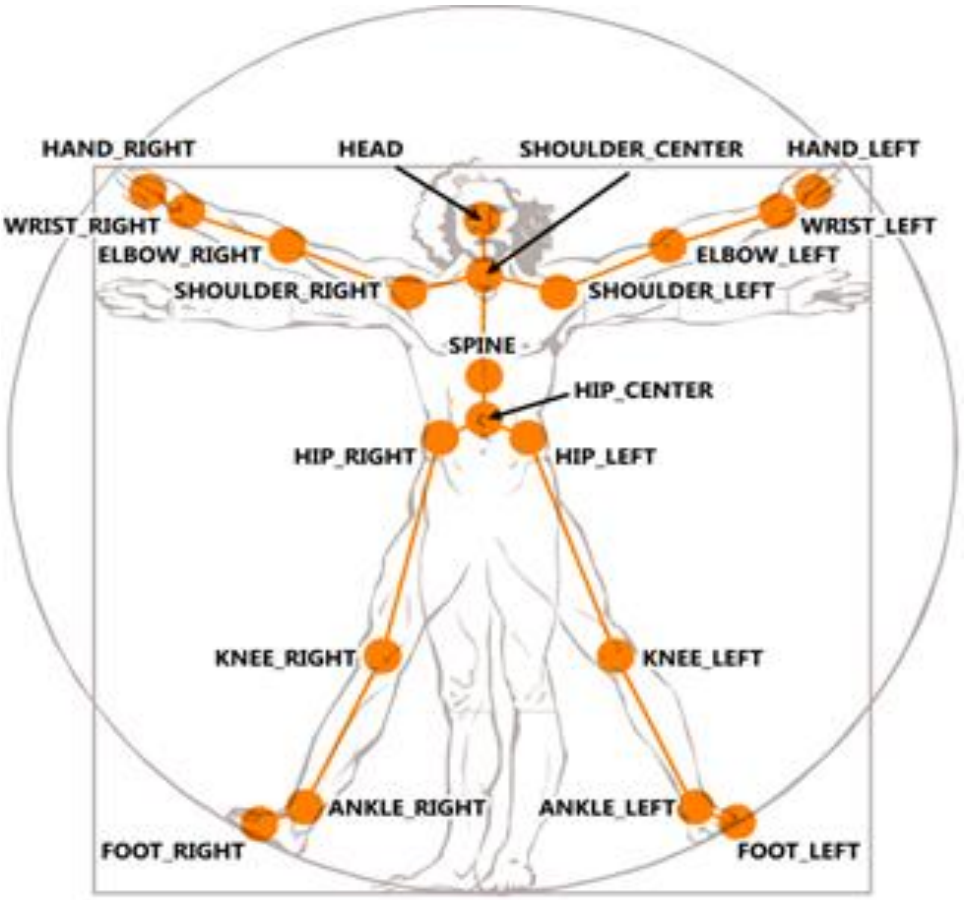

Figura 4.10. Articulaciones del cuerpo humano que reconoce el Kinect (Matlab-Simulink, 2013)

El número máximo de personas detectadas en este bloque son dos (2) personas con sus 20 articulaciones cada una.

4.2 USO DE LOS BLOQUES DE LA LIBRERÍA NID EN SIMULINK

Se refiere a la adecuada construcción del modelo de simulación tomando en cuenta la manera de conectar cada bloque para obtener las señales de salida correspondientes.

4.2.1 IMAGEN RGB

El bloque NID IMAQ sincroniza Simulink con el SDK de Kinect para Windows. El bloque NID IMAGE adquiere las imágenes RGB, porque contiene las funciones e instrucciones del SDK para interactuar con el NUI API de la cámara RGB del Kinect.

El enlace entre los bloques de NID IMAQ y NID IMAGE debe ser directo sin ningún tipo de unión con otro bloque de Simulink como lo muestra la Figura 4.11.

El bloque de Frame Rate Display permite visualizar el número de cuadros por segundo (FPS) de la simulación. Es muy importante que el tiempo de muestra del bloque NID IMAQ que es igual a 1/FPS, sea mayor al tiempo fijo de simulación.

El bloque Video Viewer permite visualizar imágenes binarias, RGB o alguna secuencia de video. El bloque posee controles de simulación como play, pause, stop y se los puede ejecutar mientras el video está activo.

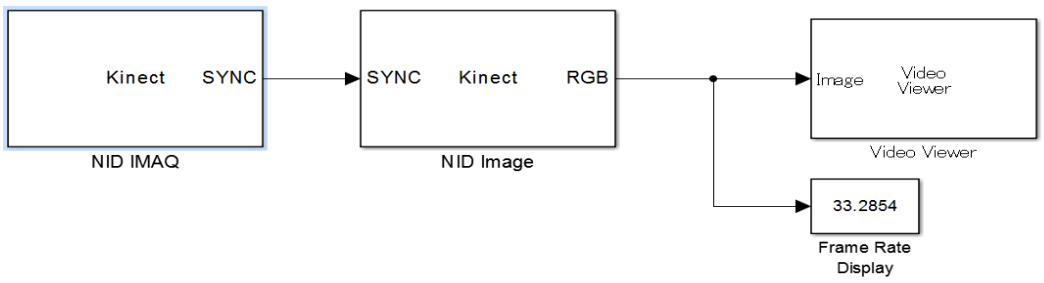

Figura 4.11. Modelo de simulación para capturar una imagen RGB en Simulink a través de la librería NID
(Matlab-Simulink, 2013)

4.2.2 IMAGEN DE PROFUNDIDAD

El bloque NID DEPTH adquiere las imágenes de profundidad, porque contiene las funciones e instrucciones del SDK para interactuar con el NUI API de los sensores IR del Kinect.

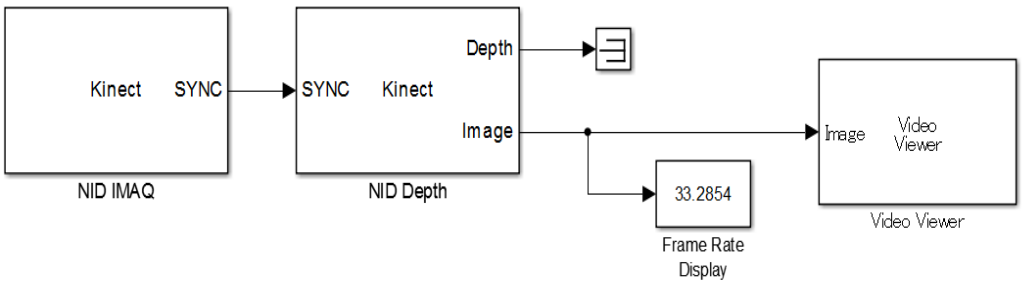

Figura 4.12. Modelo de simulación para capturar una imagen de profundidad en Simulink a través de la librería NID
(Matlab-Simulink, 2013)

4.2.3 IMAGEN EN MOVIMIENTO

El bloque NID IMAQ sincroniza Simulink con el SDK de Kinect para Windows. El bloque NID IMAGE adquiere las imágenes RGB, el bloque NID MOTION detecta el movimiento de las personas a través de un algoritmo que adquiere información de los objetos que cambian de profundidad durante el tiempo y a estos objetos les da una identificación o ID. EL ID es una numeración escalar que se encuentra entre el 0 y el 15: 0 significa que no existe movimiento y los números del 1 al 15 son los objetos detectados con movimiento.

La salida del bloque NID MOTION está conectada a dos bloques similares llamados Switch. El bloque Switch permite escoger una de las señales de entrada de acuerdo a una señal de control. Las señales de entrada se unen con los puertos que se encuentran en el extremo superior y en el extremo inferior del bloque, la señal de control se une con el puerto que se encuentra en la mitad del bloque. El puerto de control tiene un criterio de control que

permite el paso de cualquier de las dos señales de entrada, pero no ambas. El puerto de entrada superior siempre se encuentra conectado; si el criterio de control no es el correcto, el puerto de entrada superior se desconectará y el switch se conectará con el puerto inferior de entrada.

El criterio de control es el valor del lumbral > 0, esto quiere decir, existe movimiento en la imagen, porque si NID MOTION no detecta movimiento enviará un 0 al puerto de control, pero si detecta movimiento enviará un número entre el 1 y el15.

Este modelo separa el fondo (background), objetos sin movimiento del cuerpo humano, objeto con movimiento.

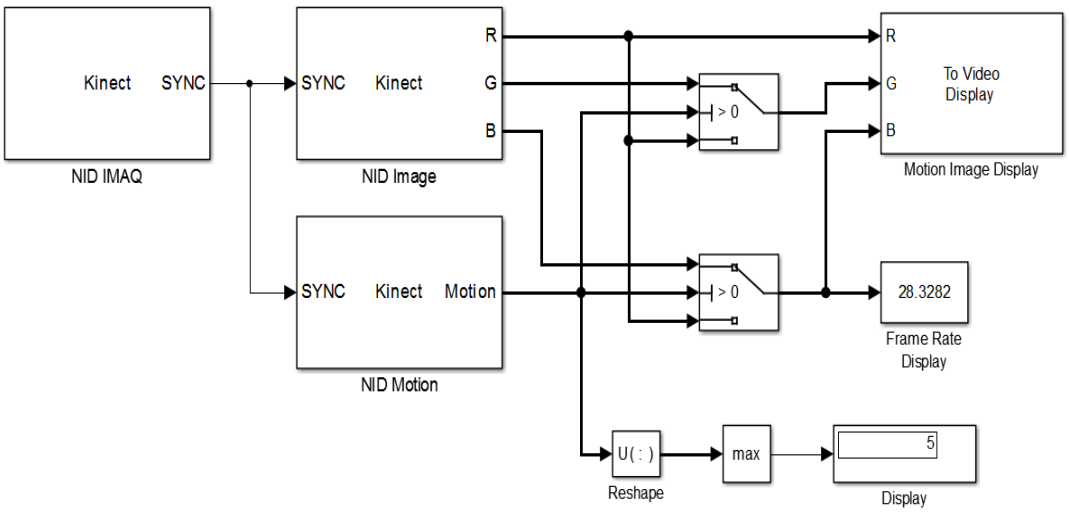

Figura 4.13. Modelo de simulación para capturar una imagen en movimiento en Simulink a través de la librería NID
(Matlab-Simulink, 2013)

4.2.4 COORDENADAS DEL ESQUELETO

El bloque NID IMAQ sincroniza Simulink con el SDK de Kinect para Windows. El bloque NID SKELETON adquiere la información de las 20 articulaciones más importantes del esqueleto en 3 dimensiones (x, y, z), esta información es guardada en un vector característico de dimensiones

[1x20x3]. Cada articulación tiene un marcador correspondiente. Cada marcador tiene la posición (x, y, z) en m de su articulación, esta posición parte de un sistema de coordenadas absolutas del Kinect que tiene como centro absoluto (0, 0, 0) el centro del mismo dispositivo.

El bloque Matlab Function adquiere el vector característico que es el dato de salida del puerto Skeleton, al vector se lo separa por marcadores, a estos marcadores se los divide en componentes de la posición (x, y, z) de la articulación y a estos componentes se les asignan salidas específicas en el bloque Matlab Function para luego poder visualizarlos en cada uno de los display correspondientes.

El enlace entre los bloques de NID IMAQ y NID SKELETON debe ser directo sin ningún tipo de unión con otro bloque de Simulink como lo muestra la Figura 4.14.

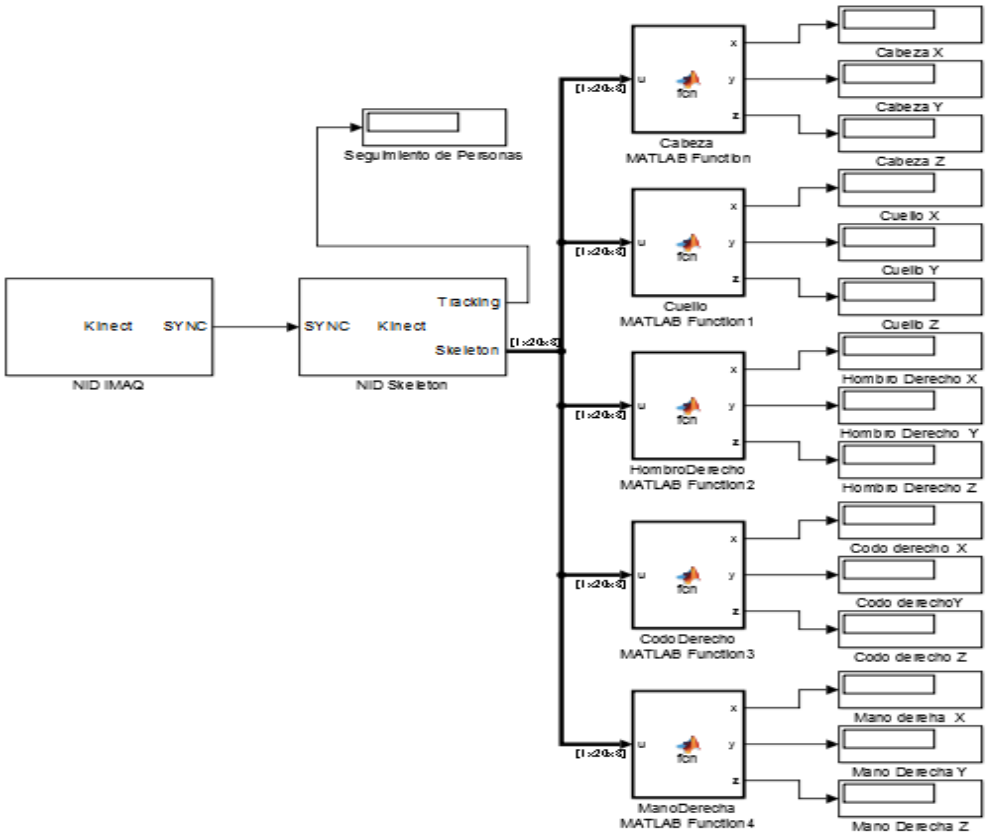

Figura 4.14. Modelo de simulación para obtener las coordenadas XYZ del esqueleto en Simulink a través de la librería NID
(Matlab-Simulink, 2013)

4.2.5 SEGUIMIENTO DEL ESQUELETO

El bloque Skeleton Viewer es una S – Function que permite componer un esqueleto uniendo todas las articulaciones mediante líneas.

El primer paso para construir el esqueleto es unir todas las articulaciones que representen la columna. La cabeza = 4, el centro de los hombros = 3, el centro de la columna vertebral= 2, el centro de la cadera = 1. Se los guarda en un vector id= [4 3 2 1] y luego se los une mediante una línea vertical.

Luego se localiza el hombro derecho e izquierdo, esta información se guarda en un vector id2= [5 9] respectivamente y se los une con una línea horizontal, esta línea debe cruzar por la primera línea vertical. Se realiza el mismo procedimiento con la cadera derecha e izquierda, esta información se guarda en el vector id3= [12 13].

Se une las articulaciones de la parte superior derecha, vector id4= [6 7 8], con el hombro derecho y las articulaciones de la parte superior izquierda, vector id5= [10 11 12], con el hombro izquierdo.

Por último se une las articulaciones de la parte inferior derecha, vector id6= [18 19 20], con la cadera derecha y las articulaciones de la parte inferior izquierda, vector id7= [14 15 16], con la cadera izquierda.

La Figura 4.15 muestra todo el proceso anteriormente descrito.

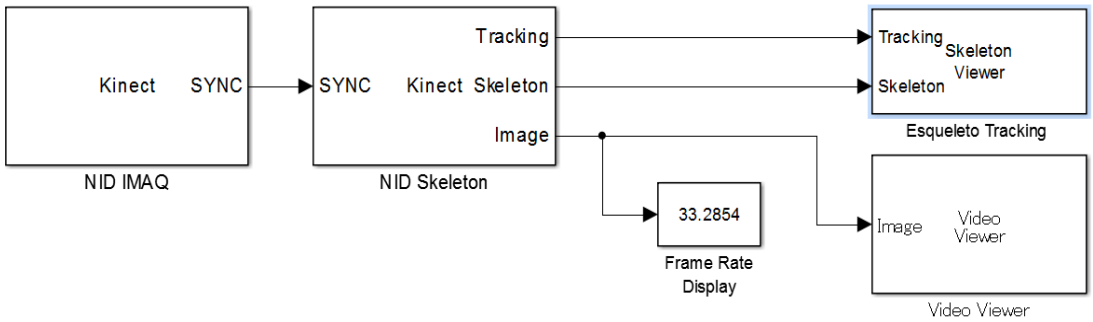

Figura 4.15. Modelo de simulación para el seguimiento del esqueleto en Simulink a través de la librería NID
(Matlab-Simulink, 2013)

4.2.6 MOVIMIENTO DEL PIVOTE MOTORIZADO

En los parámetros SDK del bloque NUI IMAQ se encuentra la opción Entrada del ángulo por sensor (Set the sensor by Input).

Al momento de activar la opción en el bloque NUI IMAQ se crea un nuevo puerto de entrada con el nombre ángulo (angle), esta entrada permite ingresar el valor de elevación de ángulo en grados del pivote motorizado.

El ángulo de elevación del pivote está entre 27º y -27º, el tipo de formato para el ingreso del ángulo es de int32.

Los bloques de Constant y Slider Gain se utilizan para variar la ganancia del ángulo. En los parámetros del bloque Slider Gain se ingresan los parámetros baja (low) y alto (high) que especifican los límites de ganancia del ángulo. Para el Kinect los parámetros en este bloque lo muestra la Figura 4.16.

Los demás bloques NID IMAGE, Video Viewer y Frame Rate Display se los utiliza en el Modelo de simulación para capturar una imagen RGB en Simulink a través de la librería NID.

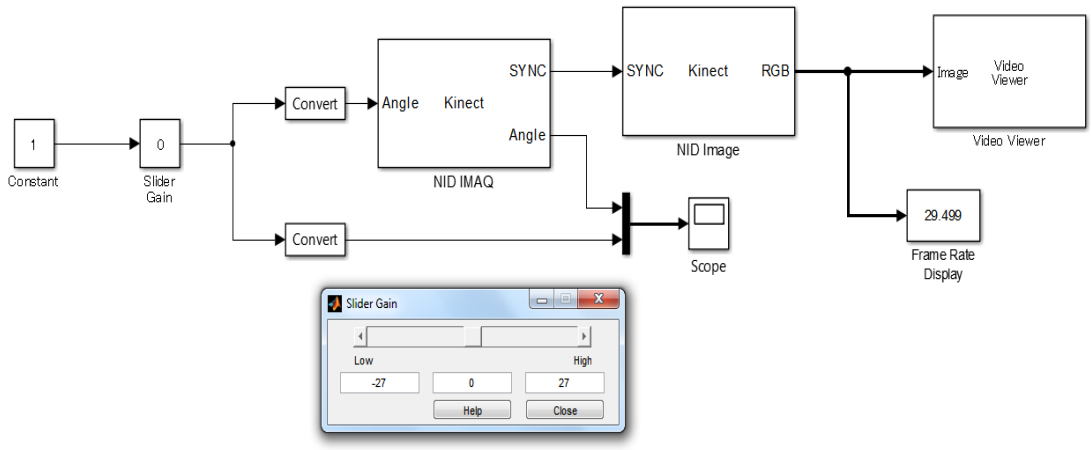

Figura 4.16. Modelo de simulación para el movimiento del pivote motorizado en Simulink a través de la librería NID
(Matlab-Simulink, 2013)

4.3 RANGOS DE VISIÓN DE LA APLICACIÓN

El Kinect es un dispositivo de visión artificial que gracias a su cámara RGB y sus sensores infrarrojos de profundidad permiten adquirir una imagen en 3D del esqueleto humano. Los ejes de las tres dimensiones son (X, Y, Z) siendo (X) el eje horizontal, (Y) el eje vertical y (Z) la profundidad.

El Kinect adquiere imágenes dependiendo de dos rangos de visión que se encuentran en el sistema de coordenadas (Z, Y) y (X, Z) el ángulo de visión en cada uno de los sistemas de coordenadas es el limitante para identificar el esqueleto. El ángulo de visión vertical es de 43.5º partiendo del centro del Kinect y con un ángulo inicial del pivote motorizado de 0º como lo muestra la Figura 4.17 y el ángulo de visión horizontal es de 57.5º como lo muestra la Figura 4.18.

En el capítulo anterior los requisitos de la aplicación mencionan que el Kinect debe detectar una persona entre (1.5 – 2) m de estatura. Partiendo de esta premisa se resolvió los rangos de visión de la siguiente manera.

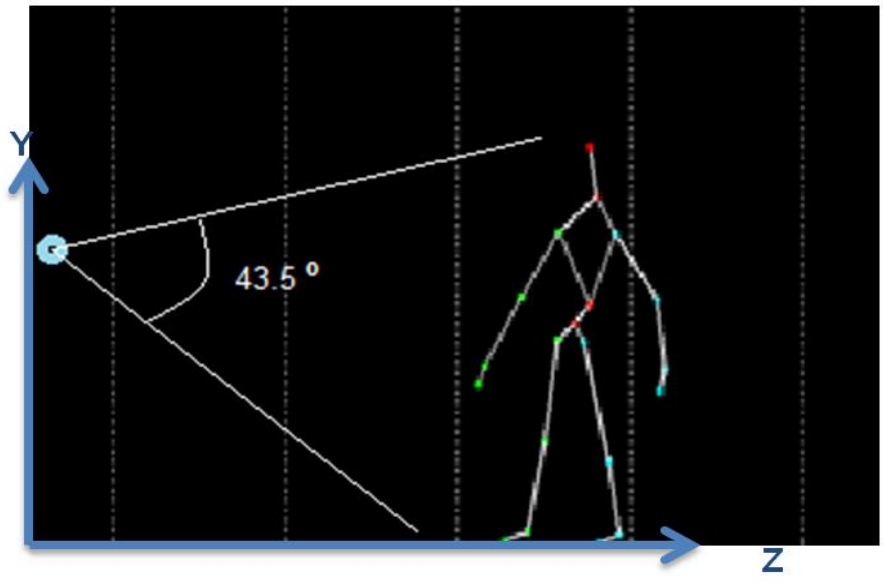

Figura 4.17. Angulo de visión ejes (Z, Y)

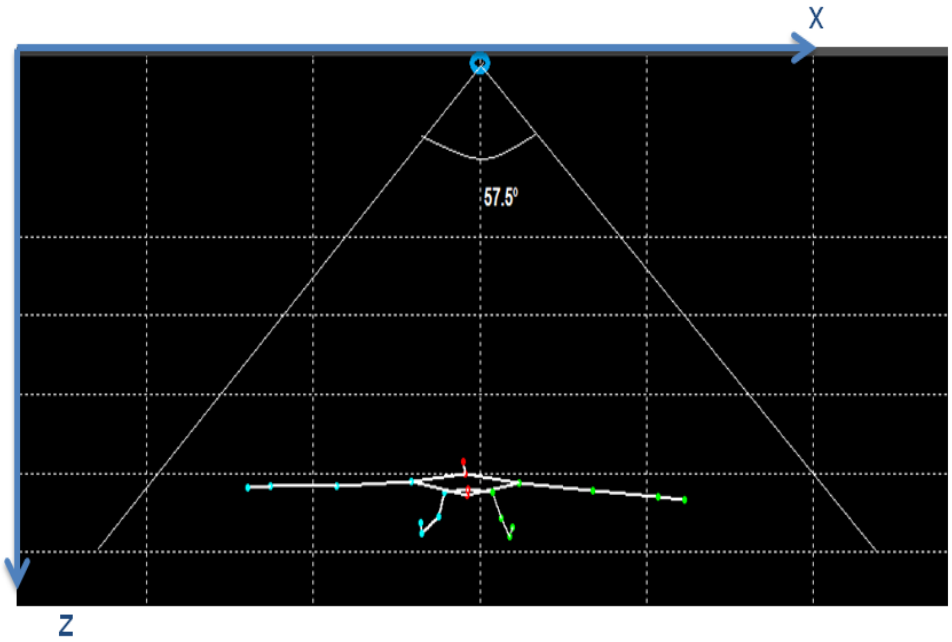

Figura 4.18. Angulo de visión ejes (X, Z)

El Kinect tiene dos modos de visión los cuales son: el modo cercano y el modo lejano. El modo cercano permite adquirir una imagen desde 0.8 m hasta los 2.5 m y el modo lejano permite adquirir una imagen desde 1.2 m hasta los 3.5m frente al Kinect. Estos límites físicos de visión del Kinect ya limitan el área activa donde el Kinect pueda sensar.

