UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

PROGRAMA DE MAESTRÍA Y DOCTORADO EN INGENIERÍA
ELÉCTRICA – TELECOMUNICACIONES

DESARROLLO DE UN SISTEMA DE AUTENTICACIÓN PARA APLICACIONES DE SEGURIDAD CON EL USO DE UN SENSOR KINECT

TESIS
QUE PARA OPTAR POR EL GRADO DE:
MAESTRO EN INGENIERÍA

PRESENTA:
LUIS ALFONSO GARCÍA VÁZQUEZ

TUTORA PRINCIPAL
DRA. FATIMA MOUMTADI
FACULTAD DE INGENIERÍA

MÉXICO, D. F. NOVIEMBRE 2015

JURADO ASIGNADO:

Presidente:	Dr. Vicente Vivas Esau
Secretario:	Dr. Pérez Alcazar Pablo Roberto
Vocal:	Dra. Moumtadi Fatima
1 er. Suplente:	M. I. Escobar Salguero Larry
2 d o. Suplente:	Dr. Rivera Rivera Carlos

Ciudad Universitaria, México, D.F.

TUTOR DE TESIS:

Dra. Fatima Moumtadi

_____

AGRADECIMIENTOS

A la Universidad Nacional Autónoma de México por ser un pilar sólido en mi formación académica y por inculcarme altos valores humanos.

A los profesores por enseñarnos a pesar de las dificultades.

Con todo mi cariño a mis amigos, familiares y personas que quiero.

A mi tutora por la motivación, el apoyo y el tiempo que me brindó.

A mis sinodales por sus valiosas observaciones.

Al Conacyt que con su apoyo y respaldo hizo posible la realización de este trabajo.

A mi país del cual siempre estaré orgulloso.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

RESUMEN

En este trabajo se desarrolló un sistema de autenticación multifactor utilizando el sensor Kinect y equipo de cómputo. Se utilizó programación en lenguaje C# con el ambiente de desarrollo y las herramientas que provee el fabricante para el sistema operativo Windows. Se eligió una combinación de cinco métodos para conseguir la autenticación multifactor, con el fin de reducir la capacidad que tiene un usuario no autorizado de ser elegible para tener acceso a un determinado recurso o lugar, cubriendo las tres categorías de métodos de autenticación: llaves de información, llaves biométricas y llaves físicas, basadas respectivamente en algo que la persona sabe, es y posee. Se pudo desarrollar un sistema de autenticación confiable, robusto y sencillo de usar, privilegiando la confiabilidad y disminuyendo la complejidad de cada uno de los métodos individuales. Se demostró que es posible desarrollar un sistema de autenticación multifactor con un sensor Kinect.

Palabras clave: Autenticación, Kinect, Multifactor

INTRODUCCIÓN

Las tecnologías de la información y la comunicación cada vez son más relevantes en la vida cotidiana, ya que las computadoras se han vuelto de uso común. Con todas las nuevas implementaciones a través de las computadoras, que continúan en aumento con el paso del tiempo, también hay una necesidad cada vez mayor de garantizar su funcionamiento libre de errores no intencionales, su operación adecuada por los usuarios, así como evitar su uso indebido.

Gestionar la infraestructura de las tecnologías de la información y la comunicación de alguna organización requiere de seguridad. Así que cada vez es más importante contar con sistemas que garanticen la seguridad de los sistemas computacionales que a su vez pueden resguardar lugares o recursos. La autenticación permite identificar si una persona es elegible para tener acceso a un lugar o recurso. Para esto existen tres categorías de métodos distintos de autenticación: Llaves de información (frases, contraseñas, cuestionarios), llaves físicas (objetos necesarios para acceder al sistema) y llaves biométricas (huellas digitales, geometría de la mano, impresiones de voz y de retina) [1]. Utilizar una combinación de varios métodos, ya sea del mismo tipo o de diferentes tipos, incrementa la solidez de la autenticación [2].

El avance en la tecnología ha permitido el desarrollo de nuevos tipos de sensores y también ha disminuido su costo, lo que ha abierto la posibilidad al desarrollo de diversos sistemas de autenticación. Entre los nuevos dispositivos disponibles se encuentra el sensor Kinect que surgió como un dispositivo para controlar juegos de video sin manos, así como el control mediante interfaces naturales de usuario para interaccionar con un sistema sin necesidad de mandos o dispositivos de entrada físicos. El sensor Kinect, introducido en el año 2010, permite el desarrollo de nuevos sistemas de autenticación utilizando varios métodos debido a su capacidad de seguir los movimientos del cuerpo humano, medir distancias y el reconocimiento de voz entre otras capacidades. Así que es posible utilizar el sensor Kinect para el desarrollo de sistemas de autenticación en los que se sigue el movimiento del cuerpo humano, se reconoce la voz y se envían notificaciones como confirmaciones [3].

Una de las ventajas de utilizar el sensor Kinect para sistemas de autenticación es que se tiene gran flexibilidad en las formas de ingresar la información, contribuyendo a la diversidad de métodos y tipos de sistemas de autenticación posibles. El abanico de posibilidades que ofrece es enorme y apenas comienza el desarrollo de su verdadero potencial.

ANTECEDENTES

Se han realizado diversos estudios sobre el uso del sensor Kinect para fines de reconocimiento, así como para la detección de las posiciones del cuerpo humano con diversas aplicaciones en áreas como la robótica, medicina, educación, seguridad, arquitectura, entre otras. Para el desarrollo de este sistema, son relevantes los avances que están relacionados con la seguridad y la detección de los movimientos del cuerpo humano utilizando el sensor Kinect.

Vinculado con el uso del sensor Kinect para aplicaciones que requieran la detección de movimiento del cuerpo humano, existe una solución para evitar que una persona no autorizada pase a través de una puerta segura utilizando la detección del número de personas que pasan por la puerta mientras permanece abierta [4]. También hay una discusión sobre cómo podría detectar un robot si un intruso se está intentando rendir así como sobre la detección de intrusos potenciales [5]. Por otro lado, existe una revisión del mundo en torno al sensor Kinect estableciendo las directrices para desarrollar aplicaciones basadas en el seguimiento de movimientos con este dispositivo [6].

En relación a la utilización del sensor Kinect para aplicaciones de autenticación con el uso de contraseñas, se plantea la creación de un gesto patrón que actúe como contraseña, de forma que el sistema de reconocimiento de gesto de mano funcione robustamente a pesar de las variaciones en la orientación de la mano, la escala o la articulación [7]. En la misma línea, se explora la autenticación de usuarios basada en gestos capturados por el sensor Kinect [8]. También en autenticación utilizando los gestos, hay una investigación sobre las ganancias en desempeño y robustez para la autenticación de usuarios basada en gestos utilizando varios sensores Kinect [9].

En autenticación biométrica, existe una exploración sobre la capacidad de reconocer la presencia de un humano y estimar sus dimensiones utilizando una cámara, efectuando una autenticación biométrica con una solución que usa el sensor Kinect para este fin [10]. También se ha considerado la viabilidad de utilizar el sensor Kinect para obtener puntos de las posiciones del esqueleto de sujetos caminando y usar estos puntos para identificación biométrica considerando la contribución de diferentes combinaciones de partes del cuerpo humano al proceso de identificación [11]. En otro caso biométrico, se propone el uso del Kinect pero en lugar de acomodar a un amplio número de usuarios se explota la unicidad de cada usuario en términos de gestos [12]. Para identificar personas de forma discreta y a distancia de forma biométrica, se construyeron y evaluaron

conjuntos de características y partes del cuerpo humano en el proceso de reconocimiento de personas a partir de su forma de caminar utilizando los datos esqueléticos obtenidos con el sensor Kinect [13].

Un sistema más relacionado con el objetivo del presente trabajo es el que permite una identificación basada en el cuerpo humano, que utiliza las diferencias individuales en longitudes de segmentos del cuerpo humano y patrones de gestos de movimientos de manos, es decir, utiliza llave biométrica y llave de información [14].

El sistema de autenticación desarrollado en este trabajo con el uso del sensor Kinect, se llevó a cabo desde una nueva perspectiva y enfoque; se cubrieron los tres tipos de métodos de autenticación, y cada uno de los métodos seleccionados es sencillo pero complementario a los demás, para lograr una autenticación multifactor.

OBJETIVO GENERAL

Desarrollar un sistema de autenticación multifactor de individuos para aplicaciones de seguridad utilizando programación en lenguaje C# y un sensor Kinect, así como el ambiente de desarrollo y las interfaces de programación de aplicaciones, a través de las posiciones del cuerpo y el reconocimiento de voz.

Objetivos particulares:

- Utilizar una combinación de diferentes métodos de autenticación
- Contemplar un escenario en el que se requiera a los individuos validar su autorización para acceder al sistema o lugar y que informe la negación o permiso de acceso
- Aprovechar los atributos físicos del individuo, así como la universalidad de los movimientos permitidos por las articulaciones del cuerpo humano, con el fin de simplificar su uso y añadir seguridad al requerir la presencia física del individuo y evitar el conocimiento previo del funcionamiento del sistema a individuos ajenos o intrusos

DEFINICIÓN DEL PROBLEMA

En el mundo se necesita seguridad en diversos campos y a través de diferentes sistemas que cada vez más están basados en equipos de cómputo. Estos equipos pueden ser usados en aplicaciones críticas, como la protección de lugares o recursos de gran importancia.

El enfoque de este trabajo se encuentra en un elemento fundamental para prevenir el acceso de intrusos, que es desarrollar un sistema de autenticación multifactor de usuarios. La autenticación de usuarios es un componente fundamental de un modelo de seguridad, ya que comprueba la identidad de un usuario, y por lo tanto tiene gran importancia en la defensa de determinado lugar o recurso.

Existen muchas áreas, como las de control de acceso de usuarios a un lugar, que no requieren que el equipo se encuentre conectado a una red así como tampoco requieren que el usuario tenga contacto físico con el sistema de autenticación. El uso de interfaces o dispositivos de entrada convencionales, como el teclado o el ratón, permiten controlar aspectos de los equipos computacionales que no se limitan a la única función a la que estén destinados. Este control de aspectos ajenos a su función asignada, representa una vulnerabilidad cuando están destinados a evitar el acceso no autorizado. Así que este tipo de dispositivos de entrada representan inherentemente una vulnerabilidad para este tipo de sistemas.

Por otro lado, los sistemas computacionales utilizan sistemas operativos complejos que involucran una gran cantidad de software y resultan prácticamente imposibles de realizar sin cometer errores. Aunque existen áreas en las que los errores no son tan importantes; para la seguridad, la presencia de un solo error puede vulnerar al sistema si el intruso consigue acceso.

En el mercado se encuentran disponibles diversos sistemas que utilizan métodos de autenticación que han conseguido establecerse como estándares o de aplicación en una escala masiva. Esto representa una ventaja, al ser sistemas que se han ido mejorando con el paso del tiempo y que cada vez son mejores, más rápidos, más accesibles y de mayor disponibilidad. Sin embargo, también es esta popularidad la que los hace más vulnerables, debido a que los intrusos han desarrollado paralelamente formas de vulnerarlos. Por ejemplo, se han encontrado formas de vulnerar los sistemas de reconocimiento de huella digital y con ello de todos los lugares y recursos que protegen, que aumentan con el incremento en popularidad de esta forma de autenticación.

Lo anterior hace necesarios nuevos sistemas de autenticación multifactor que operen de forma eficiente, sencilla, y que sean accesibles. Los sistemas de autenticación actuales tienen vulnerabilidades en ciertos escenarios, que pueden ser subsanadas con el desarrollo de nuevos sistemas. El sistema de autenticación que se desarrolló es novedoso, multifactor, no requiere contacto físico del usuario, no es vulnerable inherentemente debido al uso de una interface estándar común, evita el control del sistema sin que esto impida su funcionamiento, es robusto a errores siempre presentes en los sistemas operativos ya que sólo permite ingresar la información requerida para su funcionamiento. Una de las grandes ventajas del sistema propuesto, además de ser de alto nivel de seguridad, es que tiene grandes posibilidades de desarrollo a futuro tanto en hardware como en software, con la posibilidad de agregar funciones adicionales para mejorar el nivel de seguridad y ampliar su funcionamiento.

METODOLOGÍA

Para evaluar el sensor Kinect es necesario hacer pruebas y analizar sus características técnicas así como sus limitaciones. En pruebas iniciales se verifica que sus diversos componentes y funciones, operan de forma adecuada para poder continuar con el desarrollo del sistema. En esta fase del desarrollo se realizan observaciones y se comienza a bosquejar las posibilidades del sensor Kinect para ser usado en un sistema de autenticación. Algunas características importantes para su funcionamiento correcto son la distancia a la que tiene que encontrarse la persona para ser detectada correctamente, la altura a la que se tiene que encontrar el sensor y los ángulos para medir los diversos parámetros a utilizar. La realización de esta evaluación del sensor Kinect es fundamental para su utilización en la implementación concreta de autenticación de usuarios, ya que conocer sus limitaciones, permite descartar o tomar en cuenta estas como causas de posibles problemas en el funcionamiento del sistema.

Algunos puntos clave para el diseño, implementación y ajustes finales se enlistan a continuación:

1. Obtención del software necesario, preparación y habilitación de los diferentes elementos que componen el sistema: Equipo de cómputo y sensor Kinect.
2. Evaluación de las limitaciones de los diferentes componentes del sistema y pruebas para conocer su funcionamiento adecuado. Elección de un espacio adecuado para su instalación.

3. Selección de los métodos de autenticación que serán utilizados.

4. Comenzar el desarrollo del sistema por partes, por medio de la creación de varios códigos, utilizando el lenguaje de programación C#, con el fin de realizar pruebas sobre cada método seleccionado.

5. Programar una aplicación que integre los métodos de autenticación seleccionados tomando en cuenta la interfaz de usuario y el correcto funcionamiento de los diferentes algoritmos y procesos. Probar el código con diferentes casos para validar los diferentes algoritmos.

6. Realizar los ajustes finales a través de las variables, para conseguir que los métodos seleccionados individualmente funcionen correctamente así como un ajuste previo para el sistema de autenticación considerando todos los métodos.

7. Hacer pruebas con el sistema considerando todos los métodos y mejorarlos a través del ajuste de las diversas variables y verificación de su funcionamiento correcto.

CAPÍTULO 1. SEGURIDAD CON COMPUTADORAS Y AUTENTICACIÓN

Las computadoras día con día adquieren más importancia en la sociedad, ya que cada vez son más usadas para realizar un mayor número de funciones y se les asignan tareas de mayor relevancia. Se usan para almacenar datos para bancos, para hospitales, datos personales, secretos para gobiernos, para corporaciones, para militares, entre otros. Ayudan al control de los automóviles y de las redes de comunicaciones telefónicas. Debido a este aumento en importancia de las computadoras en la sociedad, cada vez hay una responsabilidad mayor para los administradores de estos sistemas quienes se encargan de asegurar que estén funcionando adecuadamente [1]. La seguridad se considera comúnmente como un aspecto adicional y solo cuando ocurre un incidente que conduce a alguna afectación se le suele tomar en cuenta [2]. Existen aplicaciones críticas donde el mal funcionamiento de las computadoras puede afectar nuestra integridad física [1].

1.1 Fundamentos de la seguridad con computadoras

Debido a que existen malos usos de los equipos de cómputo, cada vez es más importante contar con sistemas que garanticen su seguridad. Es necesario ocuparse de su uso inadecuado, ya que los usuarios pueden buscar beneficios a través de su uso inadecuado.

Los objetivos fundamentales de la seguridad de las tecnologías de la información son: la confidencialidad, la integridad y la disponibilidad.

La confidencialidad garantiza que la información y su existencia sean reveladas únicamente a individuos autorizados, mientras que la integridad tiene que ver con que la información se encuentre libre de modificaciones no autorizadas o eliminación. La disponibilidad garantiza que no se niegue el acceso a los usuarios autorizados de los sistemas computacionales [1].

Inicialmente la seguridad de las computadoras trataba con mecanismos de protección físicos.

Asegurar un sistema computacional es difícil y entre las causas se encuentran: los costos adicionales para la seguridad, el hecho de que el problema comúnmente son las personas que usan el sistema o se encuentran en el sistema operativo, que la seguridad se considera como un obstáculo para el usuario y que esta se agrega al final.

Los sistemas operativos son proyectos grandes y, debido a que mientras más grande es un programa más difícil es que se encuentre libre de errores, son prácticamente imposibles de realizar sin cometer errores. La existencia de un solo error puede resultar en un hoyo de seguridad a través del cual los intrusos obtengan acceso al sistema.

1.2 Autenticación de usuarios y tipos de métodos de autenticación

Es importante distinguir entre la autenticación y la autorización, que es otro elemento importante en un plan de seguridad. La autenticación comprueba la identidad de un usuario, mientras que la autorización verifica que el usuario tenga los permisos correctos y derechos para acceder a determinado lugar o recurso. En términos de la seguridad tanto física como de información, el control de acceso es la restricción selectiva de acceso a un lugar u otro recurso. El acto de acceder significa consumir, entrar o usar. Al permiso de acceder a un recurso se le llama autorización. En este trabajo se desarrollará un sistema de autenticación, aunque es posible anclar a este sistema otros que permitan la autorización y el permiso de acceder a determinado lugar o recurso.

La autenticación de usuarios es un componente fundamental de un modelo de seguridad y por lo tanto tiene gran importancia en la defensa de un sistema computacional, ya que es el primer obstáculo con el que se encuentran los intrusos. Un sistema de autenticación no puede verificar la identidad verdadera de un usuario, de forma que existen diversos métodos para asegurar que los usuarios son quienes dicen ser y de esta forma reducir la capacidad que tiene un usuario no autorizado para aparentar que se trata de un usuario autorizado y pueda acceder al sistema [1].

Existen tres tipos distintos de métodos de autenticación: Llaves de información, llaves físicas y llaves biométricas. Cualquier tipo de autenticación se basa en validar al menos uno de los siguientes elementos: Algo que la persona sabe, algo que la persona posee y algo que la persona es [15] (Ver Figura 1.1).

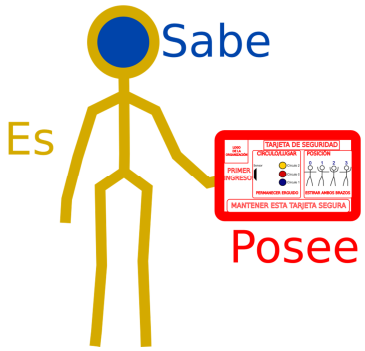

Figura 1.1 La autenticación se basa en validar al menos uno de los elementos: Algo que la persona sabe, posee y es.

El tipo más común de estos métodos es el de llaves de información. Las contraseñas son algo que el usuario conoce y es el método de autenticación menos seguro porque una contraseña puede perderse y ser usada por alguien más [1] [2]. Aunque la tecnología de autenticación continúe avanzando, la autenticación seguirá dependiendo de información que solo la persona conozca, una contraseña es esencial para un sistema de autenticación. Las llaves físicas y biométricas no suelen ser lo suficientemente efectivas por sí mismas, sin embargo se vuelven muy efectivas al combinarlas con una contraseña, [15].

Es posible utilizar uno o más de los numerosos métodos de autenticación que existen. Utilizar una combinación de métodos, del mismo o de diferente tipo, incrementa la fortaleza de la autenticación [2] (Ver Figura 1.2). Cuando se requiere de más de un elemento para realizar la autenticación, se le llama autenticación multifactor. El uso de autenticación multifactor provee varias capas de seguridad que funcionan en conjunto [15]. Por ejemplo, en la autenticación de dos factores, existen dos piezas de evidencia para validar la identidad del usuario. La gran mayoría de mecanismos de control de acceso autentican a los usuarios con una combinación de algo que saben y poseen.

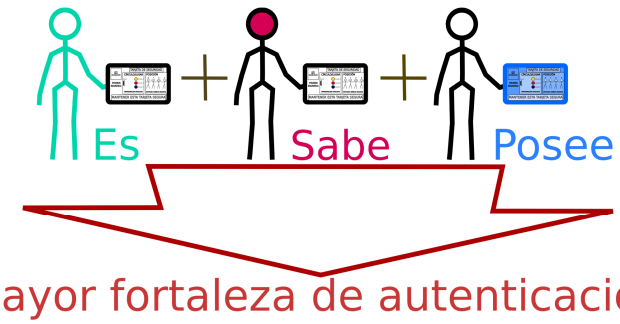

Figura 1.2 El uso de autenticación multifactor incrementa la fortaleza de la autenticación.

La autenticación multifactor se utiliza para aplicaciones que son muy sensibles y requieren una validación más fuerte de la identidad del individuo accediendo al sistema. [2]. Utilizar cualquiera de los métodos individuales no es infalible, pero al combinar dos o más de ellos la seguridad aumenta drásticamente. Sin embargo, la industria aún se encuentra en una etapa temprana y existe una carencia de estándares, lo cual conduce a costos involucrados altos, debido a que se requiere hardware especializado y la utilización generalizada de software [15].

1.2.1 Llaves de información

Las llaves de información son información específica como frases, contraseñas o cuestionarios, que un usuario autorizado conoce y puede proveer al sistema cuando se le solicite. Esto asegura que únicamente aquellos usuarios que tengan su contraseña correspondiente, puedan acceder al sistema.

Las contraseñas suelen ser consideradas como un obstáculo molesto para el usuario, ya que pueden ser difíciles de recordar y se tiene un dilema al permitir que el usuario las seleccione; si selecciona una contraseña fácil de recordar, comúnmente esto conduce a una contraseña fácil de adivinar, lo cual afecta su propósito. Por otro lado, si el usuario selecciona una cadena aleatoria de caracteres, esta se vuelve difícil de adivinar pero también se hace más difícil de recordar. A mayor número de caracteres o elementos que compongan una contraseña mayor dificultad se tiene para deducirla. También a mayor número de posibilidades para elegir cada elemento de la contraseña, mayor dificultad para deducirla. Aún con una selección de contraseña sólida, puede ser descubierta, por lo que debe evitarse escribirla o mencionarla además de cambiarla con regularidad [1].

Una extensión del concepto de cambiar la contraseña con cierta regularidad, es la contraseña de una sola vez, la cual cambia cada vez que se usa. Existen muchos parámetros para las contraseñas, entre los cuales se encuentran: Bloqueo de la aplicación después de varios intentos, longitud de la contraseña, estructura de la contraseña, campos no mostrados, validación de contraseñas antes de que puedan ser cambiadas por otras y limitaciones para compartirlas [2]. La autenticación con cuestionario comprueba la identidad de un usuario con preguntas cuyas respuestas sean muy difíciles de deducir para un intruso.

1.2.2 Llaves físicas

Las llaves físicas son objetos que debe poseer el usuario para poder acceder a un lugar o recurso, como tarjetas magnéticas o llaves. De forma similar a cómo una llave es necesaria para entrar a un cuarto cerrado, para acceder a un recurso, puede ser requerida una llave especial. La posesión de un objeto que confirma la identidad del usuario, provee una validación más fuerte de la identidad del usuario. La desventaja de las llaves físicas es que pueden dañarse o perderse [1] [2].

1.2.3 Llaves biométricas

Las llaves biométricas son mediciones de las características personales o atributos físicos del usuario como patrones de huellas digitales, geometría de la mano, tonalidad de voz e impresiones de retina (Ver Figura 1.3). Las llaves biométricas tienen varias ventajas sobre las llaves físicas o de información. Las tres ventajas más importantes son: (1) siempre están en posesión del usuario, (2) son únicas, y (3) es difícil duplicarlas o construirlas. A este tipo de autenticación se le llama también autenticación fuerte [1] [2].

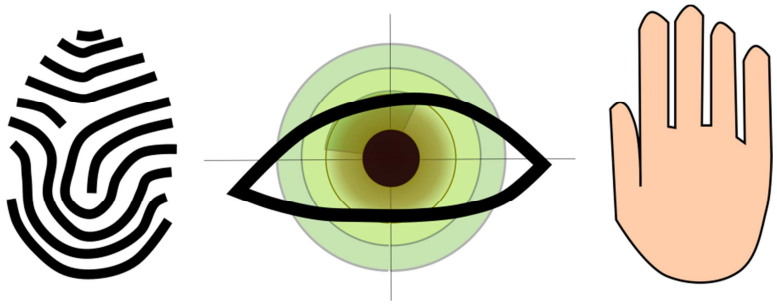

Figura 1.3 Las huellas digitales, las impresiones de retina y la geometría de la mano son ejemplos de llaves biométricas.

Existen análisis sobre acciones habituales de la persona en lugar de atributos físicos menos controlables que algunos investigadores no consideran una medición biométrica, como el de la firma de una persona. La firma de una persona depende tanto de características biométricas, como de acciones habituales [1].

CAPÍTULO 2. SENSOR KINECT Y HERRAMIENTAS DE PROGRAMACIÓN

El sensor Kinect (Ver Figura 2.1) se lanzó al mercado en Noviembre de 2010 pero fue hasta Junio de 2011 que se anunció el lanzamiento del paquete de desarrollo de software para el sistema operativo Windows [16]. El sensor Kinect captura un flujo de elementos de imagen o píxeles coloreados con información acerca de la profundidad de cada uno de ellos. Se trata de un sensor capaz de detectar la posición del cuerpo humano, así como su movimiento y voz. El sensor Kinect permite proveer una interface natural de usuario para la interacción utilizando el movimiento del cuerpo humano, además de la utilización de comandos hablados [2].

