ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

IMPLEMENTACIÓN DE UN SISTEMA DE RECONOCIMIENTO DE PATRONES DE MOVIMIENTO CON LAS EXTREMIDADES SUPERIORES DEL CUERPO HUMANO

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y CONTROL

SANTIAGO JAVIER RIOFRÍO HIDALGO

santymh.jav@gmail.com

DIRECTOR: Ing. DAVID FERNANDO POZO ESPÍN, MSc.

david_mh4485@yahoo.com

CODIRECTOR: Ing. CARLOS PATRICIO BURBANO ROMERO, MSc.

pburbanor@hotmail.com

Quito, Agosto 2014

DECLARACIÓN

Yo, Santiago Javier Riofrío Hidalgo, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

_____

Santiago Javier Riofrío Hidalgo

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Santiago Javier Riofrío Hidalgo, bajo mi supervisión.

_____
David Fernando Pozo Espín, MSc.
DIRECTOR DEL PROYECTO

AGRADECIMIENTO

A mis padres quienes me han apoyado incondicionalmente durante toda mi vida siendo un ejemplo de perseverancia, dedicación y humildad.

A mis queridas hermanas y toda mi familia quienes de uno u otro modo me han brindado su apoyo para la culminación de mi vida estudiantil y comienzo de mi vida profesional.

Un especial agradecimiento al Ing. David Pozo por su total dedicación y apoyo en la realización del presente proyecto de titulación.

Agradezco a la Escuela Politécnica Nacional y a todos mis profesores por permitirme aprender las bases técnicas y teóricas que me han servido para la elaboración de este trabajo y que me servirán durante toda mi vida profesional.

A mis entrañables amigos de facultad y de colegio por toda su confianza, aliento y ayuda.

Al talento humano de *Microsoft* cuyo innovador trabajo hace posible proyectos como este.

-Santiago Javier Riofrío Hidalgo-

DEDICATORIA

A mis queridos padres Servio y Margoth a quienes les debo mucho por todo el apoyo que me han brindado todos estos años por lo que los amo y siempre les estaré agradecido.

-Santiago Javier Riofrío Hidalgo-

CONTENIDO

CONTENIDO I

RESUMEN III

PRESENTACIÓN IV

CAPÍTULO 1 1

FUNDAMENTOS BÁSICOS 1

1.1 INTRODUCCIÓN 1

1.2 SENSORES DE CAPTURA DEL MOVIMIENTO DEL CUERPO HUMANO EN 3D 2

1.2.1 EL SENSOR KINECT 2

1.2.1.1 DESCRIPCIÓN DEL *HARDWARE* DEL SENSOR KINECT 3

1.2.1.2 DESCRIPCIÓN DEL *SOFTWARE* DEL KINECT 6

1.2.2 LOS SENSORES CARMINE Y CAPRI 10

1.3 RECONOCIMIENTO DE PATRONES 11

CAPÍTULO 2 13

ALINEAMIENTO TEMPORAL DINÁMICO 13

2.1 INTRODUCCIÓN 13

2.2 ESTUDIO DEL ALGORITMO DEL ALINEAMIENTO TEMPORAL DINÁMICO 13

2.3 RESTRICCIONES DEL DTW 29

2.3.1 RESTRICCIÓN DE MONOTONÍA 29

2.3.2 RESTRICCIÓN DE CONTINUIDAD 30

2.3.3 RESTRICCIÓN DE FRONTERA 30

2.3.4 RESTRICCIÓN DE VENTANA 31

2.3.5 RESTRICCIÓN DE PENDIENTE 33

2.4 APLICACIONES DEL DTW 36

CAPÍTULO 3 37

DESARROLLO E IMPLEMENTACIÓN DEL *SOFTWARE* 37

3.1 INTRODUCCIÓN 37

3.2 PROCESO DE RECONOCIMIENTO DE PATRONES DE MOVIMIENTO 37

3.3 DESCRIPCIÓN DE LA INTERFAZ GRÁFICA DEL PROGRAMA 38
3.4 DESARROLLO E IMPLEMENTACIÓN DEL ALGORITMO DTW 39
CAPÍTULO 4 60
PRUEBAS Y RESULTADOS 60
4.1 INTRODUCCIÓN 60
4.2 CALIBRACIÓN DE PARÁMETROS DEL DTW 60
4.3 PRUEBAS Y RESULTADOS FINALES 64
4.4 COSTOS DEL PROYECTO 68
CAPÍTULO 5 69
CONCLUSIONES Y RECOMENDACIONES 69
5.1 CONCLUSIONES 69
5.2 RECOMENDACIONES 70
TRABAJOS FUTUROS 71
REFERENCIAS BIBLIOGRÁFICAS 72
ANEXO A A-1
MANUAL DE USUARIO A-1

RESUMEN

Este proyecto presenta la implementación de un sistema de reconocimiento de patrones de movimiento con las extremidades superiores del cuerpo humano enfocado en la interacción humano-máquina utilizando nuevas herramientas para su desarrollo.

El programa es capaz de grabar patrones realizados con las extremidades superiores del cuerpo humano y luego reconocerlos cuando se los repite. El programa puede almacenar hasta 10 patrones los cuales pueden ser guardados en un archivo tipo .txt y luego empezar a grabar una nueva lista de patrones desde cero en un nuevo archivo. Además, el programa permite controlar diapositivas en una presentación realizada en "*Microsoft Power Point*" en base a la lista de patrones grabados.

Los movimientos son detectados por el sensor de movimiento "KINECT" el cual ubica las articulaciones del cuerpo humano en un sistema de coordenadas tridimensional.

El reconocimiento de patrones se realiza mediante la técnica del "ALINEAMIENTO TEMPORAL DINÁMICO" que es un método de reconocimiento geométrico basado en el cálculo de distancias, el cual ha sido utilizado en el reconocimiento de patrones de voz, imágenes y escritura.

El programa ha sido desarrollado en el entorno de *Microsoft Visual Studio* mediante programación en lenguaje *CSharp* (C#). El programa cuenta con un HMI que le permite al usuario grabar, editar, archivar y recuperar listas de patrones.

El programa también permite configurar los parámetros que intervienen en el algoritmo de reconocimiento y visualizar en pantalla los cálculos realizados para reconocer un patrón.

PRESENTACIÓN

El presente proyecto de titulación se encuentra dividido en cinco capítulos. El primer capítulo describe los fundamentos básicos para el desarrollo del sistema de reconocimiento de patrones de movimiento, se describe el *hardware* y *software* del sensor kinect así como la teoría sobre el reconocimiento de patrones.

El segundo capítulo está dedicado al estudio de la teoría de la técnica del Alineamiento Temporal Dinámico explicado mediante ejemplos, también se analizan las restricciones que intervienen en el algoritmo.

El tercer capítulo describe el diseño y desarrollo del *software* en el que se explica el funcionamiento e implementación del programa de reconocimiento de patrones. También se describe el algoritmo de reconocimiento programado en lenguaje C#.

En el cuarto capítulo se muestran las pruebas y resultados experimentales de la implementación del programa. Primero las calibraciones necesarias para obtener el comportamiento deseado del sistema y segundo las pruebas realizadas para mostrar su desempeño.

El quinto capítulo contiene las conclusiones y recomendaciones obtenidas al culminar la realización del proyecto con el objetivo de evidenciar el cumplimiento de los alcances planteados así como proponer trabajos futuros.

CAPÍTULO 1

FUNDAMENTOS BÁSICOS

1.1 INTRODUCCIÓN

Las nuevas tecnologías de captura de movimiento del cuerpo humano permiten crear nuevos e innovadores sistemas en los que las personas puedan transformar sus movimientos en acciones y los dispositivos electrónicos puedan interpretar de mejor manera el entorno que los rodea.

Los seres humanos poseen la capacidad de expresar sus ideas y sentimientos mediante lenguaje corporal, es decir, mediante movimientos corporales y gestos aprendidos a través de la percepción visual y para poder imitar esta forma de comunicación en la interacción humano-máquina es necesario hacer uso de un sensor que capture los movimientos que realiza una persona, en este caso el sensor kinect, y aplicar alguna metodología de reconocimiento de patrones, en este caso la técnica del Alineamiento Temporal Dinámico.

Existen movimientos que resultan ser familiares o más fáciles de repetir que otros, pero en términos generales cualquier serie de movimientos estructurados y organizados de forma secuencial puede ser considerado un patrón de movimiento.

Según Harrow y Sefeldt los patrones básicos de movimiento se clasifican en: locomotores, no locomotores y manipulativos. [1]

Los patrones locomotores son aquellos que implican acción de traslación del cuerpo como por ejemplo: andar, correr, saltar, deslizarse, trepar, rodar, etc. [1]

Los patrones no locomotores implican acciones realizadas en un solo lugar en los que se mueve el cuerpo alrededor de un punto fijo como por ejemplo: balancearse, inclinarse, girar, doblar, estirar, etc. [1]

Los patrones manipulativos implican acciones con las manos empleando objetos, como por ejemplo: lanzar, atrapar, golpear, patear, empujar, golpear, levantar, etc. [1]

Es necesario precisar que cuando se hace referencia a los “patrones de movimiento” a lo largo del presente trabajo se alude específicamente a los patrones de movimiento no locomotores realizados con las extremidades superiores del cuerpo humano.

En este marco de estudio el presente trabajo pretende estudiar, aplicar y evaluar la técnica del Alineamiento Temporal Dinámico aplicado al reconocimiento de patrones de movimiento mediante la implementación de una plataforma y su aplicación en el control de diapositivas en *Microsoft Power Point* utilizando el sensor kinect.

1.2 SENSORES DE CAPTURA DEL MOVIMIENTO DEL CUERPO HUMANO EN 3D

En el mercado existen sensores con la capacidad de percibir la realidad en tres dimensiones como lo hacen los ojos de los seres humanos, estos sensores han sido diseñados para reconocer la forma humana y entregar información de la ubicación de las extremidades del cuerpo humano en un sistema de coordenadas tridimensional los cuales se mencionan a continuación.

1.2.1 EL SENSOR KINECT

El kinect es un sensor de captura del movimiento del cuerpo humano en un espacio tridimensional y resulta de la combinación de un proyector y una cámara de infrarrojos con el que se puede transformar movimientos en acciones.

Desde que *Microsoft* lanzó al mercado el sensor kinect en noviembre del 2010 este sensor rápidamente se ha convertido en una herramienta que permite a los desarrolladores de *software* crear nuevas formas de interacción hombre-ordenador como por ejemplo: controlar el *mouse* de un computador mediante el movimiento de las manos, identificar un usuario mediante reconocimiento facial, controlar diapositivas mediante el movimiento de las manos, reconstruir digitalmente espacios tridimensionales, interactuar con vídeo juegos de manera más activa.

Físicamente, el kinect es una barra horizontal de aproximadamente 27 [cm] de largo y 6 [cm] de ancho conectada a una pequeña base circular mediante un eje de articulación como se aprecia en la figura 1.1.