El usuario siempre debe estar de frente al Kinect y ambos modos de visión son válidos; pero esta aplicación demandada por su función principal que es controlar la secuencia de las presentaciones, estableció que el usuario no estaría tan cerca del Kinect como para usar el modo cercano; porque necesitaría espacio para explicar sus presentaciones por ese motivo se escogió el modo lejano.

Posteriormente se realizó una investigación de cuál es la distancia óptima entre el proyector y el lugar de exposición. Esta investigación se la realizó en las aulas del bloque B y en los Laboratorios de Mecatrónica del bloque G de la Universidad Tecnológica Equinoccial obteniendo los resultados de la Tabla 4.7.

Tabla 4.7. Distancias entre el proyector y el lugar de exposición de aulas de la Universidad Tecnológica Equinoccial

Bloque	**Aula**	**Distancia entre el proyector y el lugar de exposición (m)**
B	B11	2.75
B	B13	2.75
B	B12	2.75
B	B21	2.75
B	B22	2.75
B	B23	2.75
B	B31	2.85
B	B32	2.85
B	B33	2.85
G	Lab. Mecanismos	2.83
G	Lab. Software1	2.83
G	Lab. Hardware 1	2.83
G	Lab. Software 2	2.83
G	Lab. Software3	2.83
Distancia media entre el proyector y el lugar de exposición		**2.8**

Se determinó la distancia media en la Tabla 4.7. Como los datos de la tabla no poseen una variación extremadamente grande se considera que este dato es el valor óptimo de distancia entre el proyector y el lugar de exposición. Pero se debe tomar en cuenta que el Kinect tiene algoritmos predictivos y correctivos para el seguimiento del esqueleto humano; estos algoritmos dependen de la distancia de profundidad del Kinect entre más lejos esté una persona el funcionamiento del algoritmo correctivo no es tan óptimo.

La distancia escogida entre el proyector y el lugar de exposición fue de 2.8m.

Una vez escogida la distancia de profundidad óptima para el proceso de visión se elige la altura adecuada de ubicación del Kinect.

De acuerdo a los ángulos de visión del Kinect se determina que su área activa de visión y sensado tiene una forma triangular para ser más exactos tiene una forma de triángulo isósceles; ya que la visión y el sensado comienza en un punto específico llamado centro focal y a medida que el plano de la imagen se aleja del centro focal, el área activa se expande de forma proporcional a los dos lados que forman el ángulo de visión.

De acuerdo a los corolarios de geometría plana, la bisectriz del ángulo desigual de un triángulo isósceles es mediana, altura y mediatriz; debido a esto el Kinect debe colocarse a 1m de altura, ya que uno de los requisitos de la aplicación es poder detectar a una persona de máximo 2m de altura.

Por último se procede a realizar los cálculos para determinar la altura máxima de la aplicación, la distancia máxima de movimiento horizontal de la aplicación y el área de trabajo de la aplicación.

- La altura máxima y mínima de la aplicación

  Determina el rango de altura de una persona para ser detectada por el Kinect.

  El ángulo de visión vertical del Kinect es de 43.5º. Partiendo del vértice de este ángulo se puede formar los dos lados del triángulo isósceles; la bisectriz de este ángulo es la mediana y la altura del triángulo isósceles, esta medida de la mediana es la distancia escogida entre el proyector y el lugar de exposición que es de 2.8m.

  Se forma el triángulo que se observa en la Figura 4.19. Las medidas de la Figura 4.19 se encuentran en milímetros.

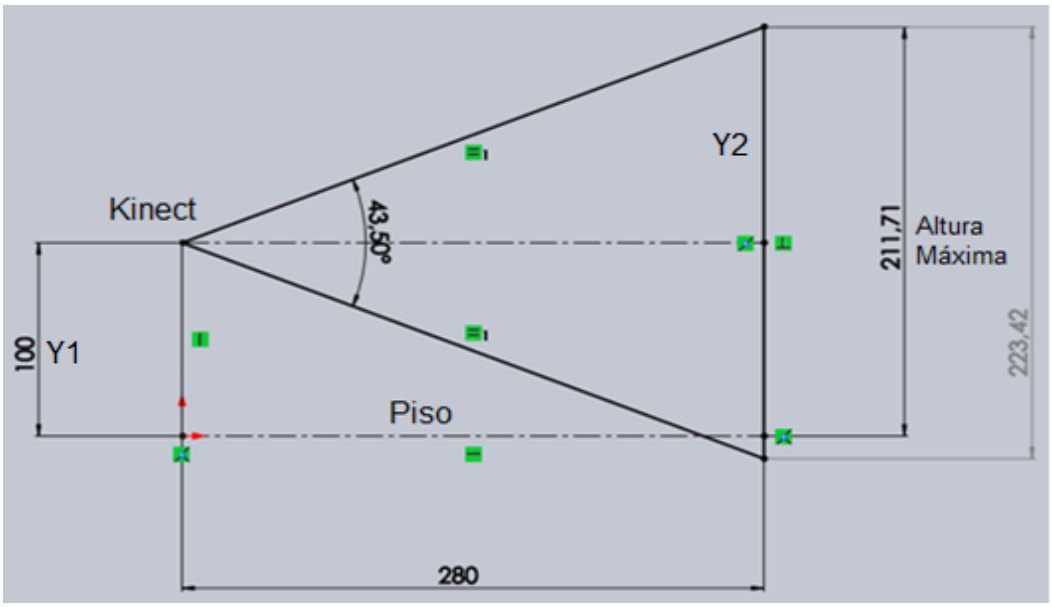

Figura 4.19. Altura máxima de la aplicación

Este triángulo isósceles se lo divide en dos triángulos rectángulos. Se trabajó con el segundo triangulo rectángulo donde se encuentra Y2; de manera sencilla se puede obtener el valor de Y2 a través de la tangente del ángulo conocido como lo muestra la ecuación 4.1.

$$\tan \emptyset = \frac{Y2}{Z} \quad [4.1]$$

Dónde:

∅: Angulo de visión vertical del Kinect divido para dos

Y2: Cateto opuesto al ángulo de visión vertical del Kinect

Z: La distancia escogida entre el proyector y el lugar de exposición

Se observa que:

$\tan \emptyset * Z = Y2$

$Y2 = 1.117$

$Altura\ Máxima = Y1 + Y2$

$Altura\ Máxima = 2.117\ m$

Debido a que la aplicación detecta sólo los movimientos de la parte superior del cuerpo del usuario y el dispositivo Kinect está colocado a una altura de 1m sobre el nivel del suelo como lo muestra la Figura 4.19; el usuario con la altura mínima deberá sobrepasar con todo su torso superior la altura del Kinect.

Según la cuadricula de proporcionalidades del cuerpo (Rafael, 2006) la altura total del cuerpo se lo puede dividir en 8 porciones iguales. De la 1º hasta la 6º porción abarca una altura que va desde los pies hasta la mitad del pecho, las otras dos porciones corresponde a la parte alta del pecho y la cabeza.

Por consiguiente: Si las 6 porciones del cuerpo llegaran hasta 1m y faltaran las 2 partes superiores del cuerpo (parte alta del pecho y

cabeza), cada porción mediría 16,67 cm por lo tanto la altura total de la persona con sus 8 porciones sería de 1.333 m; esta sería la altura mínima estimada para que la aplicación pueda detectar a la persona y sensar sus movimientos de acuerdo con el cálculo de proporcionalidades del cuerpo (Rafael, 2006).

Se determinó que el Kinect en una ubicación de 1m sobre el piso puede sensar a una persona de hasta 2.117m de altura como máximo y como mínimo una persona de hasta 1.333m si la persona está a 2.8m alejada del Kinect.

- La distancia máxima de movimiento horizontal de la aplicación

Determina la distancia máxima que una persona puede desplazarse horizontalmente hacia la derecha o hacia la izquierda mientras es detectada. Se toma siempre como origen el Kinect.

El ángulo de visión horizontal del Kinect es de 57.5º. Partiendo del vértice de este ángulo se puede formar los dos lados del triángulo isósceles; la bisectriz de este ángulo es la mediana y la altura del triángulo isósceles, esta medida de la mediana es la distancia escogida entre el proyector y el lugar de exposición que es de 2.8m.

Se forma el triángulo que se observa en la Figura 4.20. Las medidas de la Figura 4.20 se encuentran en milímetros.

El movimiento horizontal siempre será tomado desde el Kinect hacia la izquierda o hacia la derecha. Siendo el Kinect el punto de inicio 0 m.

Este triángulo isósceles se lo divide en dos triángulos rectángulos. Se trabajó con el primer triangulo porque ambos triángulos son congruentes y la medida de X1 será igual a la medida de X2. Se determinó el valor X1 a través de la tangente del ángulo conocido como lo muestra la ecuación 4.2.

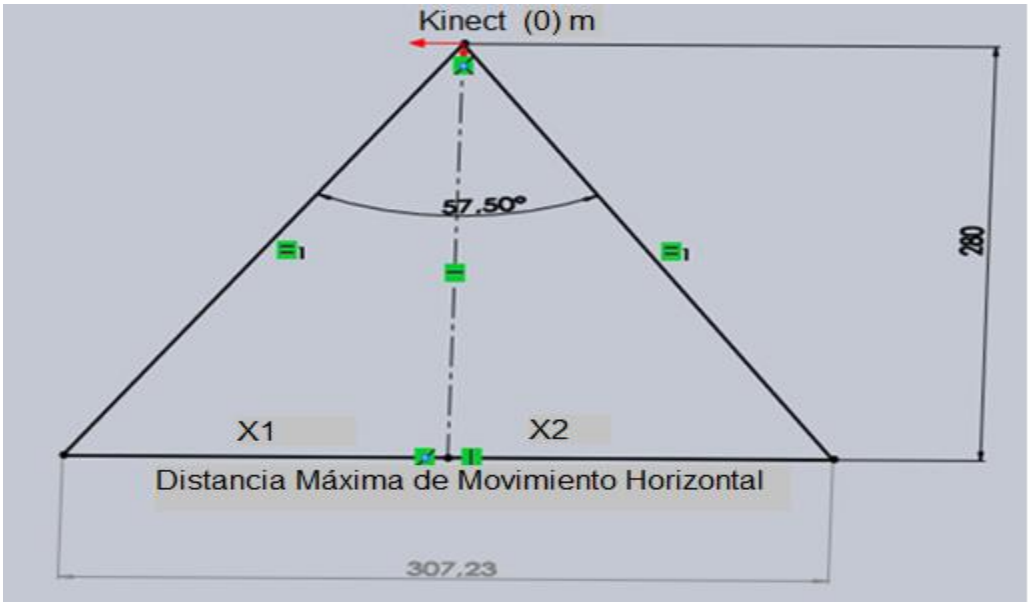

Figura 4.20. Distancia máxima de movimiento horizontal de la aplicación

$$\tan \alpha = \frac{X1}{Z} \quad [4.2]$$

Dónde:

α: Angulo de visión horizontal del Kinect divido para dos

X1: Cateto opuesto al ángulo de visión horizontal del Kinect

Z: La distancia escogida entre el proyector y el lugar de exposición

Se observa que:

$\tan \alpha * Z = X1$

$X1 = 1.536 = X2$

$Distancia\ Maxima\ Horizontal = X1 + X2$

$Distancia\ Maxima\ Horizontal = 3.072\ m$

Se determinó que el usuario puede desplazarse desde el Kinect horizontalmente hacia la derecha o hacia la izquierda 1.536 m. Teniendo una distancia total de desplazamiento de 3.072 m

- Área de trabajo de la aplicación

Determina el área donde una persona puede moverse y es detectada por el Kinect como lo muestra la Figura 4.21.

Las medidas de la Figura 4.21 se encuentran en milímetros.

El área de trabajo está dada por la ecuación 4.3.

$$área\ de\ trabajo = \frac{(b+Distancia\ Máxima\ Horizontal)*(Z-1.2)}{2} \quad [4.3]$$

Dónde:

b: Es la distancia máxima de movimiento horizontal de la aplicación teniendo como Z=1.2

$$área\ de\ trabajo = \frac{(((\tan\alpha * Z) * 2) + 3.072) * (2.8 - 1.2)}{2}$$

$$área\ de\ trabajo = 3.511\ m^2$$

Se determinó que el Kinect limita el movimiento del usuario debido a sus dos (2) ángulos de visión y al modo de visión escogido. El lugar en donde puede ser detectado el usuario es a 1.2m alejado del Kinect, este lugar tiene una forma trapezoidal con un área de 3.511 metros cuadrados como muestra la Figura 4.21.

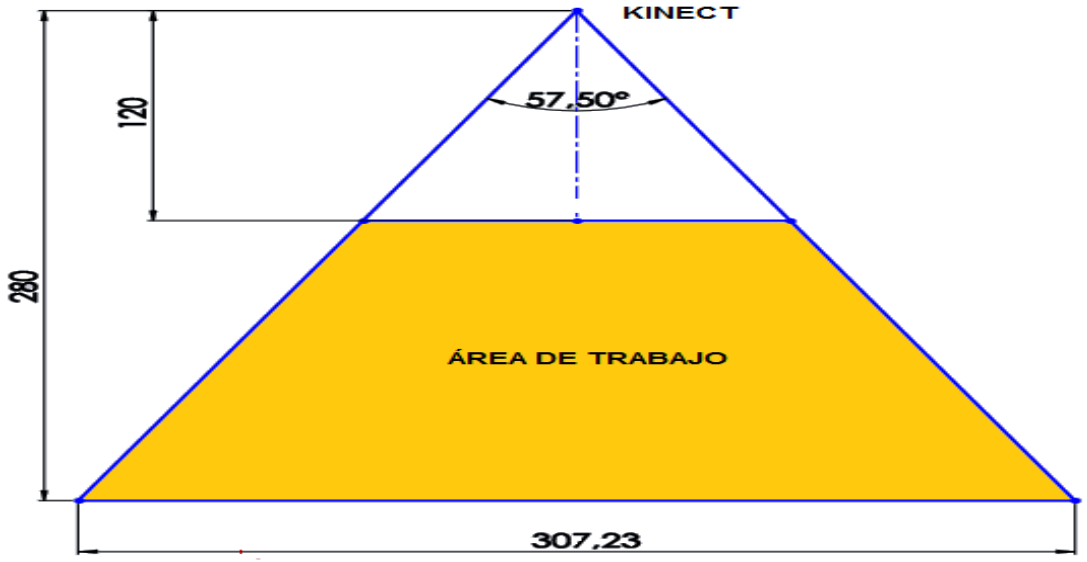

Figura 4.21. Área de trabajo de la aplicación

4.4 ETAPAS DEL SISTEMA DE VISIÓN ARTIFICIAL A TRAVÉS DE LA LIBRERÍA NID DEL KINECT EN SIMULINK

La Visión Artificial es una descripción automática de las propiedades de un mundo tridimensional a partir de una o varias imágenes bidimensionales del mundo.

La entrada de un sistema de visión artificial es una imagen obtenida por un elemento de adquisición, mientras que su salida es una descripción de la escena; esta descripción debe contener toda la información requerida para la tarea de interacción con el medio ambiente. (Sobrado, 2003)

Las etapas de un sistema de visión artificial son:

1. Adquisición
2. Segmentación
3. Descripción
4. Reconocimiento

4.4.1 ADQUISICIÓN

Es la etapa de la visión artificial que permite capturar imágenes. La aplicación necesita detectar los movimientos que hace una persona en video; esto se lo realiza a través de un seguimiento al esqueleto humano.

Para realizar un seguimiento al esqueleto humano con la librería NID en Simulink, se necesita de los bloque IMAQ Y SKELETON como lo muestra la Figura 4.22.

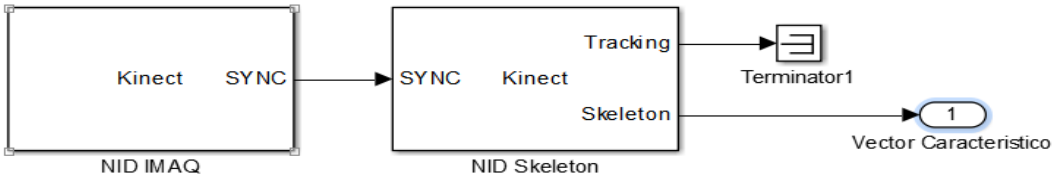

Figura 4.22. Modelo para la adquisición del esqueleto con la librería NID en Simulink

El bloque IMAQ de la librería NID en Simulink adquiere todos los datos que entrega el Kinect, se debe adquirir los datos en un tiempo de muestreo de 1/10 s como lo muestra la Figura 4.23. Este tiempo es adecuado, ya que el número de fotogramas que se van a adquirir permite captar el movimiento humano y además de esta manera no se llena la memoria virtual del PC con fotogramas innecesarios. Se escogió el modo sentado para el sensado del esqueleto como lo muestra la Figura 4.23; se determinó este modo en los requerimientos del control de eventos, ya que sólo se necesitaría obtener el vector característico con la información de la parte superior del cuerpo.

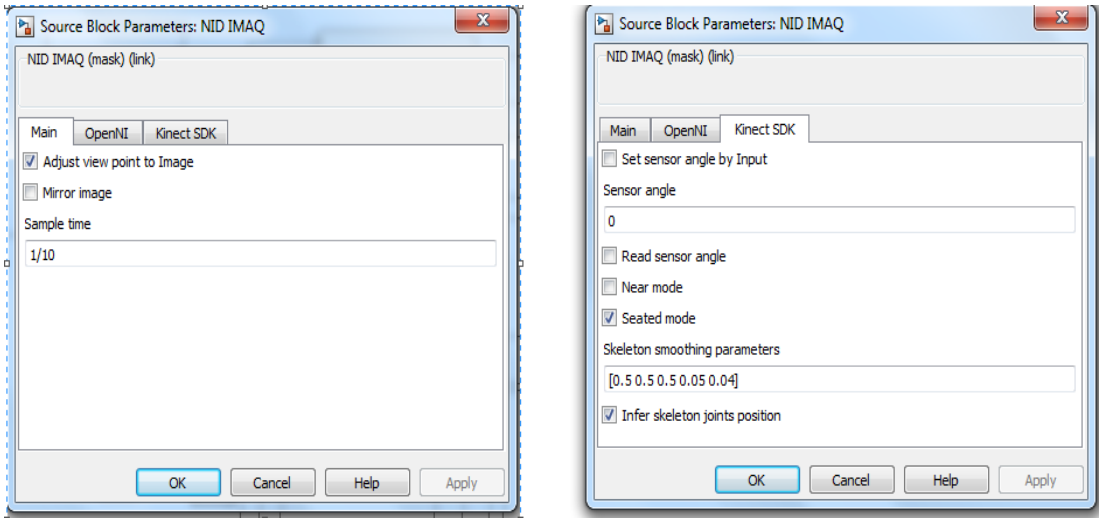

Figura 4.23. Configuración bloque IMAQ de la librería NID en Simulink

El Solver de la simulación fue el método discreto, ya que el sistema de visión artificial necesita de valores asociados en cada paso del tiempo en la simulación.

El ajuste del paso de integración del tiempo fue fijo, este paso fijo de toma de tiempo es de 1/20 s; esto significa que la simulación es más rápida que la adquisición de fotogramas como lo muestra la Figura 4.24.

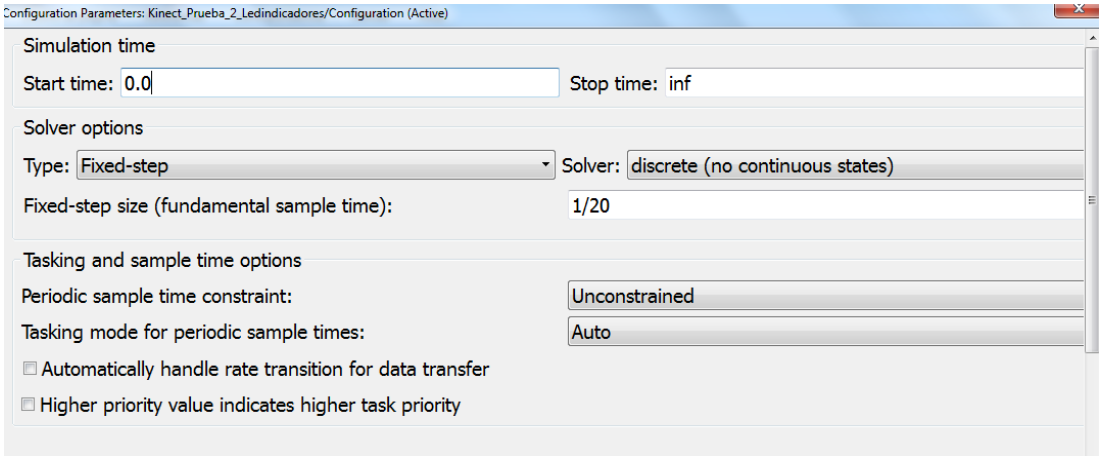

Figura 4.24. Configuración de los parámetros de simulación en Simulink

El bloque SKELETON de la librería NID en Simulink adquiere las 10 articulaciones más importantes de la parte superior del esqueleto humano, debido a que se escogió en los parámetros del Kinect SDK el modo sentado en el bloque IMAQ.