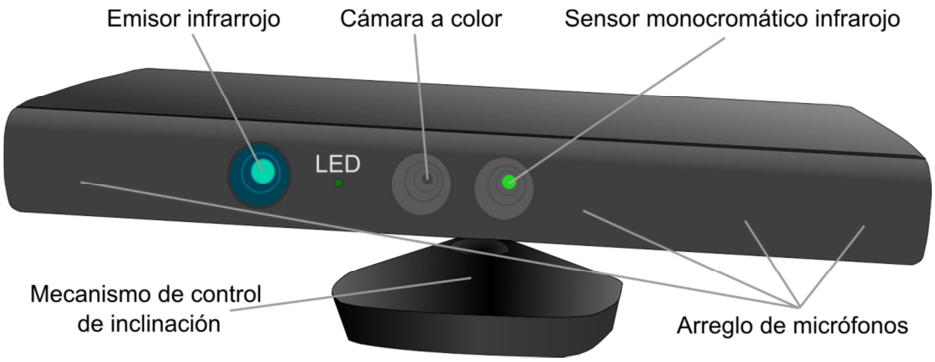

Figura 2.1 Sensor Kinect para Xbox 360 Modelo 1414 (requiere fuente de alimentación adicional Modelo DE-12WS-XK).

2.1 Sensor Kinect

El sensor Kinect cuenta con un arreglo de micrófonos, una cámara a color, un receptor infrarrojo o sensor monocromático CMOS, un emisor infrarrojo, un mecanismo con un motor para controlar la inclinación de todo el sensor y un LED. Utiliza el sistema en un chip o SoC por sus siglas en inglés PS1080 de la compañía PrimeSense, que maneja la información de audio y visual de forma independiente. Se comunica con un PC u ordenador personal por medio de un puerto bus universal en serie o USB 2.0 por sus siglas en inglés y requiere de alimentación eléctrica adicional. Dentro del sensor se encuentra el SoC PS1080, además de componentes y circuitos que le permiten funcionar [16] [2].

El procesamiento de datos es ejecutado en el ordenador personal junto con el controlador Kinect. El controlador puede ser instalado en Windows 7 o Windows 8 y opera en procesadores de 32 o 64 bits. Los requisitos mínimos son: 2 GB de memoria de acceso aleatorio o RAM y un procesador de doble núcleo con una frecuencia de operación de 2.66 GHz o mayor para el controlador Kinect [16] [2].

El mecanismo de inclinación conecta al cuerpo del sensor con la base y consta de un motor y un sistema de engranes que permiten cambiar el ángulo del sensor verticalmente 27° en cada dirección, lo cual quiere decir que los ángulos de operación especificados del sensor pueden ser cambiados 27 grados hacia arriba o hacia abajo. Esto permite que el sensor sea colocado a cualquier altura dentro de una habitación. El LED, o diodo emisor de luz, se encuentra localizado frente al sensor, entre la cámara y el proyector infrarrojo y permite conocer el estado del sensor.

El sensor Kinect cuenta con un arreglo de cuatro micrófonos colocados en orden lineal, distribuidos tres en un lado y uno en el otro, todos en la base del sensor (Ver Figura 2.2). El arreglo de micrófonos permite, además de la captura del sonido, localizar la dirección de la onda de sonido. Esto ayuda a reconocer la voz de forma más efectiva al aumentar la supresión de ruido, la cancelación de eco y la formación de haz direccional [2]. Para calcular de dónde proviene el sonido se vale del hecho de que el sonido llega a tiempos ligeramente diferentes a cada uno de los micrófonos, debido a que la fuente de sonido se encuentra a una distancia ligeramente diferente de cada micrófono. A través de la información de diferente tiempo de arribo, es posible calcular en qué posición se encuentra la fuente de sonido [17]. De esta forma el sensor Kinect opera también como un micrófono bidireccional que puede identificar la posición de la fuente del sonido independientemente del ruido y eco presentes en el ambiente.

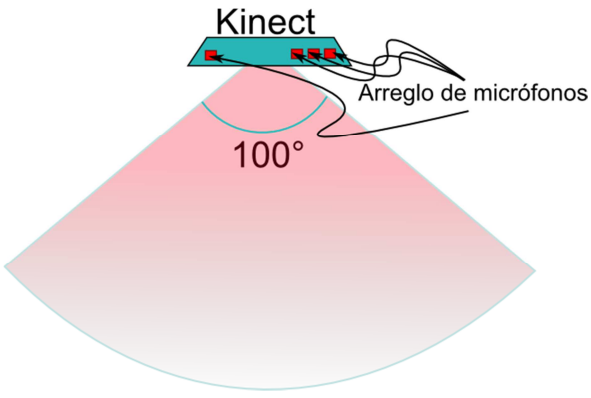

Figura 2.2 Ubicación de micrófonos individuales en el arreglo de micrófonos [2].

La cámara a color es responsable de la captura y el flujo de datos de video a color. El flujo de datos capturados por la cámara es una sucesión de cuadros de imágenes fijas a color, a una velocidad de 30 cuadros por segundo y con una resolución de 640 x 480 píxeles, pudiendo llegar a una resolución máxima de 1280 x 960 píxeles a 12 cuadros por segundo. El sensor es capaz de producir un flujo de datos RGB de 32 bits, lo cual quiere decir que la cámara a color regresa imágenes RGB a 32 bits a las resoluciones soportadas. Cada pixel RGB del cuadro de imagen es un arreglo de tamaño cuatro, los primeros tres valores representan los valores de azul, verde y rojo, mientras que el cuarto representa el valor alfa [3].

El rango de visión de las cámaras del sensor es de 43 grados verticalmente por 57 grados horizontalmente [2] (Ver Figura 2.3). El sensor Kinect tiene la habilidad de capturar una vista tridimensional de los objetos que se encuentren dentro de su rango de operación, sin importar las condiciones de iluminación.

El sensor utiliza lentes ópticos que permiten su operación correcta dentro de los siguientes rangos, además de los ángulos de operación mencionados con anterioridad: (1) Distancia entre el usuario y el sensor para los mejores resultados de 1.2m a 4m (o de 0.4m a 3m en modo cercano) (2) Rango de profundidad de 8m (4m en modo cercano) (3) Temperatura entre 5 C° y 35 C° [16].

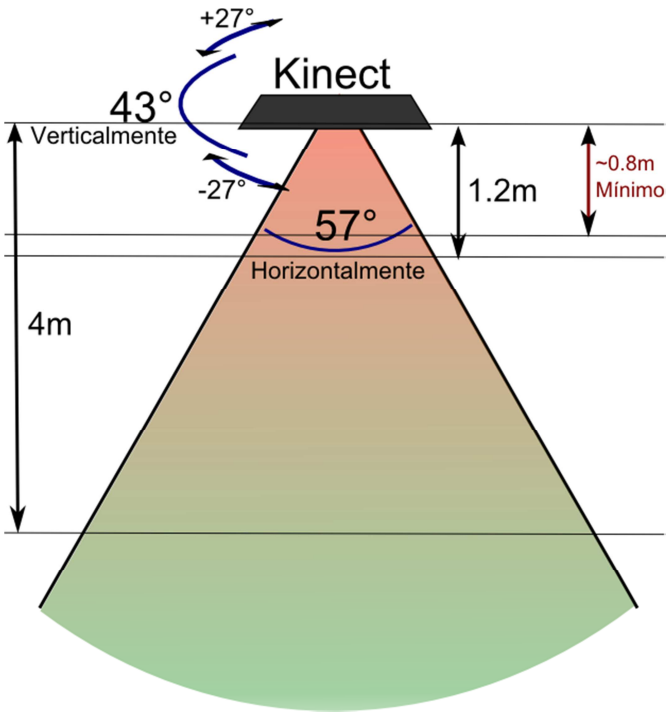

Figura 2.3 Rango de visión de las cámaras del sensor.

Para detectar profundidad se hace uso de un proyector de luz infrarroja que emite constantemente, sobre lo que se encuentre frente a él, un patrón de puntos con distribución pseudoaleatoria en el espacio y de un detector de infrarrojo que consta de un sensor monocromático CMOS, operando en conjunto. Los puntos de luz se reflejan en los diferentes objetos en frente del sensor Kinect y a través del sensor infrarrojo se convierten en información de profundidad, midiendo la distancia entre el sensor y el objeto desde el cuál se reflejó el punto infrarrojo [2]. La distribución de puntos es aparentemente aleatoria, pero en realidad se genera de forma mecánica y sencilla de repetir, y debido a que el sensor conoce el patrón y la forma en que es proyectado, este puede comparar la imagen de la cámara infrarroja con el patrón que conoce. Al conocer hacia dónde apuntan los puntos, le es posible calcular qué tan lejos se encuentra el objeto por la posición de los puntos sobre este [17].

El flujo de datos de profundidad, con 30 cuadros por segundo, soporta resoluciones de 640 x 480 píxeles, 320 x 240 píxeles y 80 x 60 píxeles. Está compuesto de 16 bits por píxel, y cada píxel en el flujo de profundidad usa 13 bits (extremo izquierdo, bits más significativos) para datos de profundidad y 3 bits (bits menos significativos) para identificar a la persona. Un valor de profundidad de 0 indica que no hay información de profundidad disponible debido a que el objeto se encuentra muy cerca o muy lejos de la cámara. Un pixel con un índice de persona con valor de 0 significa que no se ha reconocido la presencia de ninguna persona. Mientras que cada cuadro de color consiste de números de los valores de los pixeles para las componentes de color rojo, verde y azul, cada información de pixel en los datos de profundidad representa la distancia de un objeto al sensor. La información de profundidad tiene un rango de visión de 57 grados horizontales y 43 verticales, donde cada pixel de profundidad contiene la distancia en milímetros entre el sensor Kinect y el objeto frente al dispositivo. El soporte detrás de esta tecnología proviene de la compañía PrimeSense [2] [3].

El sensor Kinect permite, a través de la computadora, predecir las posiciones de las articulaciones del cuerpo humano a partir de una imagen de profundidad y sin usar información temporal. Se realiza un reconocimiento de objetos con una representación de las partes intermedias del cuerpo. De esta forma el problema de estimación de postura se convierte en uno de clasificación por pixel. Se utiliza un conjunto de datos de entrenamiento para estimar las partes del cuerpo sin importar su postura, forma o tipo de ropa. Así se obtienen propuestas tridimensionales de diversas articulaciones del cuerpo, junto con su nivel de confianza [18].

Para el reconocimiento de voz se usa la capacidad de reconocimiento del sistema operativo Windows en conjunto con el sistema de audio del sensor Kinect. El

reconocimiento del habla es una tarea de reconocimiento de patrón, que se efectúa en diferentes pasos con el motor de reconocimiento del habla, que consiste de dos módulos: (1) modelo acústico y (2) modelo de lenguaje. Las operaciones que se realizan para reconocer el habla del usuario son las siguientes:

1. El audio es capturado por los micrófonos y convertido en una señal digital

2. Las señales de sonido son enviadas al motor de reconocimiento del habla

3. El audio es analizado por el modelo acústico del motor de reconocimiento de habla y convertido en fonemas, que son las unidades del habla

4. El modelo del lenguaje analiza el contenido y trata de empatar la palabra, combinando los fonemas con un diccionario incorporado

5. Si la palabra existe en el diccionario, el motor de reconocimiento de habla reconoce lo dicho [3].

2.2 Herramientas de programación

El paquete de desarrollo de software Kinect para el sistema operativo Windows o Kinect SDK, es un conjunto de herramientas para desarrollar aplicaciones para dispositivos Kinect que provee una interface para interactuar con el sensor Kinect a través de los controladores del sistema. El SDK incluye controladores para el sensor Kinect que interactúan con el dispositivo, mientras que el sistema operativo y las interfaces de programación de aplicaciones o APIs, interactúan con el dispositivo a través del programa. En resumen, el SDK provee la posibilidad de construir una aplicación, utilizando el lenguaje de programación C# a través del entorno de desarrollo integrado Visual Studio 2010 o versiones posteriores, ejecutado sobre los sistemas operativos Windows 7 u 8 (Ver Figura 2.4).

La caja de herramientas para el desarrollo en Windows de Kinect es un instalador adicional que trae consigo un conjunto de componentes extendidos. También contiene pruebas y documentación como referencia rápida.

El software necesario para el desarrollo con el SDK de Kinect es el siguiente:

- SDK de Kinect para Windows
- Microsoft .NET Framework 4.0 o posterior
- Visual Studio 2010 o posterior
- Microsoft Speech Platform - Server Runtime y los paquetes de lenguaje que se vayan a utilizar (en este caso español de México y español de España además de inglés de E.U.).

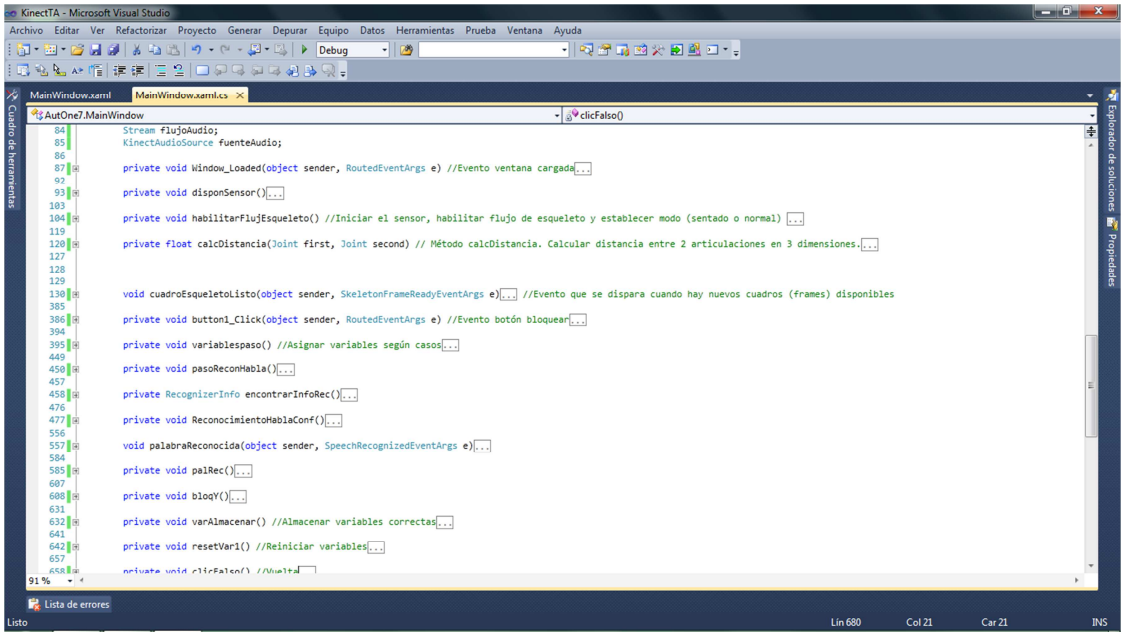

Figura 2.4 Captura de pantalla de Visual Studio 2010 mostrando el código que define la actividad .xaml.cs.

La estructura interior del kit de desarrollo de software, o SDK por sus siglas en inglés, Kinect para Windows se muestra en la Figura 2.5. El SDK instala la interfaz de programación de aplicaciones de la interface natural de usuario, o NUI por sus siglas en inglés, y los controladores de Kinect para integrar el sensor con el sistema operativo Windows.

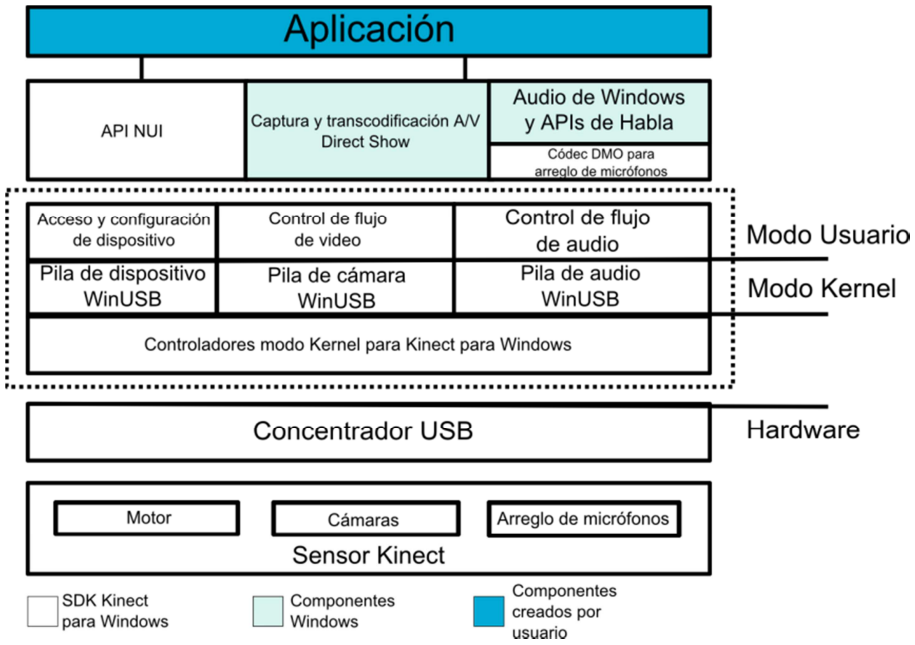

Figura 2.5 Arquitectura del kit de desarrollo de software Kinect para Windows [16].

El controlador de Kinect puede controlar la cámara, el sensor de profundidad, el arreglo de micrófonos y el motor de inclinación. El sensor Kinect es una fuente multi-flujos, ya que puede proveer un flujo para cada tipo de datos que recolecta [16]. Los datos pasan entre el sensor y la aplicación en la forma de flujos de datos de los siguientes tipos [2]:

- Flujo de datos de color
- Flujo de datos de profundidad
- Flujo de datos de audio

Los flujos son transmitidos a la PC utilizando el concentrador USB. La API NUI colecta los datos y los presenta a las aplicaciones. El SDK de Kinect para Windows puede calcular un seguimiento completo de esqueletos detectados en frente del sensor utilizando los datos de flujos de color, profundidad y audio [16].

2.2.1 Lenguaje de programación C#

El lenguaje de programación C# se puede utilizar para el desarrollo de aplicaciones a través de Visual Studio 2010. El lenguaje C# es un lenguaje de propósito general, seguro en escritura (previene errores de escritura) y orientado a objetos. Para conseguir

productividad del programador este lenguaje balancea la simplicidad, la expresividad y el desempeño.

La orientación a objetos incluye encapsulamiento, herencia y polimorfismo. La encapsulación es el procedimiento de cubrir los datos y funciones en una sola unidad o clase. La herencia es la habilidad de crear una clase a partir de otra clase, una clase padre, extendiendo la funcionalidad y estado de la clase padre en la clase derivada o hija. Implica clasificación jerárquica.

El polimorfismo se refiere al comportamiento de los objetos, no a su pertenencia a una jerarquía de clases. Permite que el mismo nombre de método sea usado para operaciones diferentes en diferentes tipos de datos.

Desde una perspectiva orientada a objetos, C# es un sistema de tipo unificado, es decir, todos los tipos (objetos o primitivos como números) comparten el mismo conjunto de funcionalidad. Es un lenguaje en el que sus reglas de escritura son muy estrictas y de propósito general, diseñado para ser usado para escribir software en una amplia variedad de dominios de aplicación [19].

2.2.2 Entorno de desarrollo integrado

Visual C# de Microsoft es un lenguaje poderoso pero simple, orientado a componentes y dirigido a desarrolladores que usan el .NET Framework. Visual Studio 2010 es un ambiente de programación que permite trabajar con C# y cuenta con muchas herramientas para permitir el desarrollo de diversos proyectos [20]. El ambiente de desarrollo que provee Microsoft Visual Studio 2010 facilita la utilización de las diversas características con el fin de mejorar la productividad del desarrollador.

Windows Presentation Foundation (WPF) permite la creación de aplicaciones nativas de Windows y proporciona un modelo de programación unificado a los programadores, incorporando un sistema moderno de gráficos para Windows que incluye aceleración por hardware e independencia de la resolución de pantalla. Con las tecnologías de visualización tradicionales no existe forma sencilla de separar el contenido gráfico del código, forzando a los diseñadores gráficos a llevar su contenido exportándolo a un formato de mapa de bits para ser usado como tapiz para las diferentes ventanas, botones y otros controles. En interfaces que cambian con el tiempo, las tecnologías de visualización tradicionales resultan muy limitantes, ya que cada elemento gráfico, como

un botón o un fondo, necesita ser exportado como un mapa de bits separado, impidiendo la combinación de mapas de bits y el uso de efectos dinámicos como transparencias y sombras. Además, el tamaño de los botones, su posición, efectos y animaciones deben ser integrados en el código por el desarrollador, de forma que el diseñador gráfico no puede controlar estos detalles. Aun cuando la interface sea diseñada por un diseñador gráfico, será necesario recrearla con código C#. WPF resuelve este problema con XAML (Extensible Application Markup Language). Cuando se diseña una aplicación WPF en Visual Studio, la ventana en la que se está trabajando no es traducida en código, sino serializada en un conjunto de etiquetas XAML, que son usadas para generar los objetos que componen la interface de usuario cuando se ejecuta la aplicación [21] (Ver Figura 2.6).

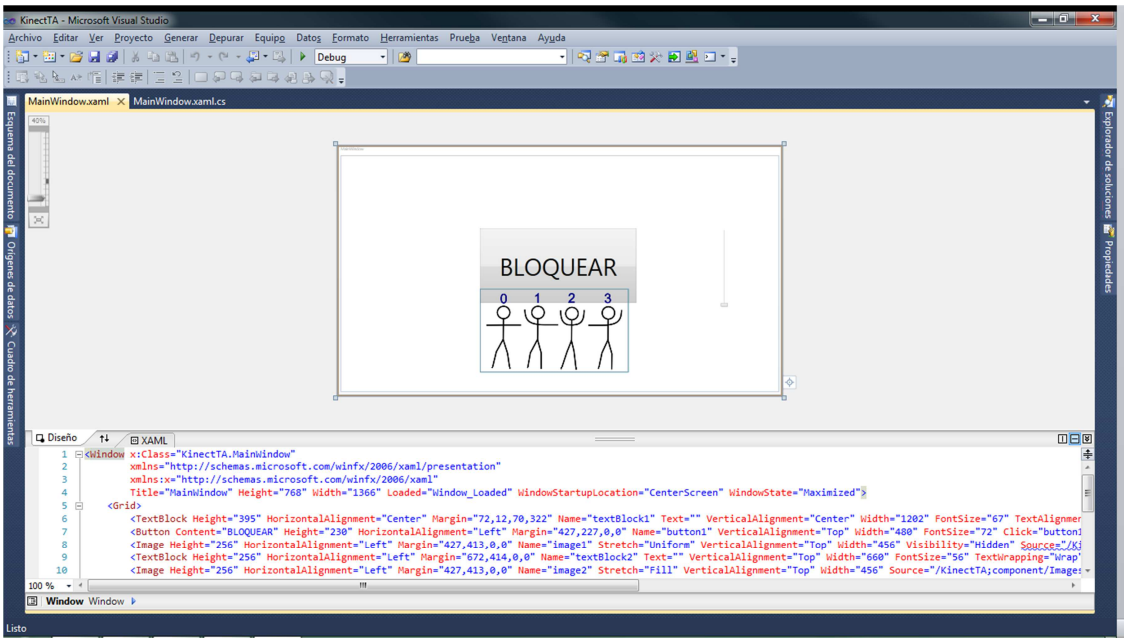

Figura 2.6 Captura de pantalla de Visual Studio 2010 mostrando el archivo que define el aspecto visual de la aplicación .xaml.

2.2.3 Preparación del ambiente de desarrollo

Para comenzar con el desarrollo del sistema se obtuvieron los diferentes elementos necesarios. Se obtuvo el software Microsoft Visual Studio 2010, el Microsoft .NET FrameWork, el Kinect for Windows SDK, el equipo de cómputo PC y el sensor Kinect con una fuente de alimentación adicional. Se descargó de la red el Microsoft Speech Platform Server Runtime, el Microsoft Speech Platform SDK y los paquetes de idiomas

para español de México y español de España. El Kinect SDK y el Developer toolkit se descargaron también de la red. Una vez instalado el SDK, se conectó el sensor Kinect en el PC y el LED se encendió con luz roja la primera vez, que es cuando el sistema empieza a instalar los controladores. Cuando el LED del sensor se enciende con luz verde significa que el dispositivo está funcionando adecuadamente y se puede comunicar con el PC. Después de instalar todo el software mencionado y verificar los controladores instalados, así como conectar el sensor a la alimentación y al PC a través de un puerto USB, es posible comenzar con el desarrollo de una aplicación.