Figura 1.1 Foto del sensor kinect

1.2.1.1 Descripción del *hardware* del sensor kinect

Las partes más importantes que conforman el kinect son: un proyector de infrarrojos, una cámara de infrarrojos, una cámara RGB, 4 micrófonos, un motor de giro y el *chip* PrimeSense PS1080-A2 – SoC como se muestra en la figura 1.2.

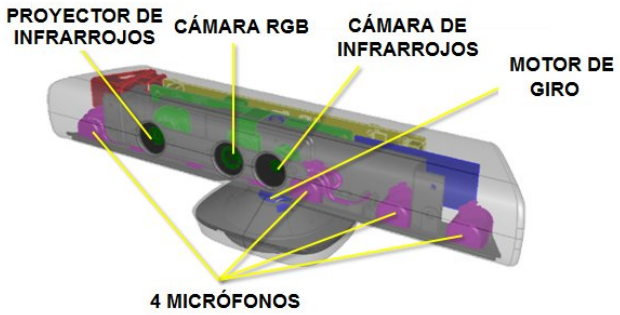


Figura 1.2 Partes del kinect, tomado de [2]

1.2.1.1.1 Cámara y Proyector de Infrarrojos

El kinect puede crear un mapa tridimensional del espacio que lo rodea sin utilizar ninguna clase de radar o sensor ultrasónico sino mediante un proyector y una cámara de infrarrojos que juntos forman el sensor de profundidad.

El proyector de infrarrojos emite miles de diminutos puntos de luz infrarroja a una longitud de onda de 830 [nm] sobre los objetos ubicados al alcance de la escena del kinect y la cámara de infrarrojos lee el reflejo de todos ellos para formar una malla de puntos y compararla con una malla predefinida mediante el *microchip* PrimeSense PS1080-A2–SoC el cual determina cuán desplazados se encuentran unos puntos de otros mediante la resolución de ecuaciones matemáticas.

El proceso de medir la disparidad entre puntos se conoce como “triangulación” en el que se obtiene el ángulo de rebote de cada uno de los puntos de luz infrarroja mediante la resolución de triángulos. La malla de puntos de luz infrarroja puede ser vista mediante una cámara de visión nocturna como se muestra en la figura 1.3. [3]

Figura 1.3 Malla de puntos de luz infrarroja que proyecta el kinect, tomado de [4]

Los rayos infrarrojos presentes en la luz solar tienen suficiente potencia en el rango de los 830 [nm] para perturbar las lecturas del sensor kinect por lo que es recomendable no usarlo en ambientes donde haya considerable exposición a la luz solar a pesar de que la cámara de infrarrojos cuente con un filtro de infrarrojos para hacer que las lecturas sean lo más confiables posibles en presencia de la luz solar o cualquier otra fuente emisora de rayos infrarrojos como por ejemplo los controles remotos de los televisores. [3]

1.2.1.1.2 Cámara RGB

La cámara RGB que posee el kinect es una cámara VGA de 640x480 píxeles de resolución a 30 [fps] ((Color CMOS) VNA38209015) y de 1280x1024 píxeles a 10 [fps]. Esta cámara permite ver imágenes y ubicar objetos en el plano “XY”. [5]

La cámara RGB y el sensor de profundidad del kinect trabajan juntos para obtener distancias medidas en un sistema de coordenadas cartesianas tridimensional. El *microchip* que posee el kinect genera toda la información y la envía a un computador a través del puerto USB cuando el usuario lo solicita. Las distancias obtenidas por el kinect son presentadas de forma muy parecida a los de una cámara común excepto que en vez de píxeles de color el kinect entrega “píxeles de distancias” del entorno que lo rodea.

1.2.1.1.3 Sensor de sonido direccional

El kinect posee 4 micrófonos separados entre sí unos de otros con el objetivo de localizar la fuente de sonido. El sonido tarda en viajar a través del aire y cuando una persona habla su voz llega a cada uno de los micrófonos con un ligero retardo, esta variación en tiempo en la lectura de los sonidos es procesada mediante *software* y de esta manera se puede localizar y escoger la fuente de sonido. También el kinect puede direccionar sus micrófonos para escuchar exclusivamente a la persona detectada por el kinect y considerar los sonidos restantes como ruido, por defecto el kinect viene calibrado para distinguir la señal de audio más alta. [6]

1.2.1.1.4 Motor de giro

El kinect posee en su base un sistema de giro que le permite regular su ángulo de elevación en un rango de -27º a +27º con el objetivo de regular el ángulo de visión y enfocar de mejor manera la escena.

Para inclinar el panel de sensores se utiliza un motor unido a engranajes los cuales generan un movimiento vertical para ajustar el ángulo de visión como se aprecia en la figura 1.4. Para determinar la posición actual en la que se encuentra la barra de sensores se utiliza un acelerómetro. [3]

Figura 1.4 Base del kinect, tomado de [3]

Un acelerómetro, en su forma más sencilla, es un dispositivo que mide la aceleración. En el caso de un sistema fijo como el kinect, el acelerómetro sirve para calcular el ángulo de elevación [3]. Esto permite que el sistema ubique y mueva la barra de sensores del kinect en ángulos específicos y además ayuda al sensor a orientarse en el espacio, especialmente cuando se halla ubicado en lugares a desnivel.

1.2.1.2 Descripción del *software* del kinect

El sensor kinect se comunica con el computador mediante el puerto USB y para verificar si ha sido reconocido se debe verificar que el led de *status* ubicado en el panel de sensores aparezca de color verde. Para que el kinect pueda comunicarse con el entorno de *Microsoft Visual Studio*, es decir, con lenguajes como: C++, C# y *Visual Basic* es necesario descargar e instalar el *driver* "Kinect *for Windows* SDK" versión 1.5 desde la dirección http://kinectforwindows.org. El esquema de comunicación entre el kinect y una aplicación se muestra en la figura 1.5.

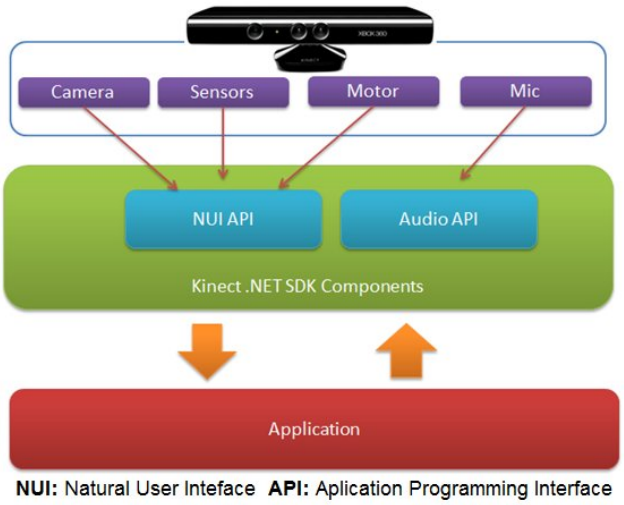


Figura 1.5 Esquema de comunicación del kinect con una aplicación mediante el *software* "Kinect para *Windows* SDK", tomado de [7]

La información que se obtiene del kinect es muy variada y a continuación se enlista los tipos de *stream* que el kinect genera:

- *Stream* de vídeo
- *Stream* de profundidad
- *Stream* de audio
- *Stream* de seguimiento "*Skeleton Tracking*"

1.2.1.2.1 Stream de vídeo

Este tipo de información proviene de la cámara RGB del kinect y es usada para generar imágenes de la escena que captura el kinect. Esta información es igual a

la que entrega una cámara común y corriente, es decir, píxeles de color que corresponden a una imagen.

El *buffer* de vídeo almacena píxeles y cada píxel se describe como un valor de 32 bits el cual está formado por 8 bits que corresponden al color azul, 8 bits al color verde, 8 bits al color rojo y 8 bits al valor de transparencia de acuerdo al formato BGR32 como se muestra en la figura 1.6.

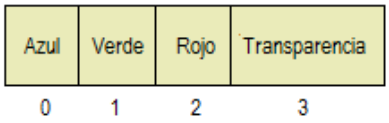


Figura 1.6 Píxel de color de la cámara RGB del kinect, tomado y traducido de [6]

1.2.1.2.2 Stream de profundidad

Este *stream* proviene del sensor de profundidad y proporciona información de cuán alejados están los objetos del kinect en el eje “Z”. Las distancias están expresadas en milímetros y están representadas en 13 bits, es decir, en teoría el kinect podría alcanzar distancias de alrededor de 8.192 [mm] que son más de 8 [m], pero en realidad el rango de visión del kinect va desde 0.8 a 4 [m], sin embargo en el reconocimiento de personas, los límites van de 1.2 a 3.5 [m] debido a que en este rango los valores de las distancias son más confiables como se ilustra en la figura 1.7. [6]

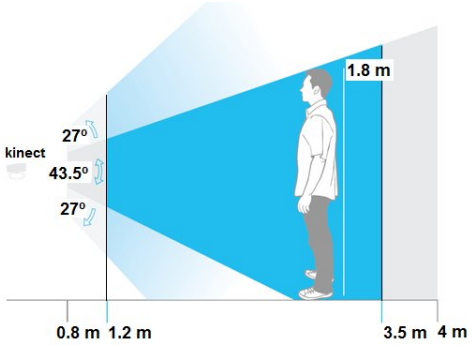


Figura 1.7 Rangos de visión del kinect, tomado de [8]

Adicionalmente a los 13 bits de distancia, existen otros 3 bits de identificación que se utilizan para conocer a quién pertenece la distancia medida, en caso de que exista más de una persona reconocida por el kinect. Cuando una persona ingresa

en la escena el kinect le asigna un número de identificación aleatoriamente, es decir, los 3 bits de identificación no expresan la identidad real de las personas sino indican una identificación relativa mientras permanecen en la escena.

Los diseñadores del kinect juntaron los 13 bits de distancias y los 3 bits de identificación para formar una sola variable de 16 bits. Los 3 bits de identificación son los menos significativos como se muestra en la figura 1.8. [6]

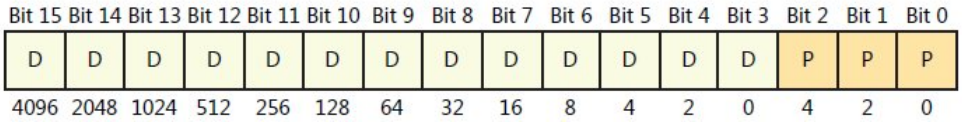


Figura 1.8 Bits del *stream* de profundidad, tomado de [6]

1.2.1.2.3 Stream de audio

El *stream* de audio permite recoger información de los 4 micrófonos del kinect y usarla para escuchar o grabar la voz del usuario. La aplicación más importante es el reconocimiento de voz en la que el usuario puede dar órdenes utilizando el habla, con la ventaja de que el kinect puede eliminar las fuentes de ruido provenientes de dispositivos u otras personas al direccionar sus micrófonos para escuchar únicamente a la persona detectada por el kinect. [6]

1.2.1.2.4 Stream de seguimiento "Skeleton Tracking"

Este *stream* tiene particular importancia para el presente trabajo ya que la información obtenida aquí es utilizada por el algoritmo de reconocimiento de patrones.