Para poder detectar los movimientos del cuerpo humano con este bloque se necesita escoger en sus parámetros la opción de coordenadas de mundo real como lo muestra la Figura 4.25; estas coordenadas son las que nos permiten realizar algoritmos comparativos de las proporcionalidades del cuerpo para reconocer un movimiento.

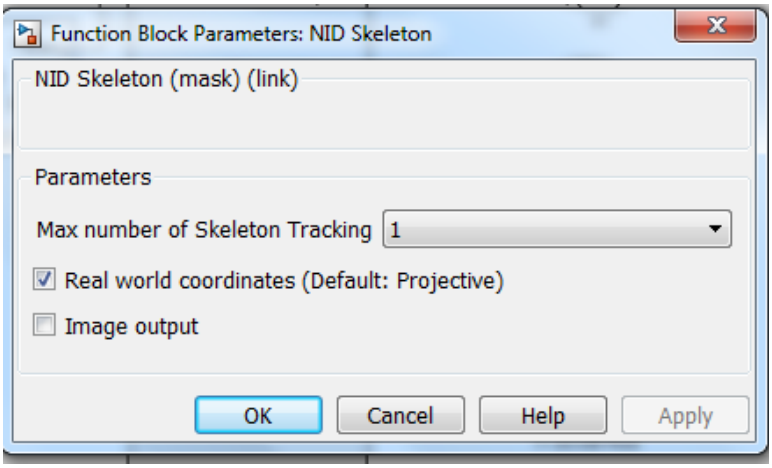

Figura 4.25. Configuración de los parámetros del bloque SKELETON de la librería NID en Simulink

El Kinect entrega las coordenadas de cada articulación de la parte superior del cuerpo en un vector característico de tres dimensiones (x, y, z). Es importante saber que estas coordenadas tienen como origen el Kinect y de acuerdo a donde se encuentren localizadas estas coordenadas respecto al Kinect se obtendrá una medida en metros de la distancia entre el Kinect y cada uno de los componentes del vector y un signo que representa la ubicación a la izquierda, a la derecha, arriba o abajo del Kinect. Esto quiere decir que el Kinect posee un sistema de coordenadas absoluto, si la articulación se encuentra por encima o por debajo del Kinect el valor de la coordenada será positivo o negativo correspondientemente y si la articulación se encuentra a la derecha o a la izquierda del Kinect de igual manera la coordenada será positiva o negativa correspondientemente.

4.4.2 SEGMENTACIÓN

La segmentación es el proceso mediante el cual una imagen se descompone en regiones o elementos que pueden corresponder a algún objeto de interés.

El vector característico está compuesto por marcadores, cada marcador contiene las coordenadas (x, y, z) de la articulación. Para detectar el movimiento se debe segmentar el vector característico y obtener las coordenadas de las articulaciones que determinan el movimiento. Las articulaciones que se necesitan son:

- Mano derecha
- Mano izquierda
- Centro del cuerpo

Existen 3 articulaciones que pueden determinar el centro del cuerpo. Estas son Hip center (Centro de caderas), Spine (Centro de Columna Vertebral), Shoulder center (Centro de los hombros). Se eligió el centro de los hombros como centro del cuerpo en esta aplicación, ya que esto facilita el reconocimiento del movimiento del brazo en la siguiente etapa por tener la misma coordenada en eje (Y) cuando los brazos se encuentran extendidos horizontalmente.

Este proceso de segmentación se realizó en Simulink con una función de Matlab que permitió escoger el marcador de cada una de las articulaciones antes mencionadas y extraer sus coordenadas como lo muestra la Figura 4.26.

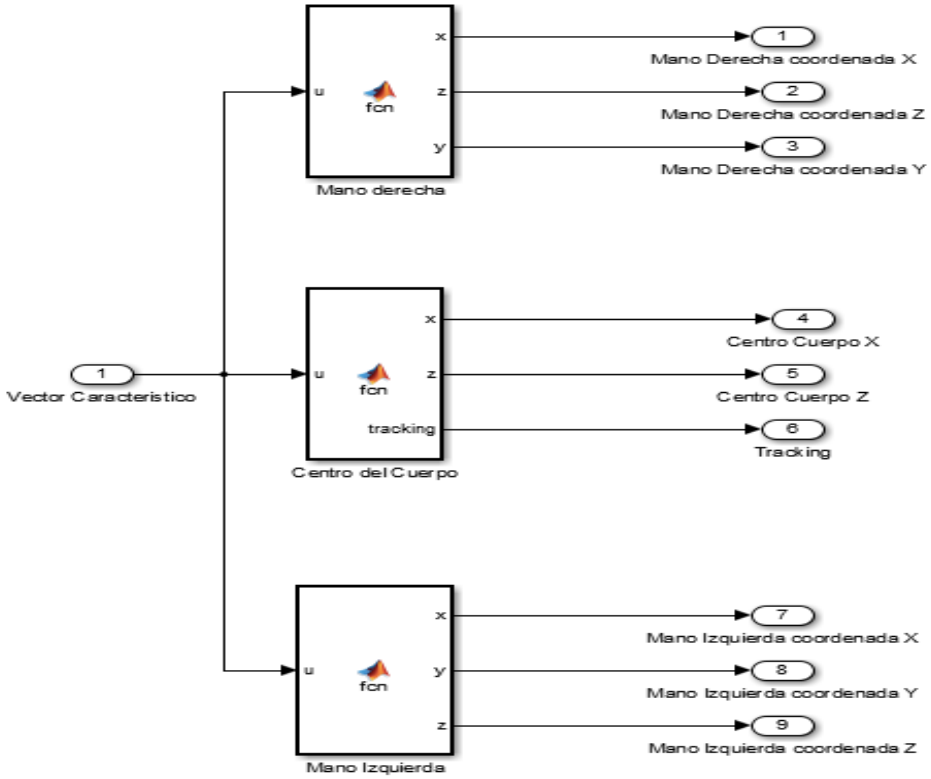

Figura 4.26. Modelo de segmentación del vector característico a coordenadas de las articulaciones mano Izquierda, derecha y centro del cuerpo

- Mano derecha

  El vector característico que proviene del bloque Skeleton de la librería NID es la entrada de la Matlab Function. Este vector tiene una dimensión de [1x10x3] lo que quiere decir que existen 10 marcadores con tres coordenadas cada uno, normalmente son 20 marcadores pero como se escogió en las características del bloque Skeleton el modo sentado los marcadores se reducen a solo los que se encuentran en la parte superior del esqueleto que son 10. Para un mejor y más rápido procesamiento se debe comprobar que solo

existan 10 marcadores, si hubiera más se tendría que cambiar el modo de sensado en las opciones del bloque Skeleton.

El primer paso para la segmentación de la mano derecha es identificar el marcador, el cual según la Figura 4.10 es el marcador número 12, luego se escoge este marcador creando un vector más pequeño de dimensión [1X1X3] este vector contiene solo la información de las coordenadas de la mano derecha. Este vector tiene distribuidas las coordenadas de la siguiente manera (x, y, z) por lo que se sabe la posición exacta que ocupa cada coordenada dentro del vector y con esto se podrá escoger la coordenada que se requiera. Por último la coordenada escogida es asignada a un puerto de salida de la Matlab Function.
Para poder realizar los algoritmos comparativos de proporcionalidades del cuerpo, se tomaron las tres coordenadas de la mano derecha (x, y, z).

Este proceso es exactamente el mismo para la mano izquierda y para el centro del cuerpo con cambios en los marcadores y en las coordenadas requeridas. El marcador de la mano izquierda es el número 8. El marcador del centro del cuerpo es el número 3, adicionalmente con este marcador se requiere realizar el seguimiento a la persona, de esta manera se identifica si la persona se encuentra en el área de trabajo; esto se realiza a través de una comparación de las coordenadas (X) de la mano izquierda y derecha con respecto a la coordenada (X) del centro del cuerpo, si alguna de ellas es 0 la persona no está identificada pero mientras sea cualquier valor diferente de 0 es identificada.

La figura 4.27 contiene el diagrama de flujo de cómo obtener las coordenadas de la mano derecha, mano izquierda y el centro del cuerpo.

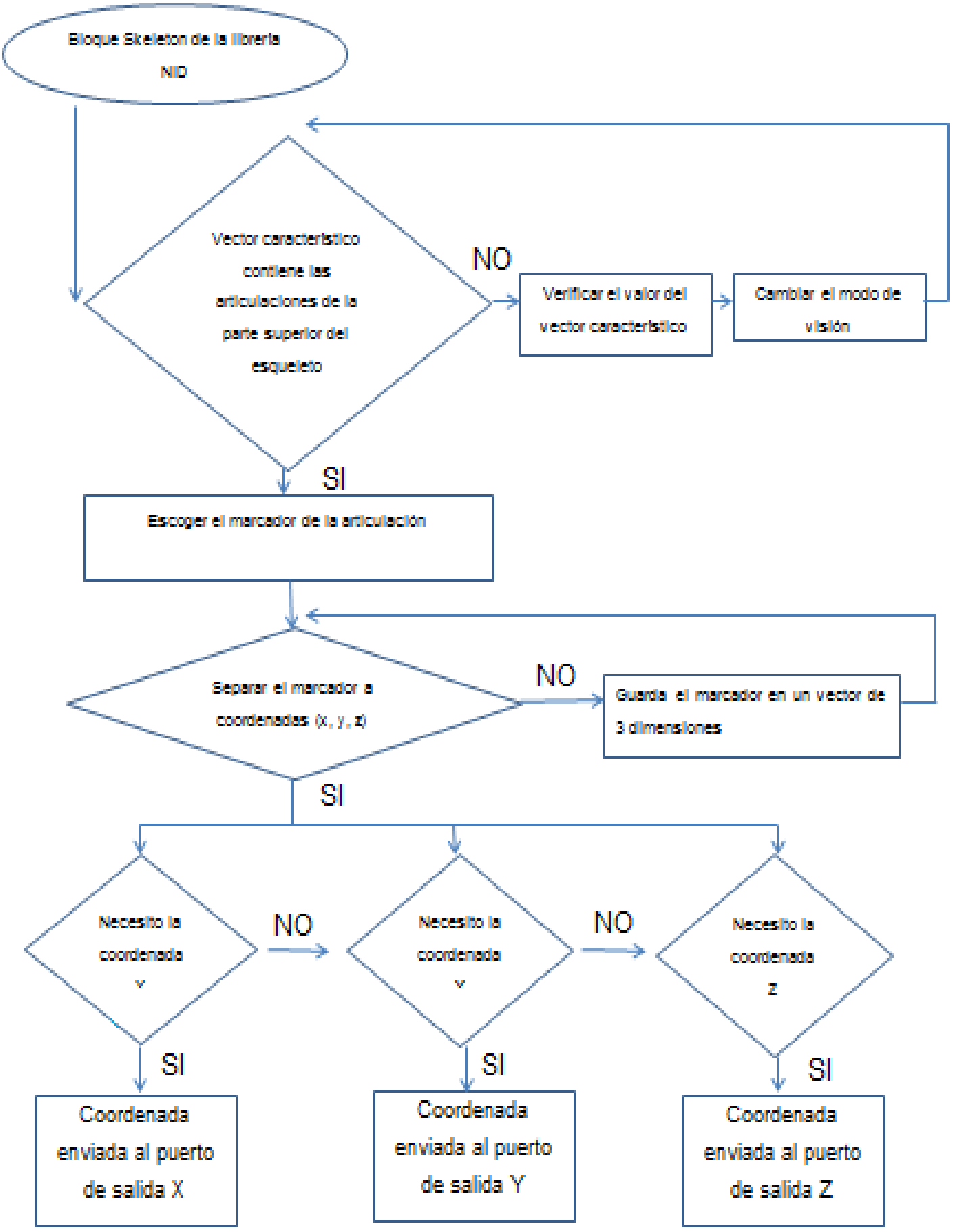

Figura 4.27. Diagrama de Flujo de la segmentación del vector característico a coordenadas de la articulación

Todas las coordenadas obtenidas de los bloques Matlab Function se encuentran en unidades del sistema internacional su distancia está en metros. Todas las medidas de distancia de profundidad y de movimiento vertical u horizontal son con respecto al Kinect que es su centro.

4.4.3 DESCRIPCIÓN

La descripción representa la obtención de características relevantes de la imagen, estas características deben ser independientes de la localización u orientación del objeto a describir; deben ser tan eficientes que puedan discriminar objetos parecidos entre sí. Los modelos que describen una imagen pueden ser estadísticos, estructurales, dimensionales, perimetrales, etc. Todos ellos se obtienen a partir de las imágenes previamente segmentadas.

Para el correcto funcionamiento de la aplicación esta debe tener la capacidad de reconocer un movimiento realizado con las manos para interpretarlo y realizar una acción determinada en el PC. Por esto es necesario saber cuáles van a ser los movimientos que deberán ser reconocidos por la aplicación. Cuando los movimientos que se desea reconocer sean elegidos, se encontrará un modelo dimensional que pueda describir el movimiento.

Los movimientos escogidos para ser descritos fueron:

- Brazo derecho arriba extendido horizontalmente
- Brazo izquierdo arriba extendido horizontalmente
- Mano derecha adelante
- Mano izquierda adelante
- Apertura y cierre de las manos adelante

Los modelos para determinar la descripción de estos movimientos fueron a través de descriptores unidimensionales y descriptores especiales para medir el flujo óptico como lo muestra la Figura 4.29.

El descriptor unidimensional permite encontrar las distancias específicas que existen entre las partes del cuerpo. La distancia es una medida de valor absoluto; ya que para un mismo usuario la medida del brazo con respecto a un origen siempre será la misma en todo lugar, este valor no se modifica aún si las coordenadas adquieren valores negativos.

El descriptor especial para medir flujo óptico mide los cambios en una imagen con respecto a un intervalo de tiempo; de esta forma se determina la dirección y velocidad del movimiento en todos los puntos de la imagen.

Se tiene que tomar en cuenta que el ser humano es un ser motor en constante movimiento; por este motivo se debe comprender bien las señales tomadas.

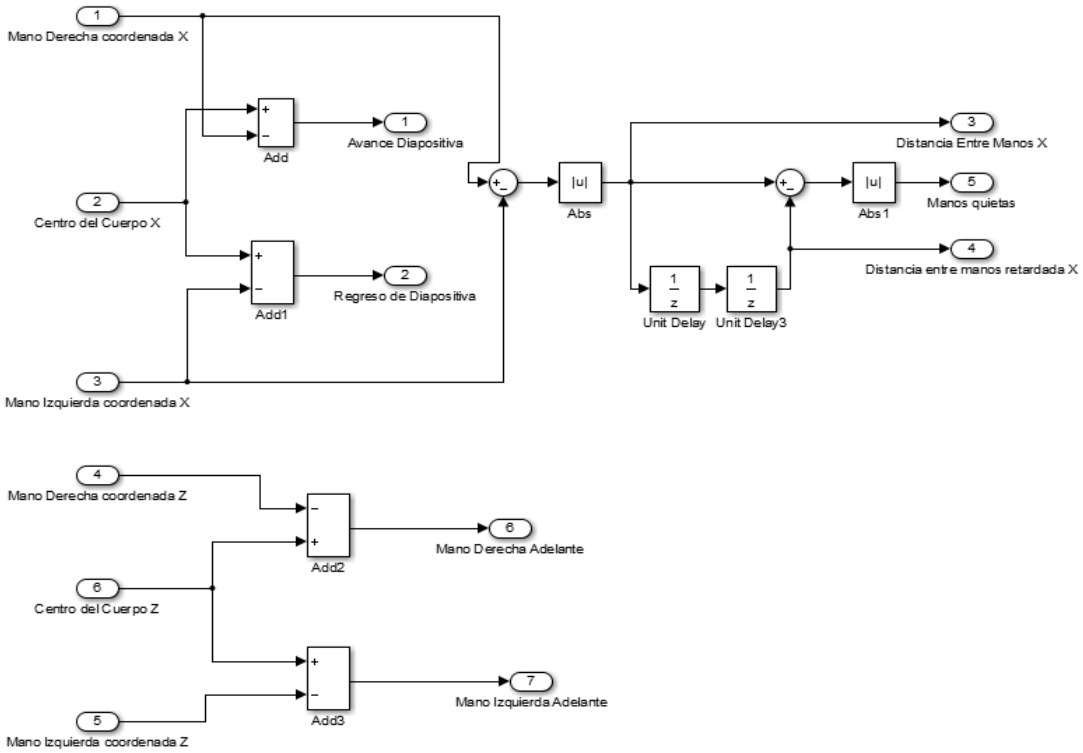

Figura 4.29. Modelo de la descripción de movimientos a través de las coordenadas de las articulaciones mano izquierda, derecha y centro del cuerpo

4.4.3.1 Descriptores unidimensionales

Se describió este movimiento a través de un algoritmo comparativo de proporcionalidades del cuerpo, que mide la distancia que existe entre la coordenada en (x) de la mano derecha con la coordenada en (x) del centro del cuerpo. Esta distancia es un valor absoluto para cada persona, independiente de donde se encuentre la personas con respecto al Kinect.

- El algoritmo comparativo de descripción de brazo derecho arriba extendido horizontalmente lo representa la ecuación 4.4.

$$d = |(coordenada\ X\ centro\ del\ cuerpo) - (coordenada\ X\ mano\ derecha)| \quad [4.4]$$

Dónde:

d= representa la distancia del brazo derecho extendido horizontalmente

En la Figura 4.30 se observa como la distancia del brazo derecho permanece constante aunque el usuario se haya movido de su posición original.

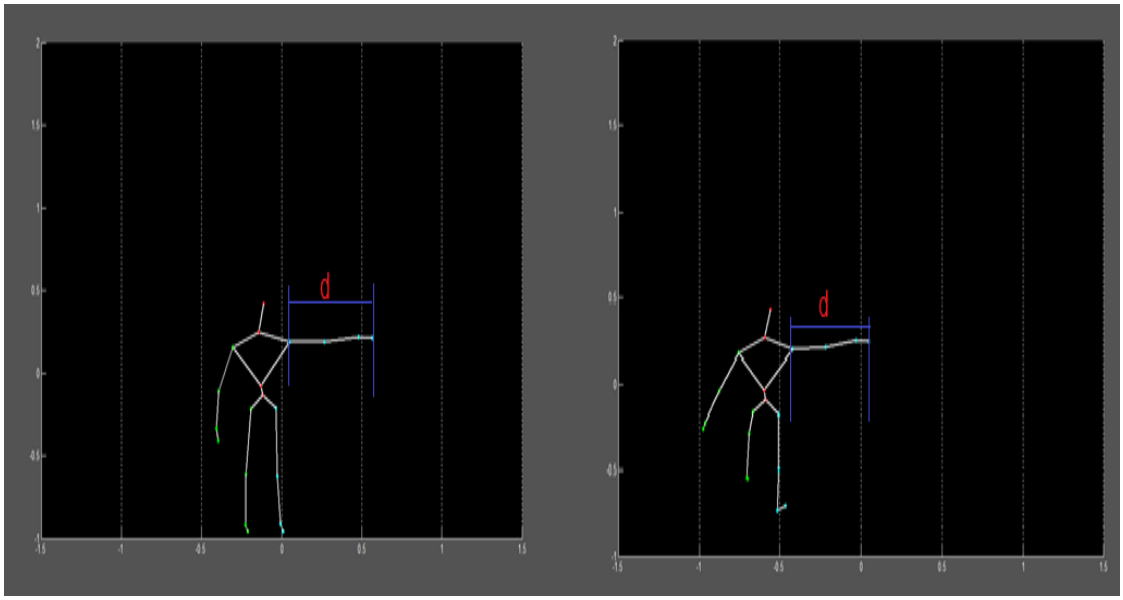

Figura 4.30. Distancia del brazo derecho arriba extendido horizontalmente

En la Figura 4.31 se observa la señal que describe el movimiento del brazo derecho cuando se lo levante horizontalmente. Como se observa esta señal es un poco ruidosa; pero se debe a que la señal capta la distancia que existe entre la mano y el centro del cuerpo. Las pequeñas variaciones significan que la mano derecha se está moviendo en relación al centro del cuerpo. También se encontró un pico alto en la señal, el cual representa la distancia máxima de separación del centro del cuerpo hacia la mano derecha y esto significa que el brazo derecho se encuentra extendido horizontalmente.

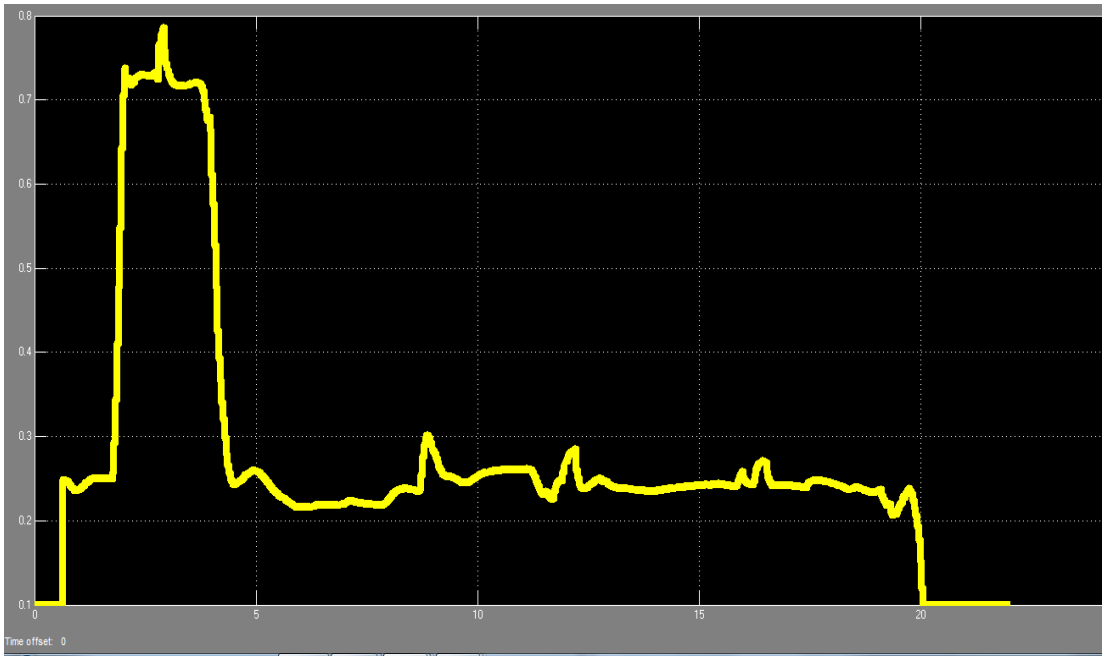

Figura 4.31. Señal de la distancia del brazo derecho arriba extendido horizontalmente

Se realiza exactamente el mismo procedimiento con el brazo izquierdo.