Para probar los sensores del Kinect se utilizó el Developer Toolkit a través de las diferentes aplicaciones de muestra que incluye. Con la aplicación Kinect Explorer fue posible observar las diferentes características del Kinect for Windows SDK, el cual obtiene los datos de color, profundidad y de esqueleto y los muestra en la interface de usuario. Con el fin de verificar el correcto funcionamiento del arreglo de micrófonos se usó el sensor Kinect como un micrófono y se verificó que el sonido, obtenido a través del sensor, al hablar correspondiera con la voz escuchada a través del parlante del sistema. Una vez que se revisaron los diferentes sensores y características, se verificaron las limitaciones técnicas establecidas en las especificaciones.

CAPÍTULO 3. DISEÑO DE LOS MÉTODOS DE AUTENTICACIÓN

El sensor Kinect permite el desarrollo de nuevos sistemas de autenticación con diversos métodos, debido a la posibilidad de seguir las posiciones del cuerpo humano y el reconocimiento de voz. Esto permite el desarrollo del sistema de autenticación en el que se sigue la posición del cuerpo humano, se reconocen palabras utilizando la voz y se muestran notificaciones como confirmaciones e instrucciones.
Con ayuda del sensor Kinect es posible obtener palabras pronunciadas por el individuo utilizando los micrófonos incorporados, mediciones de su esqueleto y las posiciones de sus diferentes articulaciones.

3.1 Esquema general del sistema de autenticación

El diseño del sistema se esquematiza en el diagrama de la Figura 3.1:

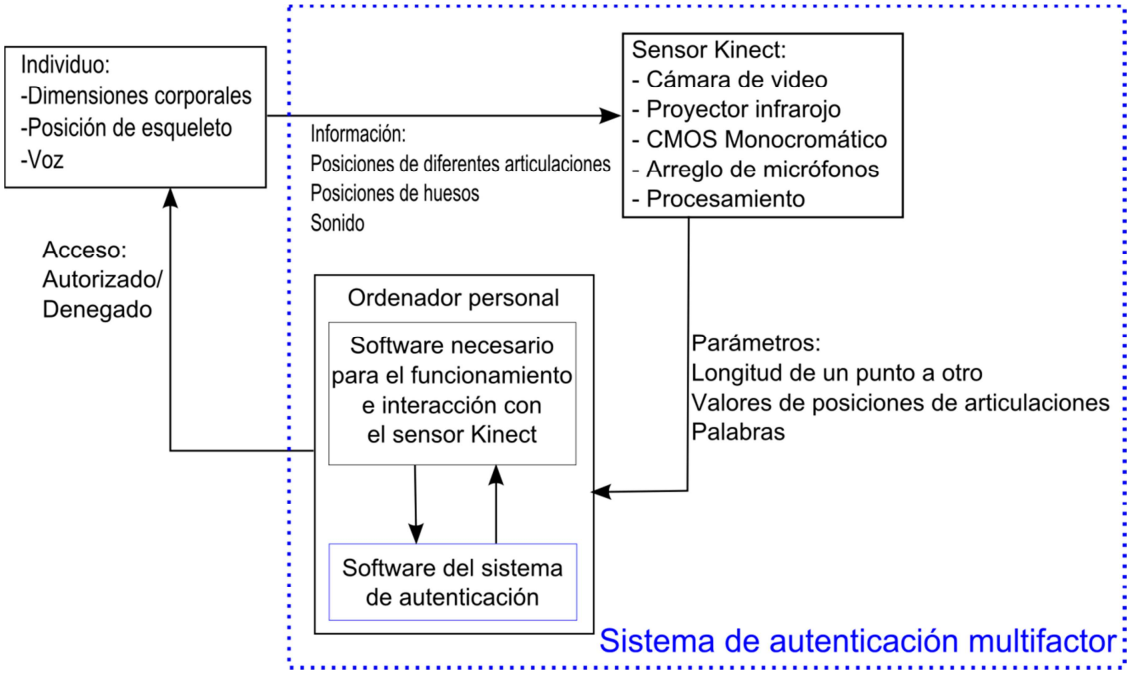

Figura 3.1 Esquema general del sistema de autenticación propuesto utilizando el sensor Kinect.

El software del sistema de autenticación, interpreta los diferentes parámetros y valida el acceso al sistema o lo niega. Lo anterior requirió programar una aplicación que integrara

los métodos de autenticación seleccionados, tomando en cuenta la interfaz de usuario y el correcto funcionamiento de los diferentes algoritmos y procesos.

El sistema puede ser protegido con una barrera resistente, que además sea transparente a la luz visible, a la luz infrarroja y permita el paso del sonido. Una barrera para proteger al sistema puede ser un vidrio laminado de seguridad a prueba de golpes con perforaciones para permitir el paso del sonido. Otra alternativa es proteger el ordenador personal en otra habitación resguardada, dejando el sensor expuesto. De esta forma se vulnera el sensor Kinect pero no la efectividad del sistema de autenticación, ya que, si el sensor es desconectado, el sistema no puede efectuar el proceso de autenticación y el sistema se bloquea.

También es importante mencionar los requisitos de espacio que este sistema requiere para operar de forma adecuada. El sistema tiene que ser instalado en una habitación que tenga las dimensiones mínimas para que la persona se pueda colocar a las distancias requeridas.

El sistema de autorización no forma parte del proceso de autenticación, sin embargo se describe un escenario sencillo en el que se otorga permiso para acceder a un lugar. En la Figura 3.2 se muestra un esquema de una habitación con dos puertas que no permite observar ni escuchar desde su exterior. La puerta 1 se abre por fuera y permite el acceso a la persona para que pueda ser autenticada. Si la persona es identificada como un usuario y se le autoriza acceder, entonces le es permitido acceder por la puerta 2. Si se trata de un intruso, la persona no es identificada como usuario elegible para entrar al lugar, no se le autoriza acceder y la puerta 2 no le permite pasar.

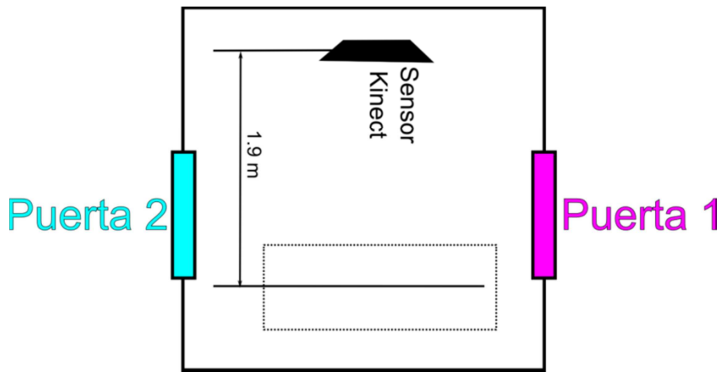

Figura 3.2 Esquema de habitación con dos puertas que no permite observar ni escuchar desde su exterior.

3.2 Selección de los métodos de autenticación que serán utilizados

Debido a los parámetros que se pueden obtener con el sensor Kinect y a las pruebas iniciales realizadas, así como a sus rangos de operación y limitaciones, se seleccionaron los siguientes métodos de autenticación:

- Método de ingreso de palabras utilizando el reconocimiento del habla. (Método de contraseña o llave de información)
- Método de introducción de contraseña utilizando la posición de alguna parte del cuerpo humano. (Método de contraseña o llave de información)
- Método de introducción de contraseña utilizando la postura del cuerpo humano (Método de contraseña y biométrica, en el sentido de que se requieren las medidas de un esqueleto humano)
- Método de medición de longitudes del esqueleto (Método biométrico)

Además se incluye un quinto método de llave física al requerir de una tarjeta impresa con información necesaria para conseguir una autenticación exitosa.

De esta forma, el sistema permite una autenticación multifactor que incrementa su robustez y cuenta con las siguientes ventajas:

- Sistema integral. Permite la utilización de varios métodos de autenticación en un solo sistema. Posibilidad de validar diversas características o parámetros simultáneamente como posiciones del esqueleto del cuerpo humano, longitudes y voz.
- Uso de métodos biométrico, llave física y tres métodos de contraseña. Al utilizar varios métodos, se tiene mayor robustez.
- Elimina necesidad de contacto físico con el sistema y puede realizarse a metros de distancia. Esto evita su modificación o vulneración por medio de software al no existir acceso directo al sistema. También de esta forma se limita el número y tipo de ingresos de información, únicamente a los necesarios para realizar la autenticación. Con la utilización de un teclado, que representa un dispositivo de entrada universal, se podría modificar el sistema o vulnerarlo.

- Sistema nuevo. Al tratarse de un sistema nuevo, del cuál no existe conocimiento generalizado sobre su funcionamiento o forma de ingresar la información, se añade una capa más de seguridad.

- Económico. Sólo es necesario un sensor Kinect y un PC. El costo por tarjeta/llave física por usuario es mínimo.

- Sencillo de utilizar. Al utilizar como forma de ingresar la información el cuerpo humano y el habla, simplifica el proceso de aprendizaje de su uso.

- Eficiente. Se simplifica el ingreso de información sin hacer más accesible el uso del sistema por intrusos. Aunque una posición del cuerpo humano sea algo fácil de conseguir para un humano, diseñar un objeto que imite su comportamiento, dimensiones y forma, resultaría poco práctico y difícil de realizar de forma discreta.

- Permite un gran número de usuarios.

- Posibilidad de alto nivel de seguridad.

- Requiere la presencia física del individuo. Requiere la forma del cuerpo humano, el uso de la pronunciación de palabras y las dimensiones del esqueleto del individuo particular.

- Verificación de usuario válido. El sistema verifica primero la presencia de un usuario válido utilizando las posiciones del cuerpo, antes de permitir el ingreso de información.

- Flexibilidad en las formas de ingreso de la información. Diversidad de tipos de métodos de autenticación.

- Gran potencial de mejoras hacia el futuro, por ejemplo mediante la inclusión de más métodos, patrones más complejos, mediciones más precisas, requerimientos de espacio menores o mayor resolución en las cámaras. Las posibilidades de mejora, al tratarse de una tecnología nueva, son enormes.

La selección de métodos de autenticación, se hizo buscando maximizar la simplicidad sin disminuir la efectividad del sistema. Se eligió evitar métodos más complejos en vista de que se utilizarían varios de ellos y se privilegió que la autenticación por cada uno fuera fiable, sobre la complejidad de cada uno de ellos. Esto debido a las limitaciones propias de la implementación. Como son métodos sencillos, dan un margen menor a errores en el reconocimiento. Aunque cada uno de los métodos difícilmente es lo suficientemente

robusto de forma individual, la combinación de todos ellos es muy robusta. Las diferentes decisiones tomadas para cada implementación particular se discuten en cada sección de los métodos de autenticación.

En el futuro, con el avance de la tecnología y con mayor desarrollo, es posible mejorar y perfeccionar cada una de los métodos, así como aumentar o disminuir la cantidad de estos.

3.3 Diseño de los métodos de autenticación utilizando el sensor Kinect

Se desarrollaron varias aplicaciones para realizar pruebas sobre cada método seleccionado para la autenticación y verificar su viabilidad así como posibles obstáculos y limitaciones no previstas. Se probaron los algoritmos para cada método, con el fin de verificar su efectividad para la autenticación en cada caso. En el capítulo IV se describe con detalle la implementación del sistema y se detalla el funcionamiento del código.

3.3.1 Método de ingreso de palabras utilizando el reconocimiento del habla

El objetivo de este método es permitir el ingreso de palabras como información que será utilizada para la autenticación de un individuo. Una cadena de caracteres forma una palabra. En este caso los elementos que forman la contraseña son palabras en lugar de caracteres, de modo que una contraseña está representada por una cadena de palabras en lugar de una cadena de caracteres.

Se seleccionó un subconjunto de palabras del total existente en el lenguaje español, con el fin de cumplir con dos objetivos: (1) Añadir seguridad al no permitir que cualquier palabra del español sea considerada como un ingreso válido y (2) Evitar que el sistema de reconocimiento confunda dos o más palabras que sean homófonas, es decir, que tengan la misma pronunciación pero su significado sea diferente y por lo tanto se trate de palabras diferentes.

Para conseguir el segundo objetivo, se definieron palabras que tuvieran pronunciación diferente a todas las demás palabras definidas. Este método de autenticación va

acompañado de otro en el que el usuario necesitará portar una llave física, que consiste de una tarjeta con una lista de las palabras que representen los ingresos posibles para el sistema, así como información relevante para los demás métodos de autenticación que serán empleados. Estas palabras son consideradas como válidas, mientras que las palabras no contenidas son consideradas como no válidas.

El código tuvo que permitir considerar un subconjunto de palabras de todas las existentes en el lenguaje español, con el fin de tener un control sobre las palabras que representen un ingreso posible de información al sistema.

Además, el código permitió el ingreso de palabras, como información, utilizando el sonido captado por los micrófonos del sensor Kinect, una a la vez y distinguir entre una palabra válida para la autenticación y una no válida.

Una vez conseguido el reconocimiento de alguna palabra válida, fue necesario que el código permaneciera a la espera de la siguiente, además de permitir la ejecución de algún evento que a su vez permitiera con la continuación del proceso de autenticación e informar al individuo sobre el estado del proceso de autenticación. También fue necesario que el código pudiera almacenar temporalmente la cadena de palabras mencionadas por el individuo, para poder compararlas de forma posterior y así permitir la simulación de la validación del acceso al sistema o negarlo.

El sensor Kinect cuenta con cuatro micrófonos en una configuración lineal. El SDK permite la captura de audio pero también permite el procesamiento de audio para supresión de ruido y cancelación de eco así como para controlar la dirección del haz del arreglo de micrófonos a través del SDK. Las capacidades de procesamiento de audio de alta calidad proporcionadas por el SDK son a través de su propio pipeline de procesamiento interno [2].

Utilizando las interfaces de programación de aplicaciones de reconocimiento del habla de Windows fue posible desarrollar aplicaciones de prueba, construyendo el vocabulario y pasándolo al motor de reconocimiento para diseñar un conjunto de comandos de voz a ser utilizados en la aplicación.

Sobre la selección del número de palabras válidas, así como el tamaño de la cadena de palabras se discute en el capítulo IV.

En la Figura 3.3 se encuentra un diagrama del algoritmo para el método de ingreso de palabras utilizando el reconocimiento del habla. Los arreglos 0 y 1 son arreglos de

palabras mientras que i es el contador de intentos y m el número de palabras necesarias para la autenticación.

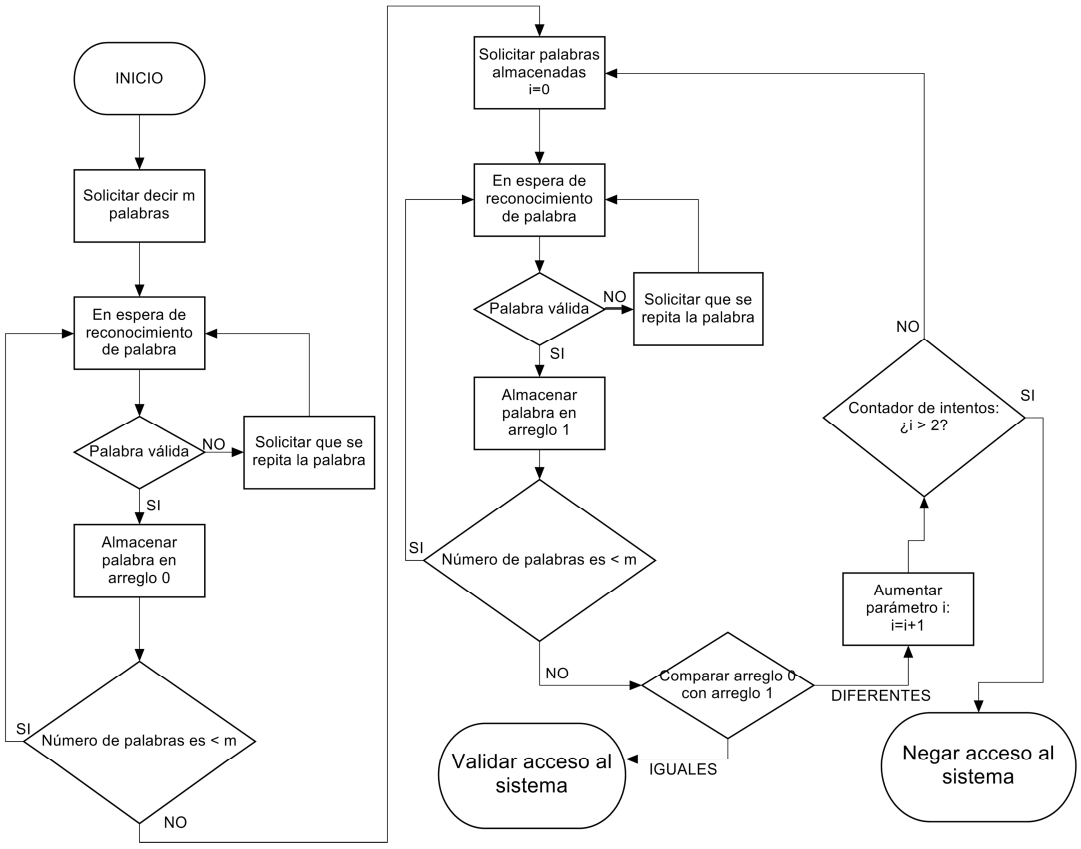

Figura 3.3 Diagrama que representa la forma de operar del algoritmo para el método de ingreso de palabras utilizando el reconocimiento del habla.

3.3.2 Método de introducción de contraseña utilizando la postura del cuerpo humano

El SDK (kit de desarrollo de software) de Kinect soporta seguimiento del esqueleto humano y es posible detectar el movimiento del esqueleto humano que se encuentre frente al sensor. Es posible seguir hasta 20 articulaciones para dos esqueletos y puede detectar hasta seis esqueletos. Sobre los otros cuatro esqueletos solo es posible obtener información acerca de la posición propuesta del centro de los hombros. El SDK también soporta el seguimiento del cuerpo humano que se encuentre sentado, pero únicamente los 10 puntos de articulaciones superiores [2] [3]. En la Figura 3.4 se muestran las 20

articulaciones que se pueden seguir para los esqueletos de máximo dos personas. Las líneas rojas muestran las 10 articulaciones que es posible seguir en modo sentado.

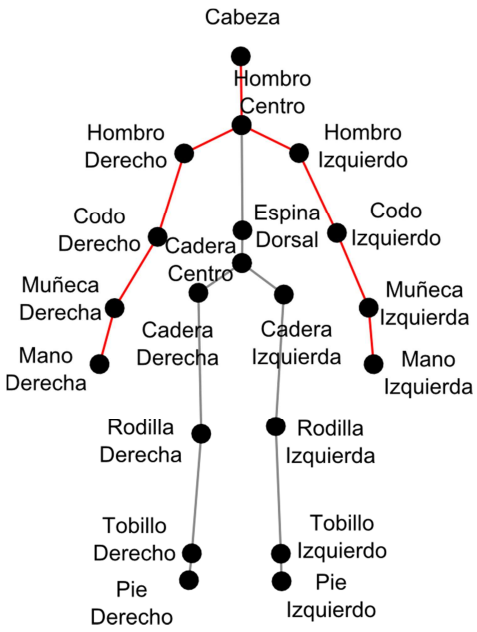

Figura 3.4 Las 20 articulaciones que se pueden seguir para los esqueletos de máximo dos personas.

Las posibilidades de autenticación a través de la introducción de claves, utilizando las posiciones del esqueleto del cuerpo humano, son muchas. Sin embargo, para cumplir con los objetivos de ser un sistema simple pero efectivo, se decidió utilizar un algoritmo que no fuera complejo, pero que pudiera garantizar que aportara una barrera importante de seguridad. El objetivo principal de este método es evitar que las personas que no conozcan el sistema puedan ingresar información al sistema para la autenticación. En un escenario hipotético, la persona podría hacer movimientos aleatorios en frente del sensor con la esperanza de obtener acceso al sistema, pero este método está diseñado para impedirlo. Entonces, el problema principal de este método es diseñar una posición del cuerpo humano tal que no sea natural pero que tampoco sea complicada o difícil de mantener. Esto es importante porque, si se trata de una posición común o natural, una persona ajena al sistema podría sortear esta barrera haciendo movimientos aleatorios o permaneciendo en una posición natural. Al mismo tiempo, es importante que no sea complicada por simplicidad y comodidad en su ejecución. La solución propuesta tomó en cuenta dos aspectos: (1) Las posiciones posibles o válidas y (2) El tiempo que deben ser mantenidas para ser consideradas como una entrada o ingreso de información al sistema.

Para el primer aspecto se consideró que a mayor complejidad en las posiciones, mayor seguridad, pero también mayor dificultad en la ejecución y por lo tanto menor efectividad del método. Se eligió que las posiciones del cuerpo se limitarían a las posiciones de los brazos, tanto por simplicidad como por razones de aplicabilidad al mayor número de personas. Si una persona por alguna razón no puede estar de pié, el algoritmo de autenticación sigue siendo válido y se puede emplear, ya que sólo considera las posiciones de los brazos.

El siguiente paso fue elegir la cantidad y tipo de posiciones válidas. Se desarrolló un algoritmo que requiriera que el individuo conociera cierta información para poder ejecutar las posiciones de forma apropiada. La información que requiere conocer el individuo es que ambos brazos deben estar estirados. Cualquier posición de los brazos en la cual estos no estén estirados simultáneamente, no será considerada como válida por el sistema y no se podrá continuar con el proceso de autenticación, de modo que el individuo no conseguirá acceso al sistema.

Para añadir seguridad al sistema, sin introducir complejidad, se eligieron cuatro posiciones válidas que se muestran en la Figura 3.5.

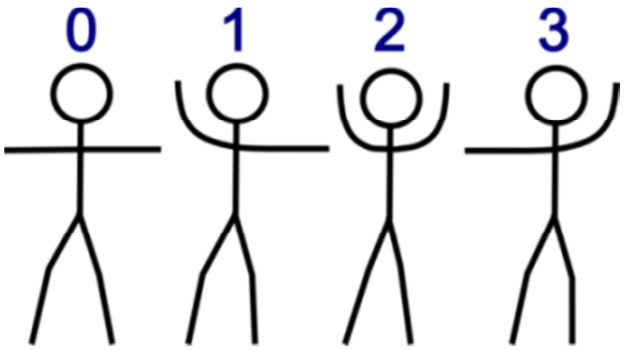

Figura 3.5 Las cuatro posiciones válidas requieren que el individuo mantenga estirados ambos brazos.

Las cuatro posiciones requieren que el individuo mantenga estirados ambos brazos y son las siguientes:

- 0: Ambos brazos a los lados u horizontales.
- 1: Brazo derecho arriba o vertical y brazo izquierdo a un lado u horizontal.
- 2: Ambos brazos arriba o verticales.
- 3: Brazo izquierdo arriba o vertical y brazo derecho a un lado u horizontal.

Es necesario mantener alguna de las posiciones válidas durante cierto tiempo para poder continuar con el proceso de autenticación. El tiempo que debe ser mantenida cualquiera de las posiciones para ser considerada como una entrada o ingreso válido de información al sistema, será controlado por un parámetro dependiente de los cuadros por segundo que son enviados sobre las posiciones del esqueleto, que será ajustado en la fase final de desarrollo.

El algoritmo funciona de la siguiente forma:

- Si el individuo tiene ambos brazos estirados el sistema libera un candado.
- Si el individuo tiene ambos brazos por encima de cierta altura, con respecto a sus hombros, el sistema libera un segundo candado.
- Si el individuo se encuentra en alguna de las cuatro posiciones válidas, el sistema libera un tercer candado.

Entonces, el individuo debe mantener los brazos estirados y en alguna de las cuatro posiciones válidas, para que el sistema comience a contar el número de cuadros del flujo de datos del esqueleto. Si el usuario adopta alguna de las cuatro posiciones pero no durante el tiempo suficiente, el sistema automáticamente reinicia el contador de cuadros y es necesario volver a mantener la posición durante el tiempo requerido inicialmente. Esto aparenta ser complejo, pero lo importante es que el sistema de autenticación sea fácil de utilizar, y desde este punto de vista el algoritmo no complica la facilidad de uso pero sí incrementa la robustez.

También, en este método, el código debe almacenar la posición elegida por la persona para su comparación futura en el proceso de autenticación.

En la Figura 3.6 se muestra el diagrama que representa la forma de operar del algoritmo para el método de introducción de contraseña, utilizando la postura o posición del cuerpo humano. El parámetro de tiempo t está relacionado con el número de cuadros del flujo de datos sobre el esqueleto y permite controlar el tiempo que es necesario mantener la posición para considerarla un ingreso de información.