El *stream* de seguimiento permite localizar y obtener información de las articulaciones del esqueleto del cuerpo humano en un eje de coordenadas tridimensional. Los huesos del ser humano están unidos mediante articulaciones y el kinect es capaz de localizar un total de 20 articulaciones en modo "*default*" y 10 articulaciones pertenecientes a las extremidades superiores en modo "sentado". [6]

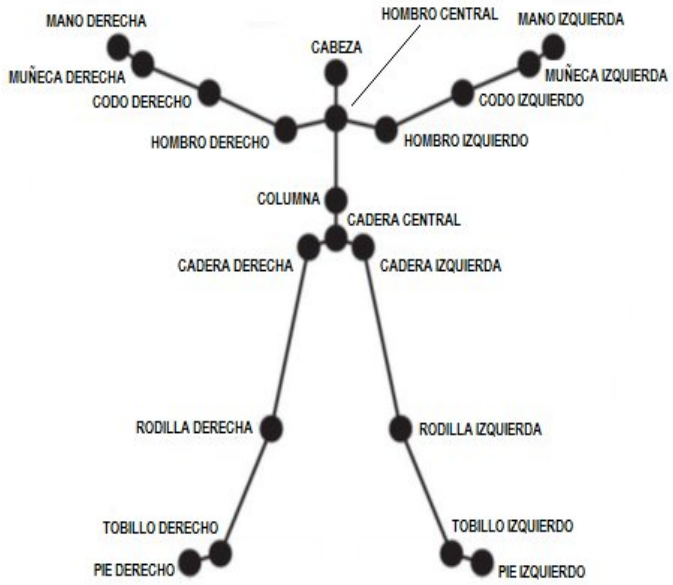


Figura 1.9 Articulaciones que detecta el kinect en el cuerpo humano, tomado y traducido de [9]

El *microchip* que posee el kinect le permite realizar varias tareas como reconocer cuando una persona entra en la escena y luego obtener la posición de sus articulaciones a lo largo de todo su cuerpo, el kinect se encarga de realizar todo este proceso y envía los datos con las posiciones de las articulaciones en los ejes “X”, “Y” y “Z” al usuario a una velocidad de 30 [fps].

La posición de cada articulación en la escena viene dada de acuerdo a un eje de referencia cuyo origen es el kinect como se muestra en la figura 1.10.

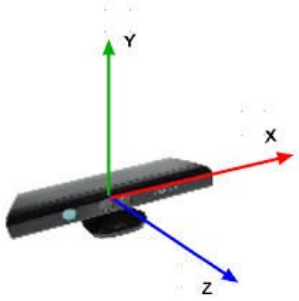


Figura 1.10 Eje de referencia del kinect

El kinect es capaz de reconocer un total de 6 personas, 2 de ellas con información detallada de sus articulaciones y las cuatro restantes únicamente la distancia al centro de masa de la persona.

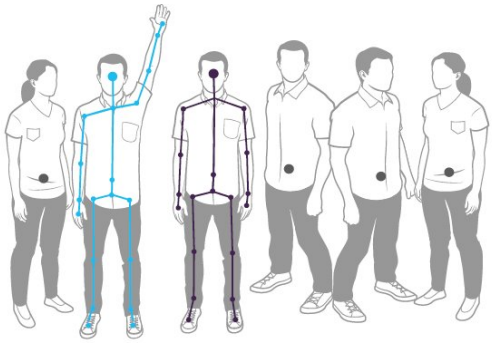


Figura 1.11 Total de personas reconocidas por el kinect, tomado de [8]

Cada articulación que el kinect determina puede tener tres estados: ubicado, inferido y no ubicado.

Si el kinect determina sin problemas la ubicación de la articulación, el estado es “ubicado”, cuando el kinect no puede determinar la ubicación exacta de la articulación tal vez pueda inferir su ubicación tomando en cuenta las articulaciones adyacentes entonces su estado es “inferido”, pero si definitivamente no puede ubicar la articulación su estado es “no ubicado”. [6]

Si bien el kinect reconoce fácilmente a las personas que entran en la escena, es recomendable no usar atuendos y artículos que cambien drásticamente la forma del cuerpo humano.

1.2.2 LOS SENSORES CARMINE Y CAPRI

Éstos sensores reconocen y capturan el movimiento del cuerpo humano en 3D al igual que lo hace el kinect y son fabricados por la compañía “PrimeSense” la misma que brindó soporte a *Microsoft* en el proceso de creación del kinect.

Como novedad el sensor Carmine 1.09 detecta objetos en un rango de corto alcance que va de 0.35 a 1.4 [m] a diferencia del sensor Carmine 1.08 cuyo rango va de 0.8 a 3.5 [m]. [10]

Figura 1.12 Sensor Carmine 1.08, tomado de [10]

El sensor Capri 1.25 es una plataforma de detección 3D más versátil por lo que es ideal para incorporarlo en dispositivos electrónicos tales como *tablets*, televisores, PCs, teléfonos móviles, robots y más como se aprecia en la figura 1.13. [10]

Figura 1.13 Sensor Capri 1.25, tomado de [10]

1.3 RECONOCIMIENTO DE PATRONES

El reconocimiento de patrones es la ciencia que se ocupa de los procesos relacionados con las ramas de la ingeniería, computación y matemáticas con el propósito de extraer información de objetos físicos o abstractos para establecer propiedades sobre conjuntos de dichos objetos. [11]

Un sistema completo de reconocimiento de patrones incluye un sensor que recoja fielmente los elementos del universo a ser clasificado, un mecanismo de extracción de características cuyo propósito es extraer la información útil eliminando la información redundante e irrelevante, y finalmente una etapa de toma de decisiones como se muestra en la figura 1.14. [11]

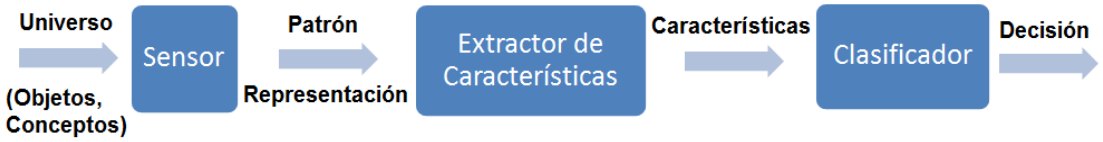


Figura 1.14 Sistema completo de reconocimiento de patrones, tomado de [11]

Sensor

El sensor es el dispositivo encargado de la adquisición de datos. Ha de ser capaz de transformar magnitudes físicas o químicas, llamadas variables de instrumentación, en magnitudes eléctricas. [11]

Extracción de características

La extracción de características es el proceso de generar características que puedan ser usadas en el proceso de clasificación de datos.

En ocasiones viene precedido por una etapa de preprocesado de la señal, necesario para corregir posibles deficiencias en los datos debido a errores del sensor, o bien para preparar los datos de cara a posteriores procesos en las etapas de extracción de características o clasificación. [11]

Selección de variables

Consiste en seleccionar cuál es el tipo de características o rasgos más adecuados para describir los objetos. Para ello, se deben localizar los rasgos que inciden en el problema de manera determinante. Esta etapa también puede ser diseñada dentro de la clasificación. [11]

Clasificación

En esta etapa se utiliza lo que se conoce como aprendizaje automático, cuyo objetivo es desarrollar técnicas que permitan a las computadoras aprender.

Habitualmente se utiliza uno de los siguientes procedimientos:

1. Geométrico (*Clustering*): En éste enfoque se emplea el cálculo de distancias, geometría de formas, vectores numéricos, puntos de atracción, etc. [11]

2. Estadístico: Se basa en la teoría de la probabilidad y la estadística, utiliza análisis de varianzas, covarianzas, dispersión, distribución, etc. [11]

Un método de reconocimiento geométrico basado en el cálculo de distancias es el método conocido como Alineamiento Temporal Dinámico el cual ha sido especialmente utilizado en el reconocimiento de voz para hacer frente a diferentes velocidades del habla [12], en el reconocimiento de la escritura para hacer frente a la semejanza entre letras, palabras o símbolos como firmas [13] [14], así como en la genética en la comparación y reconocimiento de genes [15], estas características hacen del Alineamiento Temporal Dinámico el indicado para el reconocimiento de patrones de movimiento en los que se necesita hacer frente a diferentes variaciones de velocidad y semejanza al momento de reconocer patrones.

CAPÍTULO 2

ALINEAMIENTO TEMPORAL DINÁMICO

1.4 INTRODUCCIÓN

Para que un sistema pueda reconocer un patrón es necesario implementar algún algoritmo de reconocimiento que permita procesar la información de la ubicación de las articulaciones de la persona que ha sido detectada por el sensor kinect en este caso se empleará el Alineamiento Temporal Dinámico.

En este capítulo se explica el algoritmo del Alineamiento Temporal Dinámico mediante la resolución de un problema en el que se plantea encontrar la señal más parecida a un patrón dado. También se hace una comparación con el método de Dijkstra con fines explicativos.

1.5 ESTUDIO DEL ALGORITMO DEL ALINEAMIENTO TEMPORAL DINÁMICO

El Alineamiento Temporal Dinámico (DTW por sus siglas en inglés, *Dynamic Time Warping*) es un método de reconocimiento de patrones que mide la distancia entre dos series numéricas deformadas y desplazadas en el eje del tiempo.

En otras palabras, el algoritmo es capaz de alinear dos series numéricas deformadas en el eje del tiempo hasta que se encuentre una coincidencia óptima entre ellas, es decir, este algoritmo es capaz de reconocer patrones sin importar si se los realiza a diferentes velocidades como se muestra en la figura 2.1.

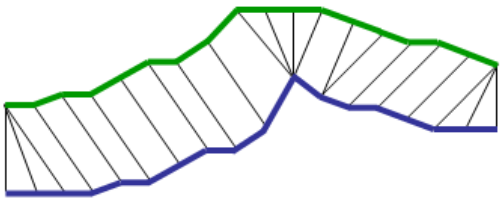


Figura 2.1 Gráfico de dos secuencias alineadas mediante el DTW, tomado de [15]

Para implementar el algoritmo del DTW es necesario seguir los siguientes pasos: 1) Calcular una matriz de distancias, 2) Calcular una matriz de costo, 3) Encontrar el camino de menor costo llamado "*warping path*", y 4) Calcular la distancia DTW.

Para explicar de mejor manera la resolución de este algoritmo se propone analizar el siguiente ejemplo:

Dadas las series A, B y P en función del tiempo:

A(t)=[2,3,3,4,5,5,6,7,6,5,5,4,3,3,3,3,3,3,3,3,3,3,3,3]

B(t)=[2,2,2,2,2,2,2,2,3,3,4,5,5,5,6,7,7,7,7,6,5,5,5,4,3]

P(t)=[1,1,1,1,1,2,2,3,4,4,5,6,5,4,4,3,2,2,2,2,2,2,2,2,2]

Figura 2.2 Gráfico de las series numéricas A, B y P en función del tiempo

Se toma la serie P(t) como patrón y las series A(t) y B(t) como series de prueba. El objetivo del ejemplo consiste en encontrar cuál serie se parece más al patrón.

A simple vista un ser humano probablemente diría que la serie A(t) se parece más al patrón P(t), pero para un computador o un robot esto no resulta tan fácil de decidir por lo que a continuación se aplica el algoritmo del DTW.