- El algoritmo comparativo de descripción de brazo izquierdo arriba extendido horizontalmente lo representa la ecuación 4.5.

$$d = |(coordenada\ X\ centro\ del\ cuerpo) - (coordenada\ X\ mano\ izquierda)| \quad [4.5]$$

Dónde:

d= representa la distancia del brazo izquierdo extendido horizontalmente

En la Figura 4.32 se observa como la distancia del brazo izquierdo permanece constante aunque el usuario se haya movido de su posición original.

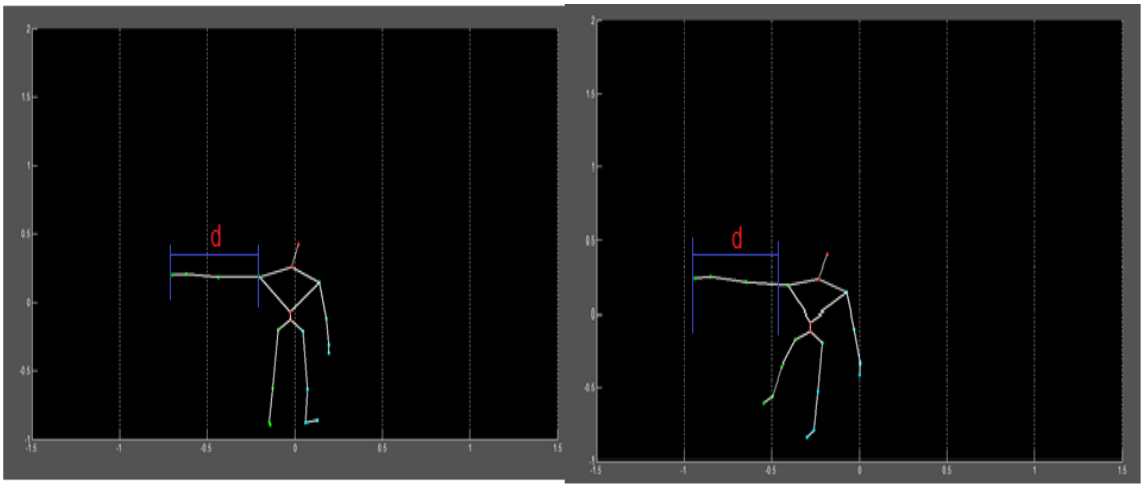

Figura 4.32. Distancia del brazo izquierdo arriba extendido horizontalmente

En la Figura 4.3 se encontró un pico bajo en la señal, el cual representa la distancia máxima de separación del centro del cuerpo hacia la mano izquierda y esto significa que el brazo izquierdo se encuentra extendido horizontalmente.

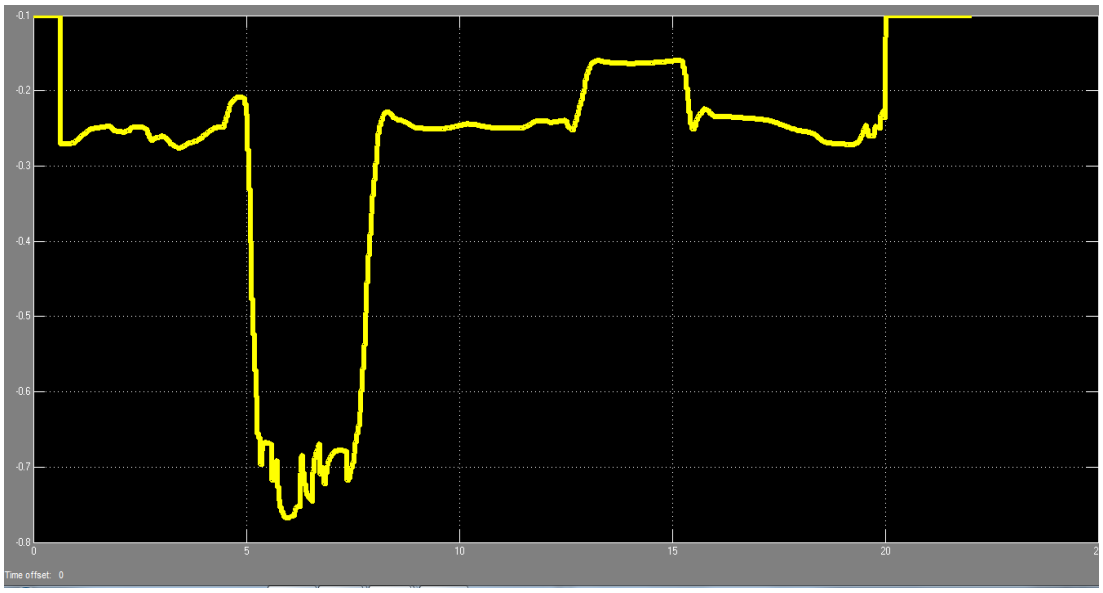

Figura 4.33. Señal de la distancia del brazo izquierdo arriba extendido horizontalmente

La descripción de mano derecha adelante se describió a través de un algoritmo comparativo de proporcionalidades del cuerpo, que mide la distancia que existe entre la coordenada en (z) de la mano derecha con la coordenada en (z) del centro del cuerpo. Esta distancia es un valor absoluto para cada persona, independiente de donde se encuentre la personas con respecto al Kinect.

El algoritmo comparativo de descripción de mano derecha adelante lo representa la ecuación 4.6.

$$d = |(coordenada\ Z\ centro\ del\ cuerpo) - (coordenada\ Z\ mano\ derecha)| \quad [4.6]$$

Dónde:

d= representa la distancia del brazo derecho extendido hacia adelante

En la Figura 4.34 se observa como la distancia de la mano derecha adelante permanece constante aunque el usuario se haya movido de su posición original.

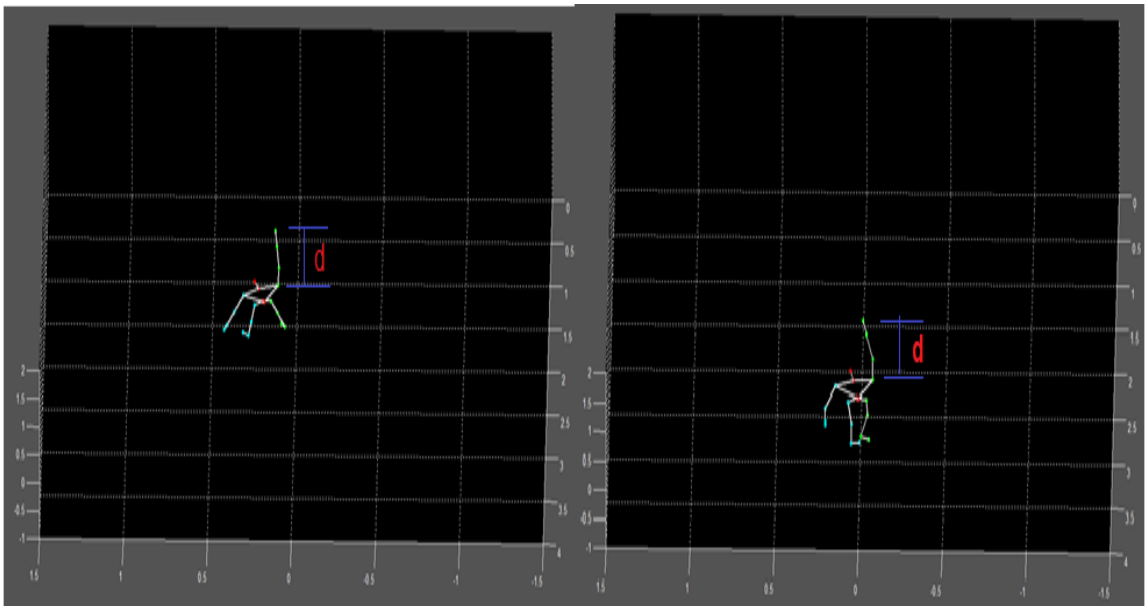

Figura 4.34. Mano derecha adelante

La Figura 4.35 muestra un pico alto en la señal, el cual representa la distancia máxima de separación del centro del cuerpo hacia la mano

derecha y esto significa que la mano se encuentra extendida hacia adelante del cuerpo humano.

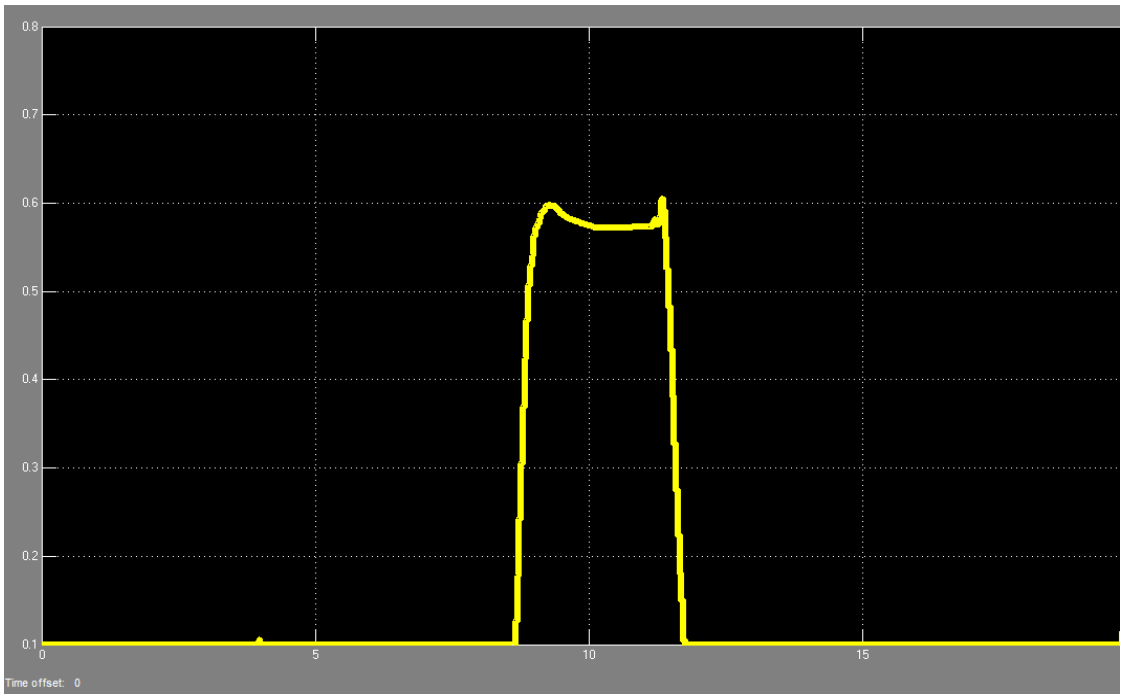

Figura 4.35. Señal de la mano extendida hacia adelante

Se realiza exactamente el mismo procedimiento con el brazo izquierdo.

- El algoritmo comparativo de descripción de mano izquierda adelante lo representa la ecuación 4.7.

$$d = |(coordenada\ Z\ centro\ del\ cuerpo) - (coordenada\ Z\ mano\ izquierda)| \quad [4.7]$$

Dónde:

d= representa la distancia del brazo izquierdo extendido hacia adelante

En la Figura 4.36 se observa como la distancia de la mano izquierda adelante permanece constante aunque el usuario se haya movido de su posición original.

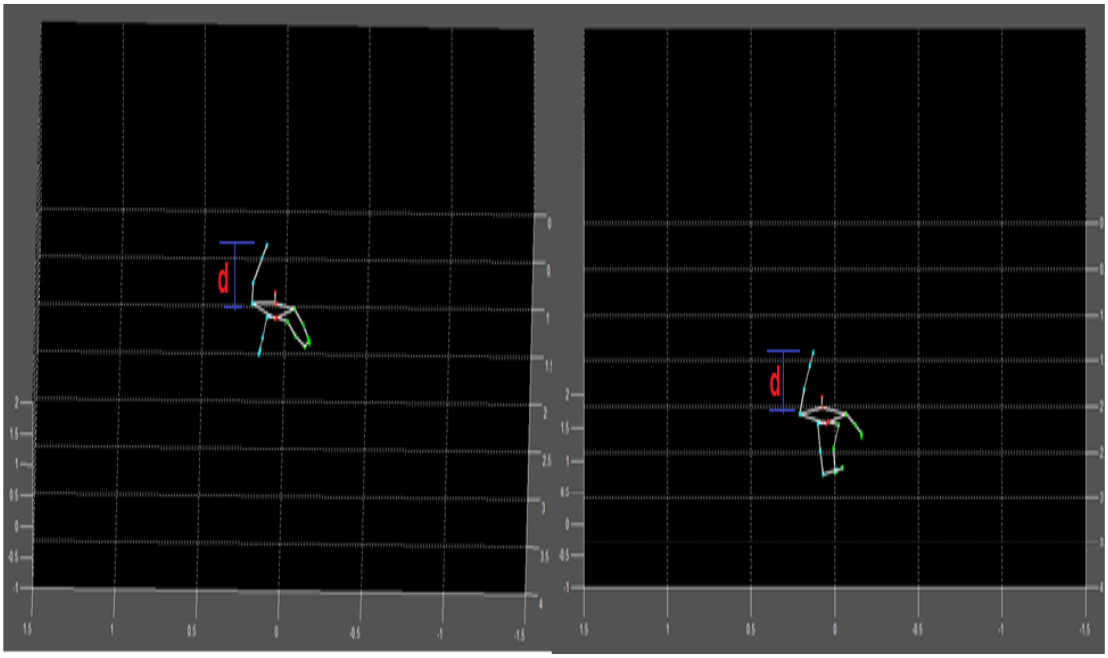

Figura 4.36. Mano derecha izquierda

La Figura 4.37 muestra un pico alto en la señal, el cual representa la distancia máxima de separación del centro del cuerpo hacia la mano izquierda y esto significa que la mano se encuentra extendida hacia adelante del cuerpo humano.

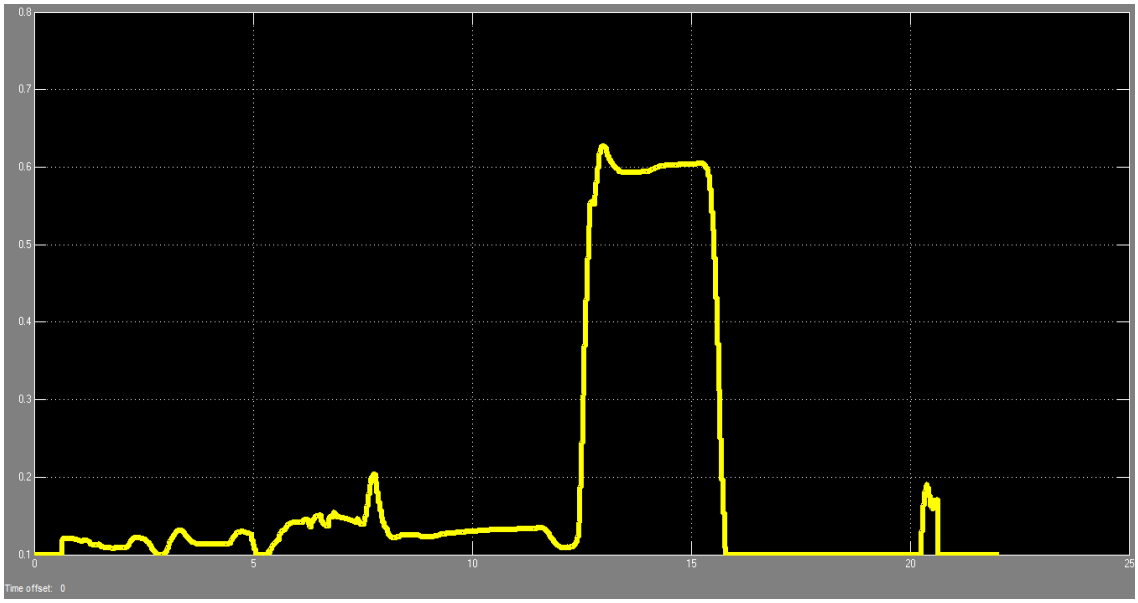

Figura 4.37. Señal de la mano izquierda extendida hacia adelante

En la Figura 4.38 se muestra las señales de los movimientos brazo derecho arriba extendido horizontalmente, brazo izquierdo arriba extendido horizontalmente, mano derecha adelante y mano izquierda adelante en un mismo intervalo de tiempo. En la figura se observa que mientras se describe un movimiento específico en el cuerpo humano también el cuerpo humano realiza otros movimientos pequeños con diferentes articulaciones.

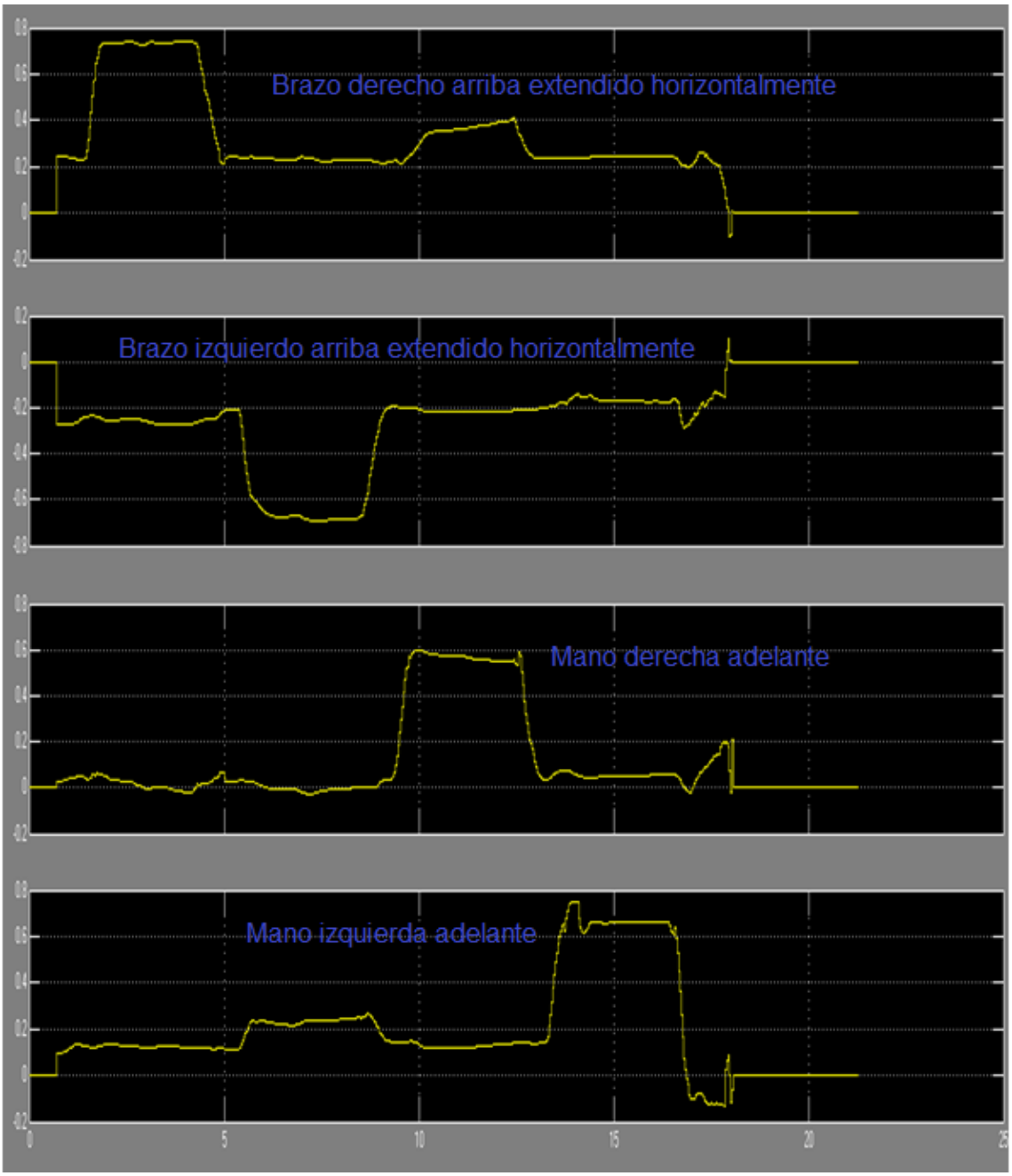

Figura 4.38. Señales de los movimientos descritos en un mismo intervalo de tiempo

Los ejes en todas las figuras tienen las siguientes escalas:

Eje Y: Longitud de los brazos (m)

Eje X: Tiempo (s)

4.4.3.2 Descriptores especiales para medir el flujo óptico

Este descriptor determina el movimiento a través de una comparación de una medida con respecto a la misma medida aplicada un filtro discreto digital, que permite que la señal de la medida se retrase en el dominio del tiempo.

La primera medida que se necesita es la distancia de separación que existe entre la coordenada (x) de la mano derecha y la coordenada (x) de la mano izquierda. Esta medida es comparada con su retraso y se obtiene la diferencia entre las dos medidas. Si la diferencia es positiva se determina que el movimiento es de apertura y si la diferencia es negativa el movimiento es de cierre. El flujo óptico es representado por la ecuación 4.8 y el algoritmo que mide el flujo óptico lo representa la ecuación 4.9.

$$\Delta imagenes = (dS) \quad [4.8]$$

(Pijares Martinsanz & De la Cruz Garcia, Vision por Computadora, 2008)

Dónde:

dS: Flujo óptico

$$dSm = |(mdX - miX) - Z^{-1}(Z^{-1}(mdX - miX))| \quad [4.9]$$

Dónde:

dSm: Flujo óptico de las manos

mdX: Coordenada X de la mano derecha

miX: Coordenada X de la mano izquierda

Z^{-1}: Retraso de una muestra a la señal de entrada

La figura 4.39 muestra dos señales ondulatorias: una señal de color amarillo y una morada. La señal amarilla determina la apertura o cierre de las manos y la señal morada es el retraso de la señal amarilla en el dominio del tiempo; debido a la aplicación de dos filtros discretos digitales como se constata en la fórmula [4.9].