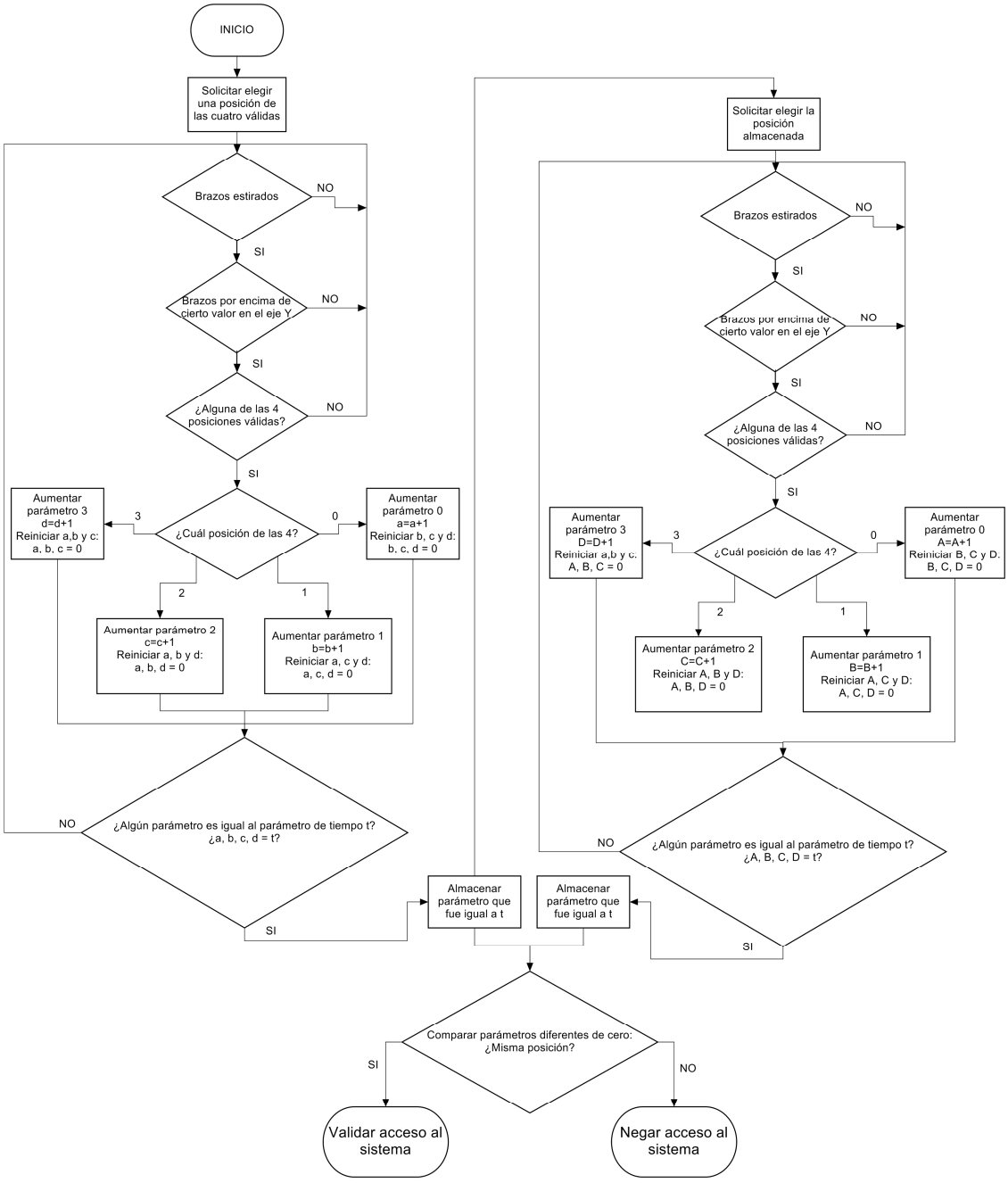

Figura 3.6 Diagrama que representa la forma de operar del algoritmo para el método de introducción de contraseña utilizando la postura del cuerpo humano.

3.3.3 Método de introducción de contraseña utilizando la posición de alguna parte del cuerpo humano.

Este método fue incluido debido a que no agrega dificultad de uso al sistema, pero constituye una barrera más y toma ventaja de la información que obtiene el sensor. Es similar al método de introducción de claves utilizando la postura del cuerpo humano pero sólo toma en cuenta la posición del individuo respecto al sensor. Se seleccionó la posición de la cabeza, en el eje X del sensor Kinect, con respecto al sensor. En la Figura 3.7 se muestran los ejes espaciales del sensor Kinect. El eje X aumenta hacia la derecha desde el punto de vista del usuario, el Y aumenta hacia arriba y el Z aumenta hacia el usuario [16].

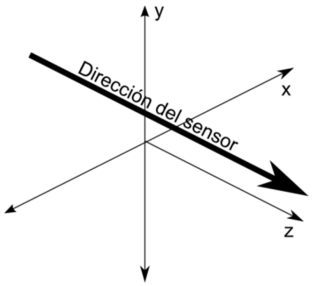

Figura 3.7 Ejes espaciales del sensor Kinect.

En este método se puede seleccionar una de tres posiciones posibles indicadas en el piso con círculos marcados. Las posiciones son: Círculo 0, círculo 1 y círculo 2. Los círculos marcados en el piso deben tener un diámetro de 45 cm y estar separados entre sí por 10 cm. En la Figura 3.8 se muestra un diagrama de la vista desde arriba, no a escala, de la distribución de los diferentes elementos, para el método de introducción de contraseña utilizando la posición de la cabeza.

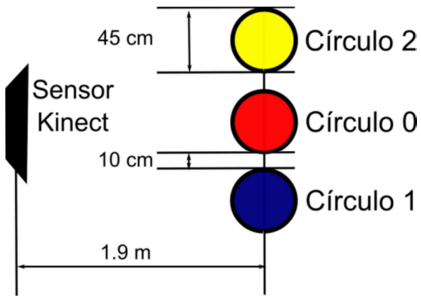

Figura 3.8 Vista desde arriba, no a escala, de la distribución de los diferentes elementos para el método de introducción de contraseña utilizando la posición de la cabeza.

Debido a que se indica al usuario las posiciones a través de unos círculos marcados en el piso, en este método es importante que la persona permanezca derecha y se coloque dentro de alguno de los círculos.

En este algoritmo se considerará la posición elegida por el usuario, y se contará el número de cuadros del flujo del esqueleto, mientras mantenga alguna de las posiciones de los brazos válida. Una vez terminada la cuenta hasta cierto valor, que será determinado en los ajustes finales, la posición quedará determinada.

En este método también se debe almacenar la posición elegida por la persona para su comparación futura en el proceso de autenticación.

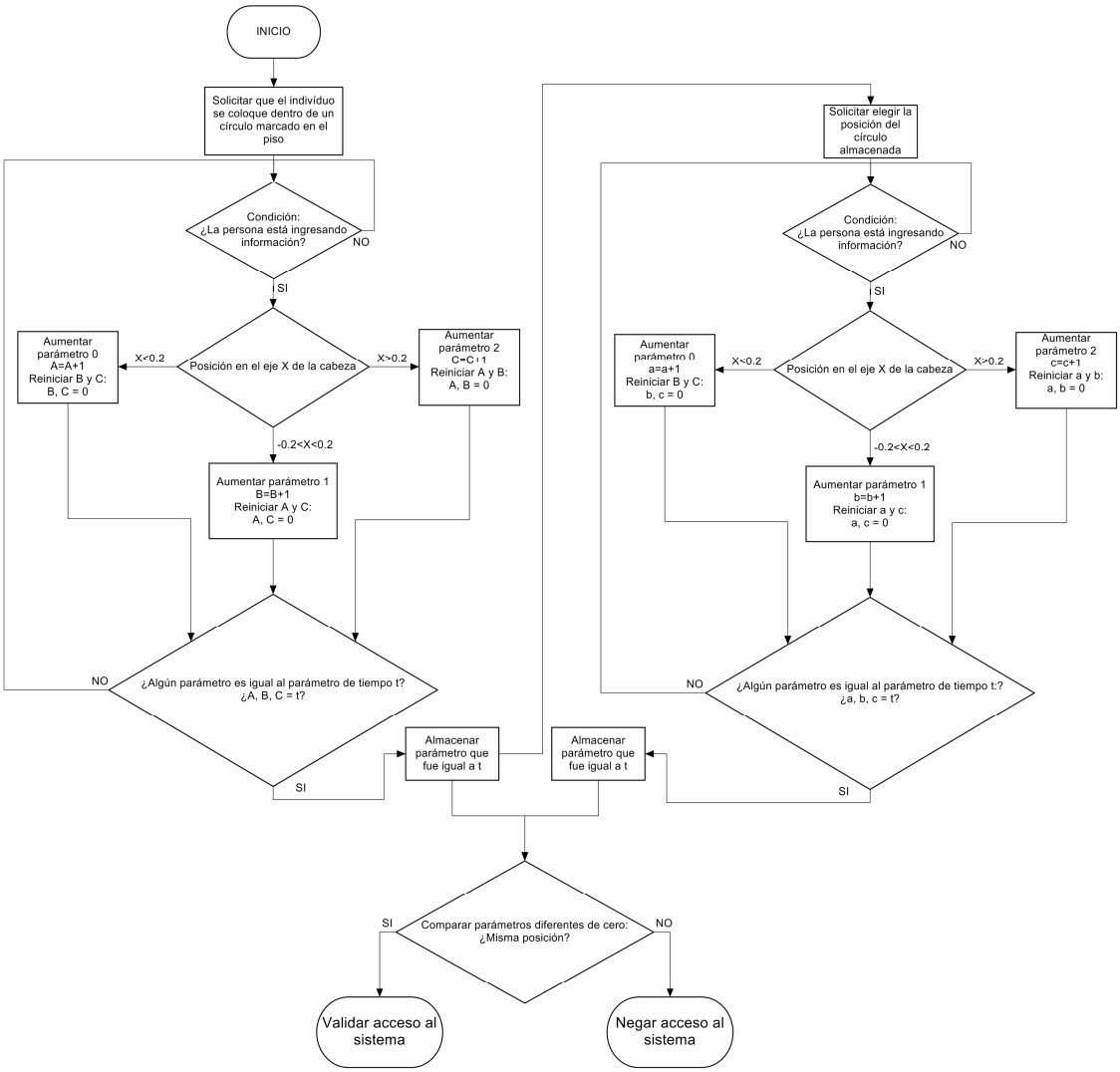

Figura 3.9 Diagrama que representa la forma de operar del algoritmo para el método de introducción de contraseña utilizando la posición de alguna parte del cuerpo humano.

En la Figura 3.9 se puede observar que se requiere saber si la persona está ingresando información para incrementar el parámetro correspondiente. En la implementación final del sistema se utiliza la detección de la posición válida de los brazos para asegurar que la persona está intentando ingresar información al sistema. Esto es necesario porque si la persona se encuentra en frente del sensor, pero no desea indicar un círculo marcado en el piso en particular, el algoritmo debe ignorar las mediciones que se realicen en ese momento. Este criterio se utilizará también en el método de medición de longitudes del esqueleto para garantizar que la persona se encuentra en la misma posición cuando se haga la medición de la longitud que se utilizará en la autenticación. El parámetro de tiempo t está relacionado con el número de cuadros del flujo de datos del esqueleto.

3.3.4 Método de medición de longitudes del esqueleto

En este método, el objetivo fue utilizar un método de autenticación biométrico a partir de la información de las dimensiones del cuerpo humano, obtenida a través del sensor Kinect. El primer aspecto importante a considerar es el sensor Kinect y en particular sus limitaciones para tomar mediciones de longitudes del cuerpo humano. Una discusión sobre esto se llevará a cabo en el cuarto capítulo.

El código en este método tiene que obtener la longitud de alguna región del cuerpo del individuo y almacenarla para compararla posteriormente y poder efectuar la autenticación. En las pruebas del funcionamiento adecuado del sensor se encontró que tomar una sola medición no es suficiente, ya que las fluctuaciones en las mediciones impiden distinguir a una persona de otra con una sola medición. Con el fin de mejorar la medición de la longitud se toman tantas mediciones como sea posible durante el intervalo de tiempo definido por el parámetro de tiempo y se promedian.

En la Figura 3.10 se muestra el diagrama que representa la forma de operar del algoritmo para el método de medición de longitudes del esqueleto. En éste se muestra tanto el modo de inscripción, en el que la medición del invivíduo es almacenada, como el modo de autenticación, en el que se verifica su identidad. El parámetro de tiempo limita el número de mediciones posibles ya que el flujo de datos del esqueleto tiene un número de cuadros por segundo máximo.

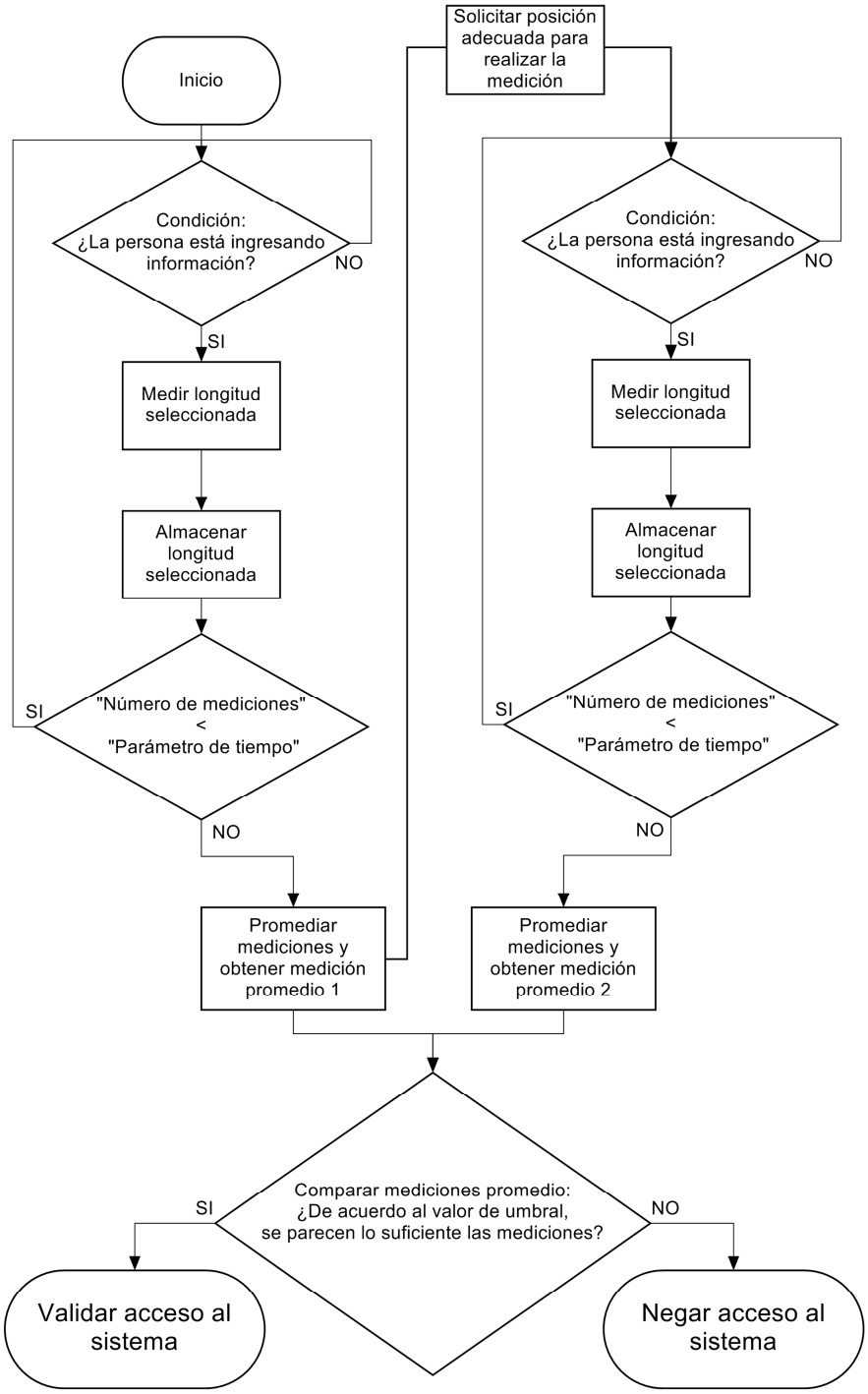

Figura 3.10 Diagrama que representa la forma de operar del algoritmo para el método de medición de longitudes del esqueleto.

Las mediciones se efectúan únicamente mientras la persona se encuentra en alguna de las posiciones válidas de brazos.

CAPÍTULO 4. IMPLEMENTACIÓN DEL SISTEMA Y RESULTADOS

Para el desarrollo de la aplicación se siguió el siguiente procedimiento:

1. Abrir el ambiente de programación Visual Studio 2010
2. Crear un nuevo proyecto eligiendo Visual C# y Aplicación WPF y dando un nombre al proyecto. Se crean 2 archivos importantes: el archivo que define el aspecto visual de la aplicación .xaml y el código que define la actividad .xaml.cs.
3. Agregar las librerías necesarias, añadir los ensamblados Microsoft.Kinect.dll y Microsoft.Speech.dll como referencia al proyecto
4. Declarar los espacios de nombres apropiados
5. Comenzar a desarrollar la aplicación.

4.1 Implementación e integración de los diferentes métodos de autenticación

A continuación se describirá el código que fue necesario para la realización de los algoritmos de todos los métodos seleccionados como un programa.

Primero se agregaron las librerías necesarias, se refirió al archivo Microsoft.Kinect.dll y al archivo Microsoft.Speech.dll. Para hacer las clases de las librerías más sencillas de utilizar se agregaron las siguientes líneas, los espacios de nombres, al inicio del código:

```
using Microsoft.Kinect;
using Microsoft.Speech.AudioFormat;
using Microsoft.Speech.Recognition;
using System.IO;
```

Las variables se fueron definiendo (e inicializando en algunos casos) conforme se necesitaron al ir construyendo el código, fuera de todos los métodos cuando se requirió que estuvieran accesibles a todos los métodos. Las variables fueron de diversos tipos: enteros, flotantes, cadenas y arreglos de cadenas. En el código 0, localizado en el anexo 1, se encuentran las variables utilizadas.

A continuación se creó el método NivelS para asignar el valor del control deslizante que controla el nivel de seguridad, se convirtió a entero y se mostró en el bloque de texto textBlock1, la leyenda “Ajuste con el control deslizante el nivel de seguridad: Baja/Media/Alta”. Cada vez que se cambia el valor del control deslizante se recurre a este método. En el anexo se encuentra el código 1 que se relaciona con el control deslizante.

En cuanto se carga la ventana, se recurre a los métodos: disponSensor y habilitarFlujEsqueleto. El método disponSensor verifica que se encuentre algún sensor Kinect y de no encontrar el sensor, muestra en una caja de mensaje, la leyenda de error “Conectar el sensor Kinect". De haber sensor Kinect, obtiene el primero y lo nombra: miKinect. El código 2 corresponde al método disponSensor.

El método habilitarFlujEsqueleto inicia el sensor, habilita el flujo de datos del esqueleto y establece el modo: sentado o normal. Cuando se habilita el flujo de datos del esqueleto, éste puede generar un evento cada vez que tiene nuevos datos de esqueleto para el programa. De no tener éxito en habilitar los flujos necesarios e iniciar el sensor, así como establecer el modo, muestra un mensaje de error y apaga la aplicación. En el código 3 se encuentra el método habilitarFlujEsqueleto.

El método calcDistancia se utilizó para calcular la distancia entre dos articulaciones en tres dimensiones, éste utilizó el teorema de Pitágoras para el caso de tres dimensiones. En el código 4 se encuentra el método calcDistancia.

A continuación se creó el botón “BLOQUEAR” y el evento asociado a éste. Al momento de dar clic en el botón, se oculta éste y el control deslizante; se muestra la leyenda “Favor de elegir un círculo en el piso y una posición de las siguientes: (IMPORTANTE: Mantener la posición hasta siguiente instrucción)”; se muestra la imagen de la Figura 3.5; y se conecta al evento que se dispara cuando hay nuevos cuadros del flujo de datos del esqueleto disponibles. El método miKinect.SkeletonFrameReady está conectado al evento cuadroEsqueletoListo y correrá cada vez que el código de seguimiento de esqueleto ha terminado el análisis de la escena en frente del sensor. En el código 5 se encuentra el código correspondiente al botón “BLOQUEAR”.

El evento cuadroEsqueletoListo es de gran importancia, ya que es aquí donde se lleva a cabo el procesamiento de la información del flujo de datos del esqueleto y donde se implementan los algoritmos relacionados con la información del flujo de datos del esqueleto.

Los puntos correspondientes a las posiciones de las articulaciones del esqueleto se localizan de forma diferente a los datos de profundidad y de video. Tanto el conjunto de datos de profundidad como el de video se definen en un plano coordenado geométrico y son medidos en pixeles, con un valor de cero en los ejes X e Y en la esquina superior izquierda. En el espacio de profundidad, la dimensión Z es medida en milímetros. El espacio del esqueleto es medido en metros y todas las articulaciones son representadas en tres dimensiones (x, y, z), en un sistema coordenado en donde el eje Z, apuntando hacia el individuo, es el producto cruz de los ejes X e Y. El eje Y aumenta hacia arriba, mientras que el X aumenta hacia la derecha. Las posiciones cero, para los ejes X e Y, se encuentran en el centro del sensor de profundidad. El eje X varía de -2.2m a 2.2m, mientras que el eje Y varía de -1.6m a 1.6m y el eje Z va de 0m a 4m [3] [22]. La Figura 4.1 ilustra los espacios coordenados para los diferentes flujos de datos.

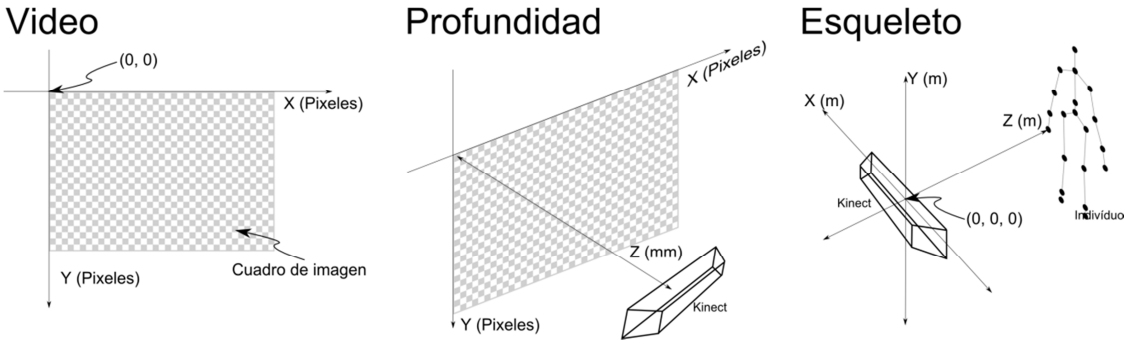

Figura 4.1 Espacios coordenados para los diferentes tipos de datos.

Cada posición se representa por medio de una estructura Joint en el kit de desarrollo de software, usando tres propiedades principales: (1) JointType, (2) Position y (3) TrackingState [3].

Lo primero que hace el código en el método cuadroEsqueletoListo es revisar los datos de esqueleto para encontrar qué se identificó en la escena de haberse encontrado algo. Después busca para cada esqueleto y si se detecta un esqueleto, se obtiene la instancia Joint que describe la posición de las diferentes partes del esqueleto a partir de la colección de articulaciones que provee la clase Skeleton. En el código 6 se encuentra el código que corresponde a esta primer parte del método cuadroEsqueletoListo.

El SDK contiene una clase (colección) llamada JointCollection, de modo que se usa el valor del tipo JointType para indexar la colección y obtener la información acerca de determinada articulación en ella. Por ejemplo, para la mano derecha se obtiene el valor JoinType.HandRight en donde la variable dJoint contiene la información de la

articulación que describe a la mano derecha, incluyendo su posición del tipo SkeletonPoint: Joint dJoint = esqueleto.Joints[JointType.HandRight]

Luego para obtener la posición de la información de la articulación de la mano derecha: SkeletonPoint dJointPosition = dJoint.Position

Y para obtener el valor, por ejemplo, de la coordenada z de la posición de la mano derecha: dJointPosition.Z

Para continuar con el método cuadroEsqueletoListo e implementar los algoritmos se continuó con el código 7.

Puede ocurrir que no todas las articulaciones sean visibles para el sensor debido a un obstáculo, que puede ser un objeto o el mismo cuerpo del individuo. En este caso se tiene que no hay objetos enfrente del sensor y el individuo no adopta posturas de forma que se obstruyan regiones de su cuerpo, así que la propiedad TrackingState permanece en estado Tracked para las articulaciones del esqueleto, lo que significa que el sensor tiene una vista clara de la posición de las articulaciones y las tres coordenadas son capturadas adecuadamente.

Si alguna de las variables utilizadas para contar, como por ejemplo DLIA o centro, llegan al valor de tiempoSilla, que es el parámetro que se utilizará para determinar el tiempo necesario para validar las posiciones y entre diferentes etapas, se recurre al método "variablespaso". Para una longitud dada, la variable distancia = "valor de longitud promedio de j mediciones de longitud". El método variablespaso asigna las variables de la forma mostrada en la tabla 1.