Primero se comparan las series A(t) vs P(t).

El primer paso es obtener la matriz de distancias.

La matriz de distancias no es más que una matriz formada por distancias locales entre los elementos del patrón y los elementos de la serie de prueba, así la matriz de distancias entre las series A(t) y P(t) es la que se muestra en la figura 2.3.

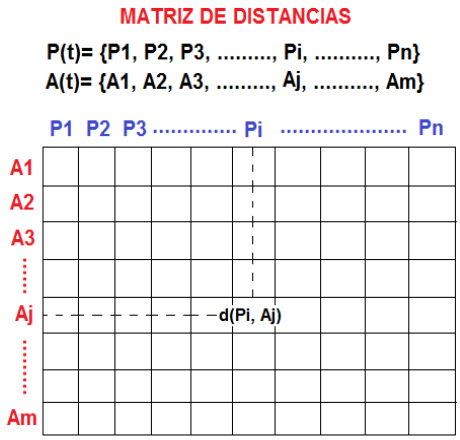


Figura 2.3 Descripción de la matriz de distancias entre las series P(t) y A(t)

Para calcular la distancia $d(Pi, Aj)$ usualmente se emplea la distancia Manhattan o la distancia Euclidiana.

La distancia Manhattan por definición es:

$$d(x, y) = |x - y| \text{ [13]}$$

Aplicando la distancia Manhattan en la figura 2.3 se tiene que:

$$d(Pi, Aj) = |Pi - Aj|$$

En definitiva la distancia Manhattan es el valor absoluto entre dos puntos, y para el ejemplo planteado la matriz de distancias es la que se observa en la figura 2.4.

Por ejemplo, calculando las distancias $\mathrm{d}(P_1, A_1)$ y $\mathrm{d}(P_6, A_1)$ en la figura 2.4 se tiene lo siguiente:

$$\mathrm{d}(P_1, A_1) = |1 - 2| = |-1| = 1 \text{ [cm]}$$

$$\mathrm{d}(P_6, A_1) = |2 - 2| = |0| = 0 \text{ [cm]}$$

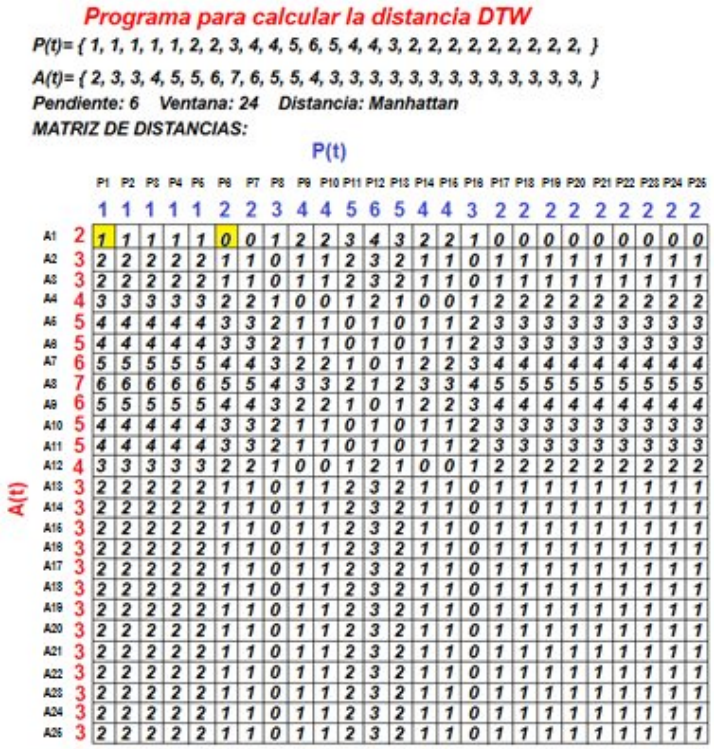


Figura 2.4 Matriz de distancias entre las series A(t) y P(t)

Adicionalmente es necesario considerar la posibilidad de que cada elemento de las series sean vectores de más de una dimensión, por ejemplo considérese que los elementos de cada serie sean vectores en 3 dimensiones, en este caso se tendría lo siguiente:

$$P(t) = \left\{ [\overbrace{3,2,5}^{x\ y\ z}], [1,3,2], [6,2,3], \dots, P_n[\,] \right\}$$

$$A(t) = \left\{ [\overbrace{1,4,2}^{x\ y\ z}], [1,2,1], [4,3,2], \dots, A_m[\,] \right\}$$

Figura 2.5 Ejemplo de dos series P(t) y A(t) en tres dimensiones

En este caso para hallar la distancia $d(Pi, Aj)$ en la figura 2.5 se emplea la distancia Euclidiana que por definición es:

$$d(x, y) = \sqrt{\sum_i (x_i - y_i)^2}$$

Para el ejemplo de la figura 2.5 se tiene:

$$d(Pi, Aj) = \sqrt{(Pix - Ajx)^2 + (Piy - Ajy)^2 + (Piz - Ajz)^2}$$

Si se calcula la distancia entre el primer elemento del patrón P(t) con el primer elemento de la serie A(t) en la figura 2.5 se obtiene:

$$d(P_1, A_1) = \sqrt{(P1x - A1x)^2 + (P1y - A1y)^2 + (P1z - A1z)^2}$$

$$d(P_1, A_1) = \sqrt{(3-1)^2 + (2-4)^2 + (5-2)^2}$$

$$d(P_1, A_1) = 4.12[u]$$

En definitiva lo que se ha hecho es calcular la distancia entre dos puntos en el espacio.

Ahora, también es posible ampliar el concepto de la distancia Manhattan a más de una dimensión así:

$$d(x, y) = \sum_i |x_i - y_i|$$

De igual manera para el ejemplo de la figura 2.5 se tiene:

$$d(P_1, A_1) = |P_{1x} - A_{1x}| + |P_{1y} - A_{1y}| + |P_{1z} - A_{1z}|$$

$$d(P_1, A_1) = |3-1| + |2-4| + |5-2| = 7[u]$$

De esta manera se encuentra la matriz de distancias. Generalmente se usa la distancia Manhattan cuando las series tienen elementos en una dimensión y la distancia Euclidiana cuando existen elementos de más de una dimensión.

El siguiente paso para resolver el algoritmo del DTW consiste en hallar la matriz de costo en base a la matriz de distancias.

La matriz de costo es una matriz de distancias acumuladas cuyo concepto se expone mediante el siguiente ejemplo: Un viajero desea ir de A a F recorriendo la menor distancia posible, entonces ¿Cuál es la menor distancia desde A hasta F? en la figura 2.6.

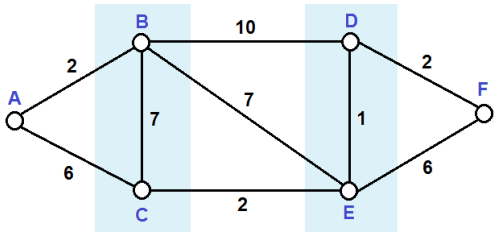


Figura 2.6 Grafo del problema del viajero, menor distancia de A a F

Probablemente, la primera respuesta es 12 [u] siguiendo la ruta señalada en la figura 2.7.

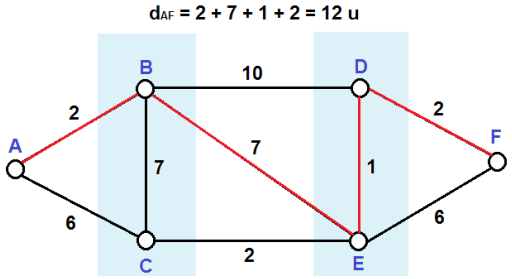


Figura 2.7 Distancia de A a F en el problema del viajero

Pero, la menor distancia de A hasta F es: 11 [u] como se muestra en la figura 2.8.

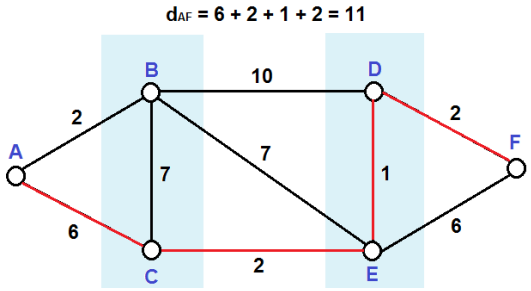


Figura 2.8 Respuesta de la menor distancia de A a F en el problema del viajero

Para resolver problemas de mayor complejidad con un grafo en el que intervienen más nodos es necesario plantear algún método para obtener la menor distancia de un punto a otro, un método muy utilizado es el de Dijkstra el cual consiste en ir encontrando distancias acumuladas mínimas en cada nodo. Este método permite explicar el concepto de la matriz de costo y el "*warping path*" por lo que se lo aplicará en el ejemplo del viajero en la figura 2.9.

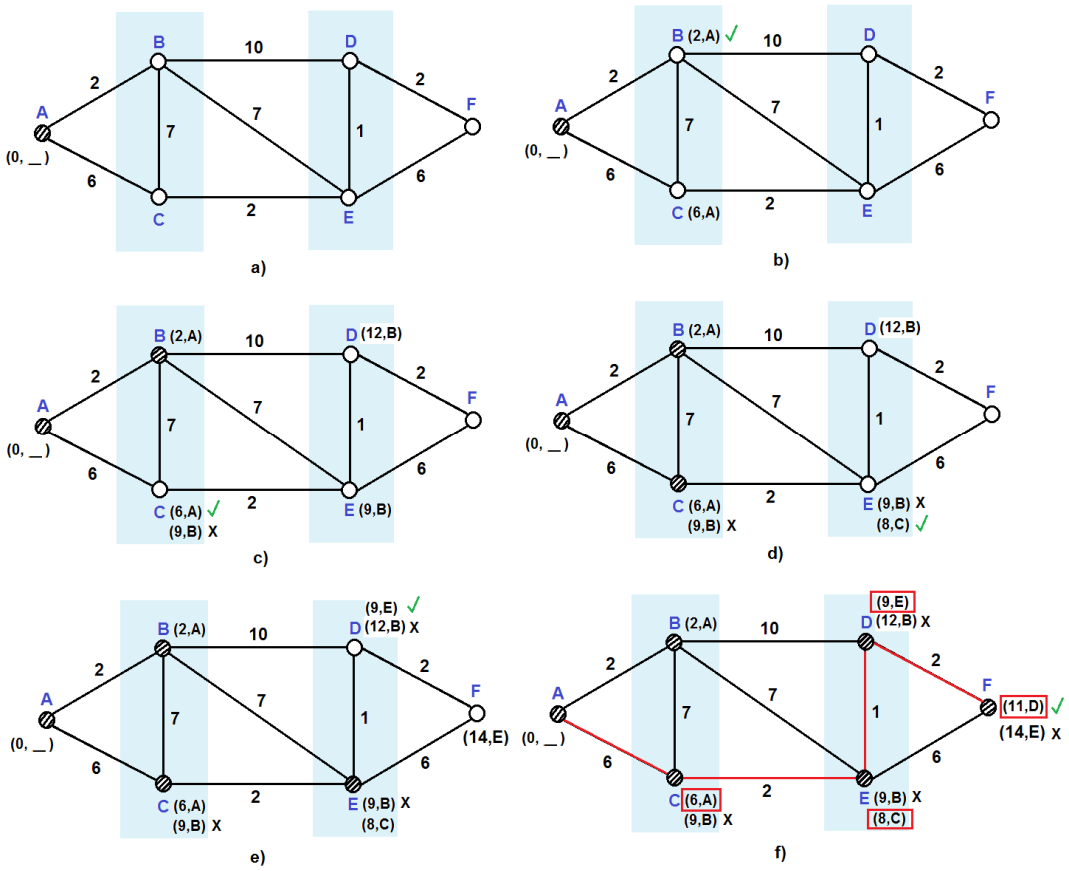


Figura 2.9 Menor distancia de A a F mediante el algoritmo de Dijkstra

Ahora, se procede a explicar los pasos que se han dado para obtener la menor distancia de A a F en la figura 2.9.