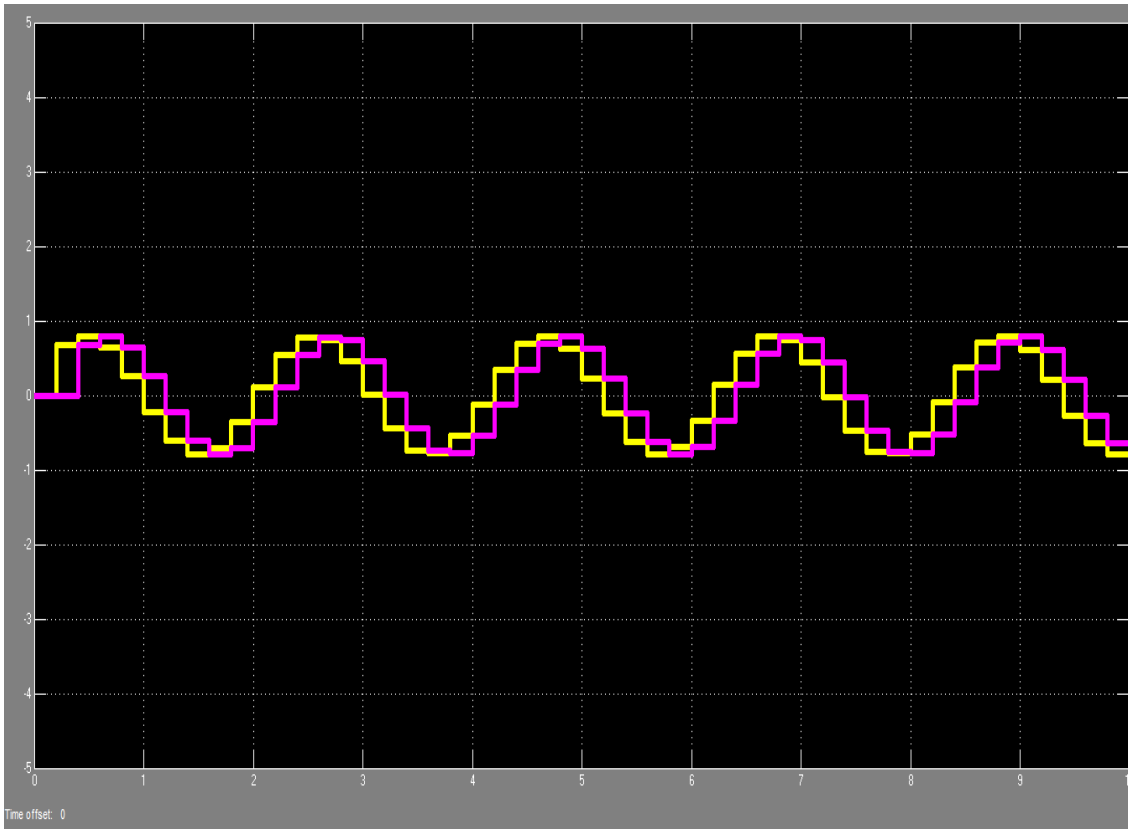

Figura 4.39. Señal de la diferencia entre las manos y su señal retrasada

En la figura 4.40 se observa que las señales en cierto momento tienden a subir y en otro tiende a bajar. El movimiento apertura de las manos es el que provoca que las señales tienda a subir y el movimiento de cierre de las manos es el que provoca que las señales tiendan a bajar.

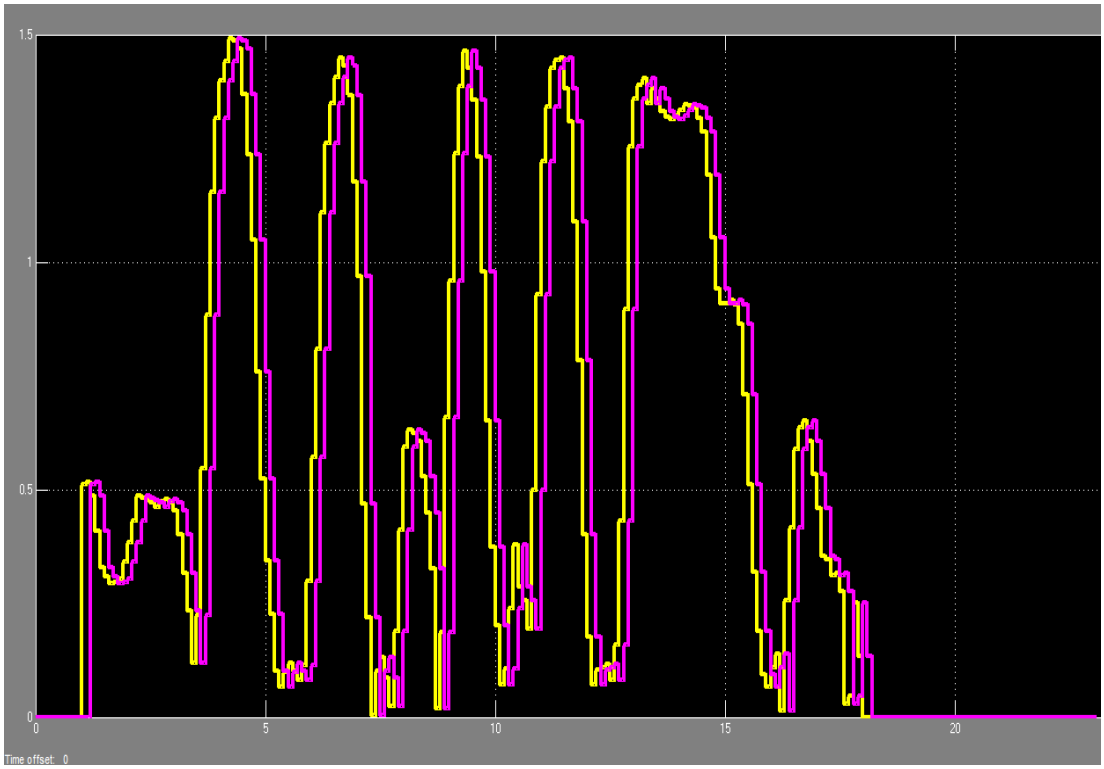

Figura 4.40. Señales de los movimientos de apertura y cierre realizados con las manos

Los ejes en todas las figuras tienen las siguientes escalas:

Eje Y: Longitud de los brazos (m)

Eje X: Tiempo (s)

4.4.4 RECONOCIMIENTO

Es la categorización de datos de entrada en clases identificadas por medio de la descripción de características significativas o atributos de los datos extraídos de un medio ambiente. Matemáticamente hablando, la clasificación consiste en la partición del espacio n-dimensional definido por las características de un objeto en varias regiones donde cada región corresponde a una clase. (Sobrado, 2003)

La aplicación necesita reconocer ciertos movimientos que realiza el usuario para interactuar con el PC y estos movimientos se los describieron a través de algoritmos comparativos de proporcionalidades del cuerpo y algoritmos que miden el flujo óptico. La aplicación al usar la interfaz natural del usuario (NUI) sensa cualquier persona que se encuentre en su área de trabajo; pero no todas las personas son de igual tamaño o tienen las mismas medidas corporales. Por ese motivo al usar dichos algoritmos cada persona tendrá diferente valor de resultado y esto no permite que el movimiento descrito corresponda a una sola clase.

La solución para esto es realizar una parametrización a cada usuario antes de usar la aplicación. Con esto se obtendrán las medidas corporales de cada usuario a las cuales se les aplicarán métodos estadísticos, para determinar cuál es el valor óptico que describe cada movimiento y este valor se enviará a la aplicación primaria. Con estos datos únicos de cada usuario adquiridos de la parametrización, la descripción de movimiento de la aplicación primaria podrá categorizar en clases los movimientos para su posterior interpretación.

4.4.4.1 Parametrización del usuario

La parametrización es la obtención de valores constantes llamados parámetros dentro del cuerpo humano. Para la aplicación se obtendrá las proporcionalidades de ciertas partes del cuerpo humano. La aplicación de parametrización se la puede observar en la Figura 4.41.

La aplicación de parametrización en relación con la aplicación primaria solamente desarrolla las etapas de: adquisición, segmentación y descripción del proceso de visión artificial de las siguientes articulaciones:

- Mano derecha
- Mano izquierda
- Centro del cuerpo

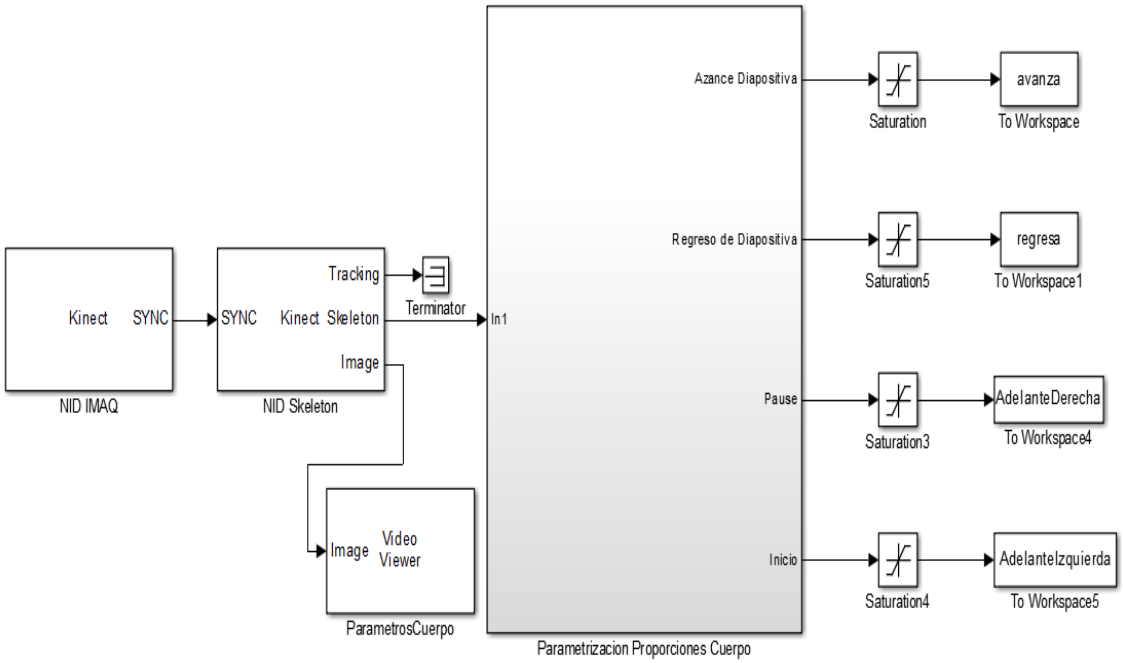

Figura 4.41. Aplicación de parametrización del cuerpo humano

Obtenidas las coordenadas de las articulaciones se realiza un proceso de descripción de los movimientos a través de la posición de estas coordenadas. Como lo muestra la Figura 4.42.

Las señales que se obtienen de la descripción se las limita con los rangos máximos de altura y de distancia horizontal máxima del Kinect. Por último estos datos se los clasifican y se los envía a la aplicación primaria.

Los movimientos que se describe en esta aplicación son los mismos que en la aplicación primaria; ya que sólo tienen diferentes nombres y estos nombres están relacionados con las acciones a realizar dentro del PC.

La clave de la parametrización es que se enfoca en cada usuario. Todas estas etapas de visión artificial deberán realizarlas antes de ejecutar la aplicación primaria, para determinar los parámetros del usuario que va a usar la aplicación.

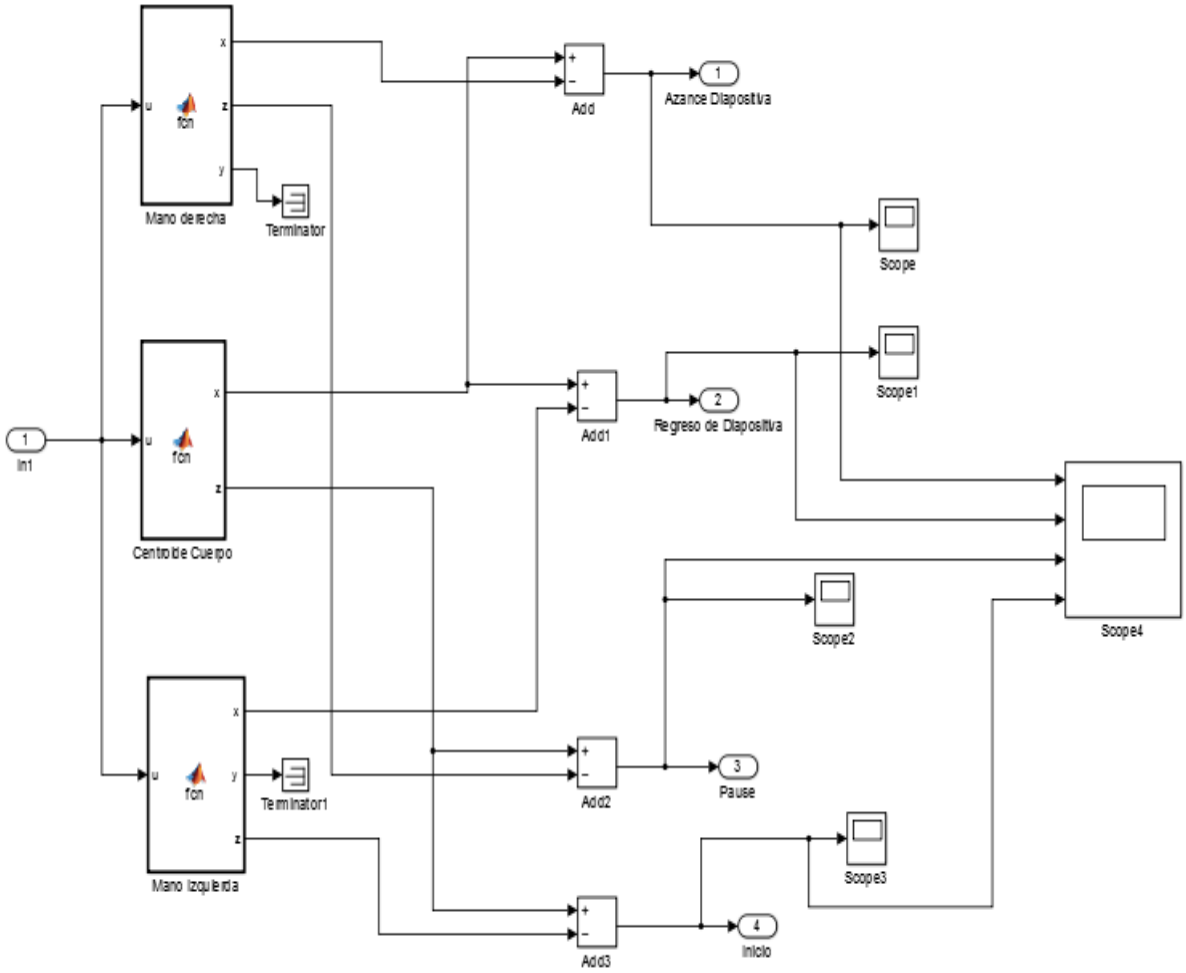

Figura 4.42. Etapas de la visión artificial de la aplicación de parametrización del cuerpo humano

4.4.4.2 Obtención de valores óptimos

Como se puede observar en la Figura 4.38 siempre existe una señal, la señal óptima que describe el movimiento es un pico dentro de la señal normal; pero el resto de la señal hace mención a que existe un pequeño movimiento de esa articulación. Por esta razón los valores que se toman para enviar a la aplicación primaria son extensos; por ello se necesita distribuir estos valores para comprenderlos de una mejor manera.

Para esto se usará la distribución continua de probabilidad que permite modelar numerosos fenómenos naturales, de la industria, y de la

investigación (Walpole R. E., 2007). Las señales obtenidas son variables aleatorias continuas que significa que pueden adquirir un número infinito de valores en un determinado rango de tiempo. Pero todos los valores obtenidos se distribuyen uniformemente alrededor de una media (σ) esto es lo que permite evaluar la distribución a través de la función gaussiana como lo muestra la Figura 4.43 que tiene forma de campana, y que sus propiedades son:

- El campo de existencia es cualquier valor real (-∞; ∞)
- Todos los datos son simétricos respecto a la media
- El eje de abscisas es una asíntota de la curva
- La probabilidad equivale al área bajo la curva

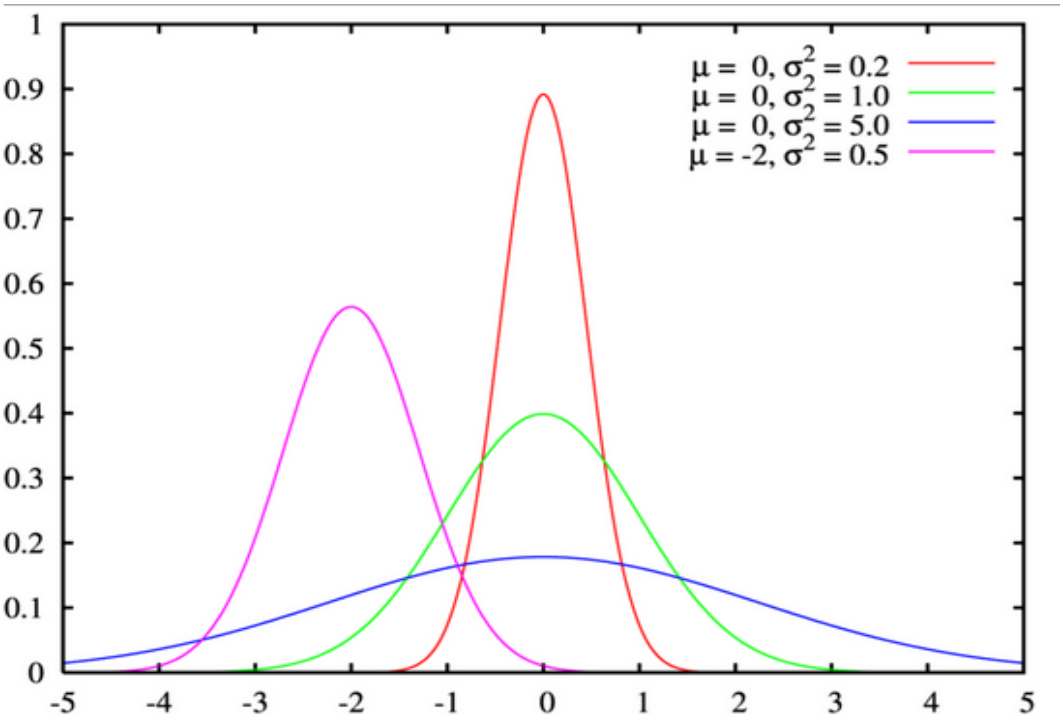

Figura 4.43. Función Gaussiana con diferentes parámetros
(Walpole, Myers, Myers, & Ye, 2007)

Para la aplicación y para describir de forma correcta el movimiento, se basó el análisis del parámetro óptimo a través de la probabilidad estadística de que el movimiento esté pasando. Para esto se transformó el pico de la señal

que es una variable aleatoria continua (X) a un nuevo conjunto de observaciones de una variable aleatoria normal (Z). El valor Z mide la distancia entre un valor especificado de X y la media aritmética en las unidades de la desviación estándar. Al determinar el valor Z utilizando la ecuación 4.10, es posible encontrar el área de probabilidad bajo cualquier curva normal.

$$Z = \frac{X-\mu}{\sigma} \quad [4.10]$$

(Walpole R. E., 2007)

Dónde:

Z: Valor Optimo de la variable aleatoria normal

X: Valor Optimo de la variable aleatoria de las señales

μ: Media de todas las señales

σ: Desviación Estándar de las señales.

Para obtener el valor óptimo se partirá de una manera inversa a la ecuación anterior. Primero se buscará una probabilidad adecuada para luego encontrar el valor óptimo que englobe esta probabilidad.