TABLA I: ASISGNACIÓN DE VARIABLES

Selección	Variable	Valor de variable
Círculo 0	pasoSilla	0
Círculo 1	pasoSilla	1
Círculo 2	pasoSilla	2
Posición de brazos 0	pasoBrazo	0
Posición de brazos 0	pasoBrazo	1
Posición de brazos 0	pasoBrazo	2
Posición de brazos 0	pasoBrazo	3

Además de mostrar en pantalla la posición seleccionada, el círculo/lugar seleccionado y recurrir al método pasoReconHabla. El código correspondiente al método variablespaso se encuentra en el anexo como código 8.

El método pasoReconHabla muestra la leyenda: "Diga en voz alta “m” palabras de la tarjeta", donde el valor de m depende del valor del control deslizante y por lo tanto del nivel de seguridad elegido. A mayor seguridad mayor número de palabras. También, el método pasoReconHabla oculta la imagen que se encontraba visible, deshabilita el flujo de datos del esqueleto y recurre al método ReconocimientoHablaConf. El método pasoReconHabla se encuentra como código 9 en el anexo.

El método ReconocimientoHablaConf recurre al método encontrarInfoRec en el que se busca el paquete de lenguaje y de no encontrarse se muestra la leyenda: "Error en reconocimiento de habla". De encontrarse el paquete de lenguaje, se intenta cargar el motor de reconocimiento de habla y si no se tiene éxito en cargar el motor de reconocimiento, se muestra la leyenda: "No pudo ser cargado el motor de reconocimiento de habla". A continuación se configuran las palabras que podrán ser reconocidas o válidas para el sistema, usando la clase Choices, creando una colección de palabras que serán reconocidas. Para agregar palabras, se crea la variable “palabras” para almacenar las palabras que se podrán reconocer: `Choices palabras = new Choices();`

Luego se agregan las palabras: `palabras.Add("Purpura"); palabras.Add("Azul"),…`

Después se utiliza la clase GrammarBuilder para crear la gramática del lenguaje que se esté utilizando, creando una GrammarBuilder llamada constructorGramatica. Enseguida se usa para crear un valor gram que se carga en el reconocedor.

Por último se configura el audio. En el código 10 (Ver anexo) se permite acceder al flujo de audio y establecer el modo de ángulo del haz como adaptativo. Los valores posibles de este modo del arreglo de micrófonos son los siguientes [16]:

- Manual. La aplicación selecciona el haz
- Automático. El sistema selecciona el haz
- Adaptativo. Un localizador de fuente interno selecciona el haz.

La clase KinectAudioSource tiene diversas propiedades para controlar el flujo de audio. Por ejemplo, para obtener la dirección de la fuente de audio, se recurriría a la propiedad

KinectAudioSource.SoundSourcePosition, la cual corresponde a un valor que representa a un ángulo, en radianes, relativo al eje z de la cámara, que es perpendicular al sensor Kinect. También es posible obtener el nivel de confianza de este valor. La clase SpeechRecognitionEngine permite el reconocimiento del habla.

El código correspondiente a los métodos ReconocimientoHablaConf y encontrarInfoRec se encuentra en el anexo como código 11. Al final del código 11 se muestra el evento SpeechRecognized, el cual se ejecuta cada vez que se detecta una palabra. En el método palabraReconocida, se revisa el valor de confianza del resultado obtenido. La propiedad Result es del tipo clase RecognitionResult y contiene lo que se ha identificado y la calidad en términos de nivel de confianza. La propiedad Confidence es un valor flotante que va del 0 al 1 y es asignado por el reconocedor. Si el valor es menor a 0.9 (90%) se muestra en pantalla la leyenda: “Favor de repetir la palabra” y se borra el texto en el bloque de texto textBlock3. Si el valor es mayor a 0.9 (90%), se considera un reconocimiento correcto de la palabra, se muestra en pantalla, se almacena en la variable palabratemp, se cambian los valores de algunas variables de conteo y se continúa con el método palRec. El código correspondiente al método palabraReconocida se muestra en el anexo como código 12.

Los métodos palRec, bloqY, varAlmacenar, resetVar1, clicFalso y fin() permiten continuar con el proceso de autenticación. El programa se bloquea después de tres intentos fallidos y muestra en pantalla que el sistema se encuentra bloqueado, representando esto que el sistema ha negado el acceso al sistema. El código comentado, correspondiente a los métodos palRec, bloqY, varAlmacenar, resetVar1, clicFalso y fin(), se encuentra en el anexo como código 13. El código .xaml de la interface de usuario se encuentra en el anexo como código 14.

4.2 Integración y operación del sistema

El diagrama de la aplicación que integra los métodos de autenticación seleccionados, tomando en cuenta la interfaz de usuario, y que corresponde al software del sistema de autenticación que interpreta los diferentes parámetros y valida el acceso al sistema o lo niega, se encuentra en la Figura 4.2:

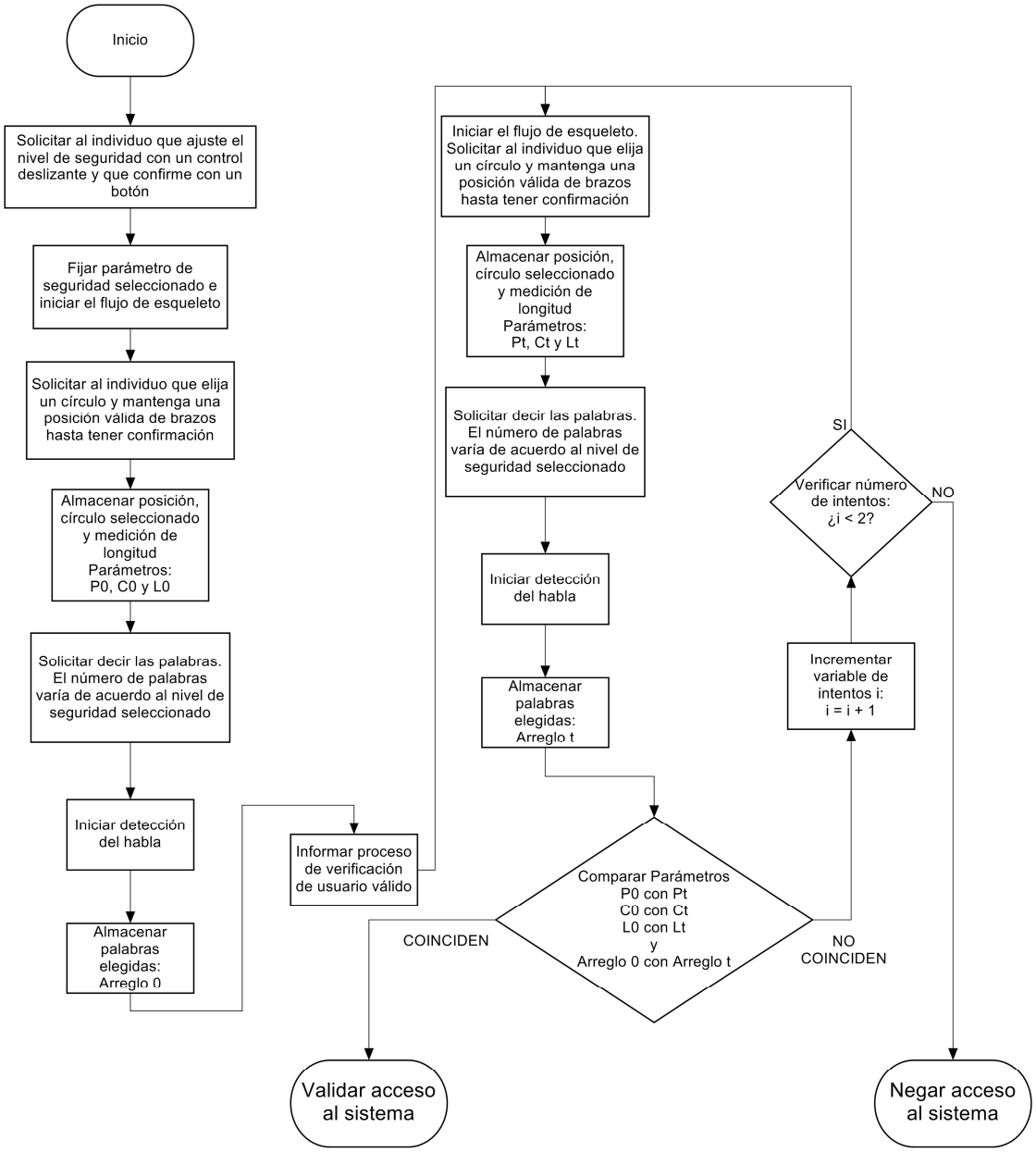

Figura 4.2 Diagrama que representa la forma de operar de la aplicación que integra los métodos de autenticación seleccionados.

El programa opera de la forma siguiente:

Al iniciar el programa se muestra en pantalla la leyenda “Ajuste con el control deslizante el nivel de seguridad: Alta/Media/Baja“. Del lado derecho se muestra un control deslizante que permite cambiar el nivel de seguridad entre: Alta, media o baja. Al cambiar el control deslizante cambia la leyenda en pantalla para indicar la configuración seleccionada. Al centro de la pantalla se encuentra un botón nombrado “BLOQUEAR”.

Esto indica que se debe ajustar el nivel de seguridad con el control deslizante y utilizar el botón “BLOQUEAR” para confirmar (Ver Figura 4.3).

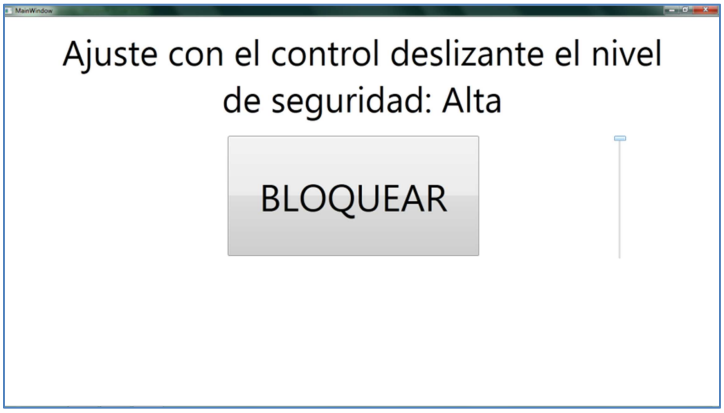

Figura 4.3 Captura de pantalla al iniciar el programa.

Una vez usado el botón “BLOQUEAR”, se muestra en pantalla la leyenda “Favor de elegir un círculo en el piso y una posición de las siguientes: (IMPORTANTE: Mantener la posición hasta siguiente instrucción)” y una imagen en la parte de abajo que muestra las posiciones de los brazos posibles (Ver Figura 4.4).

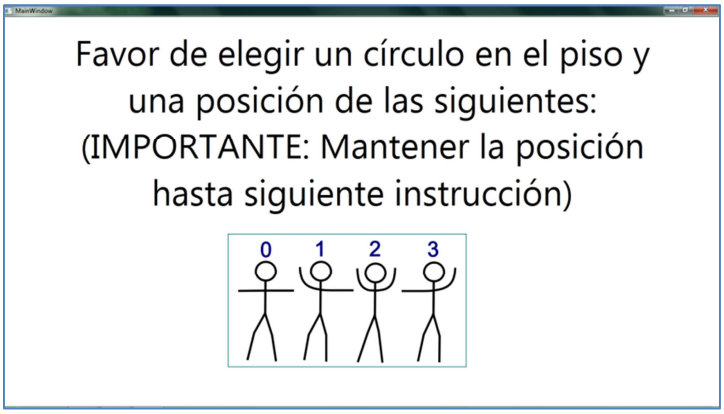

Figura 4.4 Captura de pantalla una vez usado el botón “BLOQUEAR”.

El individuo se coloca frente al equipo a una distancia de 1.9m, respecto al sensor, dentro de alguno de los círculos marcados en el piso, dándole al individuo la menor cantidad de información posible. Si la persona frente al sensor desconoce el mecanismo para ingresar la contraseña (estirar ambos brazos), y por lo tanto el sistema no responde, se niega el acceso y de esta forma se constituye una primer barrera de seguridad. Quienes están familiarizados con este sistema, debido a una previa

capacitación para su uso y el conocimiento del lugar adecuado, así como de la posición válida y correspondiente, podrán sortear esta primera barrera.

Una vez que el individuo elige una posición y un círculo y los mantiene un breve periodo de tiempo, se muestra en pantalla su selección de lugar y de posición de brazos con un número y se muestra la leyenda "Diga en voz alta 3/4/5 palabras de la tarjeta". Si el usuario seleccionó un nivel de seguridad bajo, medio o alto la leyenda solicita 3, 4 o 5 palabras respectivamente (Ver Figura 4.5).

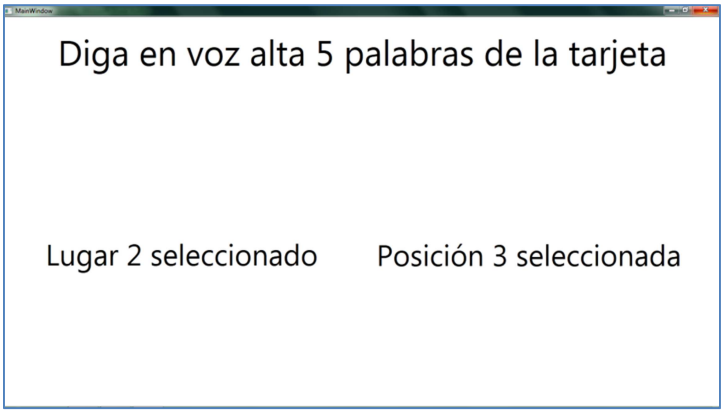

Figura 4.5 Captura de pantalla en donde se muestra el lugar seleccionado y la posición seleccionada, además de solicitar las palabras.

Cuando el individuo dice la primera palabra, esta se muestra en pantalla (Ver Figura 4.6).

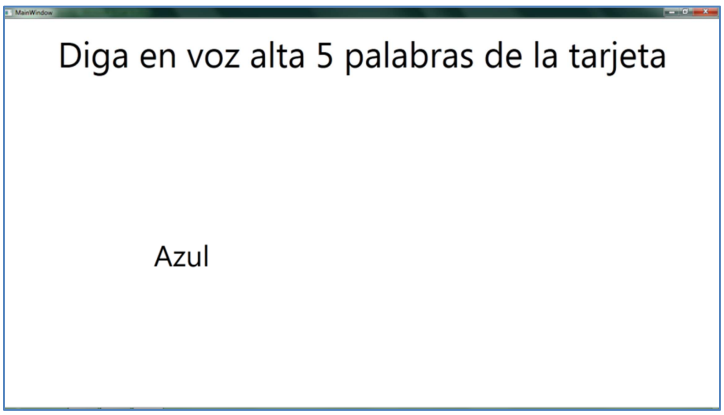

Figura 4.6 Captura de pantalla donde se muestra la palabra mencionada.

Si la palabra no se encuentra entre las válidas, no es reconocida por mala pronunciación, por mucho ruido, o cualquier otra causa, entonces se solicita que sea repetida con la leyenda "Favor de repetir la palabra" (Ver Figura 4.7). En caso de reconocimiento

correcto de la palabra, aunque no necesariamente un ingreso correcto, se muestra en pantalla la palabra reconocida.

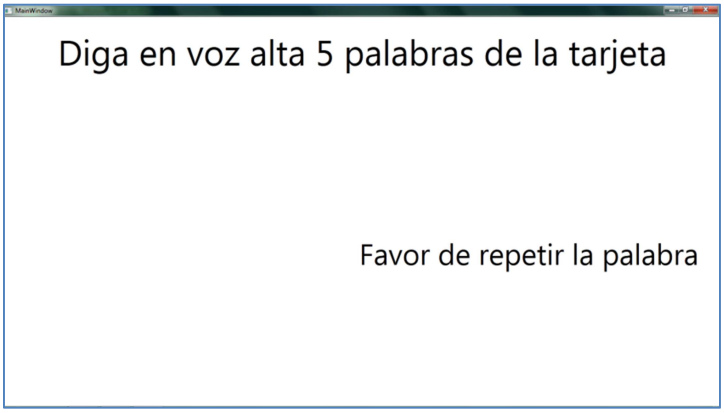

Figura 4.7 Captura de pantalla donde se muestra la leyenda “Favor de repetir la palabra”.

Cuando la persona menciona las palabras requeridas de forma exitosa, se muestra en pantalla “SISTEMA BLOQUEADO. Para desbloquear favor de realizar PRIMER INGRESO”. En la tarjeta se encuentra la información sobre el primer ingreso, sin embargo, esta vez no se le da información al usuario sobre cómo se desbloquea el sistema para mejorar la seguridad de éste (Ver Figura 4.8 y 4.9).

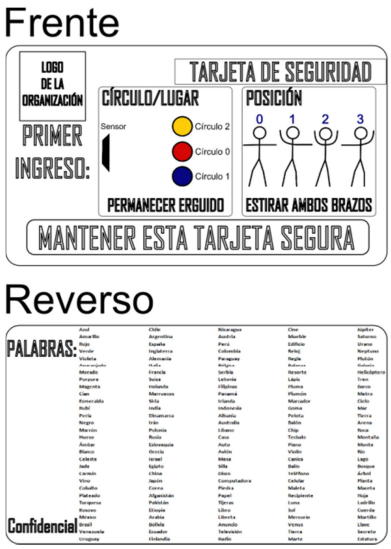

Figura 4.8 Tarjeta con información necesaria para la autenticación.

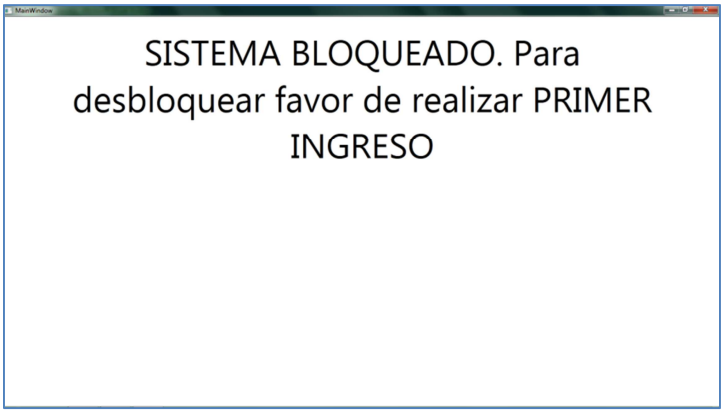

Figura 4.9 Captura de pantalla donde se muestra la leyenda "SISTEMA BLOQUEADO. Para desbloquear favor de realizar PRIMER INGRESO".

El individuo recurre a la tarjeta en la que se encuentra la información sobre el primer ingreso y elige una posición y un círculo y los mantiene un breve periodo de tiempo, se muestra en pantalla su selección de lugar y de posición de brazos con un número y se muestra la leyenda "Diga en voz alta 3/4/5 palabras de la tarjeta".

Si la persona elige un círculo, una posición de brazos, o palabras diferentes o si la medición de la longitud no corresponde a la de esa persona, se muestra en pantalla la leyenda "FALLÓ. Intente de nuevo. Para desbloquear favor de realizar PRIMER INGRESO".

Se tienen tres oportunidades para lograr validar el acceso, de no tener éxito y agotar las oportunidades, se muestra en pantalla la leyenda "BLOQUEADO. Intentos agotados" (Ver Figura 4.10). En caso de tener éxito y validar el acceso la aplicación se cierra. Se intentó que se bloqueara la computadora además de mostrarlo en pantalla, pero el sistema operativo no permite mostrar imágenes de la interfaz de usuario del programa mientras se encuentra bloqueado, de modo que se recurrió a la simulación de este estado. Sin embargo, el desarrollo del sistema de bloqueo no forma parte de este sistema.

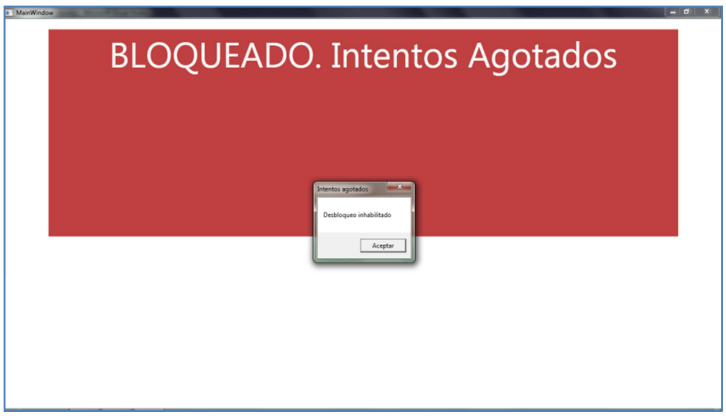

Figura 4.10 Captura de pantalla donde se muestra la leyenda que se obtiene al agotar las oportunidades para lograr validar el acceso.

4.3 Ajustes de parámetros, pruebas y resultados

En esta sección se encuentran los ajustes finales de los parámetros y los resultados, tanto cualitativos como cuantitativos, de la evaluación del sistema y se comparan con algunos de los sistemas existentes en el mercado.

Para el método de autenticación por medición de una longitud, se eligió considerar la longitud obtenida de la siguiente forma:

$$\begin{aligned} "Longitud" = (&"Distancia\ de\ hombro\ derecho\ a\ centro\ de\ hombros" \\ &+ "Distancia\ de\ centro\ de\ hombros\ a\ hombro\ izquierdo" \\ &+ "Distancia\ de\ codo\ izquierdo\ a\ hombro\ izquierdo" \\ &+ "Distancia\ de\ codo\ derecho\ a\ hombro\ derecho") \end{aligned}$$

El diagrama que muestra esta longitud se encuentra en la Figura 4.11.

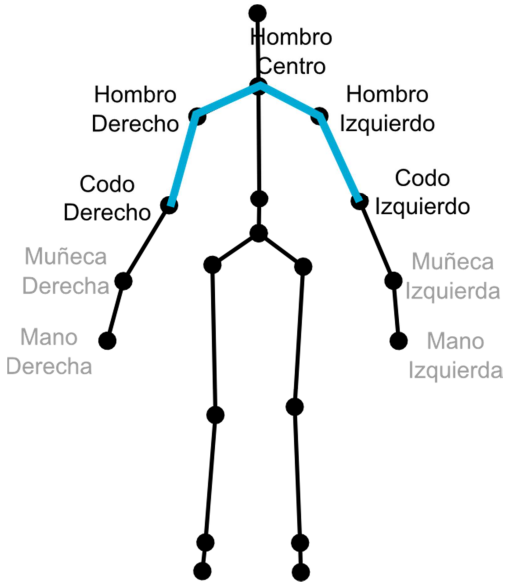

Figura 4.11 Forma de medir la longitud. Longitud mostrada en color azul.

Este valor de longitud se obtiene para cada cuadro del flujo esqueleto y se promedia con todos los obtenidos durante el periodo de reconocimiento de la posición de brazos.

El método biométrico desarrollado en este sistema permite diferenciar a personas que sean lo suficientemente diferentes en sus distancias entre articulaciones, pero como método único para distinguir entre dos personas cualesquiera no es suficiente.

Las manos y muñecas se excluyeron de la medición de longitud porque los valores de las posiciones fluctúan demasiado y no se consiguen diferencias de valores, entre diferentes personas, útiles para su empleo en el sistema.

Se hicieron pruebas considerando la estatura del individuo pero nuevamente las variaciones en las mediciones son muy grandes. Por ejemplo, si la persona trae algún sombrero o gorro, esto afecta la medición de la estatura. De igual forma, si la persona usa zapatos muy altos, este valor cambia. Por lo tanto, esta medición de longitud quedó descartada.

También se intentó medir diferentes partes del esqueleto y realizar diversas comparaciones usando proporciones entre ellas, pero se encontró que estos valores de las proporciones no varían de forma significativa entre diferentes personas.

Todos los sistemas biométricos están sujetos a errores intrínsecos, pero es posible calcular probabilidades de error. Las mediciones más comunes, como criterios para la evaluación de la autenticación biométrica y describir la exactitud real de un algoritmo biométrico, son las siguientes:

- Tasa de falsa coincidencia o FMR por sus siglas en inglés. Es la razón entre el número de muestras que se consideran coincidentes sin serlo y el número total de pruebas
- Tasa de falso negativo o FNMR por sus siglas en inglés. Es la razón entre el número de muestras que son coincidentes pero que no se consideran coincidentes por el sistema y el número total de pruebas

Para realizar una evaluación del método biométrico propuesto se obtuvieron estas dos mediciones para diferentes valores de umbral, con 30 muestras de registro para cada valor de umbral. Las mediciones se realizaron, en la posición de brazos 0 y círculo 0, con dos personas de diferente estatura: 1.71m y 1.65m. En las tablas II y III se encuentran las mediciones obtenidas. En la figura 4.12 se encuentra la gráfica obtenida a partir de estos datos, en esta se muestran las tasas de error como función del valor de umbral.