Junto a cada nodo se escribe entre paréntesis dos elementos, el primero indica la distancia acumulada y el segundo la letra del nodo del que proviene.

Se analizan todos los nodos y cuando se escoge uno se le llama “nodo permanente”. Para empezar se toma el “nodo A” junto al que se ha escrito (0, _) en la figura 2.9 a. Esto significa que la distancia al mismo nodo es cero y que no proviene de ningún otro nodo.

Luego se consideran todos los nodos adyacentes al “nodo A”, es decir, los nodos B y C. Se halla la distancia del “nodo A” a dichos nodos y se tiene que: B(2, A) y C(6, A), se toma el nodo B porque es el de menor distancia como se observa en la figura 2.9 b.

Entonces, el “nodo B” pasa a ser el nuevo nodo permanente y nuevamente se toman todos los nodos libres adyacentes al “nodo B”, es decir, los nodos C, D y E. Se halla las distancias acumuladas obteniendo: C(9, B), D(12, B), E(9, B) y C(6, A), por lo que el nuevo nodo permanente es C(6, A) como se aprecia en la figura 2.9 c.

El “nodo E” es el único nodo libre adyacente al “nodo C” por lo que se procede a calcular la distancia acumulada al “nodo E” la cual es (8, C) se nota que ahora el “nodo E” tiene dos distancias acumuladas una anterior (9, B) y la nueva (8, C), de igual manera se escoge la menor distancia acumulada, es decir, E(8, C). De igual forma se procede a buscar el siguiente nodo permanente comparando los nodos E(8, C) y D(12, B), como resultado se tiene que el “nodo E” es el nuevo nodo permanente como lo indica la figura 2.9 d.

Ahora los nodos libres adyacentes al “nodo E” son los nodos D y F, por lo que la distancia acumulada en ellos es D(9, E), F(14, E) y D(12, B), la menor distancia resultante es D(9, E), ahora para escoger el nuevo nodo permanente resta comparar los nodos libres D(9, E) y F(14, E), D tiene la menor distancia acumulada por lo que pasa a ser el nuevo nodo permanente como se puede ver en la figura 2.9 e.

El último nodo libre adyacente al nodo D(9, E) es el “nodo F” por lo que sumando la distancia local da como resultado F(11, D) y F(14, F), por lo que la menor distancia al “nodo F” es 11 como se puede apreciar en la figura 2.9 f.

De esta manera se puede conocer no sólo la menor distancia del “nodo A” al “nodo F”, sino a cualquier otro nodo, por ejemplo la menor distancia del “nodo A” al “nodo D” es 9. Además, se puede ver que el nodo F(11, D) proviene del nodo D(9, E) que a su vez proviene del nodo E(8, C) que de igual manera proviene del nodo C(6, A) el cual proviene del “nodo A”, es decir, se conoce la ruta tomada para obtener la menor distancia, es decir, FDECA o ACEDF en sentido inverso.

Ahora se realiza una analogía entre el método de Dijkstra y el algoritmo del DTW. Las distancias locales de nodo a nodo mostradas en el problema del viajero son los elementos que conforman la matriz de distancias en el DTW que se halló en el paso anterior, y las distancias mínimas acumuladas que se calculan en cada nodo

mediante el método de Dijkstra son las distancias que conforman la matriz de costo. Por último, el camino de distancias mínimas que en el método de Dijkstra era FDECA en el DTW se llama "*warping path*" como se ilustra en la figura 2.10.

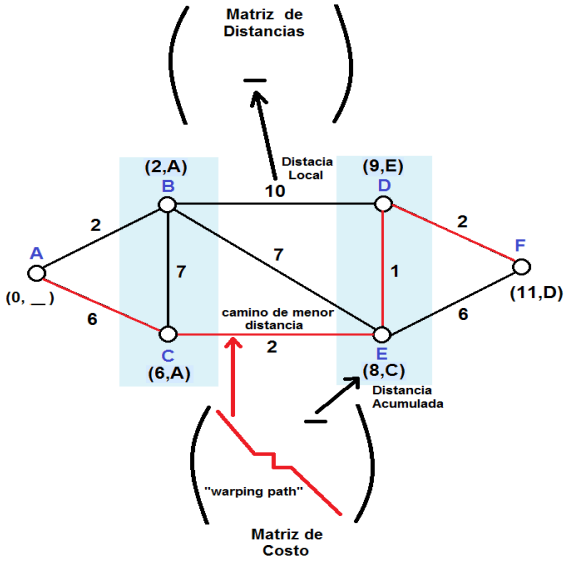


Figura 2.10 Analogía entre el DTW y método de Dijkstra

Ahora se puede decir que la matriz de costo es una matriz de distancias acumuladas en el que cada elemento está definido por la siguiente expresión:

$$c(i,j) = d_{i,j} + min\left(c_{i-1,j-1}, c_{i-1,j}, c_{i,j-1}\right)$$ [12]

Para las series A(t) y P(t) se tiene que:

$$c(i,j) = d\left(P_i, A_j\right) + min(c_{i-1,j-1}, c_{i-1,j}, c_{i,j-1})$$

Donde $d\left(P_i, A_j\right)$ es la distancia local proveniente de la matriz de distancias obtenida anteriormente, en otras palabras, cada nuevo elemento de la matriz de costo es la suma de la distancia local más la menor distancia acumulada en el paso anterior, de esta manera la matriz de costo entre las series A(t) y P(t) es la matriz de la figura 2.11.

Programa para calcular la distancia DTW

P(t)= { 1, 1, 1, 1, 1, 2, 2, 3, 4, 4, 5, 6, 5, 4, 4, 3, 2, 2, 2, 2, 2, 2, 2, 2, }

A(t)= { 2, 3, 3, 4, 5, 5, 6, 7, 6, 5, 5, 4, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, }

Pendiente: 6 Ventana: 24 Distancia: Manhattan

MATRIZ DE COSTO:

A(t)		P(t) P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25
		1	1	1	1	1	2	2	3	4	4	5	6	5	4	4	3	2	2	2	2	2	2	2	2	2
A1	2	01	02	03	04	05	05	05	06	08	10	13	17	20	22	24	25	25	25	25	25	25	25	25	25	25
A2	3	03	03	04	05	06	06	06	05	06	07	09	12	14	15	16	16	17	18	19	20	21	22	23	24	25
A3	3	05	05	05	06	07	07	07	05	06	07	09	12	14	15	16	16	17	18	19	20	21	22	23	24	25
A4	4	08	08	08	08	09	09	09	06	05	05	06	08	09	09	09	10	12	14	16	18	20	22	24	25	26
A5	5	12	12	12	12	12	12	12	08	06	06	05	06	06	07	08	10	13	15	17	19	21	23	25	27	28
A6	5	16	16	16	16	16	15	15	10	07	07	05	06	06	07	08	10	13	16	18	20	22	24	26	28	30
A7	6	21	21	21	21	21	19	19	13	09	09	06	05	06	08	09	11	14	17	20	22	24	26	28	30	32
A8	7	27	27	27	27	27	24	24	17	12	12	08	06	07	09	11	13	16	19	22	25	27	29	31	33	35
A9	6	32	32	32	32	32	28	28	20	14	14	09	06	07	09	11	14	17	20	23	26	29	31	33	35	37
A10	5	36	36	36	36	36	31	31	22	15	15	09	07	06	07	08	10	13	16	19	22	25	28	31	34	37
A11	5	40	40	40	40	40	34	34	24	16	16	09	08	06	07	08	10	13	16	19	22	25	28	31	34	37
A12	4	43	43	43	43	43	36	36	25	16	16	10	10	07	06	06	07	09	11	13	15	17	19	21	23	25
A13	3	45	45	45	45	45	37	37	25	17	17	12	13	09	07	07	06	07	08	09	10	11	12	13	14	15
A14	3	47	47	47	47	47	38	38	25	18	18	14	15	11	08	08	06	07	08	09	10	11	12	13	14	15
A15	3	49	49	49	49	49	39	39	25	19	19	16	17	13	09	09	06	07	08	09	10	11	12	13	14	15
A16	3	51	51	51	51	51	40	40	25	20	20	18	19	15	10	10	06	07	08	09	10	11	12	13	14	15
A17	3	53	53	53	53	53	41	41	25	21	21	20	21	17	11	11	06	07	08	09	10	11	12	13	14	15
A18	3	55	55	55	55	55	42	42	25	22	22	22	23	19	12	12	06	07	08	09	10	11	12	13	14	15
A19	3	57	57	57	57	57	43	43	25	23	23	24	25	21	13	13	06	07	08	09	10	11	12	13	14	15
A20	3	59	59	59	59	59	44	44	25	24	24	25	27	23	14	14	06	07	08	09	10	11	12	13	14	15
A21	3	61	61	61	61	61	45	45	25	25	25	26	28	25	15	15	06	07	08	09	10	11	12	13	14	15
A22	3	63	63	63	63	63	46	46	25	26	26	27	29	27	16	16	06	07	08	09	10	11	12	13	14	15
A23	3	65	65	65	65	65	47	47	25	26	27	28	30	29	17	17	06	07	08	09	10	11	12	13	14	15
A24	3	67	67	67	67	67	48	48	25	26	27	29	31	31	18	18	06	07	08	09	10	11	12	13	14	15
A25	3	69	69	69	69	69	49	49	25	26	27	29	32	33	19	19	06	07	08	09	10	11	12	13	14	15

Figura 2.11 Matriz de costo entre A(t) y P(t)

Aplicando la definición de la matriz de costo en la figura 2.11 se tiene:

$$c(i,j) = d(Pi, Aj) + min(c_{i-1,j-1}, c_{i-1,j}, c_{i,j-1})$$

$$c(6,7) = d(6,7) + min(c_{5,6}, c_{5,7}, c_{6,6})$$

$$c(6,7) = 4 + min(16, 21, 15)$$

$$c(6,7) = 4 + 15$$

$$c(6,7) = 19\ [cm]$$

Ahora, se ve cómo se hallan elementos que están ubicados al borde de la matriz, así:

$$c(11,1) = d(11,1) + min(c_{10,0}, c_{10,1}, c_{11,0})$$

$$c(11,1) = 3 + min(\infty, 10, \infty)$$

$$c(11,1) = 3 + 10$$

$$c(11,1) = 13\ [cm]$$

Se nota que las posiciones c(10, 0) y c(11, 0) no existen por lo que su valor se toma como infinito.