En la tabla de áreas de la curva normal (Walpole, 2007) se buscó una probabilidad del 95% de que el movimiento este pasando. Se escogió el 95%, ya que el movimiento es descrito por las proporcionalidades que existen entre dos articulaciones del cuerpo previamente determinadas. Estas proporcionalidades pueden adquirir innumerables valores en un rango de tiempo; por lo tanto si se usaba una probabilidad mayor podría suceder que el movimiento nunca se reconozca o si se usaba una probabilidad menor cualquier movimiento de las articulaciones podía ser interpretado como el movimiento específico. Como lo muestra la ecuación 4.10 a la ecuación 4.11.

$$(Z * \sigma) + \mu = Xoptimo \quad [4.11]$$

(Walpole R. E., 2007)

El 95 % de la probabilidad corresponde a un Z= 1.65 como lo muestra la ecuación 4.12.

$$(1.65 * \sigma) + \mu = Xoptimo \quad [4.12]$$

(Walpole R. E., 2007)

La media es igual a la suma de todas las variables aleatorias continuas, divididas entre el número de variables aleatorias continuas, como lo muestra la ecuación 4.13.

$$(1.65 * \sigma) + \left(\frac{1}{n} * \sum_{i=1}^{n} Xi\right) = Xoptimo \quad [4.13]$$

Los picos de las señales representa la existencia del movimiento, por eso se tomará la desviación estándar como la desviación que existe desde un valor promedio de movimiento al valor máximo de movimiento como lo muestra la ecuación 4.14.

$$\left(1.65 * (Xmax - (\frac{1}{n} * \sum_{i=1}^{n} Xi))^2\right) + (\frac{1}{n} * \sum_{i=1}^{n} Xi) = Xoptimo \quad [4.14]$$

La Figura 4.44 muestra la distribución de la ecuación 4.14, con las señales de mi movimiento brazo derecho arriba extendido horizontalmente. Dónde:

Xmax: 0.7498

Z: 1.65

μ: 0.2932

σ: 0.2085, Xóptimo: 0.6371

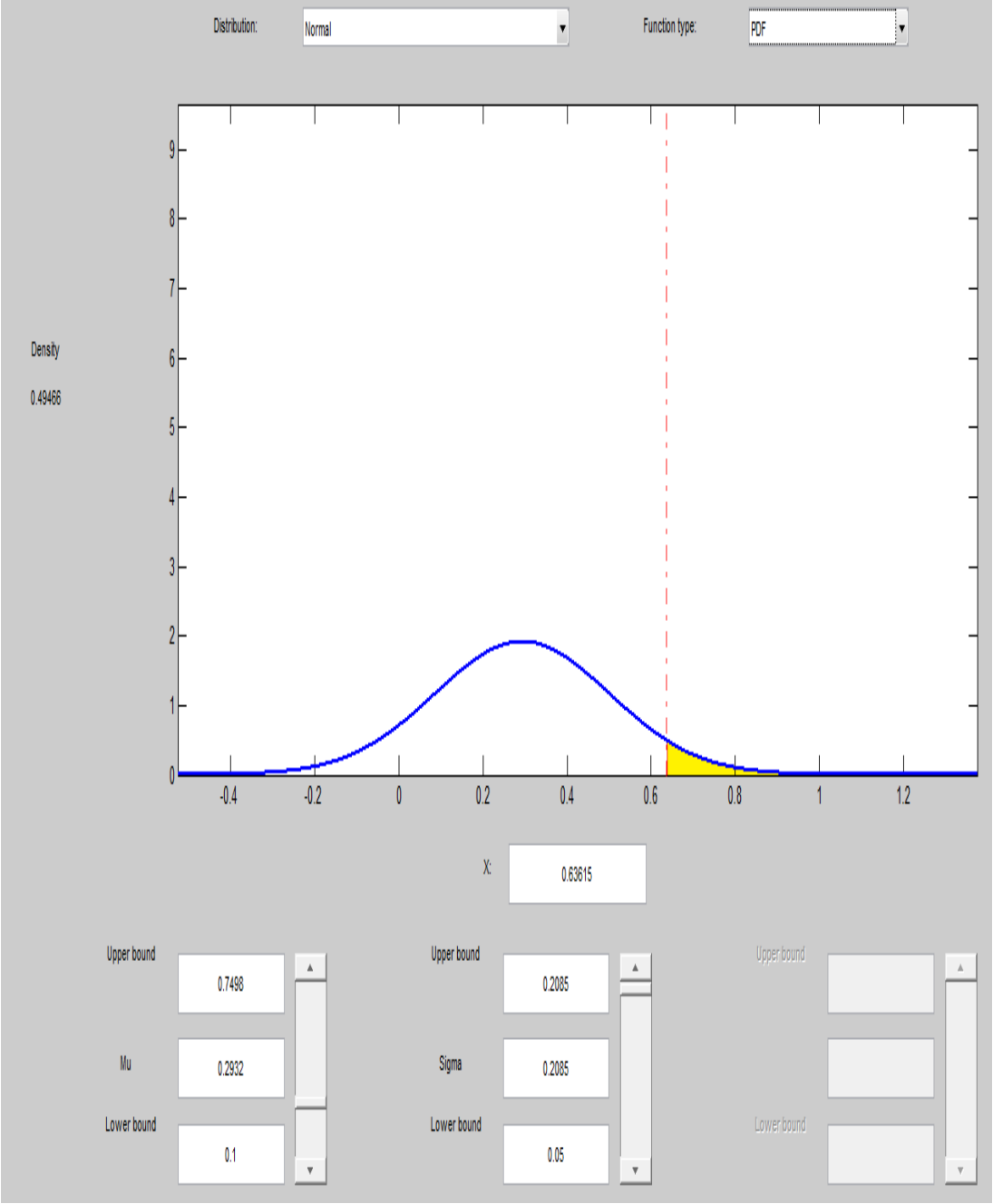

Figura 4.43. Distribución Normal para el movimiento brazo derecho arriba extendido horizontalmente

En la figura 4.44 se observa la distribución acumulada de variable aleatoria X que representa la probabilidad de la variable X. En el extremo izquierdo de la imagen se puede observar la probabilidad del valor 0.6371.

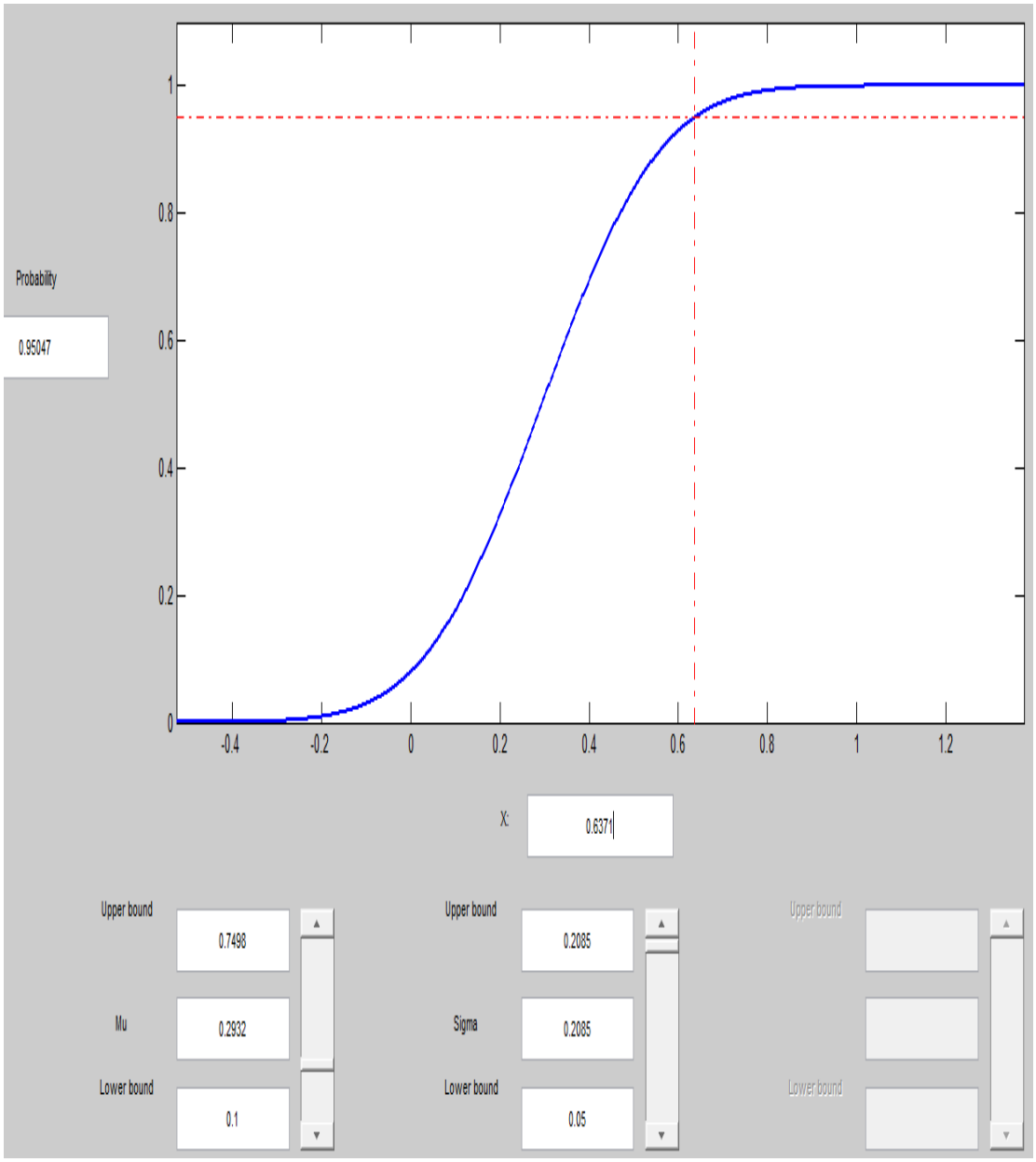

Figura 4.44. Distribución acumulada para el movimiento brazo derecho arriba extendido horizontalmente

Todo este proceso lo realiza el subsistema de la aplicación primaria de reconocimiento de datos característicos mostrado en la Figura 4.45. En el cual los valores enviados por la aplicación de parametrización son recibidos y son la entrada de una Matlab Function donde se realiza todo el proceso anterior de búsqueda del valor óptimo a través de la distribución normal. Vale recalcar que el mismo proceso de búsqueda del valor óptimo del movimiento brazo derecho arriba extendido horizontalmente se debe realizar

para todos los movimientos descritos unidimensionalmente como son: Brazo derecho arriba extendido horizontalmente, Brazo izquierdo arriba extendido horizontalmente, Mano derecha adelante y Mano izquierda adelante.

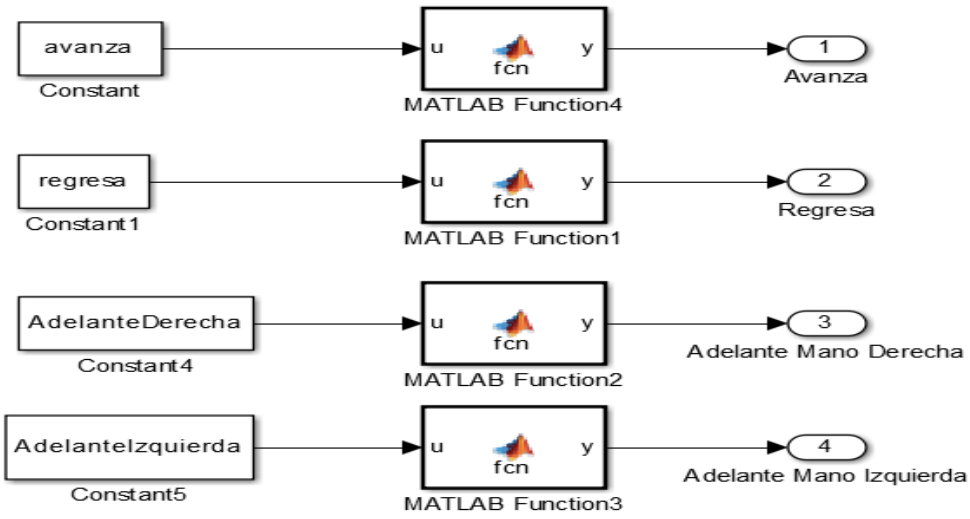

Figura 4.45. Reconocimiento de datos característicos enviados por la aplicación de parametrización en la aplicación primaria

4.5 INTERPRETACIÓN DE LOS DATOS CARACTERÍSTICOS

Es la comprensión de los datos obtenidos a través de la visión artificial para usarlos en condiciones o fórmulas que demuestren alguna acción o decisión determinada.

Los valores óptimos de reconocimiento de cada movimiento se convertirán en las entradas de un entorno de modelado y simulación lógica de decisión combinatoria y secuencial basado en máquinas de estado y diagramas de flujo. Los valores óptimos son elementos de una lógica de condiciones que permiten la transición de un estado a otro como lo muestra la Figura 4.46.

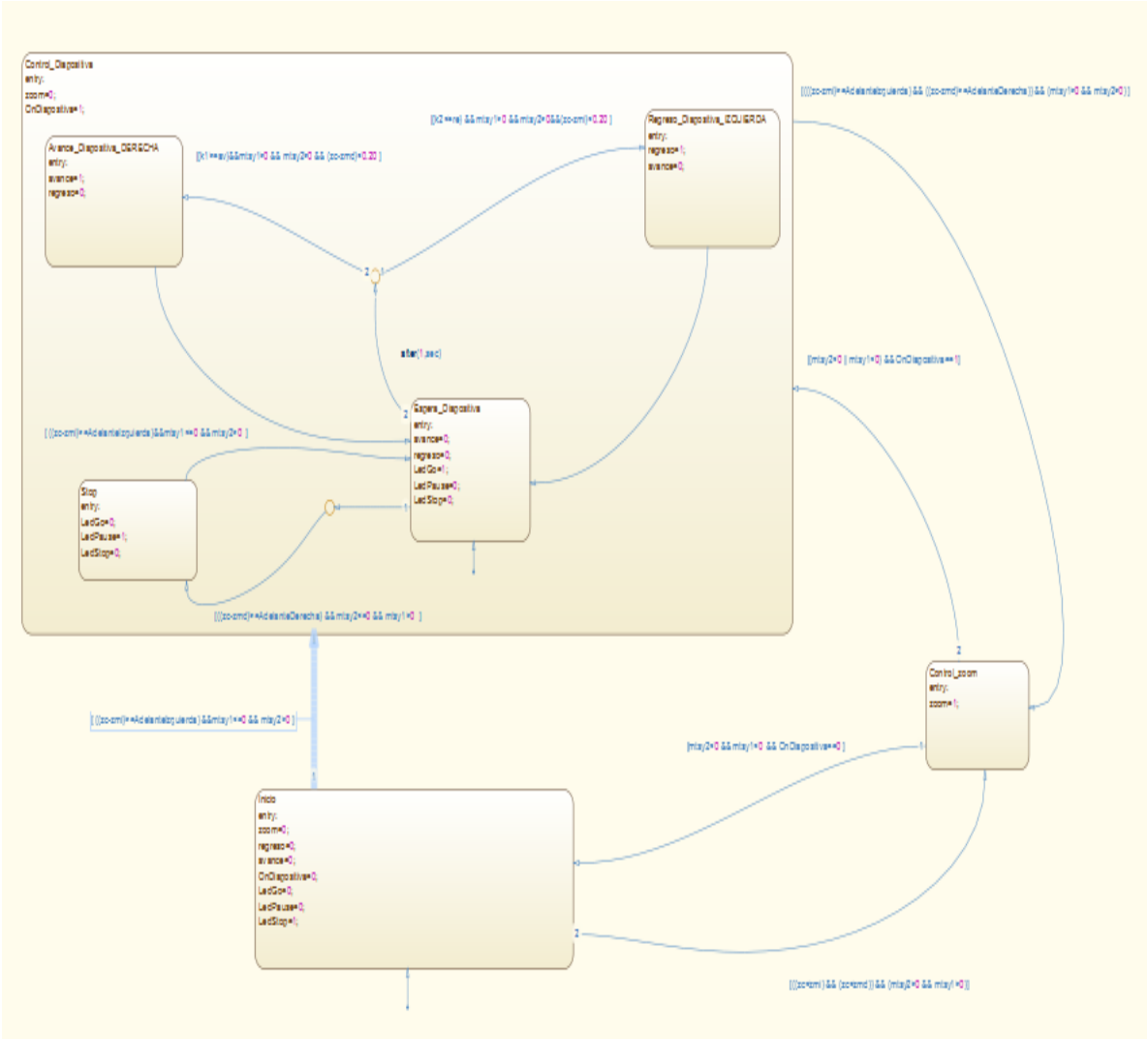

Figura 4.46. StateFlow de la aplicación

En la Figura 4.46 se ven líneas de color azul que representan las condiciones de transición de un estado al otro y los cuadros grandes y pequeños demuestran los diferentes estados que tiene la aplicación. Dentro de estos cuadros se encuentra una sintaxis que determina las acciones que debe realizar el PC.

Para explicar el funcionamiento del State Flow se partirá de una explicación de qué contiene cada estado y cuáles son sus transiciones relacionadas al reconocimiento del movimiento como lo muestra la Tabla 4.8.

Tabla 4.8. Contenido de cada estado del State Flow y transiciones con relación a los movimientos del cuerpo establecidos

Estado	Contenido	Transiciones
Inicio	• Salida de zoom apagada • Salida de regreso de la diapositiva apagada • Salida de avance de diapositiva apagada • Salida de LED Verde apagada • Salida de LED Amarillo encendida • Salida de LED Rojo apagada	• Mano izquierda adelante • Apertura y cierre de manos adelante
Control de diapositivas	• Salida zoom apagada	• Apertura y cierre de manos adelante
Avance de diapositivas	• Salida de avance de diapositiva encendida • Salida de regreso de la diapositiva apagada	• Brazo derecho arriba extendido horizontalmente • Apertura y cierre de manos adelante
Regreso de diapositivas	• Salida de avance de diapositiva apagada • Salida de regreso de la diapositiva encendida	• Brazo izquierdo arriba extendido horizontalmente • Apertura y cierre de manos adelante
Espera de diapositivas	• Salida de zoom apagada • Salida de regreso de la diapositiva apagada • Salida de avance de diapositiva apagada • Salida de LED Verde encendida • Salida de LED Amarillo apagada • Salida de LED Rojo apagada	• Después de 2 segundos de activado el Estado
Stop	• Salida de LED Verde apagado • Salida de LED Amarillo apagada • Salida de LED Rojo encendido • Salida de regreso de la diapositiva apagada • Salida de avance de diapositiva apagada	• Mano izquierda adelante • Mano derecha adelante
Control de zoom	• Salida de zoom encendida	• Apertura y cierre de manos adelante

Según la tabla 4.8 la aplicación tiene siete estados que son: inicio, control de diapositivas, avance de diapositivas, regreso de diapositivas, espera de

diapositivas, stop y control de zoom. Las transiciones permiten pasar de un estado al otro cumpliendo con las condiciones propias de cada transición.

- Inicio

Es el estado inicial donde la aplicación comienza. En este estado todas las salidas del State Flow se encuentran apagadas excepto la salida para un LED indicador de color amarillo, el cual indica que el usuario se encuentra en ese estado.

Sus transiciones pueden cambiar al estado de control de zoom y control de diapositivas como lo muestra la Tabla 4.9.

Tabla 4.9. Condiciones de transición del estado inicio

Estado Actual	**Condición de la transición**	**Estado Final**
Inicio	La mano izquierda adelante debe ser igual o mayor al parámetro de mano izquierda adelante **Y** La altura de la mano izquierda es mayor a la altura de ubicación del Kinect **Y** La altura de la mano derecha es menor a la altura de la ubicación del Kinect	Control de diapositivas
Inicio	Las dos manos se deben encontrar adelante **Y** La altura de las 2 manos debe ser mayor a la altura del Kinect	Control Zoom

- Control de diapositivas

Es el estado que contiene a los subestados de avance de diapositiva: regreso de diapositiva, parar y espera de diapositivas. Además es el estado que permite la interacción con el control del zoom como lo muestra la Tabla 4.10.

Tabla 4.10. Condiciones de transición del estado control de diapositivas

Estado Actual	**Condición de la transición**	**Estado Final**
Control de diapositivas	Las dos manos se deben encontrar adelante **Y** La altura de las 2 manos debe ser mayor a la altura del Kinect	Control Zoom

- Espera de diapositiva

Es el estado que identifica si el usuario puede seguir pasando las diapositivas o debe parar. En este estado se encuentran todas las salidas

del State Flow apagadas excepto la salida para el LED indicador de color verde; si el LED se encuentra activado el usuario puede avanzar o retroceder de diapositiva.

Sus transiciones pueden cambiar al estado avance de diapositiva, regreso de diapositivas, Stop y control del zoom, como lo muestra la Tabla 4.11.

Tabla 4.11. Condiciones de transición del estado espera de diapositivas

Estado Actual	**Condición de la transición**	**Estado Final**
Espera de diapositivas	Después de 1 segundo de activado este estado **Y** el brazo derecho arriba extendido horizontalmente debe ser mayor o igual al parámetro de brazo derecho arriba extendido horizontalmente **Y** la Altura del brazo derecho debe ser mayor a la altura de ubicación del Kinect **Y** La coordenada en (y) de la mano izquierda debe estar a 10 cm por debajo de la coordenada en (y) del centro del cuerpo	Avance de diapositiva
Espera de diapositivas	Después de 1 segundo de activado este estado **Y** el brazo izquierdo arriba extendido horizontalmente debe ser mayor o igual al parámetro del brazo izquierdo arriba extendido horizontalmente **Y** la Altura del brazo izquierdo debe ser mayor a la altura de ubicación del Kinect **Y** La coordenada en (y) de la mano derecha debe estar a 10 cm por debajo de la coordenada en (y) del centro del cuerpo	Regreso de diapositivas
Espera de diapositivas	La mano derecha adelante debe ser igual o mayor al parámetro de mano derecha adelante **Y** La altura de la mano derecha es mayor a la altura de ubicación del Kinect **Y** La altura de la mano izquierda es menor a la altura de la ubicación del Kinect	Stop
Espera de diapositivas	Las dos manos se deben encontrar adelante **Y** La altura de las 2 manos debe ser mayor a la altura del Kinect	Control Zoom

- Parar (Stop)

Es el estado de pausa de la aplicación, no realiza ninguna acción, se lo usó por seguridad cuando no quiere el usuario que ningún movimiento de él sea detectado. En este estado se encuentran todas las salidas del State Flow apagadas excepto la salida para el LED indicador de color rojo; si el LED se encuentra activo el usuario entiende que se encuentra en Stop. Su única

transición es el regreso al estado de espera de diapositiva como lo muestra la Tabla 4.12.

Tabla 4.12. Condiciones de transición del estado stop

Estado Actual	Condición de la transición	Estado Final
Stop	La mano izquierda adelante debe ser igual o mayor al parámetro de mano izquierda adelante **Y** La altura de la mano izquierda es mayor a la altura de ubicación del Kinect **Y** La altura de la mano derecha es menor a la altura de la ubicación del Kinect	Espera diapositivas

- Avance de diapositivas

Es el estado que permite avanzar la diapositiva. El estado tiene una de las salidas del State Flow que proporciona un dato de valor 1 activado y 0 desactivado. Existe la condición de que cuando esté activado el estado avance de diapositiva, el estado de regreso de diapositiva deba ser 0. Este estado no tiene una transición definida por los parámetros de proporcionalidad del cuerpo sólo cambia de estado un segundo después de que el mismo estado avance de diapositiva fue activado y regresa al estado de espera de diapositiva.

- Regreso de diapositivas

Es el estado que permite regresar a una diapositiva. El estado tiene una de las salidas del State Flow que proporciona un dato de valor 1 activado y 0 desactivado. Existe la condición de que cuando esté activado el estado regreso de diapositiva el estado de avance de diapositiva deba ser 0. Este estado no tiene una transición definida por los parámetros de proporcionalidad del cuerpo sólo cambia de estado un segundo después de que el mismo estado regreso de diapositiva fue activado y luego pasa al estado de espera de diapositiva.

- Control del zoom

Es el estado que permite controlar el nivel de zoom de las diapositivas. Además el estado tiene una de las salidas del State Flow que proporciona un dato de valor 1 activado y 0 desactivado.

Sus transiciones pueden cambiar al estado inicio y control de diapositivas como lo muestra la Tabla 4.13.

Tabla 4.13. Condiciones de transición del estado control de zoom

Estado Actual	Condición de la transición	Estado Final
Control de zoom	La altura de las 2 manos debe ser menor a la altura d la ubicación Kinect	Inicio Control de diapositivas

Los datos de salida del State Flow son los eventos para controlar el PC y para indicar al usuario lo que tiene que hacer. La Tabla 4.14 muestra dichos datos de salida.

Tabla 4.14. Tabla de datos de salida del State Flow

Nombre del dato de salida	Tipo de dato	Estado de referencia
Avanza	Binario	Avance de diapositivas
Regreso	Binario	Regreso de diapositivas
Zoom	Binario	Control de zoom
LED GO	Binario	Espera de diapositivas
LED STOP	Binario	Stop
LED INICIO	Binario	Inicio

4.6 CONTROL DE EVENTOS EN EL PC

Todo el proceso de visión artificial más su interpretación, entrega datos binarios; estos datos binarios representan eventos que tiene dos estados los cuales son: se realiza el evento o no se realiza el evento.