TABLA II: TASA DE FALSO NEGATIVO

Tasa de falso negativo	
Umbral de decisión (cm)	Tasa de falso negativo (%)
0.5	80.0
1.0	73.3
2.0	56.7
3.0	40.0
4.0	16.7
5.0	10.0
7.0	3.3

TABLA III: TASA DE FALSA COINCIDENCIA

Tasa de falsa coincidencia	
Umbral de decisión (cm)	Tasa de falsa coincidencia (%)
3.0	3.3
4.0	6.7
5.0	13.3
7.0	20.0
10.0	33.3
15.0	70.0

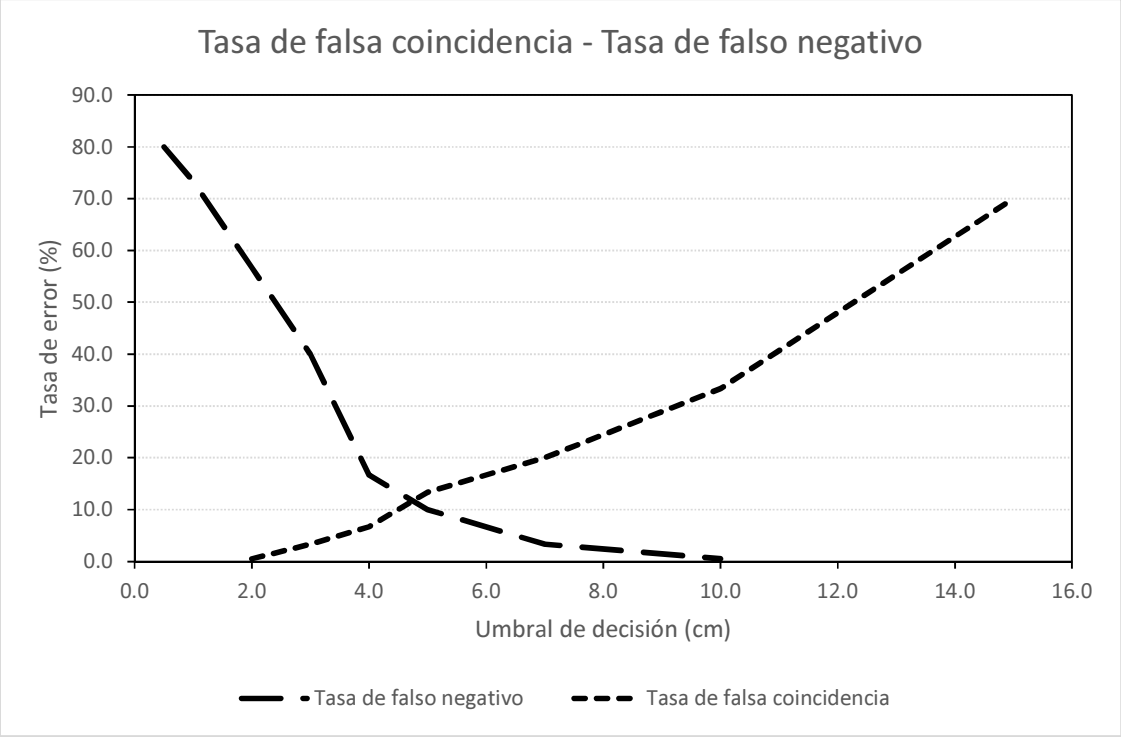

Figura 4.12 Gráfica de las tasas de error como función del valor de umbral.

Como se puede observar en la gráfica, a la disminución de la tasa de falso negativo, corresponde el aumento en la tasa de falsa coincidencia. Un punto significativo, es en el cual FMR es igual a FNMR, llamada tasa de error igual o EER por sus siglas en inglés. En este caso la tasa de error igual se encuentra entre los valores de umbral 4 y 6. Sin embargo no existe justificación práctica para operar un sistema de autenticación biométrico a exactamente ese punto [23]. Un valor alto de FNMR causa descontento para los usuarios auténticos, ya que se incrementa la probabilidad de falsos rechazos, mientras que un valor alto de FMR conduce a un aumento en la probabilidad de falsa aceptación, disminuyendo la efectividad del sistema. En vista de que este método biométrico es sólo uno de los utilizados en este sistema multifactor, se decidió disminuir la probabilidad de falso rechazo, estableciendo un umbral de 10cm para la comparación de promedios de longitudes.

Debido a que la medición de longitud se lleva a cabo mientras se detecta la posición de brazos, la comparación entre las mediciones de longitud se realiza entre dos mediciones tomadas con la misma posición de brazos y en el mismo círculo. Aun así, se realizó la medición de longitud en las diferentes posiciones de los brazos con el fin de conocer si estos valores cambiaban de forma significativa. En la tabla IV se encuentran las diferentes mediciones de longitud en las diferentes posiciones de los brazos.

TABLA IV: MEDICIONES DE LONGITUD EN DIFERENTES POSICIONES

Valor de longitud medido (m)	Medición de longitud en posición 0 (m)	Medición de longitud en posición 1 (m)	Medición de longitud en posición 2 (m)	Medición de longitud en posición 3 (m)	Máxima diferencia respecto a valor medido (m)
0.92	0.891	0.879	0.881	0.829	0.091

El sistema de autenticación fue ajustado para que funcionara adecuadamente. El método que impide una autenticación más confiable es el biométrico, ya que distinguir entre personas de dimensiones similares, con el sensor Kinect actual y utilizando el método de medición de longitud propuesta, tiene gran dificultad. En la tabla V se muestra una comparación cualitativa de los métodos seleccionados.

TABLA V: COMPARACIÓN CUALITATIVA DE LOS MÉTODOS

Método / Característica	Medición longitud (Método 4)	Posición círculo (Método 2)	Voz habla (Método 1)	Posición brazos (Método 3)	Tarjeta física (Método 5)
Confiabilidad	No	Sí	Sí	Sí	Sí
Facilidad de uso	Sí	Sí	Sí	Sí	Sí
Costo Adicional	No	No	No	No	No
Factor sorpresa	Sí	Sí	Sí	Sí	Sí
Requiere tiempo adicional	No	No	Sí	No	No
Alto número de configuraciones posibles	No aplica	No	Sí	No	Complementaria
Interferencias	Precisión y exactitud del sensor	No atender instrucciones	Ruido, pronunciación, eco	No atender instrucciones	Daño, extravío
Requerimiento de espacio adicional	No	No	No	No	No
Posibilidad de mejoría en el futuro	Sí	Sí	Sí	Sí	Sí
Posibilidad de reconocimientos falsos	Sí	No	Sí	No	No aplica
Afectación o interferencia con las demás Métodos	No	No	No	No	No
Ventaja principal / Aporte	Autenticación biométrica	Facilidad de uso	Número de configuraciones posibles	Conocimiento / Factor sorpresa	Llave física

Si se aumenta la tolerancia a diferencias en las mediciones de longitud, se incrementa la probabilidad de falsa aceptación. Aunque los demás métodos contribuyen para aumentar la robustez de la autenticación, esto disminuye la efectividad del sistema. Si por el contrario, se disminuye la tolerancia a diferencias en las mediciones de longitud, se incrementa la probabilidad de falso rechazo, lo cual conduce a un mayor descontento de los usuarios del sistema.

Existe la posibilidad de mejorar cada uno de los métodos aquí elegidos. Por ejemplo, se ha considerado la viabilidad de utilizar el sensor Kinect para obtener y usar los puntos de las posiciones del esqueleto de sujetos caminando para la identificación biométrica, considerando la contribución de diferentes combinaciones de partes del cuerpo al proceso de identificación [11].

El umbral para la posición de la cabeza se obtuvo al definir el diámetro que deben de tener los círculos marcados en el suelo para que una persona pueda caber dentro y las diferentes distancias requeridas. Para el círculo central se definió un valor en el eje X entre -0.2 y 0.2, que corresponde a un intervalo entre -20 cm y 20 cm. Para los otros dos círculos se seleccionaron los intervalos complemento. Por simplicidad de implementación se eligieron tres círculos, ya que de esta forma se admite un rango de error mayor en la posición de las marcas de los círculos en el piso. Además, el aporte en el nivel de seguridad tanto del método de posiciones de los brazos como del de la elección del círculo, es adicional y su principal fortaleza es la facilidad de uso y su novedad. La seguridad, desde el punto de vista de la cantidad de configuraciones posibles, la aporta el método de reconocimiento del habla. Por facilidad de uso, es conveniente limitar la cantidad de ingresos válidos, tanto en el método de posición de brazos como en el de selección de círculo, y aumentar el número de palabras válidas del método de reconocimiento del habla así como el número de palabras requeridas para la autenticación. En la tabla VI se muestran los diferentes números de configuraciones posibles para los métodos mencionados.

TABLA VI: NÚMERO DE CONFIGURACIONES POSIBLES

Número de ingresos	Número de configuraciones posibles		
	Posición círculo	Voz/habla	Posición brazos
1	3	150	4
2	9	22,500	16
3	27	3,375,000	64
4	81	506,250,000	256
5	243	75,937,500,000	1,024

Para seleccionar las posiciones de los brazos válidas, se tuvo un compromiso entre facilidad de ejecutarlas y la posibilidad de reconocimientos falsos. Para que no fueran difíciles de realizar, al mismo tiempo que se evitaran reconocimientos falsos, se ajustaron los parámetros experimentalmente de forma cuidadosa. Las posiciones se eligieron de forma que implicaran un conocimiento previo sobre cómo realizarlas, además de requerir mantenerlas durante cierto tiempo. El conocimiento necesario para su ejecución correcta consiste en que ambos brazos deben estar estirados, de lo contrario el sistema no permite el ingreso de información y de esta forma se niega el acceso (Ver Figura 4.13).

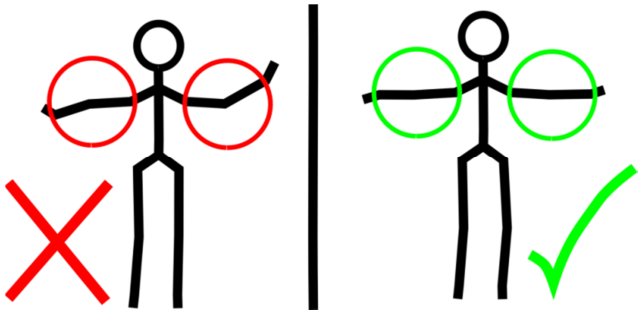

Figura 4.13 Ambos brazos deben estar estirados para que el sistema considere la posición.

Para el método de posiciones de los brazos es necesario liberar tres candados. El primer candado se libera si el individuo tiene tanto el brazo derecho como el izquierdo estirado. Para determinar si un brazo se encuentra estirado se compararon dos longitudes, si la diferencia es mayor a 3cm se considera que el brazo no se encuentra estirado. Si la diferencia es menor a 3cm se considera que el brazo se encuentra estirado. Este valor se determinó experimentalmente. Para determinarlo se mantuvo estirado el brazo, de forma cómoda, durante 1 minuto y se revisó que el sistema no lo registrara como no estirado durante ese intervalo de tiempo, buscando el valor de diferencia mínimo. Las longitudes se encuentran en la Figura 4.14.

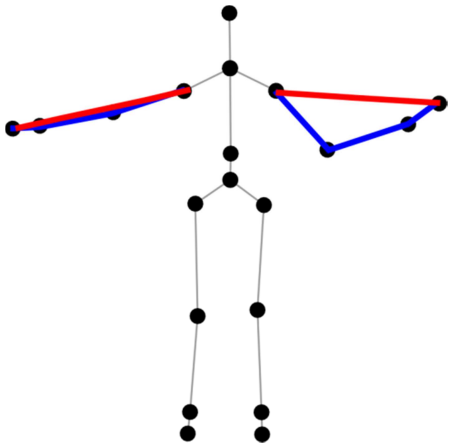

Figura 4.14 Forma de determinar si los brazos se encuentran estirados. La longitud mostrada en color azul es comparada con la mostrada en rojo.

Para liberar el segundo candado, ambos brazos deben encontrarse por encima de cierta altura con respecto a los hombros. Para conseguir esto se considera la coordenada Y de la posición del hombro y se compara con la coordenada Y de la mano, para ambos brazos. Es necesario que la diferencia entre la coordenada Y de la posición del hombro y la coordenada Y de la posición de la mano sea menor que 25cm para que se considere el brazo arriba. Este valor se determinó experimentalmente, a valores menores la tolerancia disminuye y a valores mayores la tolerancia aumenta. Nuevamente se encontró el valor más adecuado, ya que con este valor se garantiza que el usuario no adopte la posición adecuada al azar. En la Figura 4.15 se encuentran las dos longitudes que se comparan para liberar este segundo candado.

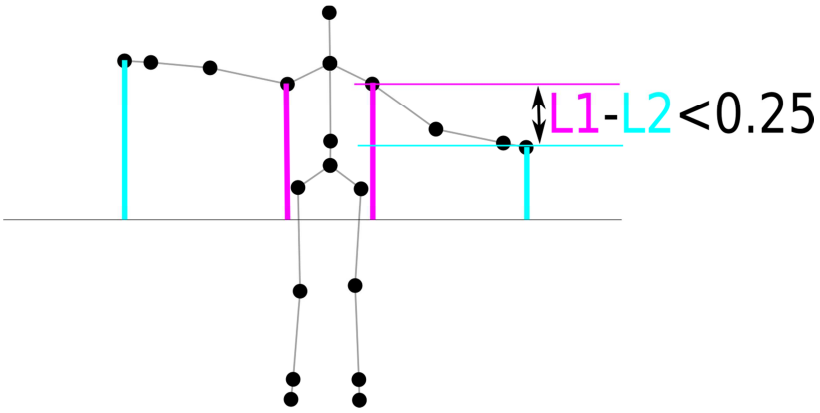

Figura 4.15 Forma de determinar si el brazo se encuentra por encima de cierta altura con respecto a los hombros.

Para liberar el tercer candado es necesario que se cumpla uno de los cuatro casos posibles para las posiciones de los brazos:

1. Posición 0: Ambos brazos a los lados u horizontales.
2. Posición 1: Brazo derecho arriba o vertical y brazo izquierdo a un lado u horizontal.
3. Posición 2: Ambos brazos arriba o verticales.
4. Posición 3: Brazo izquierdo arriba o vertical y brazo derecho a un lado u horizontal.

Para determinar si alguno de los brazos se encuentra a un lado se considera la coordenada Y del hombro y la coordenada Y de la mano del mismo brazo. Se calcula el valor absoluto de la diferencia entre la coordenada Y de la posición del hombro y la coordenada Y de la posición de la mano y se pide que éste sea menor que 30cm para que se considere el brazo a un lado. Se encontró que éste valor es adecuado para reconocer con certeza y estabilidad la posición. La Figura 4.16 muestra la condición necesaria para considerar alguno de los brazos a un lado.

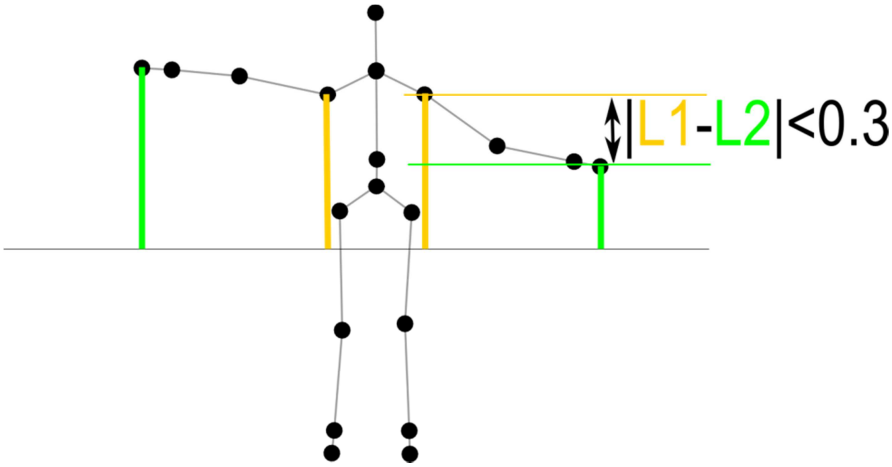

Figura 4.16 Forma de determinar si el brazo se encuentra a un lado.

Finalmente, para determinar si alguno de los brazos se encuentra arriba, se considera la coordenada X del hombro y la coordenada X de la mano del mismo brazo. Se calcula el valor absoluto de la diferencia entre la coordenada X de la posición del hombro y la coordenada X de la posición de la mano y se pide que este sea menor que cierto valor para que se considere el brazo arriba. Se seleccionó el mismo valor de 30 cm ya que es también un valor adecuado para este caso, análogo al anterior, debido a que se tiene simetría. La Figura 4.17 muestra la condición necesaria para considerar alguno de los brazos arriba.

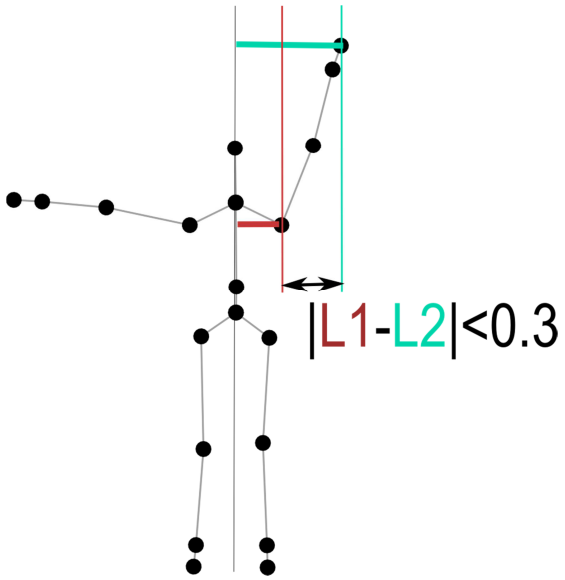

Figura 4.17 Forma de determinar si el brazo se encuentra arriba.

El tiempo que debe ser mantenida cualquiera de las posiciones, para ser considerada como una entrada o ingreso válido de información al sistema, es controlado por el "parámetro de tiempo".

Por otro lado, para el método de ingreso de palabras utilizando el reconocimiento del habla, el universo de palabras tuvo que ser seleccionado, de ser posible, evitando palabras que tuvieran la misma pronunciación en el idioma español. Algo que vale la pena mencionar es que la selección de palabras es susceptible del país en el que se hable el idioma. Por ejemplo, las palabras casa y caza en español de España se pronuncian diferente, mientras que en el español de México no. Esto fue verificado cambiando los paquetes de lenguaje en el código. También se comprobó que a mayor número de palabras válidas o reconocibles por el sistema, mayor dificultad para el reconocimiento y de que el sistema confunda unas con otras. Lo mismo ocurre al elegir palabras cuya pronunciación sea similar.

Experimentalmente se encontró que alrededor de 250 palabras son reconocidas de forma exitosa si son seleccionadas cuidadosamente para evitar palabras con pronunciación similar. Esto se hizo experimentalmente, mencionando cada una de las 250 palabras y cuando se requirió más de un intento para ser reconocidas o fueron reconocidas incorrectamente se cambiaron y se repitió el proceso. Al llegar a 300 palabras empiezan a ocurrir reconocimientos equivocados de palabras, aunque se seleccionen cuidando evitar su pronunciación similar. Debido a esto se seleccionó una lista de 150 palabras válidas, dejando un margen de tolerancia mayor, para que de esta

forma se puedan seleccionar las palabras con mayor libertad y de ser necesario en diversos idiomas. En la tabla VII se muestra la relación entre el número de palabras válidas y la dificultad para su reconocimiento.

TABLA VII: NÚMERO DE PALABRAS VÁLIDAS Y LA DIFICULTAD PARA SU RECONOCIMIENTO

Número de palabras válidas	Dificultad para el reconocimiento	Número de configuraciones posibles para 1 palabra	Número de configuraciones posibles para 2 palabras
250	Mayor	250	62,500
200	Menor	200	40,000
150	No	150	22,500

Para la elección final de las 150 palabras se realizó una primera selección de palabras, las cuales se encuentran en el anexo 2, en el inciso A. Posteriormente se fueron eliminando las palabras que tenían pronunciación similar una a una y cambiándolas por palabras de pronunciación diferente. Después del proceso de selección, se llegó a la lista de palabras contenida en el anexo 2, en el inciso B. Con el fin de conocer la probabilidad de que una palabra sea mal reconocida, se mencionaron 3000 palabras aleatoriamente de la lista de palabras del anexo 2, en el inciso B. Se encontró que para 3000 palabras fue necesario realizar 70 repeticiones y ocurrieron 6 errores, es decir, una probabilidad del 0.2% de reconocer equivocadamente una palabra.

En la tabla VIII se encuentran los resultados de realizar la selección de palabras. Se mencionaron las 150 palabras hasta que fueran reconocidas todas, contando el número total de veces que fue necesario repetirlas así como el número de veces que fueron confundidas o se realizaron reconocimientos equivocados.

TABLA VIII: SELECCIÓN DE PALABRAS

Palabras	Número total de palabras	Número de Repeticiones	Reconocimientos equivocados
Antes	150	25	7
Después	150	10	0

A partir de esta limitación se seleccionó el número de palabras para los diferentes niveles de seguridad. Para el menor nivel de seguridad se requieren 3 palabras y cada

una se puede seleccionar de entre 150 palabras válidas. De forma que, para el menor nivel de seguridad, se tienen 3,375,000 posibilidades para elegir una contraseña con este método. Para el nivel medio se requieren 4 palabras y se tienen 506,250,000 posibilidades. Para el nivel de alta seguridad se tienen 5 palabras y 75,937,500,000 posibilidades.

Si se hubiera permitido la selección de 2 palabras, se tendrían 22500 posibilidades, lo cual no representa una barrera de seguridad suficiente si se considera que es el método que aporta en mayor parte la robustez por método de contraseña o llave de información. Desde el punto de vista de eficiencia tampoco tiene sentido seleccionar pocas palabras ya que por cada palabra agregada se aumenta de manera exponencial el número de posibilidades. Por otro lado, con 5 palabras el número de posibilidades, y con ello el nivel de seguridad, se dispara. Sin embargo, más de 5 palabras hacen que el proceso de autenticación sea largo y fastidioso. Otro aspecto importante es que, a diferencia de las contraseñas en las que se trata de cadenas de caracteres en lugar de cadenas de palabras, las palabras son seleccionadas por el usuario de un modo que es difícil encontrar patrones en su selección. Con la indicación única de que no se repitan las palabras, es suficiente para que el usuario elija una contraseña fuerte.

Entre los parámetros elegidos se encuentra el “Parámetro de tiempo”, nombrado tiempoSilla en el código y que controla el número de cuadros del flujo que serán considerados. Si se disminuye este parámetro se hace más rápido el proceso, ya que el número de cuadros del flujo esqueleto necesarios para considerar una posición de brazos o de la elección del círculo, es menor. Sin embargo, al disminuir este valor, se corre el riesgo de obtener falsos reconocimientos de posición. Además de que, al disminuir el “Parámetro de tiempo”, es menor el número de mediciones de la longitud, de forma que se tiene un mayor error en el cálculo del promedio de la medición de longitud. Aumentar este valor hace que sea incómodo mantener la posición por tanto tiempo, además de aumentar el tiempo requerido para realizar la autenticación. Se eligió un “Parámetro de tiempo” correspondiente a 2.5 segundos o 75 mediciones de longitud. En la tabla IX se muestra el tiempo requerido según el número de cuadros o valor del parámetro de tiempo.

TABLA IX: PARÁMETRO DE TIEMPO

Número de cuadros	Tiempo requerido (segundos)
15	0.5
75	2.5
120	4

Un elemento importante para este sistema es la tarjeta. La tarjeta incluye información fundamental para realizar la autenticación, de forma que es necesario portarla. Esto hace que represente un método de autenticación del tipo de llave física. Esta tarjeta incluye información sobre qué es primer ingreso, un diagrama con las posiciones de los brazos y de círculos con el número que las identifica, indica que los círculos son un lugar así como que hay que permanecer erguido, nota que los brazos tienen que estar estirados, así como la lista de 150 palabras válidas para el sistema. La lista final de palabras válidas se incluye en el anexo en el inciso B, mientras que el diseño de la tarjeta se encuentra en la Figura 4.8. En la tabla X se encuentra el tipo de información contenido en la tarjeta así como su importancia para realizar una autenticación exitosa.

TABLA X: IMPORTANCIA DE LA INFORMACIÓN CONTENIDA EN TARJETA

Información	Importancia
Qué es primer ingreso	Conocer qué es lo que solicita el sistema en ese momento
Diagramas de posiciones de los brazos y círculos así como el número que las identifica	Facilitar memorización y ejecución adecuada así como confirmación de que el sistema ha registrado la información correcta
Que los círculos son un lugar	La persona debe conocer que ha de permanecer dentro de alguno de los círculos
Que hay que permanecer erguido	De no permanecer erguido el sistema podría no registrar el círculo correcto
Que los brazos tienen que estar estirados	Si ambos brazos no están estirados, el sistema no responderá
Lista de 150 palabras válidas	Aun suponiendo que la persona sabe que tiene que mencionar palabras en español, el diccionario de la Real Academia Española tiene unas 80,000 palabras, a las que hay que añadir unos 70,000 americanismos. En inglés, se encuentran fácilmente unas 350,000 palabras en el diccionario Oxford [24] [25].