Ahora, se muestra cómo se halla otro elemento:

$$c(1,1) = d(1,1) + min(c_{0,0}, c_{0,1}, c_{1,0})$$

$$c(1,1) = 1 + min(0, \infty, \infty)$$

$$c(1,1) = 1 + 0$$

$$c(1,1) = 1\ [cm]$$

Se aprecia que la posición c(0, 0) es cero lo cual permite empezar a calcular los demás elementos de la matriz de costo.

Al igual que se hizo en el algoritmo de Dijkstra, una vez encontradas las distancias acumuladas en cada nodo es posible encontrar un camino de menor costo del nodo inicial a cualquier otro nodo, es decir, ahora se puede encontrar el "*warping path*" desde el elemento (P_n, A_m) al elemento (P_1, A_1) como se lo realizó en la figura 2.9 f. al momento de hallar la menor distancia de A a F la cual era FDECA.

Para hallar los elementos que conforman el "*warping path*" primero se toma el elemento (P_n, A_m) y los demás elementos se encuentran aplicando la siguiente fórmula en la matriz de costo de manera sucesiva:

$$min(c_{i-1,j-1}, c_{i-1,j}, c_{i,j-1}) \quad [12\ ]$$

De esta manera se hallan los elementos que conforman el "*warping path*" hasta llegar al elemento (P_1, A_1).

En consecuencia, en la figura 2.12 se muestra el "*warping path*" para las series A(t) y P(t).

Programa para calcular la distancia DTW

P(t)= { 1, 1, 1, 1, 1, 2, 2, 3, 4, 4, 5, 6, 5, 4, 4, 3, 2, 2, 2, 2, 2, 2, 2, 2, }

A(t)= { 2, 3, 3, 4, 5, 5, 6, 7, 6, 5, 5, 4, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, }

Pendiente: 6 Ventana: 24 Distancia: Manhattan

MATRIZ DE COSTO:

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25
A1	[01]	[02]	[03]	[04]	[05]	[05]	[05]	06	08	10	13	17	20	22	24	25	25	25	25	25	25	25	25	25	25
A2	03	03	04	05	06	06	06	[05]	06	07	09	12	14	15	16	16	17	18	19	20	21	22	23	24	25
A3	05	05	05	06	07	07	07	[05]	06	07	09	12	14	15	16	16	17	18	19	20	21	22	23	24	25
A4	08	08	08	08	09	09	09	06	[05]	[05]	06	08	09	09	09	10	12	14	16	18	20	22	24	25	26
A5	12	12	12	12	12	12	12	08	06	06	[05]	06	06	07	08	10	13	15	17	19	21	23	25	27	28
A6	16	16	16	16	16	15	15	10	07	07	[05]	06	06	07	08	10	13	16	18	20	22	24	26	28	30
A7	21	21	21	21	21	19	19	13	09	09	06	[05]	06	08	09	11	14	17	20	22	24	26	28	30	32
A8	27	27	27	27	27	24	24	17	12	12	08	[06]	07	09	11	13	16	19	22	25	27	29	31	33	35
A9	32	32	32	32	32	28	28	20	14	14	09	[06]	07	09	11	14	17	20	23	26	29	31	33	35	37
A10	36	36	36	36	36	31	31	22	15	15	09	07	[06]	07	08	10	13	16	19	22	25	28	31	34	37
A11	40	40	40	40	40	34	34	24	16	16	09	08	[06]	07	08	10	13	16	19	22	25	28	31	34	37
A12	43	43	43	43	43	36	36	25	16	16	10	10	07	[06]	[06]	07	09	11	13	15	17	19	21	23	25
A13	45	45	45	45	45	37	37	25	17	17	12	13	09	07	07	[06]	07	08	09	10	11	12	13	14	15
A14	47	47	47	47	47	38	38	25	18	18	14	15	11	08	08	[06]	07	08	09	10	11	12	13	14	15
A15	49	49	49	49	49	39	39	25	19	19	16	17	13	09	09	[06]	07	08	09	10	11	12	13	14	15
A16	51	51	51	51	51	40	40	25	20	20	18	19	15	10	10	[06]	07	08	09	10	11	12	13	14	15
A17	53	53	53	53	53	41	41	25	21	21	20	21	17	11	11	06	[07]	08	09	10	11	12	13	14	15
A18	55	55	55	55	55	42	42	25	22	22	22	23	19	12	12	06	07	[08]	09	10	11	12	13	14	15
A19	57	57	57	57	57	43	43	25	23	23	24	25	21	13	13	06	07	08	[09]	10	11	12	13	14	15
A20	59	59	59	59	59	44	44	25	24	24	25	27	23	14	14	06	07	08	09	[10]	11	12	13	14	15
A21	61	61	61	61	61	45	45	25	25	25	26	28	25	15	15	06	07	08	09	10	[11]	12	13	14	15
A22	63	63	63	63	63	46	46	25	26	26	27	29	27	16	16	06	07	08	09	10	11	[12]	13	14	15
A23	65	65	65	65	65	47	47	25	26	27	28	30	29	17	17	06	07	08	09	10	11	12	[13]	14	15
A24	67	67	67	67	67	48	48	25	26	27	29	31	31	18	18	06	07	08	09	10	11	12	13	[14]	15
A25	69	69	69	69	69	49	49	25	26	27	29	32	33	19	19	06	07	08	09	10	11	12	13	14	[15]

Figura 2.12 Matriz de costo y "*warping path*" entre A(t) y P(t)

Los elementos de la matriz que pertenecen al "*warping path*" se encuentran encerrados entre corchetes en la figura 2.12, el primer elemento es $(P_{25}, A_{25}) = 15\ [cm]$, el siguiente elemento es el menor entre los puntos:

$$(P_{24}, A_{24}) = 14\ [cm], (P_{24}, A_{25}) = 14\ [cm], (P_{25}, A_{24}) = 15\ [cm]$$

En este caso los puntos (P_{24}, A_{24}) y (P_{24}, A_{25}) tienen valores iguales, pero se da prioridad al elemento $(P_{24}, A_{24}) = 14\ [cm]$ ya que si las dos series estuvieran alineadas el "*warping path*" tomaría la forma de la diagonal de la matriz que va desde el punto (P_{25}, A_{25}) al punto (P_1, A_1), esta igualdad entre elementos al momento de encontrar el "*warping path*" abre la posibilidad de que existan varios caminos que permiten encontrar uno de menor costo ya que hasta este punto tendríamos dos caminos, uno por $(P_{24}, A_{24}) = 14\ [cm]$ y otro por $(P_{24}, A_{25}) = 14\ [cm]$, podría ser que al escoger uno u otro elemento se esté influyendo en el resultado final por lo que es necesario analizar todos los caminos posibles hasta obtener el menor de todos, sin embargo, hay ocasiones en las que esto se simplifica simplemente dando prioridad a la formación de la diagonal ya que si la

serie de prueba tiende a parecerse al patrón entonces el "*warping path*" no debería alejarse demasiado de la diagonal que va de (P_{25}, A_{25}) a (P_1, A_1).

Al momento de escoger los elementos que conforman el "*warping path*" es necesario hacer alusión al principio de optimalidad dado por Richard Bellman el cual dice: "Una vez conocida la solución óptima global (matriz de costo), cualquier solución parcial ("*warping path*") que involucre sólo una parte de las etapas es también una solución óptima, es decir, todo subconjunto de una solución óptima global es a su vez una solución óptima para un problema parcial". [16]

De igual manera analizando los puntos:

$$(P_{14}, A_{11}) = 7\ [cm], (P_{14}, A_{12}) = 6\ [cm], (P_{15}, A_{11}) = 8\ [cm]$$

Se tiene que el menor es $(P_{14}, A_{12}) = 6\ [cm]$ encerrado en corchetes en la figura 2.12.

En el elemento $(P_7, A_1) = 5\ [cm]$, el siguiente elemento se analiza escogiendo el menor entre los puntos:

$$(P_6, A_0) = \infty, (P_6, A_1) = 5\ [cm], (P_7, A_0) = \infty$$

Es decir, $(P_6, A_1) = 5\ [cm]$ encerrado en corchetes en la figura 2.12.

El último paso que se debe dar es aplicar el "*warping path*" en la matriz de distancias para hallar finalmente la distancia DTW como se aprecia en la figura 2.13.

Como resultado se tiene que la distancia DTW entre A(t) y P(t) es 0,45 [cm] la cual se obtiene al aplicar el "*warping path*" en la matriz de distancias y sacar el promedio entre todos sus elementos encerrados entre corchetes como se puede apreciar en la figura 2.13.

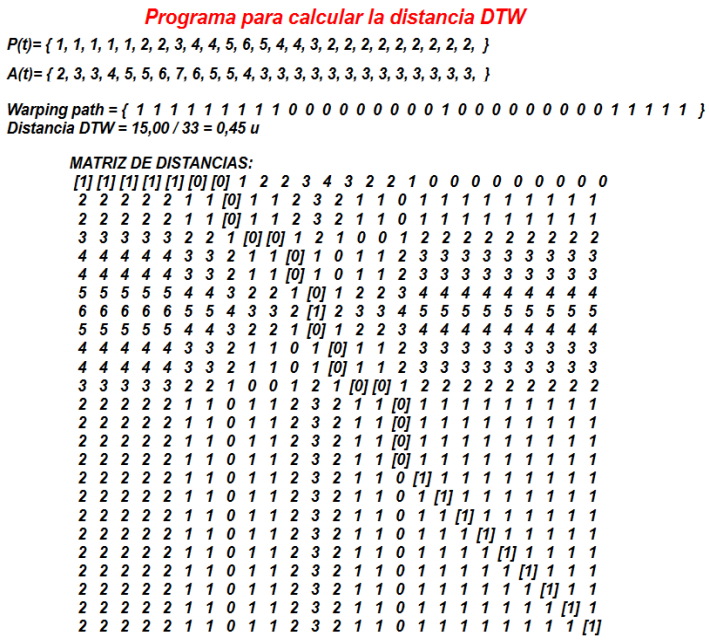


Figura 2.13 Matriz de distancias, "*warping path*" y distancia DTW entre A(t) y P(t)

De igual manera se procede a comparar las series B(t) y P(t).

Repitiendo el proceso anterior lo primero que se debe hacer es encontrar la matriz de distancias como se aprecia en la figura 2.14. Luego se halla la matriz de costo y el "*warping path*" para las series B(t) y P(t) mostrado en la figura 2.15.