Estos eventos son las entradas en el bloque de control de eventos, que es una S-Function nivel 2. Una S-Function es similar a una Matlab Function solo que en ésta, se debe configurar todos los parámetros de recepción, proceso, análisis, resolución y de salida del bloque.

Este bloque se usó porque permite emular clases y funciones de otro lenguaje de programación para de esa forma interactuar con el PC.

4.6.1 PARÁMETROS DEL BLOQUE S-FUNCTION NIVEL 2 DE SIMULINK

- Parámetros de configuración de puertos

Estos parámetros permiten determinar cuántos puertos de entrada y salida tendrá el bloque. Además permite escoger si el puerto será dinámico o estático como lo muestra la Tabla 4.15.

Tabla 4.15. Parámetros de configuración de puertos del bloque S-Function

Parámetros	**Configuración**
Numero de puertos de entrada	6
Numero de puertos de salida	0
Propiedades dinámicas o heredadas	Dinámico

- Parámetros de configuración de datos

Estos parámetros permiten determinar todas las propiedades de los datos que va a contener el bloque como: tipo de dato, complejidad, etc como lo muestra la Tabla 4.16.

Tabla 4.16. Parámetros de configuración de los datos del bloque S-Function

Parámetros	**Configuración**
Tipo de dato	Double
Complejidad del dato	Real
Modo de muestreo del dato	Simple
Sobre escritura del dato	Desactivada
Ajuste del dato	Por defecto

- Parámetros de los métodos del bloque

Estos parámetros permiten registrar los métodos opcionales y necesarios con los que el bloque puede resolver, analizar y procesar los datos como lo muestra la Tabla 4.17.

Tabla 4.17. Parámetros de configuración de los métodos del bloque S-Function

Parámetros	**Configuración**	**Descripción**
Método Write RTW	Activado	Permite compilar la función
Método Outputs	Activado	Permite generar la función

4.6.2 FUNCIÓN DE CONTROL DE EVENTOS EN EL BLOQUE S-FUNCTION DE SIMULINK

Los eventos que determinan un movimiento se encuentran en las salidas del State Flow, tienen una entrada determinada en el bloque S-Function. No se puede unir cualquier salida del bloque State Flow con cualquier entrada del Bloque S-Function. Con esto la función primaria del bloque S-Funtion reconoce que dato se necesita para realizar el control de eventos.

Esta función primaria es un software embebido, debido a que se usa la clase Robot del paquete java.awt del lenguaje de programación Java. AWT es un conjunto de herramientas GUI (Interfaz Gráfica con el Usuario) diseñadas para interactuar con el PC.

La clase robot se utiliza para generar eventos de entrada del sistema para la automatización de pruebas, demos autoejecutables, y otras aplicaciones donde se requiere el control del ratón y el teclado. Para la aplicación se importará la clase Robot.

- import java.awt.Robot

La clase Robot tiene tres funciones importantes: el control del teclado, el control del mouse y el control de la pantalla. La aplicación usará el control del teclado.

Se importa la función que controla el teclado.

- import java.awt.event.KeyEvent

Se asigna una variable para inicializar la clase.

- control = Robot()

Dentro de tres estructuras de selección se utilizarán los eventos obtenidos del State Flow y otras condiciones para interactuar con el PC.

4.6.2.1 Control del avance de las diapositivas

Si el evento avanza es igual a uno (1), se ejecutarán las líneas de código:

- control.keyPress(KeyEvent.VK_RIGHT);
- control.keyRelease(KeyEvent.VK_RIGHT)

Las cuales permiten interactuar con el teclado, pulsando el cursor derecho.

4.6.2.2 Control del regreso de las diapositivas

Si el evento regreso es igual a uno (1), se ejecutarán las líneas de código:

- control.keyPress(KeyEvent.VK_LEFT);
- control.keyRelease(KeyEvent.VK_LEFT)

Las cuales permiten interactuar con el teclado, pulsando el cursor izquierdo.

4.6.2.3 Control del aumento de zoom de las diapositivas

Si el evento zoom es igual a uno (1) y si el descriptor para medir el flujo óptico es positivo se ejecutarán las líneas de código:

- control.keyPress(KeyEvent.VK_CONTROL)
- control.keyPress(107)
- control.keyRelease(107)
- control.keyRelease(KeyEvent.VK_CONTROL)

Las cuales permiten interactuar con el teclado, pulsando la teclas (CTRL +)

4.6.2.4 Control de la reducción zoom de las diapositivas

Si el evento zoom es igual a uno (1) y si el descriptor para medir el flujo óptico es negativo se ejecutarán las líneas de código:

- control.keyPress(KeyEvent.VK_CONTROL)
- control.keyPress(109)
- control.keyRelease(109)
- control.keyRelease(KeyEvent.VK_CONTROL)

Las cuales permiten interactuar con el teclado, pulsando las teclas (CTRL -)

4.7 LED INDICADORES

Este bloque permite al usuario saber qué está sucediendo en la aplicación; ya que identifica los estados que contienen los eventos para interactuar con el PC.

Tres de los seis eventos son dirigidos al bloque de control de eventos, los otros tres eventos restantes son las salidas a los LED INDICADORES de los estados de inicio, stop y espera de diapositivas.

Estas salidas entran a interactuar con un hardware de adquisición y transmisión de datos llamados Arduino.

Se usó para la adquisición y transmisión de datos el Arduino mega 2560, ya que este tiene una velocidad de procesamiento de 16MHz, velocidad necesaria para el tiempo de muestra del Kinect.

Al evento LEDGO se le asignó los puertos digitales 12 y 11 del Arduino mega 2560.

Al evento LEDSTOP se le asignó los puertos digitales 10 y 9 del Arduino mega 2560.

Al evento LEDINICIO se le asignó los puertos digitales 8 y 7 del Arduino mega 2560.

La Figura 4.28 muestra que la segmentación del centro del cuerpo permite identificar si la persona se encuentra dentro del área de trabajo o no. A este dato de salida se le asignó el puerto digital 5 y 6 del Arduino mega 2560.

La figura 4.47 muestra la distribución de los LED indicadores.

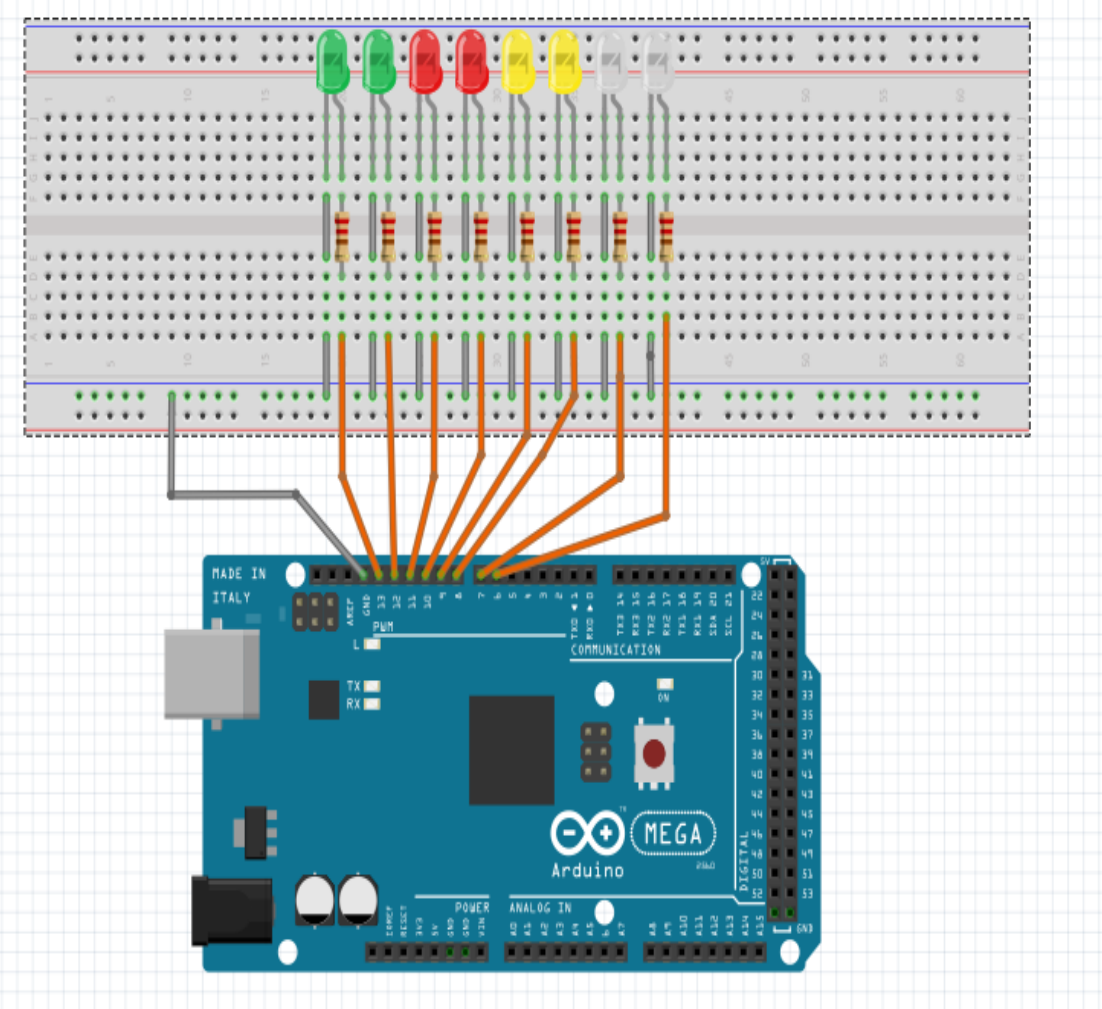

Figura 4.47. Distribución de los LED indicadores

5. ANÁLISIS DE RESULTADOS

Continuando con la metodología propuesta para el desarrollo de la aplicación, en este capítulo se analiza lo siguiente: todos los resultados obtenidos en el diseño, el cumplimiento de los requerimientos, el funcionamiento y los problemas que tiene la aplicación.

5.1 ANÁLISIS DEL USO DE KINECT EN MATLAB-SIMULINK

La mayoría de aplicaciones de Kinect se las realiza en lenguajes de programación C y Java, por esta razón existe una gran variedad de aplicaciones para el seguimiento del cuerpo humano y para el reconocimiento de movimientos. Se desarrolló esta aplicación de Kinect en Matlab-Simulink con el objetivo de interactuar el Hardware Kinect con la plataforma Matlab –Simulink, que es un software que se enseña en la carrera de mecatrónica de la Universidad Tecnológica Equinoccial.

Las ventajas y desventajas de usar Kinect en Matlab-Simulink para el desarrollo de aplicaciones se presenta en la Tabla 5.1.

Tabla 5.1. Ventajas y Desventajas del uso de Kinect en la plataforma Matlab-Simulink para el desarrollo de aplicaciones.

Ventajas	Desventajas
• Facilidad de programación • Permite la visualización del comportamiento de la señales • Permite la visualización resultados numéricos • Fácil visualización de video • Permite escoger el modelo de simulación • Permite el uso de máquina de estados	• Las aplicaciones deben usar muchos programas en su desarrollo

El cuadro de ventajas de la Tabla 5.1 determina que el uso de Kinect en Matlab-Simulink para el desarrollo de aplicaciones es de fácil programación, debido a que en Simulink se usó una programación por bloques. Además durante el proceso de desarrollo se necesitó algunos visualizadores, se utilizó varios bloques Scope para observar el comportamiento de las señales digitales de los movimientos realizados por el usuario. Se utilizó varios

bloques Display para observar los resultados numéricos específicos como la distancia de profundidad o las coordenadas de las posiciones de las articulaciones. Varios bloques Viewer para visualizar el video que se obtiene de la adquisición de imágenes. El uso de Pseudocódigo para una lógica combinacional de eventos es muy complejo en lenguajes C y Java, Simulink permite el uso de StateFlow que reduce pseudocódigo a través de diagramas de flujo, transiciones y estados.

La Tabla 5.1 determina como desventaja la instalación y vinculación de algunos programas extras a la plataforma Matlab-Simulink. Estos programas serán determinados según la función que realice la aplicación. Para esta aplicación se instaló y vinculó tres programas a la plataforma Matlab-Simulink los cuales son: Computer Vision System Toolbox, Microsoft Visual Studio y Microsoft Windows SDK.

5.2 ANÁLISIS DE LOS REQUERIMIENTOS DE LA APLICACION

Los requerimientos de la aplicación son mencionados en el capítulo de Metodología de esta tesis. Estos requerimientos se dividen en: requerimientos del control de eventos, requerimientos de los datos característicos de la aplicación y requerimientos dimensionales.

5.2.1 ANÁLISIS DE LOS REQUERIMIENTOS DEL CONTROL DE EVENTOS A TRAVÉS DE LA NUI

Este análisis determina cómo se logró controlar los eventos que se muestran en la Figura 3.3 del capítulo de Metodología. La Tabla 5.2 muestra como cada movimiento de la Figura 3.3 es calculado, luego se usa cada movimiento calculado en una transición de estado en el StateFlow y cada estado controla un evento. De esta forma se logra analizar que cada evento, tuvo un cálculo para determinar su movimiento. Se asignó una condición para su funcionamiento y tiene un estado que controla el evento.

Movimientos	**Fórmulas para calcular el movimiento**	**Condición de transición del StateFlow para que el evento ocurra**	**Eventos**	**Estado del StateFlow**
Manos Adelante Abriéndose	$dSm = \lvert(mdX - miX) - Z^{-1}(Z^{-1}(mdX - miX))\rvert$ $dSm > 0$	La altura de las 2 manos debe ser mayor a la altura d la ubicación Kinect	Control de aumento de zoom	Control zoom
Manos Adelante Cerrándose	$dSm = \lvert(mdX - miX) - Z^{-1}(Z^{-1}(mdX - miX))\rvert$ $dSm < 0$	La altura de las 2 manos debe ser mayor a la altura d la ubicación Kinect	Control de reducción de zoom	Control zoom
Mano Derecha Extendida	$d = \lvert(coordenada\ X\ centro\ del\ cuerpo) - (coordenada\ X\ mano\ derecha)\rvert$ $\left(1.65 * (dmax - (\frac{1}{n} * \sum_{i=1}^{n} di))^2\right) + \left(\frac{1}{n} * \sum_{i=1}^{n} di\right) = d.optimo$ $d \geq d.optimo$	Después de 1 segundo de activado este estado **Y** el brazo derecho arriba extendido horizontalmente debe ser mayor o igual al parámetro de brazo derecho arriba extendido horizontalmente **Y** la Altura del brazo derecho debe ser mayor a la altura de ubicación del Kinect **Y** La coordenada en (y) de la mano izquierda debe estar a 10 cm por debajo de la coordenada en (y) del centro del cuerpo	Control de avance de la presentación	Avance de Diapositiva
Mano Izquierda Extendida	$d = \lvert(coordenada\ X\ centro\ del\ cuerpo) - (coordenada\ X\ mano\ derecha)\rvert$ $\left(1.65 * (dmax - (\frac{1}{n} * \sum_{i=1}^{n} di))^2\right) + \left(\frac{1}{n} * \sum_{i=1}^{n} di\right) = d.optimo$ $d \geq d.optimo$	Después de 1 segundo de activado este estado **Y** el brazo izquierdo arriba extendido horizontalmente debe ser mayor o igual al parámetro del brazo izquierdo arriba extendido horizontalmente **Y** la Altura del brazo izquierdo debe ser mayor a la altura de ubicación del Kinect **Y** La coordenada en (y) de la mano derecha debe estar a 10 cm por debajo de la coordenada en (y) del centro del cuerpo	Control de regreso de la presentación	Regreso de diapositiva

5.2.2 ANÁLISIS DE LOS REQUERIMIENTOS DE LOS DATOS CARACTERÍSTICOS DE LA APLICACIÓN

Los datos característicos son la información que describe a cada uno de los patrones, para este caso los patrones son las coordenadas de las articulaciones de interés que realizan los movimientos predeterminados para el control de la aplicación.

En las etapas de segmentación y descripción del proceso de visión artificial se determinan estos datos característicos; los requerimientos y la evaluación de estos se muestra en la Tabla 5.3.

Tabla 5.3. Análisis de los requerimientos de los datos característicos

Requerimiento	**Evaluación del requerimiento**
Discriminación	Cada articulación usada posee diferente marcador en el vector característico
Fiabilidad	Determinado por la precisión del Kinect
Incorrelación	Para describir el movimiento de las articulaciones se usó diferentes algoritmos comparativos
Dimensionalidad	La dimensión del vector característico de [1x10x3], se reduce a [1x1x3] al encontrar las coordenadas de las articulaciones de interés

5.2.3 ANÁLISIS DIMENSIONALES

Se tomó datos de las distancias entre el proyector y el lugar de exposición en las aulas del bloque B y G y se realizó la Tabla 4.7, en esta tabla se determinó la distancia media y se observó si esta medida estaba en el rango de visión del Kinect en modo lejano. La distancia cumplía con el rango de visión y se la escogió para ser la medida entre el proyector y el lugar de exposición, la medida fue de 2.8m.

Posteriormente se realizó los cálculos para determinar la altura máxima de una persona para ser detectada por la aplicación, la distancia máxima que una persona puede desplazarse horizontalmente hacia la derecha o hacia la izquierda mientras es detectada por la aplicación y el área donde una persona puede moverse y es detectada por la aplicación. Tomando como constante para estos cálculos la distancia de 2.8 m como la distancia

máxima entre el proyector y el lugar de exposición como lo muestra la Tabla 5.4.

Tabla 5.4. Análisis de los requisitos dimensionales

Rangos	Ángulos	Eje de los ángulos	Formula	Resultados
Altura máxima de la aplicación	43.5º	Vertical (Z,Y)	$\tan \emptyset * Z = Y2$ $Altura\ Máx = Y1 + Y2$	2.117 m
Distancia máxima horizontal	57.5º	Horizontal (X,Z)	$\tan \alpha * Z = X1$ $Dist.max.h. = X1 + X2$	3.072 m
Área de trabajo	57.5º	Horizontal (X,Z)	$\frac{(((\tan \alpha * Z) * 2) + 3.072) * (2.8 - 1.2)}{2}$	3.511 m^2

El requisito de altura máxima era (1.50 a 2) m, la aplicación cumplió el requisito debido a que puede sensar hasta 2.117 m como máximo y 1.333m como mínima altura.

5.3 ANÁLISIS DE LA INFERENCIA DE POSICIONES

La inferencia de posiciones hace referencia a la deducción intuitiva que tiene la aplicación para determinar las posiciones. Esta inferencia se lleva a cabo mediante un filtrado en conjunto de ciertos parámetros para suavizar la imagen, mejorar su estabilidad y estabilizar las posiciones comunes en el tiempo.

Estos parámetros se los mencionó brevemente en el capítulo cuatro en el tema de parámetros SDK del bloque IMAQ de la librería NID. Estos parámetros proporcionan un mecanismo para suavizar y estabilizar las posiciones conjuntas en una imagen. El Kinect puede ajustar la inferencia de las posiciones de acuerdo al comportamiento de estos parámetros. Estos parámetros combinados crean el filtro conjunto para suavizar y estabilizar las posiciones, este filtro utiliza el método de Holt (doble exponencial) utilizado para el análisis estadístico de datos económicos. Este algoritmo proporciona menor latencia que otros algoritmos de suavizado. La latencia puede ser definida como el tiempo que transcurre desde cuando una persona hace un

movimiento, hasta el momento en que la persona ve la respuesta a su movimiento del cuerpo en la pantalla. La latencia degrada la experiencia tan pronto como la gente comienza a darse cuenta de que hay un retraso en la respuesta a sus movimientos.

La investigación realizada por Microsoft Developer Network muestra que el 72% de la gente comienza a darse cuenta de esta demora cuando la latencia es más de 100 ms, y por lo tanto, se sugiere que los desarrolladores apuntan a una latencia total de 100 ms o menos. (Microsoft, 2013)

Los parámetros que determinan el filtro conjunto con sus respuestas en el sistema de visión artificial lo muestra la Tabla 5.5.

Tabla 5.5. Análisis de los parámetros del filtro conjunto

Parámetro	**Descripción**	**Respuesta**	**Rango**
Suavizado	Reduce la cantidad de variaciones que existen entre pixeles vecinos.	A mayor suavizado mayor latencia.	[0,1]
Corrección	Reduce la variación que existe al tomar el valor de la posición en cada frame.	A mayor corrección mayor suavizado.	[0,1]
Predicción	Determina una razón del número de fotogramas para predecir el futuro	Debe ser mayor a 0 o igual a 0. Valores mayores a 0.5 puede ocasionar un sobre muestreo.	[0,1]
Radio Jitter	Determina en metros la desviación que existe entre dos momentos de efecto máximos o mínimos.	-	-
Max. Desviación del Radio	Determina el radio máximo en metros que puede filtrar la desviación de dos momentos de efecto máximos o mínimos.	-	-

La descripción de cada parámetro como su respuesta en el sistema de visión artificial permitió realizar pruebas cambiando los valores de los parámetros para determinar el mejor filtro conjunto para la detección y reconocimiento de los movimientos como lo muestra la Tabla 5.6.

Tabla 5.6. Resultados de la inferencia de posiciones con diferentes parámetros del filtro conjunto

Número de filtro	Suavizado	Corrección	Predicción	Radio Jitter	Max. Desv. Radio	Resultado
1	0.5	0.5	0.5	0.05	0.04	Poco suavizado y muy baja latencia, bueno para sistemas con reconocimiento de movimientos.
2	0.5	0.1	0.5	0.1	0.1	Suavizado con latencia alta, bueno para sistemas de selección con detección baja de movimientos.
3	0.7	0.3	1	1	1	Mucho suavizado con una gran latencia, bueno para sistemas de precisión donde se necesite detectar un movimiento durante un largo tiempo.

El mejor filtro para la detección de movimientos es el número 1 debido a que tiene poco suavizado y baja latencia para que pueda seguir los movimientos del cuerpo humano.

5.4 ANÁLISIS DEL PROCESAMIENTO DE IMÁGENES

5.4.1 PROBLEMAS CON HARDWARE

Un sistema típico de visión artificial, además de un dispositivo de captura, cuenta con al menos otros cuatro elementos: un dispositivo de conversión de analógico a digital (A/D), una memoria de video, un elemento de procesamiento y un monitor. Por este motivo el procesamiento de imágenes o video es una tarea pesada dentro del PC debido a que se utilizan muchos recursos para capturar los fotogramas de vídeo en forma análoga, convertirlos en señales digitales, almacenarlos, transmitirlos, procesarlos y visualizarlos en forma digital comprimida.