Para evaluar la seguridad del sistema de autenticación completo es necesario considerar las configuraciones posibles así como la información contenida en la tarjeta. También es importante tomar en cuenta el hecho de que es necesario contar con la figura humana de ciertas dimensiones además de la capacidad de realizar movimientos de acuerdo a

las características fisionómicas del cuerpo humano así como la capacidad de pronunciar palabras con voz y de comprender lo que se solicita a cada paso del proceso de autenticación. En este caso el sistema se bloquea después de tres intentos fallidos e impide que la información biométrica sea conocida o transferida de una persona a otra, con lo cual incrementa su robustez.

Para obtener resultados cuantitativos del sistema, se considera el número de configuraciones posibles considerando todos los métodos incluidos en este sistema. En la tabla XI se encuentra el número de configuraciones posibles de acuerdo al nivel de seguridad seleccionado así como el aporte de cada uno de los métodos.

TABLA XI: CONFIGURACIONES POSIBLES DE ACUERDO AL NIVEL DE SEGURIDAD SELECCIONADO

Método	Nivel de seguridad Bajo (3 Palabras)	Nivel de seguridad Medio (4 Palabras)	Nivel de seguridad Alto (5 Palabras)
Brazos	4	4	4
Círculo	3	3	3
Habla	3,375,000	506,250,000	75,937,500,000
Conjunto	40,500,000	6,075,000,000	911,250,000,000

A partir de las configuraciones posibles, se calculó la probabilidad de acertar en tres intentos de acuerdo al nivel de seguridad. Para seguridad baja se tiene una probabilidad de $7x10^{-6}$ %, mientras que para el nivel medio se tiene una probabilidad de $5x10^{-8}$ % y para el nivel alto de $3x10^{-10}$ %.

Con los métodos de brazos, habla y elección de círculo se garantiza una probabilidad muy baja de que un intruso acierte. Por otro lado la tasa de falso negativo del método biométrico es menor al 3.3% con el umbral seleccionado.

También se realizó una medición del tiempo aproximado que se requiere para llevar a cabo la autenticación de acuerdo a los diferentes procesos involucrados. En la tabla XII se encuentra el tiempo requerido para realizar la autenticación de acuerdo al nivel de seguridad seleccionado.

TABLA XII: TIEMPO REQUERIDO PARA REALIZAR LA AUTENTICACIÓN

Proceso	Tiempo requerido (s) (Seguridad baja)	Tiempo requerido (s) (Seguridad media)	Tiempo requerido (s) (Seguridad alta)
Brazos, Círculo y Biométrico	≈3	≈3	≈3
Configurar voz	≈6	≈6	≈6
Reconocimiento de voz	≈4	≈6	≈8
Total	≈13	≈15	≈17

De forma que el tiempo requerido para realizar la autenticación de una persona con este sistema va de los 13 a los 17 segundos, si esta se lleva a cabo de forma exitosa. En caso de ser necesaria la repetición de palabras será necesario agregar por cada palabra el tiempo necesario para reconocer una palabra, que es de aproximadamente 2 segundos.

TABLA XIII: COMPARACIÓN TEMPORAL ENTRE VARIOS SISTEMAS DE AUTENTICACIÓN

Método de autenticación	Modelo/Tipo	Tiempo requerido (s)	Distancia (m)	¿Necesario contacto físico?
Geometría de la mano (Biométrico)	Hand Punch 3000 [26]	≈12	0	Sí
	GT-4000 [27]	≈10	0	Sí
Cajero automático: Tarjeta y PIN (Físico y de contraseña)	NCR [28]	≈20	0	Sí
	State Bank of India [29]	≈17	0	Sí
Reconocimiento de Iris	Aoptix InSight VM Iris [30]	≈2	2	No
	id1 [31]	≈3	2	No
	Fujitsu Iris Recognition [32]	≈3	0.3	No
Contraseña (Teclado)	Genérico	≈4 a ≈6	0	Sí
Huella digital	iPhone 6	<1	0	Sí
	AVI-FTA-507-C [33]	<1	0	Sí
	Galaxy S5	<1	0	Sí
Llave mecánica (Chapa y llave)	Genérico	≈4	0	Sí
Multifactor Kinect	Seguridad Baja	≈13	1.9	No
	Seguridad Alta	≈17	1.9	No

En la tabla XIII se encuentra una comparación temporal entre varios sistemas de autenticación actuales y el sistema desarrollado en este trabajo. Una de las ventajas principales del sistema desarrollado es que no se requiere contacto físico con el sistema para poder realizar la autenticación. Esta ventaja también la tienen los sistemas modernos de reconocimiento de iris, además de que el tiempo requerido para la autenticación es menor. Así que existen sistemas en el mercado que compiten con el desarrollado en el presente trabajo, sin embargo su costo es mucho mayor a pesar de contar con el respaldo de la empresa que los desarrolló. Otra desventaja es su consumo energético, por ejemplo el modelo Aoptix InSight VM Iris tiene un consumo de energía nominal de 650 W (100 W en espera) [34]. Por otro lado, el sistema de autenticación por huella digital modelo AVI-FTA-507-C de AVI TEC solamente soporta 1500 usuarios [35], mientras que utilizando el sistema desarrollado se podría soportar tantos usuarios como número de configuraciones posibles, ya que el almacenamiento de un equipo de cómputo moderno, en el que se basa, cuenta con suficiente capacidad para almacenar gran cantidad de datos, a diferencia de algunos de los sistemas de autenticación disponibles en el mercado actualmente.

Entre los posibles desarrollos a futuro se encuentra el incorporar el reconocimiento de locutor o persona a través de su voz, además del reconocimiento del habla, incrementar el número de posiciones, añadir gestos más complejos, mediciones más precisas de las partes del cuerpo humano e incluso utilizar mediciones y gestos con el movimiento de los dedos de las manos por ejemplo. La lista de mejorías posibles para este sistema es muy extensa.

4.4 Mecanismo adicional y desarrollos a futuro

Se desarrolló el mecanismo de captura de imagen, adicional al sistema de autenticación, con el fin de mejorar la seguridad y demostrar que existen otras posibilidades. Lo importante de esta adición es que no aumenta la dificultad de su utilización ni incrementa los costos, ya que utiliza el mismo sensor y el mismo equipo de cómputo.

El programa se opera de la misma forma, así que la adición del mecanismo de captura de imagen no altera la experiencia del individuo.

El funcionamiento de este mecanismo se consigue agregando el código necesario como se describe a continuación.

Primero se agregó el método captFoto(), incluido en el anexo como código 15, que activa el flujo de video y crea la conexión al manejador del evento.c

El manejador del evento se encuentra en el código 16 y permite obtener las imágenes para que en el momento que la variable tomarFoto tome el valor de verdadero, se copie el siguiente mapa de bits y se almacene en la ubicación que se establezca en el código.

Por último se agregan las líneas contenidas en el código 17 para elegir los momentos en los cuales será obtenida o eliminada la imagen.

Al agregar el código del método de captura de imagen, el programa realiza una captura de imagen de la persona frente al sensor, después del primer intento fallido, para la autenticación. Si se agota el número de intentos, el sistema se bloquea y almacena la imagen para su futura consulta acerca de la persona que intentó validar el acceso. En caso de tener éxito y validar el acceso, la aplicación se cierra y elimina la imagen previamente almacenada.

Este mecanismo de captura de imagen permite conocer la apariencia de la persona que intentó validar el acceso y que agotó sus tres intentos, para poder realizar una valoración posterior acerca de la situación particular con el uso de esta información adicional.

El diagrama del funcionamiento del mecanismo de captura de imagen se encuentra en la Figura 4.18.

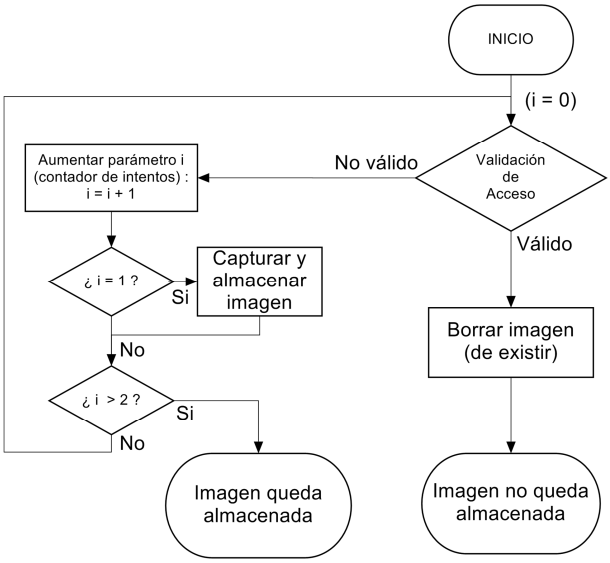

Figura 4.18 Diagrama que representa el funcionamiento del mecanismo de captura de imagen.

En la Figura 4.19 se puede observar la calidad de la imagen obtenida con este mecanismo y el sensor Kinect.

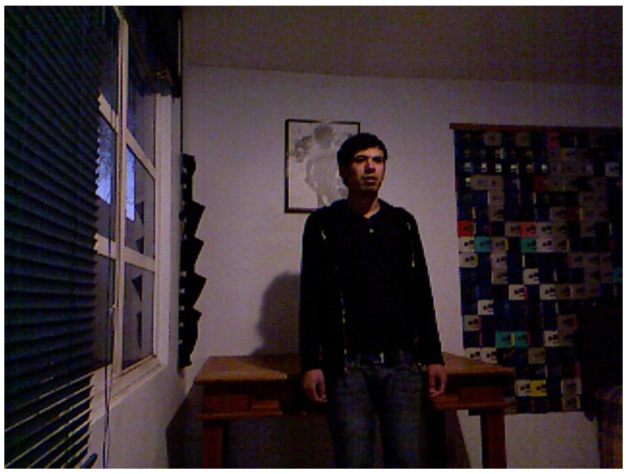

Figura 4.19 Imagen obtenida con el sensor Kinect.

Este mecanismo es solo un ejemplo funcional del gran abanico de posibilidades que permite este tipo de sistema de autenticación y prueba su posibilidad de expansión en el futuro con un mayor desarrollo.

Por ejemplo, una de las formas de vulnerar este tipo de sistemas de seguridad es siguiendo a la persona autorizada y que ésta sostenga la puerta de entrada al lugar para que entre el intruso. Comúnmente esto se previene creando un vestíbulo de seguridad, donde es necesaria la intervención de un operador. Para evitar esto, es posible crear un mecanismo simple, ya que el sensor permite detectar dos personas e incluso más. El mecanismo funcionaría de la siguiente forma: si hay más de una persona en la habitación, el sistema se bloquea automáticamente.

CONCLUSIONES

El objetivo planteado en este documento se cumplió de manera satisfactoria, ya que se logró desarrollar el sistema de autenticación multifactor utilizando un sensor Kinect que es robusto, fácil de utilizar y económico. Este sistema es novedoso; multifactor; no requiere contacto físico evitando el ingreso de virus, así como su modificación o daño; no es vulnerable debido al uso de una interface estándar común; evita el control del sistema sin que esto impida su funcionamiento; es robusto a errores en el sistema operativo, al sólo permitir ingresar la información relevante para su funcionamiento; y cuenta con la posibilidad de ser mejorado mediante software sin la necesidad de actualizar el hardware.

El proceso de la autenticación es simple pero sólido, aprovechando las características físicas del individuo y la universalidad de los movimientos permitidos por sus articulaciones, requiriendo su presencia física e información de la tarjeta. Los métodos utilizados se diseñaron de forma que en conjunto consiguieran una autenticación robusta, pero conservando la facilidad de uso y no interfiriendo entre sí de forma negativa.

Al ajustar cada uno de los métodos se logró un sistema de autenticación con 40,500,000 configuraciones posibles, que toma ≈13 segundos, en el nivel de seguridad mínimo y con 911,250,000,000 configuraciones posibles, requiriendo de ≈17 segundos, para el nivel de seguridad alto. Para aumentar el nivel de seguridad, es necesario conocer la información contenida en la tarjeta (posiciones válidas de los brazos, lista de palabras válidas, etcétera). Utilizando el sistema desarrollado se podría soportar tantos usuarios como número de configuraciones posibles, ya que el almacenamiento en un equipo de cómputo moderno, en el que se basa, cuenta con suficiente capacidad para almacenar gran cantidad de datos, a diferencia de algunos de los sistemas de autenticación disponibles actualmente en el mercado.

Se encontró que el número de cuadros por segundo máximo permitido por el sensor limita al sistema, existiendo un compromiso entre el tiempo requerido y la precisión de los datos que se pueden obtener. Si bien sus limitaciones están relacionadas con el hardware y las capacidades técnicas del sensor, como las distancias y ángulos de operación, existe la posibilidad de implementar un método biométrico diferente por medio de un desarrollo de software alternativo. También es posible mejorar, agregar, quitar, o combinar de forma diferente los demás tipos de métodos.

Algunas de sus ventajas se deben a que se trata de un sistema de autenticación nuevo, con un gran potencial de mejoría para compensar por el posible conocimiento de su funcionamiento. Con el avance de la tecnología será posible obtener mejores mediciones, disminuir los requerimientos de espacio, permitir la detección de movimientos más sutiles, aumentar el número de cuadros por segundo máximo y reducir el tiempo requerido para la autenticación, entre otras mejorías.

El potencial de desarrollo a futuro para este tipo de sistemas de autenticación multifactor utilizando un sensor Kinect es enorme, además de que se cuenta con la posibilidad de agregar funciones adicionales para mejorar el nivel de seguridad, como la captura de imagen del intruso o la negación de acceso si se encuentra presente más de una persona durante el proceso de autenticación.

Se recomienda que los desarrollos a futuro se centren en mejorar cada uno de los métodos por medio de software o con la utilización de dispositivos de próxima generación, sin que esto implique aumentar la dificultad de uso o reducir su confiabilidad. Es posible aumentar la complejidad de cada uno de los métodos utilizados, sin embargo debido a que su robustez y fortaleza de autenticación están basados en la utilización de varios métodos, es conveniente fijar la atención en la compatibilidad y refuerzo de seguridad entre los diversos métodos seleccionados, poniendo especial énfasis en evitar que la utilización de varios de ellos represente mayor dificultad de uso para el usuario o aumente el tiempo requerido para realizar la autenticación.

REFERENCIAS

[1]Fisch, Eric. *Secure Computers and Networks. Analysis, Design and Implementation*. CRC. 1999. PP. 1-4, 53-67.

[2]Killmeyer, Jan. *Information Security Architecture. An integrated Approach to Security in the Organization*. Auerbach. 2006. PP. 101, 114-116.

[3]Jana, Abhijit. *Kinect for Windows SDK Programming Guide*. Birmingham-Mumbai. Packt Publishing. 2012. PP. 7-18, 19, 21, 37-39, 47-53, 109, 121-123, 141, 157-162, 164, 188, 189, 237-247.

[4]Jiří Přinosil, Kamil Říha, Fu Dongmei, "Kinect Based Automated Access Control Systems", Department of Telecommunications, Brno University of Technology, Department of Automation, University of Science and Technology Beijing. Latest Trends in Information Technology, November 2012.

[5]Sean McSheehy, Erik Cowley, "Home Security Prototype Device", University of Massachusetts Lowell, May 2012.

[6]M. Martínez-Zarzuela, F.J. Díaz-Pernas, A. Tejero de Pablos, F. Perozo-Rondón, M. Antón-Rodríguez and D. González-Ortega, "Monitorización del cuerpo humano en 3D mediante tecnología Kinect", SAAEI, 747-752, 2011.

[7]Mohd Afizi Mohd Shukran, Mohd Suhaili Bin Ariffin, "Kinect-based Gesture Password Recognition", Faculty of Science and Defence Technology, Universiti Pertahanan Nasional Malaysia, Australian Journal of Basic and Applied Sciences, 6(8): 492-499, 2012.

[8]Liang Li, "Gesture-based User Authentication with Kinect", Boston University, Department of Electrical and Computer Engineering, Technical Report No. ECE-2013-2, April 7 2013.

[9]Wu, Jonathan, Janusz Konrad, and Prakash Ishwar, "The Value of Multiple Viewpoints in Gesture-Based User Authentication", Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on. IEEE, 2014.

[10]S. Joseph Fluckiger, "Security with Visual Understanding: Kinect Human Recognition Capabilities Applied in a Home Security System", The University of Texas at Austin, May 2012.

[11]Araujo, Ricardo M., Gustavo Graña, and Virginia Andersson, "Towards skeleton biometric identification using the microsoft kinect sensor", Proceedings of the 28th Annual ACM Symposium on Applied Computing, ACM, 2013.

[12]Wu, Jonathan, Janusz Konrad, and Prakash Ishwar, "Dynamic time warping for gesture-based user identification and authentication with Kinect", Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, 2013.

[13]Dikovski, Bojan, Gjorgji Madjarov, and Dejan Gjorgjevikj, "Evaluation of different feature sets for gait recognition using skeletal data from Kinect", Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014 37th International Convention on. IEEE, 2014.

[14]Hayashi, Eiji, Manuel Maas, and Jason I. Hong, "Wave to me: user identification using body lengths and natural gestures", Proceedings of the 32nd annual ACM conference on Human factors in computing systems, ACM, 2014.

[15]Burnett, Mark. *Perfect Passwords: Selection, Protection, Authentication*. Rockland. Syngress Publishing. 2006. PP. 76, 131, 133, 134.

[16]Catuhe, David. *Programming with the Kinect for Windows Software Development Kit*. Redmond Washington. Microsoft Press. 2012. PP. 11, 12, 3-5, 27, 32.

[17] Miles, Rob. *Start Here! Learn Microsoft Kinect API.* California. O'Reilly Media. 2012. PP. 5-9, 57-59.

[18] Shotton, Jamie, et al., "Real-time human pose recognition in parts from single depth images", Communications of the ACM 56.1, 2013. PP. 116-124.

[19]Albahari, Joseph. *C# In a Nutshell*. O'Reilly Media. Sebastopol. 2012. PP. 1-3.

[20]Sharp, John. *Microsoft Visual C# 2010 Step by Step.* Redmond Washington. Microsoft Press. 2010. P. 3.

[21]MacDonald, Matthew. *Pro WPF 4.5 in C#: Windows Presentation Foundation in .NET 4.5.* New York. Apress. 2012. PP. 3, 22.

[22] Webb, Jarrett, and Ashley, James. *Beginning Kinect Programming with the Microsoft Kinect SDK*. Apress. 2012. P. 113.

[23] Vielhauer, Claus. *Biometric user authentication for IT security: from fundamentals to handwriting*. Vol. 18. Springer Science & Business Media. 2005. PP. 1-31

[24] http://www.oed.com/ (Fecha de consulta: 15/04/2015)

[25] http://www.rae.es/ (Fecha de consulta: 15/04/2015)

[26] HandPunch - Biometric Hand Reader Employee Enrollment. https://www.youtube.com/watch?v=vdV45_gXvXI. (Fecha de consulta: 28/05/2015)

[27] SwipeClock GT-400 Hand Geometry Clock. https://www.youtube.com/watch?v=nf8gwV-AXIw. (Fecha de consulta: 28/05/2015)

[28] Using a bank machine (ATM) to make a withdrawal. https://www.youtube.com/watch?v=5PGEWejcmNU. (Fecha de consulta: 28/05/2015)

[29] How to withdraw Money from ATM (State Bank of India)?. https://www.youtube.com/watch?v=O2HEuC0htuM. (Fecha de consulta: 28/05/2015)

[30] Biometrics at Airports: Iris Recognition demo. https://www.youtube.com/watch?v=xrk7hwqKCZw. (Fecha de consulta: 28/05/2015)

[31] 2 Meter Iris Recognition - Live Demo at iD1. https://www.youtube.com/watch?v=ZqQ2Q48MZ74. (Fecha de consulta: 28/05/2015)

[32] Fujitsu iris recognition live demo. https://www.youtube.com/watch?v=XPABpehFjug. (Fecha de consulta: 28/05/2015)

[33] Time Attendance Dubai UAE, Fingerprint System, Technical Demo for "AVI-FTA-507-C", Dubai UAE. https://www.youtube.com/watch?v=Z9E5lEnrC2U. (Fecha de consulta: 28/05/2015)

[34]http://www.vimago.net/pdf_files/InSight%20VM%20Data%20Sheet%20Espanol%20(2).pdf (Fecha de consulta: 31/05/2015)

[35] http://www.alibaba.com/product-detail/AVI-FTA-507-C_138591624.html (Fecha de consulta: 31/05/2015)

ANEXOS

Anexo 1: Códigos

Código 0:

```
    int centro = 0;    int izquierda = 0;    int derecha = 0;    int tiempoSilla = 75;    int j = 0;    float ba2AB;
    float ba2ABsum = 0;    float ba2ABprom;    int pasoSilla;    int pasoBrazo;    float distancia;    int DLIL = 0;    int DAIL =
0;
    int DAIA = 0;    int DLIA = 0;    string[] palabras = new string[10];    string[] palabrasSto = new string[10];
    string palabratemp;    int m = -1;    int hs;    int regreso = 0;    int pasoSillaSto;    int pasoBrazoSto;    float
distanciaSto;
    int u = 0;
```

Código 1:

```
private void NivelS() //Nivel de seguridad
    {
      double h = slider1.Value; //Asignar a h el valor del control deslizante
      int hstemp = Convert.ToInt32(h); //Convertir a entero el valor de h
      hs = hstemp; //Asignar a hs el valor de hstemp
      if (hstemp == 3) //Seguridad baja (3 palabras)
      {
        textBlock1.Text = string.Format("Ajuste con el control deslizante el nivel de seguridad: Baja"); //Mostrar leyenda para el
caso de seguridad baja
      }
      else if (hstemp == 4) //Seguridad media (4 palabras)
      {
        textBlock1.Text = string.Format("Ajuste con el control deslizante el nivel de seguridad: Media"); //Mostrar leyenda para el
caso de seguridad media
      }
      else if (hstemp == 5) //Seguridad alta (5 palabras)
      {
        textBlock1.Text = string.Format("Ajuste con el control deslizante el nivel de seguridad: Alta"); //Mostrar leyenda para el
caso de seguridad alta
      }
    }
    private void slider1_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e) //Evento para cambio en el
valor del control deslizante
    {
      NivelS();
    }
```

Código 2:

```
 private void disponSensor()
    {
      if (KinectSensor.KinectSensors.Count == 0) //Si no hay sensores Kinect disponibles
      {
        MessageBox.Show("Conectar el sensor Kinect", "ERROR"); //Solicitar conectar el sensor
        Application.Current.Shutdown(); //Terminar aplicacion
        return;
      }
      miKinect = KinectSensor.KinectSensors[0]; //Obtener el primer sensor Kinect conectado (0 para el primero)
    }
```

Código 3:

```
private void habilitarFlujEsqueleto() //Iniciar el sensor, habilitar flujo de datos del esqueleto y establecer modo (sentado o normal)
```

```
    {
        try //Intentar habilitar los flujos necesarios e iniciar el sensor así como establecer el modo
        {
            miKinect.SkeletonStream.Enable(); //Habilitar flujo de datos del esqueleto
            miKinect.Start(); //Iniciar sensor
            miKinect.SkeletonStream.TrackingMode = SkeletonTrackingMode.Default;
        }
        catch //En caso de no tener éxito mostrar mensaje de error y apagar aplicación
        {
            MessageBox.Show("La inicialización del sensor Kinect falló. Revise que el sensor se encuentre conectado a la alimentación e
intente de nuevo.", "ERROR DE INICIALIZACION");
            Application.Current.Shutdown();
        }
    }
```

Código 4:

```
  private float calcDistancia(Joint prim, Joint segu) // Método calcDistancia. Calcular distancia entre 2 articulaciones en 3
dimensiones.
    {
        float dX = prim.Position.X - segu.Position.X;
        float dY = prim.Position.Y - segu.Position.Y;
        float dZ = prim.Position.Z - segu.Position.Z;
        return (float)Math.Sqrt((dX * dX) + … ));
    }
```

Código 5:

```
    private void button1_Click(object sender, RoutedEventArgs e) //Evento botón bloquear
    {
        button1.Visibility = Visibility.Hidden; //ocultar botón 1
        textBlock1.Text = "Favor de elegir un círculo en el piso y una posición de las siguientes: (IMPORTANTE: Mantener la posición
hasta siguiente instrucción)";
        image2.Visibility = Visibility.Visible; //mostrar imagen de opciones 0-3 (Casos brazos)
        slider1.Visibility = Visibility.Hidden; //ocultar control deslizante
        miKinect.SkeletonFrameReady += new EventHandler<SkeletonFrameReadyEventArgs>(cuadroEsqueletoListo); //Conectar al
evento que se dispara cuando hay nuevos cuadros (frames) disponibles
    }
```