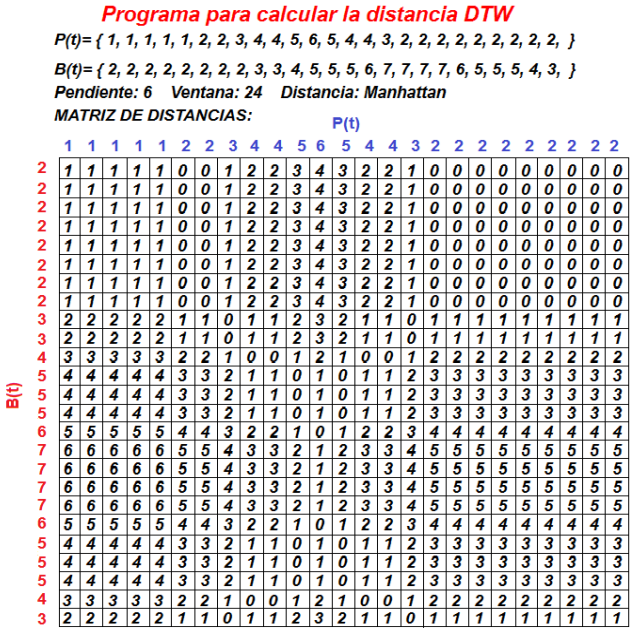


Figura 2.14 Matriz de distancias entre las series B(t) y P(t)

Programa para calcular la distancia DTW

P(t)= { 1, 1, 1, 1, 1, 2, 2, 3, 4, 4, 5, 6, 5, 4, 4, 3, 2, 2, 2, 2, 2, 2, 2, 2, 2, }

B(t)= { 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 4, 5, 5, 5, 6, 7, 7, 7, 7, 6, 5, 5, 5, 4, 3, }

Pendiente: 6   Ventana: 24   Distancia: Manhattan

MATRIZ DE COSTO y WARPING PATH:

B(t) \ P(t)	1	1	1	1	1	2	2	3	4	4	5	6	5	4	4	3	2	2	2	2	2	2	2	2	2
2	[01]	02	03	04	05	05	05	06	08	10	13	17	20	22	24	25	25	25	25	25	25	25	25	25	25
2	02	[02]	03	04	05	05	05	06	08	10	13	17	20	22	24	25	25	25	25	25	25	25	25	25	25
2	03	03	[03]	04	05	05	05	06	08	10	13	17	20	22	24	25	25	25	25	25	25	25	25	25	25
2	04	04	04	[04]	05	05	05	06	08	10	13	17	20	22	24	25	25	25	25	25	25	25	25	25	25
2	05	05	05	05	[05]	05	05	06	08	10	13	17	20	22	24	25	25	25	25	25	25	25	25	25	25
2	06	06	06	06	06	[05]	05	06	08	10	13	17	20	22	24	25	25	25	25	25	25	25	25	25	25
2	07	07	07	07	07	[05]	05	06	08	10	13	17	20	22	24	25	25	25	25	25	25	25	25	25	25
2	08	08	08	08	08	05	[05]	06	08	10	13	17	20	22	24	25	25	25	25	25	25	25	25	25	25
3	10	10	10	10	10	06	06	[05]	06	07	09	12	14	15	16	16	17	18	19	20	21	22	23	24	25
3	12	12	12	12	12	07	07	[05]	06	07	09	12	14	15	16	16	17	18	19	20	21	22	23	24	25
4	15	15	15	15	15	09	09	06	[05]	[05]	06	08	09	09	09	10	12	14	16	18	20	22	24	25	26
5	19	19	19	19	19	12	12	08	06	06	[05]	06	06	07	08	10	13	15	17	19	21	23	25	27	28
5	23	23	23	23	23	15	15	10	07	07	[05]	06	06	07	08	10	13	16	18	20	22	24	26	28	30
5	27	27	27	27	27	18	18	12	08	08	[05]	06	06	07	08	10	13	16	19	21	23	25	27	29	31
6	32	32	32	32	32	22	22	15	10	10	06	[05]	06	08	09	11	14	17	20	23	25	27	29	31	33
7	38	38	38	38	38	27	27	19	13	13	08	[06]	07	09	11	13	16	19	22	25	28	30	32	34	36
7	44	44	44	44	44	32	32	23	16	16	10	[07]	08	10	12	15	18	21	24	27	30	33	35	37	39
7	50	50	50	50	50	37	37	27	19	19	12	[08]	09	11	13	16	20	23	26	29	32	35	38	40	42
7	56	56	56	56	56	42	42	31	22	22	14	[09]	10	12	14	17	21	25	28	31	34	37	40	43	45
6	61	61	61	61	61	46	46	34	24	24	15	[09]	10	12	14	17	21	25	29	32	35	38	41	44	47
5	65	65	65	65	65	49	49	36	25	25	15	10	[09]	10	11	13	16	19	22	25	28	31	34	37	40
5	69	69	69	69	69	52	52	38	26	26	15	11	[09]	10	11	13	16	19	22	25	28	31	34	37	40
5	73	73	73	73	73	55	55	40	27	27	15	12	[09]	10	11	13	16	19	22	25	28	31	34	37	40
4	76	76	76	76	76	57	57	41	27	27	16	14	10	[09]	[09]	[10]	[12]	[14]	[16]	18	20	22	24	26	28
3	78	78	78	78	78	58	58	41	28	28	18	17	12	10	10	09	10	11	12	[13]	[14]	[15]	[16]	[17]	[18]

Figura 2.15 Matriz de costo y "*warping path*" entre las series B(t) y P(t)

Programa para calcular la distancia DTW

P(t)= { 1, 1, 1, 1, 1, 2, 2, 3, 4, 4, 5, 6, 5, 4, 4, 3, 2, 2, 2, 2, 2, 2, 2, 2, 2, }

B(t)= { 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 4, 5, 5, 5, 6, 7, 7, 7, 7, 6, 5, 5, 5, 4, 3, }

Warping path = { 1 1 1 1 1 1 2 2 2 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 }
Distancia DTW = 22,00 / 36 = 0,61 u

MATRIZ DE DISTANCIAS:

[1]	1	1	1	1	0	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0	0	0
1	[1]	1	1	1	0	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0	0	0
1	1	[1]	1	1	0	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0	0	0
1	1	1	[1]	1	0	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0	0	0
1	1	1	1	[1]	0	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0	0	0
1	1	1	1	1	[0]	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0	0	0
1	1	1	1	1	[0]	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0	0	0
1	1	1	1	1	0	[0]	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0	0	0
2	2	2	2	2	1	1	[0]	1	1	2	3	2	1	1	0	1	1	1	1	1	1	1	1	1
2	2	2	2	2	1	1	[0]	1	1	2	3	2	1	1	0	1	1	1	1	1	1	1	1	1
3	3	3	3	3	2	2	1	[0]	[0]	1	2	1	0	0	1	2	2	2	2	2	2	2	2	2
4	4	4	4	4	3	3	2	1	1	[0]	1	0	1	1	2	3	3	3	3	3	3	3	3	3
4	4	4	4	4	3	3	2	1	1	[0]	1	0	1	1	2	3	3	3	3	3	3	3	3	3
4	4	4	4	4	3	3	2	1	1	[0]	1	0	1	1	2	3	3	3	3	3	3	3	3	3
5	5	5	5	5	4	4	3	2	2	1	[0]	1	2	2	3	4	4	4	4	4	4	4	4	4
6	6	6	6	6	5	5	4	3	3	2	[1]	2	3	3	4	5	5	5	5	5	5	5	5	5
6	6	6	6	6	5	5	4	3	3	2	[1]	2	3	3	4	5	5	5	5	5	5	5	5	5
6	6	6	6	6	5	5	4	3	3	2	[1]	2	3	3	4	5	5	5	5	5	5	5	5	5
6	6	6	6	6	5	5	4	3	3	2	[1]	2	3	3	4	5	5	5	5	5	5	5	5	5
5	5	5	5	5	4	4	3	2	2	1	[0]	1	2	2	3	4	4	4	4	4	4	4	4	4
4	4	4	4	4	3	3	2	1	1	0	1	[0]	1	1	2	3	3	3	3	3	3	3	3	3
4	4	4	4	4	3	3	2	1	1	0	1	[0]	1	1	2	3	3	3	3	3	3	3	3	3
4	4	4	4	4	3	3	2	1	1	0	1	[0]	1	1	2	3	3	3	3	3	3	3	3	3
3	3	3	3	3	2	2	1	0	0	1	2	1	[0]	[0]	[1]	[2]	[2]	[2]	2	2	2	2	2	2
2	2	2	2	2	1	1	0	1	1	2	3	2	1	1	0	1	1	1	[1]	[1]	[1]	[1]	[1]	[1]

Figura 2.16 Matriz de distancias, "*warping path*" y distancia DTW entre B(t) y P(t)

Como resultado se tiene que la distancia DTW entre B(t) y P(t) es 0,61 [cm].

En resumen se obtiene que:

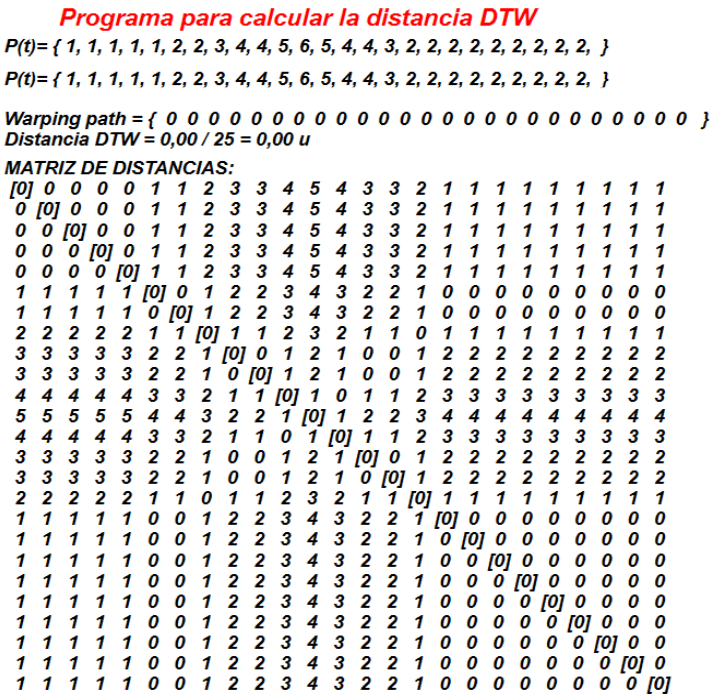
La distancia DTW entre A(t) y P(t) es 0,45 [cm]

La distancia DTW entre B(t) y P(t) es 0,61 [cm]

El resultado obtenido mediante el algoritmo del DTW es el esperado ya que efectivamente la serie A(t) es la que más se parece a la serie P(t).

Ahora, se analiza cómo varía el "*warping path*" si se compara la serie P(t) consigo misma como se aprecia en la figura 2.17.

Como era de esperarse la distancia DTW es cero y el "*warping path*" es la diagonal que va de (P_{25}, P_{25}) a (P_1, P_1) en la matriz de distancias en la figura 2.17.