Los microchips de procesamiento grafico permiten visualizar las imágenes, mostrar aplicaciones, jugar video juegos de baja resolución, interactuar con software de diseño mecánico, eléctrico, visual, publicitario, simular procesos de manufactura, entre otros. Pero la funcionalidad de estos microchips no está dedicada a la aceleración de procesamiento de imágenes o video. Por este motivo surge el problema de hardware con la aplicación. Como se muestra en las Figura 5.1. y 5.2.

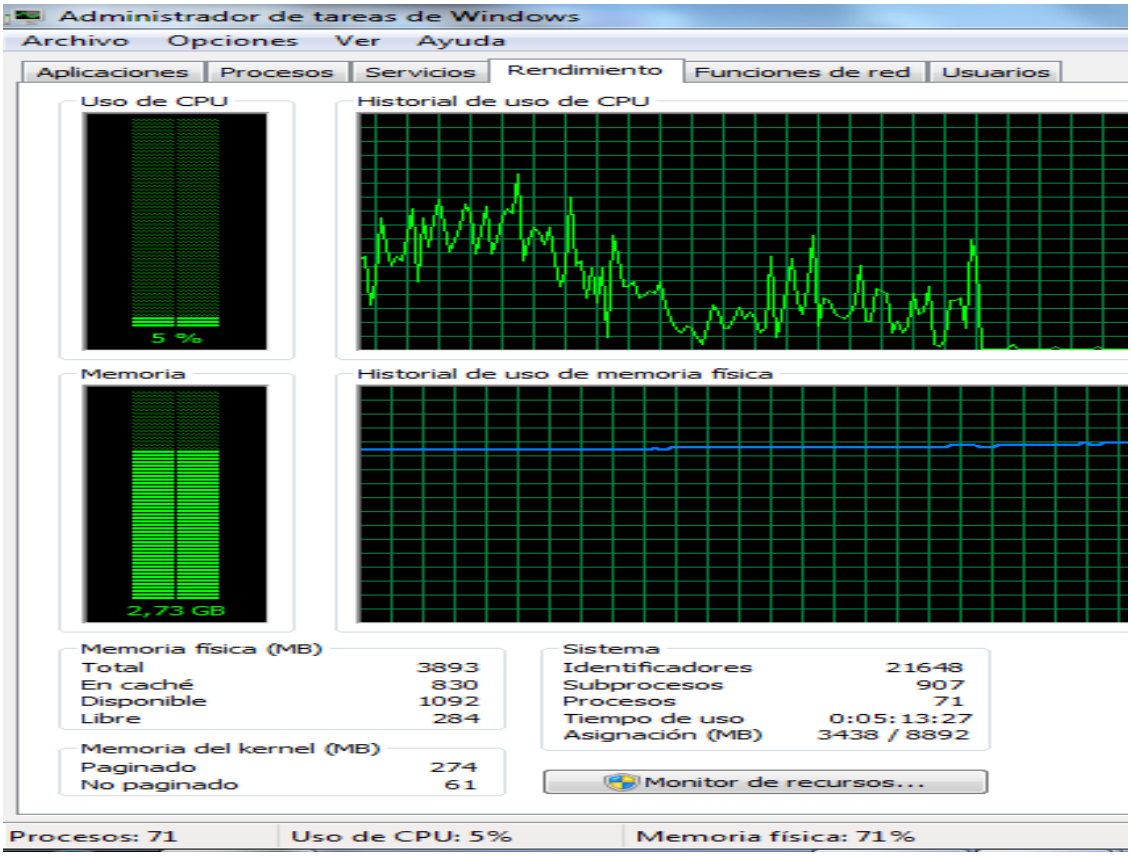

Figura 5.1. Rendimiento del PC sin la aplicación.

En la Figura 5.1 muestra el rendimiento normal del PC, sin el uso de la aplicación, donde se puede observar que el uso de la memoria RAM es de un 5% y tiene una memoria física libre de 284 MB.

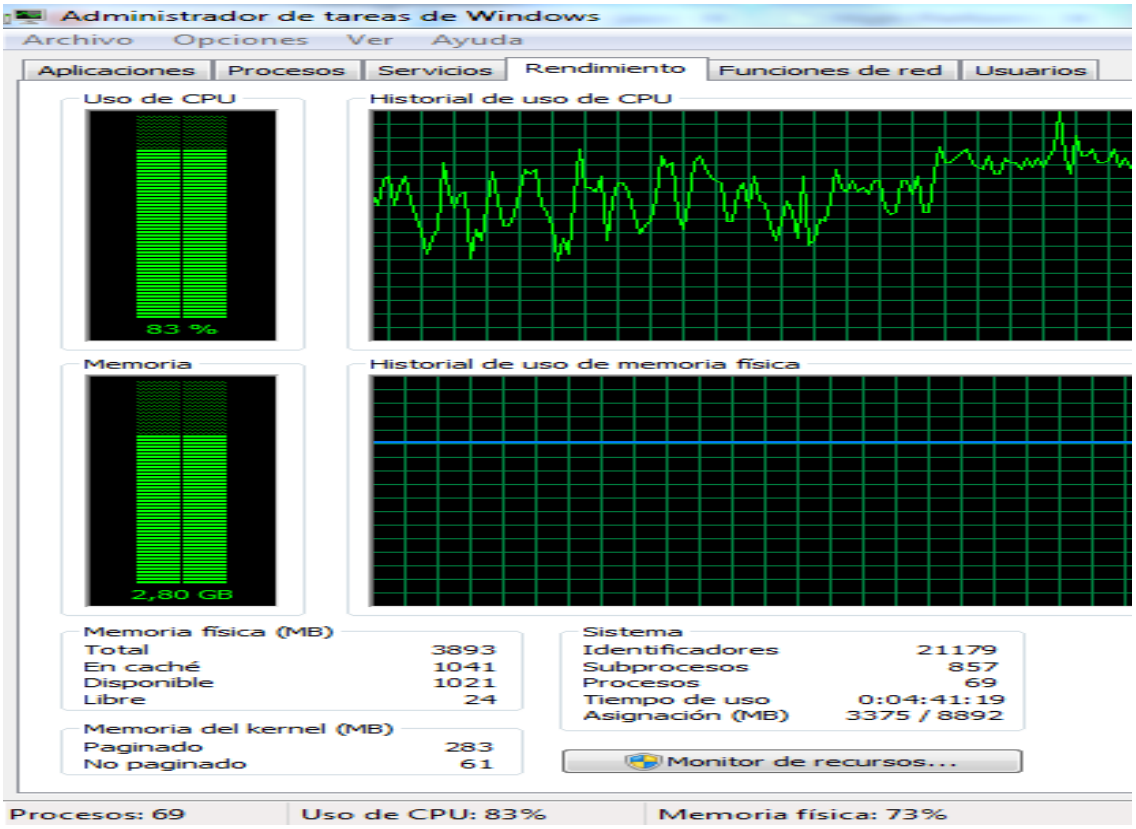

Figura 5.2. Rendimiento del PC con la aplicación.

La figura 5.2 muestra el rendimiento del PC con el uso de la aplicación, donde se puede observar que el uso de la memoria RAM es de 83 % y tiene una memoria física libre de 24 MB.

Estos resultados indican que existe un considerable incremento del uso de la memoria de RAM al usar la aplicación y este es su principal problema.

El menú de información y detalle de rendimiento del PC, proporciono la información del componente del PC con menor puntuación, el cual es la GPU del PC como lo muestra la Figura 5.3.

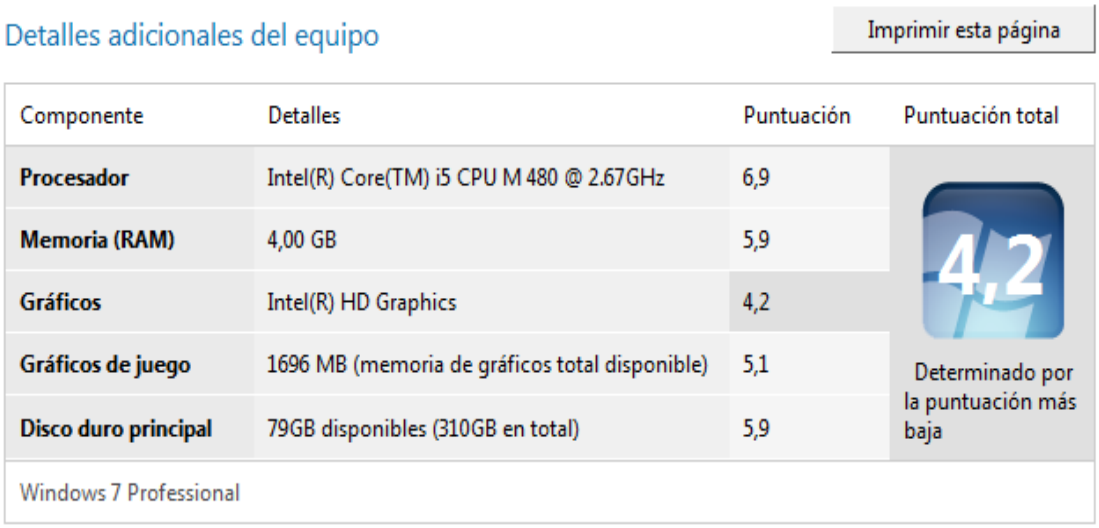

Figura 5.3. Menú de información y rendimiento del PC

La GPU usada en la aplicación es la Intel HD Graphics 3000, esta GPU no permite la aceleración de procesamiento de imágenes que necesita la aplicación y por esta razón el PC se apaga después de 10 minutos usando la aplicación, el PC al apagarse protege la memoria RAM y sus recursos.

Se comparó la GPU usada en la aplicación, versus las GPU de nueva tecnología y se concluyó que nuestra GPU no es la adecuada, ni es apta para realizar esta aplicación, debido a que no tiene la suficiente capacidad de procesamiento, ni memoria para realizar los demandantes cálculos y eventos que necesita la aplicación. Como lo muestran las Figuras 5.4. y 5.5.

Para las Figuras 5.4 y 5.5 cada columna representa información de las GPU que se detalla.

1. Ranking
2. Modelo
3. Arquitectura
4. Pixel shaders
5. Vertex shaders
6. Velocidad del núcleo
7. Velocidad del shader
8. Velocidad de memoria

9. Bus de memoria
10. DirectX
11. 3D Mark Ice Strom GPU
12. 3D Mark11P GPU
13. 3DM Vant. P GPU
14. 3D Mark06

367*	AMD Radeon HD 7430M	Terascale 2	160	600	600	1800	64	11				
370	Intel HD Graphics (Haswell)	Haswell	10	200 - 1000				11.1	30194 [2]	443 [2]	1859 [2]	3357.5 [2]
373	AMD Radeon HD 8240	GCN	128	400	400			11.1	25296	429	1247	2636
374	AMD Radeon HD 8250	GCN	128	300 - 400	300 - 400			11.1	19669 [2]	422.5 [2]	1112.5 [2]	2570 [2]
377*	ATI Mobility Radeon HD 5450	Terascale 2	80	675	675	800	64	11				3775.5 [2]
380	Intel HD Graphics 3000	Sandy Bridge	12	350-1350	350-1350			10.1	27124 [4]		1240 [89]	3286.5 [112]
385	NVIDIA GeForce 405M	GT2xx	16	606	1212	1600	64	10.1				
395*	AMD Radeon HD 6430M	Terascale 2	160	480	480	800	64	11				
396*	AMD Radeon HD 6380G	Terascale 2	160	400	400			11				2639
397	ATI Mobility Radeon HD 5430	Terascale 2	80	550	550	800	64	11			750	3082
398	AMD Radeon HD 8210	GCN	128	300	300			11.1	20119.5 [6]	345 [5]	912 [3]	2177.5 [4]
401	Intel HD Graphics 2500	Ivy Bridge	6	650 -	650 -			11.0	24058 [4]	350.5 [4]	1400.5 [4]	3816 [5]

Figura 5.4. Tabla de análisis del GPU usado en la aplicación
(NVIDIA, 2013)

1. High-End Graphics Cards

These graphics cards are able to play the latest and most demanding games in high resolutions and full detail settings with enabled Anti-Aliasing.

1	NVIDIA GeForce GTX 880M SLI	Kepler	3072	954		5000	256	11	316040	15823	42967	
2	NVIDIA GeForce GTX 780M SLI	Kepler	3072	823	823	5000	256	11	333674	14139.5 [2]	43525 [2]	27957
3	AMD Radeon R9 M290X Crossfire	GCN	2560	850 - 900	850 - 900	4800	256	11.2 (Tier 1)	284150	14147	37644	
4	AMD Radeon HD 8970M Crossfire	GCN	2560	850 - 900	850 - 900	4800	256	11.1	327560 [2]	12392.5 [2]	39191 [2]	26430 [2]
5	NVIDIA GeForce GTX 680M SLI	Kepler	2688	720	720	3600	256	11		10952 [2]	36213.5 [2]	26983.5 [2]
6	AMD Radeon HD 7970M Crossfire	GCN	2560	850	850	4800	256	11.1		11119 [2]	35208.5 [2]	27647 [2]
11	NVIDIA GeForce GTX 770M SLI	Kepler	1920	811	811	4000	192	11				

13	NVIDIA GeForce GTX 880M	Kepler	1536	954		5000	256	11	184375 [2]	8451 [2]	29698	
16	NVIDIA GeForce GTX 780M	Kepler	1536	823	823	5000	256	11	118950 [7]	7776.5 [8]	27258 [8]	23381.5 [6]
17	NVIDIA Quadro K5100M	Kepler	1536	771	771	3600	256	11	137763	7148	24457	25193
18	NVIDIA GeForce GTX 680MX	Kepler	1536	720	720	5000	256	11		6736	25270	
20	AMD Radeon R9 M290X	GCN	1280	850 - 900	850 - 900	4800	256	11.2 Tier 1	125311	6817	24205	
21	AMD Radeon HD 8970M	GCN	1280	850 - 900	850 - 900	4800	256	11.1	110127 [2]	6818 [3]	20876 [2]	18211.5 [2]
22*	NVIDIA GeForce GTX 870M	Kepler	1344	941		5000	192	11	187697	7270	24065	

Figura 5.5. Tabla de análisis del GPU más modernos.
(NVIDIA, 2013)

6. CONCLUSIONES Y RECOMENDACIONES

6.1 CONCLUSIONES

- Utilizando el dispositivo Kinect y los bloques de la librería NID de Kinect en Simulink se realizó una aplicación que permitió el seguimiento al cuerpo humano, el reconocimiento de sus movimientos y el control de cuatro (4) eventos específicos en el PC los cuales fueron avanzar de diapositiva, retroceder de diapositiva, aumentar zoom y disminuir zoom de una forma remota sin interacción directa del usuario con el PC.

- La revisión bibliográfica llevada a cabo permitió concluir que el dispositivo Kinect con la tecnología NUI de reconocimiento y seguimiento al cuerpo humano es el hardware más apto para el desarrollo de la aplicación, debido a que el usuario tiene acceso libre a su SDK (Software Development Kit) lo que permite usar sus librerías y configurar sus sensores de la mejor manera para el usuario.

- El entorno de simulación Simulink permitió integrar el dispositivo Kinect al PC a través de diferentes modelos de interacción con los bloques de la librería NID los cuales permitieron la adquisición, la segmentación, la descripción y el reconocimiento de los movimientos realizados por el usuario.

- La aplicación usó algoritmos comparativos de proporcionalidades del cuerpo humano que por medio de descriptores unidimensionales y descriptores de flujo óptico reconocieron los movimientos que el usuario realizaba con sus brazos, estos movimientos por medio de una lógica combinatoria se pudieron interpretar y relacionar con el control de un evento particular en el PC con cada movimiento, los movimientos fueron: brazo derecho arriba extendido horizontalmente con avance de diapositiva, brazo izquierdo arriba extendido

horizontalmente con retroceso de diapositiva, apertura de manos adelante con aumento de zoom y cierre de manos adelante con disminución de zoom.

- Como se analizó en la sección 5.1 el entorno de simulación Simulink tiene muchas ventajas con respecto a lenguajes de programación como C o JAVA al momento de programar, visualizar y simular aplicaciones para la detección e interpretación de movimientos del cuerpo humano esto disminuye el tiempo de desarrollo y vuelve a las aplicaciones más versátiles a cambios del usuario.

- Se determinó que el dispositivo Kinect funcionó de manera correcta al ser ubicado a 1.20m por encima del suelo teniendo una altura máxima de visión de 2.117 m y una distancia máxima de visión horizontal de 3.072 m por lo que el usuario tuvo un área total de trabajo de $3.511\ m^2$

- El filtro conjunto suavizó y estabilizó la inferencia de las posiciones en la aplicación, los parámetros del mejor filtro fueron: 0.5 en suavizado, 0.5 en corrección, 0.5 predicción, 0.05 en radio jitter y 0.04 en máxima desviación de radio, estos permitieron poco suavizado y muy baja latencia, bueno para sistemas con reconocimiento de movimientos debido a que permiten una respuesta inmediata del movimiento realizado con el evento en el PC, el tiempo de respuesta es de 100 ms.

- Se concluyó que la aplicación necesita una GPU Nvidia GeForce GTX 870M como mínimo para evitar el problema de apagado inesperado del PC, debido a que posee una GPU Intel HD Graphics 3000 que no permite la aceleración de procesamiento de imágenes.

6.2 RECOMENDACIONES

- Utilizar una GPU de alto rendimiento para cualquier aplicación que se realice para Kinect for Windows.
- Utilizar Matlab 2013 a o versiones superiores para aplicaciones que necesiten Kinect for Windows.
- Calibrar la cámara RGB del dispositivo Kinect a través de las App que poseen las nuevas versiones de Matlab.
- Revisar conceptos de geometría y parámetros de las cámaras para poder posicionar una cámara y orientarla en el espacio relativo con algún sistema de coordenadas.
- Se recomienda siempre tener una metodología clara de todas las etapas de lo que se va a realizar en un sistema de visión artificial.
- Realizar un cuadro estadístico porcentual de cuantos movimientos se identifica en un determinado tiempo variando el filtro conjunto.
- Investigar la manera de identificar gestos con Kinect 1.
- Se recomienda que se si realiza alguna identificación de colores a través de Kinect con Matlab se tome en cuenta que la luminosidad y la distancia son factores que modifica el proceso de identificación en visión artificial.
- Desarrollar la manera de embeber este código Matlab en alguna tarjeta de adquisición para innovar con un Hardware autónomo de identificación de movimientos con Kinect.
- Es recomendable que entre el Kinect y el usuario no exista ningún objeto que obstaculicé el campo de visión.
- Realizar un cálculo compensatorio integral al momento de convertir las coordenadas de pixeles a coordenadas en mm en el mundo real.
- Tomar en cuenta que la imagen es un sistema bidimensional y el mundo real es tridimensional, al trabajar en conjunto con pixeles y milímetros.

BIBLIOGRAFÍA

- Cuevas, E., Zaldivar, D., & Perez, M. (2010). Procesameinto Digital de imagenes con Matlab y Simulink. México: Alfaomega.
- Araujo, B. (2012). Parametrizacion de una protesis tumoral no convencional de hombro utilizando un enfoque sistematico. México DF, México.
- Cuevas, E., Zaldivar, D., & Perez, M. (2010). Procesameinto Digital de imagenes con Matlab y Simulink. México: Alfaomega.
- De Giusti, A., & Lanzarini, L. (2010). Reconocimiento de Patrones en Imágenes Médicas. Laboratorio de Investigación y Desarrollo en Informática, Universidad Nacional de La Plata, Argentina.
- Fan, J., Xu, W., Wu, Y., & Gong, Y. (208). Human Tracking Using Convolutional. University, Beijing,, China, Japon.
- Hernandez Toalla, L. A., & Herrera Rodrigez, J. D. (Enero de 2013). Analisis de los codigos fuente SDK. Quito, Pichinca, Ecuador.
- Hninn Maung, T. (2009). Real-Time Hand Tracking and Gesture. Department of Engineering Physics,, Mandalay Technological University,.
- K. J. Chen, E. A. (2011). MODELING HEAD TRACKING OF VISUAL TARGETS. Department of Biomedical Engineering, Northwestern University,, Evanston, Estados Unidos.
- Kauleshwar, P., Devvrat, N., Lakhotiya, A., & Umre, D. (2013). Character Recognition Using Matlab's Neural Network Toolbox. Durg, India.
- KinectForWindows. (2013). KinectForWindows. Obtenido de KinectForWindows: www.microsoft.com

- Microsoft, K. (2013). Human Interfaces Guidelines.
- Moore, H. (2010). MATLAB para Ingenieros. México: Prentice Hall.
- NVIDIA.(2013).Obtenido de:

  http://www.nvidia.com/content/global/global.php
- Pavon Serrano, L. (Septiembre de 2011). Virtual Blackboard: colour and. Caceres.
- Pelz, G. (2006). Sistemas Mecatronicos. México: Limusa Wiley.
- Pijares Martinsanz, G., & De la Cruz Garcia, J. (2008). Ejercicios de Vision Artificial Por Computadora. Madrid: Alfaomega.
- Pijares Martinsanz, G., & De la Cruz Garcia, J. (2008). Vision por Computadora. Madrid: Alfaomega.
- Pinto Bermudez, E., & Matia Espada, F. (2010). Fundamentos de control de MATLAB. Madrid: Pearson.
- Ponce Cruz, P. (2010). Inteligencia Artificial . México: Alfaomega.
- Sobrado, E. (2003). Sistemas de vision artifiical y manipulacion de objetos. Lima, Peru.
- Sucar , E. (2012). Visión Computacional. Puebla, México.
- Tang, M. (2011). Recognizing Hand Gestures with Microsoft's Kinect. Stanford University, Estados UnidosDepartment of Electrical Engineering.
- Torres , J. M. (2010). Reconocimiento gestual mediante técnicas. Universidad de Laguna, España.
- Walpole, R. E. (2007). PROBABILIDA Y ESTADISTICA.

- Walpole, R. E., Myers, R. H., Myers, S. L., & Ye, K. (2007). Probabilidad y Estadistica. México: Prentice Hall.
- Yuste Padilla, D. (Enero de 2012). Sistema de reconocimiento de gestos para un visor de imágenes. Cataluya, España.
- Alciatore, D. G., & Histand, M. B. (2008). Introduccion a la Mecatronica. México: Mc Graw Hill.