Código 6:

```
Skeleton[] esqueletos = null;
        using (SkeletonFrame frame = e.OpenSkeletonFrame())
        {
            if (frame != null) //Si la variable frame ya fue inicializada i.e. no es null
            {
                esqueletos = new Skeleton[frame.SkeletonArrayLength];
                frame.CopySkeletonDataTo(esqueletos);
            }
        }
        if (esqueletos == null) return;
        foreach (Skeleton esqueleto in esqueletos) //Construcción foreach. Busca en cada uno de los esqueletos (y busca los que
estan "seguidos")
        {
            if (esqueleto.TrackingState == SkeletonTrackingState.Tracked) //Si se detecta esqueleto
            {
            // La información de articulaciones se encuentra disponible para este esqueleto
            }
        }
```

Código 7:

```
Joint cabezaJoint = esqueleto.Joints[JointType.Head]; //cabeza
            Joint aJoint = esqueleto.Joints[JointType.ShoulderRight]; Joint bJoint = esqueleto.Joints[JointType.ElbowRight]; Joint
cJoint = esqueleto.Joints[JointType.WristRight];
            Joint dJoint = esqueleto.Joints[JointType.HandRight]; Joint AJoint = esqueleto.Joints[JointType.ShoulderLeft]; Joint BJoint
= esqueleto.Joints[JointType.ElbowLeft];
            Joint CJoint = esqueleto.Joints[JointType.WristLeft]; Joint DJoint = esqueleto.Joints[JointType.HandLeft];
            SkeletonPoint cabezaPosicion = cabezaJoint.Position; //posición de cabeza
            SkeletonPoint aJointPosition = aJoint.Position; SkeletonPoint bJointPosition = bJoint.Position; SkeletonPoint
cJointPosition = cJoint.Position; SkeletonPoint dJointPosition = dJoint.Position;
            SkeletonPoint AJointPosition = AJoint.Position; SkeletonPoint BJointPosition = BJoint.Position; SkeletonPoint
CJointPosition = CJoint.Position; SkeletonPoint DJointPosition = DJoint.Position;
            float Dab = calcDistancia(esqueleto.Joints[JointType.ShoulderRight], esqueleto.Joints[JointType.ElbowRight]) ; float Dbc =
calcDistancia(esqueleto.Joints[JointType.ElbowRight], esqueleto.Joints[JointType.WristRight]) ;
            float Dcd = calcDistancia(esqueleto.Joints[JointType.WristRight], esqueleto.Joints[JointType.HandRight]) ; float DadCalc =
(Dab + …);
            float DAB = calcDistancia(esqueleto.Joints[JointType.ShoulderLeft], esqueleto.Joints[JointType.ElbowLeft]) ; float DBC =
calcDistancia(esqueleto.Joints[JointType.ElbowLeft], esqueleto.Joints[JointType.WristLeft]) ;
            float DCD = calcDistancia(esqueleto.Joints[JointType.WristLeft], esqueleto.Joints[JointType.HandLeft]) ; float DADCalc =
(… + DCD);
            float Dad = calcDistancia(esqueleto.Joints[JointType.ShoulderRight], esqueleto.Joints[JointType.HandRight]) ;
            float DAD = calcDistancia(esqueleto.Joints[JointType.ShoulderLeft], esqueleto.Joints[JointType.HandLeft]) ;
            if (DLIL < tiempoSilla && DAIL < tiempoSilla && DAIA < tiempoSilla && DLIA < tiempoSilla)
            {
               if ((DadCalc - Dad) < “valor” && …) //Si ambos brazos están estirados
               {
                  if (aJointPosition.Y - dJointPosition.Y <= “valor” && AJointPosition.Y - DJointPosition.Y <= “valor”) //Si ambos brazos
no están abajo
                  {
                     if (Math.Abs(aJointPosition.Y - dJointPosition.Y) < “valor” && Math.Abs(AJointPosition.Y - DJointPosition.Y) <
“valor”) //Caso brazos: DERECHO LADO. IZQUIERDO LADO.
                     {
                        DLIL = DLIL + 1; //Incrementar en 1 el contador DLIL
                        DAIL = 0; //Reiniciar contador DAIL
                        DAIA = 0;
                        DLIA = 0;
                        if (j <= tiempoSilla)
                        {
                           j = j + 1;
                           ///Calcular distancia biométrica
                           ba2AB = (calcDistancia(esqueleto.Joints[JointType.ShoulderRight],
esqueleto.Joints[JointType.ShoulderCenter]) )+
                                 …);
                           ba2ABsum = ba2AB + ba2ABsum;
                        }
                        if (centro < tiempoSilla && izquierda < tiempoSilla && derecha < tiempoSilla)
                        {
                           if (Math.Abs(cabezaPosicion.X) < 0.2) //Caso circulo
                           {
                              izquierda = 0;
                              derecha = 0;
                              centro = centro + 1;
                           }
                           else if (cabezaPosicion.X < 0.2) //Caso circulo
                           {
                              centro = 0;
                              derecha = 0;
                              izquierda = izquierda + 1;
                           }
                           else //Caso circulo
                           {
                              centro = 0;
                              izquierda = 0;
                              derecha = derecha + 1;
                           }
                        }
                     }
```

```
                    else if (Math.Abs(aJointPosition.X - dJointPosition.X) < “valor” && Math.Abs(AJointPosition.Y - DJointPosition.Y) <
“valor”) //Caso brazos
…
                    else if (Math.Abs(aJointPosition.X - dJointPosition.X) < “valor” && Math.Abs(AJointPosition.X - DJointPosition.X) <
“valor”) //Caso brazos: DERECHO ARRIBA. IZQUIERDO ARRIBA.
…
                    else if (Math.Abs(aJointPosition.Y - dJointPosition.Y) < “valor” && Math.Abs(AJointPosition.X - DJointPosition.X) <
“valor”) //Caso brazos
…
                  }
                }
              }
            }
            if (centro == tiempoSilla) //Si contador centro llega al valor de tiempo establecido (tiempoSilla)
            {
              //Medir distancia ba2ABprom=ba2ABsum/j y pasar parámetros: centro=95, izquierda=0, derecha=0,
ba2ABprom=valorusuario (promedio de mediciones)
              ba2ABprom = ba2ABsum / j;
              variablespaso();
            }
            else if (izquierda == tiempoSilla) //Si contador izquierda llega al valor de tiempo establecido (tiempoSilla)
            {
              ba2ABprom = ba2ABsum / j;
              variablespaso();
            }
            else if (derecha == tiempoSilla) //Si contador derecha llega al valor de tiempo establecido (tiempoSilla)
            {
              ba2ABprom = ba2ABsum / j;
              variablespaso();
            }
            if (DLIA == tiempoSilla) //Si contador DLIA (Brazos: Derecho Lado, Izquierdo Arriba) llega al valor de tiempo establecido
(tiempoSilla)
            {
              variablespaso();
            }
            else if (DAIA == tiempoSilla)
            {
              variablespaso();
            }
            else if (DAIL == tiempoSilla) //Caso DAIL: DAIL=95, el resto = 0 //textBlock2.Text = "Derecho Arriba::Izquierdo Lado";
            {
              variablespaso();
            }
            else if (DLIL == tiempoSilla)
            {
              variablespaso();
            }
          }
```

Código 8:

```
private void variablespaso() //Asignar variables según casos
      {
        if (centro == tiempoSilla && izquierda == 0 && derecha == 0) //Caso centro => pasoSilla = 0
        {
          pasoSilla = 0; distancia = ba2ABprom;
        }
        else if (izquierda == tiempoSilla && centro == 0 && derecha == 0) //Caso izquierda => pasoSilla = 1
        {
          pasoSilla = 1; distancia = ba2ABprom;
        }
        else if (derecha == tiempoSilla && centro == 0 && izquierda == 0) //Caso derecha => pasoSilla = 2
        {
          pasoSilla = 2; distancia = ba2ABprom;
        }
```

```
        if (DLIL == tiempoSilla && DAIL == 0 && DAIA == 0 && DLIA == 0)
        {
            pasoBrazo = 0; //distancia = ba2ABprom; pasoReconHabla();
            textBlock2.Text = string.Format("Posición {0} seleccionada", pasoBrazo);
            textBlock3.Text = string.Format("Lugar {0} seleccionado", pasoSilla.ToString());
        }
        else if (DLIL == 0 && DAIL == tiempoSilla && DAIA == 0 && DLIA == 0)
        {
            pasoBrazo = 1; pasoReconHabla();
            textBlock2.Text = string.Format("Posición {0} seleccionada", pasoBrazo);
            textBlock3.Text = string.Format("Lugar {0} seleccionado", pasoSilla.ToString());
        }
        else if (DLIL == 0 && DAIL == 0 && DAIA == tiempoSilla && DLIA == 0)
        {
            pasoBrazo = 2; pasoReconHabla();
            textBlock2.Text = string.Format("Posición {0} seleccionada", pasoBrazo);
            textBlock3.Text = string.Format("Lugar {0} seleccionado", pasoSilla.ToString());
        }
        else if (DLIL == 0 && DAIL == 0 && DAIA == 0 && DLIA == tiempoSilla)
        {
            pasoBrazo = 3; pasoReconHabla();
            textBlock2.Text = string.Format("Posición {0} seleccionada", pasoBrazo);
            textBlock3.Text = string.Format("Lugar {0} seleccionado", pasoSilla.ToString());
        }
    }
```

Código 9:

```
    private void pasoReconHabla()
    {
        textBlock1.Text = string.Format("Diga en voz alta {0} palabras de la tarjeta", hs.ToString());
        image2.Visibility = Visibility.Hidden;
        miKinect.SkeletonStream.Disable();
        ReconocimientoHablaConf();
    }
```

Código 10:

```
var miKinect = KinectSensor.KinectSensors[0];
        fuenteAudio = miKinect.AudioSource;
        fuenteAudio.BeamAngleMode = BeamAngleMode.Adaptive;
        flujoAudio = fuenteAudio.Start();
```

Código 11:

```
private RecognizerInfo encontrarInfoRec()
    {
        var reconocedores = SpeechRecognitionEngine.InstalledRecognizers();
        foreach (RecognizerInfo reconInfo in reconocedores)
        {
            if (reconInfo.AdditionalInfo.ContainsKey("Kinect")) //Encontrar el valor (mirando info de cada reconocedor) que funciona
para "Kinect"
            {
                string detalles = reconInfo.AdditionalInfo["Kinect"];
                if (detalles == "True" && reconInfo.Culture.Name == "es-MX") //Para español de México
                {
                    // Información encontrada
                    return reconInfo;
                }
            }
        }
        return null; //Si no se encuentra ningun paquete de lenguaje se retorna valor nulo
    }
```

```
    private void ReconocimientoHablaConf()
    {
        infoRec = encontrarInfoRec();
        if (infoRec == null)
        {
            MessageBox.Show("Error en reconocimiento de habla", "ERROR");
            Application.Current.Shutdown();
            return;
        }
        try
        {
            reconocedor = new SpeechRecognitionEngine(infoRec);
        }
        catch
        {
            MessageBox.Show("No pudo ser cargado el motor de reconocimiento de habla", "ERROR");
            Application.Current.Shutdown();
        }
        Choices palabras = new Choices(); //Se crea variable palabras para almacenar las palabras que se podrán reconocer//Agregar
palabras
        //palabras.Add("Caza"); palabras.Add("Casa"); //En español de España se pronuncian diferente pero en español de México no.
        //Colores:
        palabras.Add("Verde"); palabras.Add("Purpura"); palabras.Add("Azul");…
        GrammarBuilder constructorGramatica = new GrammarBuilder();
        constructorGramatica.Culture = infoRec.Culture;
        constructorGramatica.Append(palabras);
        Grammar gram = new Grammar(constructorGramatica);
        reconocedor.LoadGrammar(gram);

        //Configurar audio:
        miKinect = KinectSensor.KinectSensors[0]; miKinect.Start();
        fuenteAudio = miKinect.AudioSource;
        fuenteAudio.BeamAngleMode = BeamAngleMode.Adaptive;
        flujoAudio = fuenteAudio.Start();
        reconocedor.SetInputToAudioStream(flujoAudio, new SpeechAudioFormatInfo(EncodingFormat.Pcm, 16000, 16, 1, 32000, 2,
null));
        reconocedor.RecognizeAsync(RecognizeMode.Multiple);
        //Evento SpeechRecognized el cual se ejecuta cada vez que se detecte una palabra
        reconocedor.SpeechRecognized +=
            new EventHandler<SpeechRecognizedEventArgs>(palabraReconocida);
    }
```

Código 12:

```
void palabraReconocida(object sender, SpeechRecognizedEventArgs e)
    {
        if (e.Result.Confidence < 0.9f) //Si  <90 % (Palabra que no está en el diccionario, mala pronunciación, mucho ruido, etcétera)
        {
            textBlock2.Text = "Favor de repetir la palabra";
            textBlock3.Text = "";
        }
        if (e.Result.Confidence > 0.9f) //Si  >90 %, Reconocimiento correcto de palabra, no necesariamente entrada correcta
        {
            textBlock3.Text = e.Result.Text;
            textBlock2.Text = "";
            palabratemp = e.Result.Text;
            m = m + 1;
            if (regreso >= 1)
            {
                u = 1;
            }
            paIRec();
        }
    }
```

Código 13:

```
void palabraReconocida(object sender, SpeechRecognizedEventArgs e)
{
   if (e.Result.Confidence < 0.9f) //Si 70 % (Palabra que no está en el diccionario, mala pronunciación, mucho ruido, etcétera)
   {
      textBlock2.Text = "Favor de repetir la palabra";
      textBlock3.Text = "";
   }
   if (e.Result.Confidence > 0.9f) //Si 90 %, Reconocimiento correcto de palabra, no necesariamente entrada correcta
   {
      textBlock3.Text = e.Result.Text;
      textBlock2.Text = "";
      palabratemp = e.Result.Text;
      m = m + 1;
      if (regreso >= 1)
      {
         u = 1;
      }
      palRec();
   }
}
private void palRec()
{
   if (regreso == 0) //regreso = 0 => Almacenar
   {
      palabrasSto[m] = palabratemp;
      if (m == (hs - 1))
      {
         bloqY();
      }
   }
   if (regreso >= 1 && u == 1) //Regreso >= 1 => Asignar variables a comparar con almacenadas
   {
      palabras[m] = palabratemp;
      if (m == (hs - 1))
      {
         bloqY();
      }
   }
}
private void bloqY()
{
   varAlmacenar();
   if (regreso >= 1) //Comparar variables con las almacenadas
   {
      if (pasoSillaSto == pasoSilla && Math.Abs(distanciaSto - distancia) <= 0.1 && pasoBrazoSto == pasoBrazo
         && palabrasSto[0] == palabras[0] && palabrasSto[1] == palabras[1] && palabrasSto[2] == palabras[2]
         && palabrasSto[3] == palabras[3] && palabrasSto[4] == palabras[4] && palabrasSto[5] == palabras[5]
         && palabrasSto[6] == palabras[6] && palabrasSto[7] == palabras[7] && palabrasSto[8] == palabras[8]
         && palabrasSto[9] == palabras[9])
      {
         fin();
      }
   }
   clicFalso();
   regreso = regreso + 1;
   resetVar1();
}
private void varAlmacenar() //Almacenar variables correctas
{
   if (regreso == 0) //Almacenar variables
   {
      pasoSillaSto = pasoSilla;
      distanciaSto = distancia;
```

```
            pasoBrazoSto = pasoBrazo;
        }
    }
    private void resetVar1() //Reiniciar variables
    {
        m = -1; centro = 0; izquierda = 0; derecha = 0; j = 0; ba2ABsum = 0; DLIL = 0; DAIL = 0; DAIA = 0; DLIA = 0;
    }
    private void clicFalso() //Vuelta
    {
        disponSensor();
        habilitarFlujEsqueleto();
        textBlock1.Text = "SISTEMA BLOQUEADO. Para desbloquear favor de realizar PRIMER INGRESO";
        textBlock3.Text = "";
        if (regreso >= 1)
        {
            if (pasoSillaSto != pasoSilla || Math.Abs(distanciaSto - distancia) > 0.1 || pasoBrazoSto != pasoBrazo || palabrasSto !=
palabras)
            {
                textBlock1.Text = "FALLÓ. Intente de nuevo. Para desbloquear favor de realizar PRIMER INGRESO";
            }
            if (regreso >= 3)
            {
                textBlock1.Text = "BLOQUEADO. Intentos Agotados";
                textBlock1.Foreground = Brushes.White;
                textBlock1.Background = Brushes.Red;
                MessageBox.Show("Desbloqueo inhabilitado", "Intentos agotados");
            }
        }
        miKinect.SkeletonStream.Enable();
        miKinect.SkeletonFrameReady += new EventHandler<SkeletonFrameReadyEventArgs>(cuadroEsqueletoListo);
    }
    private void fin() //Sistema desbloqueado: Cerrar aplicación
    {
        Application.Current.Shutdown();
    }
```

Código 14:

```
<Window x:Class="KinectTA.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="MainWindow" Height="768" Width="1366" Loaded="Window_Loaded" WindowStartupLocation="CenterScreen"
WindowState="Maximized">
  <Grid>
    <TextBlock Height="395" HorizontalAlignment="Center" Margin="72,12,7O,322" Name="textBlock1" Text=""
VerticalAlignment="Center" Width="1202" FontSize="67" TextAlignment="Center" TextWrapping="Wrap" />
    <Button Content="BLOQUEAR" Height="230" HorizontalAlignment="Left" Margin="427,227,0,0" Name="button1"
VerticalAlignment="Top" Width="480" FontSize="72" Click="button1_Click" Visibility="Visible" />
    <Image Height="256" HorizontalAlignment="Left" Margin="427,413,0,0" Name="image1" Stretch="Uniform"
VerticalAlignment="Top" Width="456" Visibility="Hidden" Source="/KinectTA;component/Images/Paso1SentSill.png"
DataContext="{Binding}" />
    <TextBlock Height="256" HorizontalAlignment="Left" Margin="672,414,0,0" Name="textBlock2" Text=""
VerticalAlignment="Top" Width="660" FontSize="56" TextWrapping="Wrap" TextAlignment="Center" />
    <Image Height="256" HorizontalAlignment="Left" Margin="427,413,0,0" Name="image2" Stretch="Fill"
VerticalAlignment="Top" Width="456" Source="/KinectTA;component/Images/Personadepie.png" Visibility="Hidden" />
    <TextBlock Height="256" HorizontalAlignment="Left" Margin="12,413,0,0" Name="textBlock3" Text="" VerticalAlignment="Top"
Width="654" FontSize="56" TextWrapping="Wrap" TextAlignment="Center" FontStretch="Normal" />
    <Slider Height="240" HorizontalAlignment="Left" Margin="1165,227,0,0" Name="slider1" VerticalAlignment="Top" Width="25"
Orientation="Vertical" SmallChange="1" IsSnapToTickEnabled="True" Minimum="3" Maximum="5"
ValueChanged="slider1_ValueChanged" DataContext="{Binding}" />
  </Grid>
</Window>
```

Código 15:

```
private void captFoto()
    {
        miKinect.ColorStream.Enable(); //Activar stream de video
        miKinect.ColorFrameReady += new EventHandler<ColorImageFrameReadyEventArgs>(miKinect_CuadroColorListo); //Crear
conexion al manejador de evento
        miKinect.Start(); //Encender sensor
    }
```

Código 16:

```
    bool tomarFoto;
    BitmapSource bitmapTemporal = null;
void miKinect_CuadroColorListo(object sender, ColorImageFrameReadyEventArgs e) //Manejador de evento
    {
        using (ColorImageFrame cuadroColor = e.OpenColorImageFrame())
        {
            if (cuadroColor == null) return;
            byte[] datosColor = new byte[cuadroColor.PixelDataLength];
            cuadroColor.CopyPixelDataTo(datosColor);
            kinectVideo.Source = BitmapSource.Create(
            cuadroColor.Width, cuadroColor.Height, // Dimensiones de la imagen
            96, 96, // resolucion (96 dpi para cuadros de video. X e Y respectivamente)
            PixelFormats.Bgr32, // Formato de video
            null, // paleta: ninguna
            datosColor, // datos de video
            cuadroColor.Width * cuadroColor.BytesPerPixel // stride
            );
            if (tomarFoto == true) //Copiar proximo bitmap capturado y almacenarlo
            {
                //Tomar un cuadro. bitMapTemporal mantiene la imagen para que pueda ser salvada.
                bitmapTemporal = BitmapSource.Create(
                    cuadroColor.Width, cuadroColor.Height,
                    96, 96, PixelFormats.Bgr32, null,
                    datosColor, cuadroColor.Width * cuadroColor.BytesPerPixel);
                tomarFoto = false; //Regresa la bandera para que la imagen no se sobrescriba cada vez

                //Salvar imagen a un archivo.jpg
                FileStream flujo = new FileStream("c:\\Users\\Usuario\\Desktop\\Foto.jpg", FileMode.Create);
                JpegBitmapEncoder codificador = new JpegBitmapEncoder();
                codificador.Frames.Add(BitmapFrame.Create(bitmapTemporal));
                codificador.Save(flujo);
            }
            miKinect.ColorStream.Disable();
        }
    }
```

Código 17

```
            if (regreso ==1)
            {
                captFoto();
                tomarFoto = true; //Capturar imagen
            }
    private void borrarFoto()
    {
        if (File.Exists("c:\\Users\\Usuario\\Desktop\\Foto.jpg"))
        {
            File.Delete("c:\\Users\\Usuario\\Desktop\\Foto.jpg");
        }
    }
    private void fin() //Sistema desbloqueado: Cerrar aplicación
```

```
{
    borrarFoto(); //Borrar imagen
    Application.Current.Shutdown();
}
```

Anexo 2: Listas de palabras

A) Antes:

Azul	China	Marcador
Amarillo	Japón	Goma
Rojo	Corea	Pelota
Verde	Afganistán	Balón
Violeta	Pakistán	Chip
Anaranjado	Etiopía	Teclado
Morado	Arabia	Piano
Purpura	Bolivia	Violín
Magenta	Ecuador	Canica
Cian	Finlandia	Balín
Esmeralda	Nicaragua	Teléfono
Rubí	Austria	Celular
Perla	Perú	Maleta
Negro	Colombia	Recipiente
Marrón	Paraguay	Luna
Hueso	Bélgica	Sol
Ámbar	Serbia	Mercurio
Blanco	Letonia	Venus
Celeste	Filipinas	Tierra
Jade	Panamá	Marte
Carmín	Irlanda	Júpiter
Vino	Indonesia	Saturno
Cobalto	Albania	Urano
Plateado	Australia	Neptuno
Turquesa	Líbano	Plutón
Kosovo	Casa	Galaxia
México	Auto	Helicóptero
Brasil	Avión	Tren
Venezuela	Mesa	Barco
Uruguay	Silla	Metro
Chile	Disco	Cielo
Argentina	Computadora	Mar
España	Piedra	Tierra
Inglaterra	Papel	Arena
Alemania	Tijeras	Roca
Italia	Libro	Montaña
Francia	Libreta	Monte
Suiza	Anuncio	Río
Holanda	Televisión	Lago
Marruecos	Radio	Bosque
Siria	Cine	Árbol
India	Mueble	Planta
Dinamarca	Edificio	Maceta
Irán	Reloj	Hoja
Polonia	Regla	Ladrillo
Rusia	Balanza	Cuerda
Eslovaquia	Resorte	Martillo
Grecia	Lápiz	Llave
Israel	Pluma	Secreto
Egipto	Plumón	Estatura

B) Después:

Azul
Amarillo
Circular
Verde
Magenta
Caoba
Morado
Púrpura
Magenta
Cian
Gamuza
Rubí
Gris
Oscuro
Transparente
Olivo
Ámbar
Claro
Celeste
Amatista
Tenue
Rojizo
Cobre
Plateado
Turquesa
Kosovo
México
Brasil
Venezuela
Uruguay
Chile
Argentina
España
Inglaterra
Alemania
Italia
Francia
Suiza
Holanda
Marruecos
Siria
India
Dinamarca
Irán
Polonia
Rusia
Eslovaquia
Antártida
Israel
Egipto

China
Japón
Corea
Afganistán
Pakistán
Etiopía
Arabia
Bolivia
Ecuador
Finlandia
Nicaragua
Austria
Perú
Camerún
Paraguay
Bélgica
Serbia
Letonia
Filipinas
Panamá
Irlanda
Indonesia
Croacia
Australia
Líbano
Hogar
Auto
Avión
Mesa
Silla
Disco
Computadora
Ángulo
Papel
Tijeras
Papeles
Libreta
Ventana
Televisión
Música
Cine
Ropero
Edificio
Tiempo
Compás
Balanza
Resorte
Lapicero
Bolígrafo
Plumón

Marcador
Nota
Cesto
Balón
Proceso
Teclado
Piano
Clavicordio
Canica
Balín
Teléfono
Mensaje
Maletín
Recipiente
Polar
Sol
Mercurio
Venus
Tierra
Marte
Júpiter
Saturno
Planeta
Neptuno
Plutón
Galaxia
Helicóptero
Tren
Barco
Metro
Cielo
Océano
Selva
Arena
Caída
Monte
Bosque
Caudal
Lago
Desierto
Árbol
Planta
Maceta
Mosaico
Ladrillo
Cuerda
Martillo
Llave
Contraseña
Estatura