Programa para calcular la distancia DTW

P(t)= { 1, 1, 1, 1, 1, 2, 2, 3, 4, 4, 5, 6, 5, 4, 4, 3, 2, 2, 2, 2, 2, 2, 2, 2, }

P(t)= { 1, 1, 1, 1, 1, 2, 2, 3, 4, 4, 5, 6, 5, 4, 4, 3, 2, 2, 2, 2, 2, 2, 2, 2, }

Warping path = { 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 }
Distancia DTW = 0,00 / 25 = 0,00 u

MATRIZ DE DISTANCIAS:

[0]	0	0	0	0	1	1	2	3	3	4	5	4	3	3	2	1	1	1	1	1	1	1	1
0	[0]	0	0	0	1	1	2	3	3	4	5	4	3	3	2	1	1	1	1	1	1	1	1
0	0	[0]	0	0	1	1	2	3	3	4	5	4	3	3	2	1	1	1	1	1	1	1	1
0	0	0	[0]	0	1	1	2	3	3	4	5	4	3	3	2	1	1	1	1	1	1	1	1
0	0	0	0	[0]	1	1	2	3	3	4	5	4	3	3	2	1	1	1	1	1	1	1	1
1	1	1	1	1	[0]	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0	0
1	1	1	1	1	0	[0]	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0	0
2	2	2	2	2	1	1	[0]	1	1	2	3	2	1	1	0	1	1	1	1	1	1	1	1
3	3	3	3	3	2	2	1	[0]	0	1	2	1	0	0	1	2	2	2	2	2	2	2	2
3	3	3	3	3	2	2	1	0	[0]	1	2	1	0	0	1	2	2	2	2	2	2	2	2
4	4	4	4	4	3	3	2	1	1	[0]	1	0	1	1	2	3	3	3	3	3	3	3	3
5	5	5	5	5	4	4	3	2	2	1	[0]	1	2	2	3	4	4	4	4	4	4	4	4
4	4	4	4	4	3	3	2	1	1	0	1	[0]	1	1	2	3	3	3	3	3	3	3	3
3	3	3	3	3	2	2	1	0	0	1	2	1	[0]	0	1	2	2	2	2	2	2	2	2
3	3	3	3	3	2	2	1	0	0	1	2	1	0	[0]	1	2	2	2	2	2	2	2	2
2	2	2	2	2	1	1	0	1	1	2	3	2	1	1	[0]	1	1	1	1	1	1	1	1
1	1	1	1	1	0	0	1	2	2	3	4	3	2	2	1	[0]	0	0	0	0	0	0	0
1	1	1	1	1	0	0	1	2	2	3	4	3	2	2	1	0	[0]	0	0	0	0	0	0
1	1	1	1	1	0	0	1	2	2	3	4	3	2	2	1	0	0	[0]	0	0	0	0	0
1	1	1	1	1	0	0	1	2	2	3	4	3	2	2	1	0	0	0	[0]	0	0	0	0
1	1	1	1	1	0	0	1	2	2	3	4	3	2	2	1	0	0	0	0	[0]	0	0	0
1	1	1	1	1	0	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	[0]	0	0
1	1	1	1	1	0	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	[0]	0
1	1	1	1	1	0	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0	[0]

Figura 2.17 Matriz de distancias, "*warping path*" y distancia DTW entre P(t) y P(t)

Hay ocasiones en las que la serie de prueba y el patrón resultan ser muy diferentes excepto por pequeños segmentos que resultan ser bastante similares dando como resultado una comparación errada ya que el algoritmo del DTW podría estancarse en esas pequeñas similitudes y no analizar las dos series de forma global y uniforme, para que esto no ocurra es necesario aplicar algunas restricciones al momento de resolver el algoritmo del DTW.

1.6 RESTRICCIONES DEL DTW

Las restricciones que se aplican en el DTW son: restricción de monotonía, restricción de continuidad, condiciones de frontera, restricción de ventana y restricción de pendiente.

1.6.1 RESTRICCIÓN DE MONOTONÍA

La restricción de monotonía se aplica de manera implícita al momento de encontrar los elementos que conforman el "*warping path*" en la matriz de costo, así:

$$min(c_{i-1,j-1}, c_{i-1,j}, c_{i,j-1})$$

Es decir, cada nuevo elemento del "*warping path*" se elige entre los inmediatos anteriores y no se toman en cuenta elementos como:

$$min(c_{i+1,j+1}, c_{i+1,j}, c_{i,j+1})$$

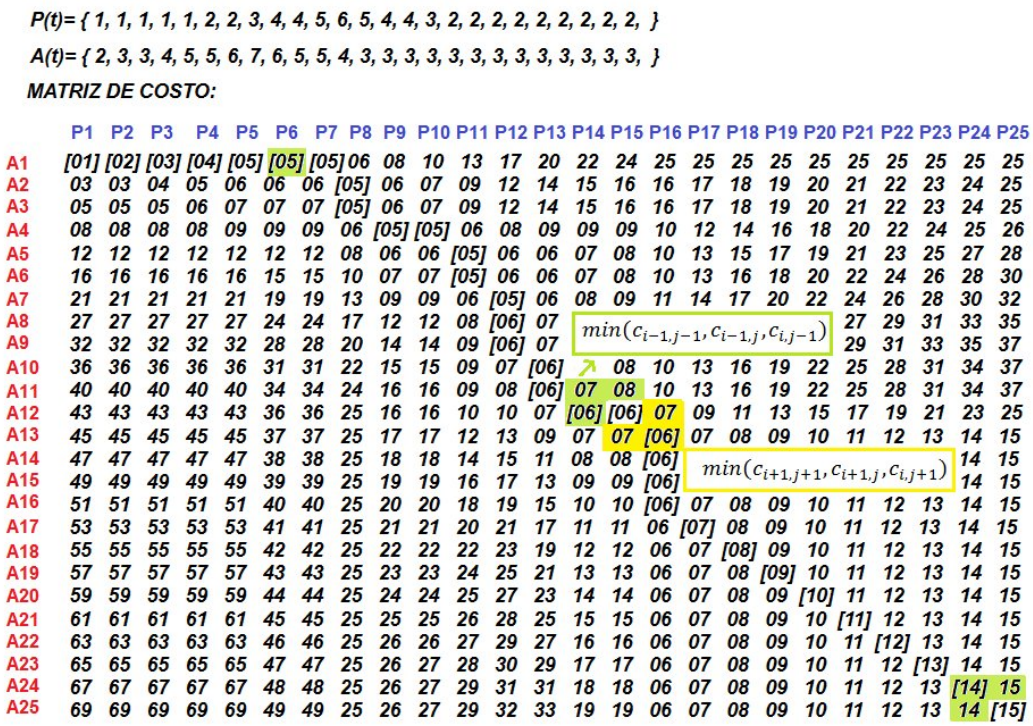


Figura 2.18 Restricción de monotonía aplicado al "*warping path*" entre las series A(t) y P(t)

Esto asegura que el "*warping path*" no regrese en el tiempo y que ciertas características se repitan, como se aprecia en la figura 2.18.

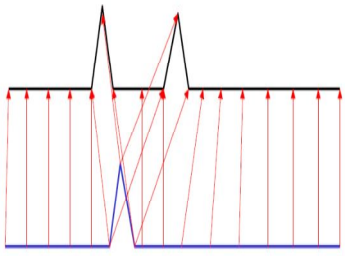


Figura 2.19 Ejemplo de la restricción de monotonía, tomado de [17]

En el ejemplo de la figura 2.19 se puede apreciar que si no se aplica esta restricción el algoritmo del DTW podría estancarse en pequeños segmentos parecidos y como consecuencia decir que las dos series son iguales.

1.6.2 RESTRICCIÓN DE CONTINUIDAD

Esta restricción dice que no se puede saltar ningún valor al momento de obtener el "*warping path*" y está implícita en la fórmula:

$$min(c_{i-1,j-1}, c_{i-1,j}, c_{i,j-1})$$

Es decir, no se puede hacer lo siguiente:

$$min(c_{i-2,j-2}, c_{i-2,j}, c_{i,j-2})$$

Esto garantiza que no se omita ninguna característica que podría ser importante como se ilustra en la figura 2.20.

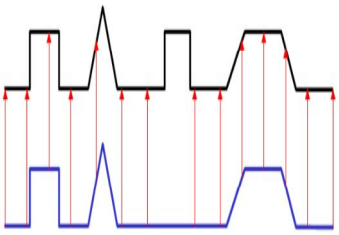


Figura 2.20 Ejemplo de la restricción de continuidad, tomado de [17]

1.6.3 RESTRICCIÓN DE FRONTERA

Esta restricción fue aplicada cuando se comparó las series A(t) y B(t) con P(t) y dice que el "*warping path*" va de (P_n, A_m) a (P_1, A_1) en la matriz de costo.

Esto asegura que se considere toda la secuencia y no sólo parte de ella como se muestra en la figura 2.21.

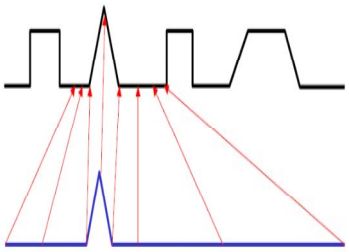


Figura 2.21 Ejemplo de la condición de frontera, tomado de [17]

1.6.4 RESTRICCIÓN DE VENTANA

Esta restricción no se aplica directamente en el "*warping path*" sino en la matriz de costo y de distancias, la idea es recortarlas para forzar que el "*warping path*" no se aleje demasiado de la diagonal que va desde (P_n, A_m) hasta (P_1, A_1).

Para comprender la utilidad de esta restricción se muestra el siguiente ejemplo:

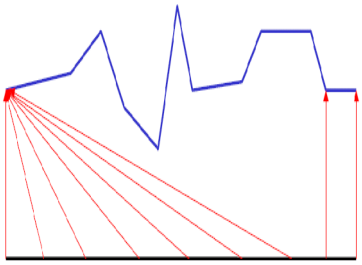


Figura 2.22 Ejemplo de la restricción de ventana, tomado de [17]

Si no se aplica la restricción de ventana en la matriz de costo el DTW podría concluir que las dos series de la figura 2.22 son semejantes.

En otras palabras, esta restricción asegura que el "*warping path*" no se quede estancado en un solo valor.

Aplicando esta restricción en la matriz de costo se puede decir que para que un elemento exista debe cumplir con la siguiente condición:

$$|i_c - j_c| \leq r, \;\; r > 0$$

$i_c, es\ la\ pocisión\ en\ i\ de\ cualquier\ elemento\ en\ la\ matriz\ de\ costo$

$j_c, es\ la\ pocisión\ en\ j\ de\ cualquier\ elemento\ en\ la\ matriz\ de\ costo$

Aplicando esta restricción en el ejemplo de las series A(t) y P(t) se tiene la matriz que se observa en la figura 2.23.

Aunque la matriz de distancias es la que aparece recortada con una ventana de valor 9 en la figura 2.23, hay que tomar en cuenta que es la matriz de costo la que en primera instancia es recortada ya que es en ella en la que se halla el "*warping path*" que luego es aplicado a la matriz de distancias.

Analizando algunos elementos en la matriz de la figura 2.23 empleando la restricción de ventana.

Para el punto (P_1, A_{10}) su ubicación en la matriz es (1,10) y la ventana se obtiene así:

$$|1 - 10| \leq 9$$

$$9 \leq 9$$

Como la condición de ventana se cumple, entonces el elemento (P_1, A_{10}) si existe.

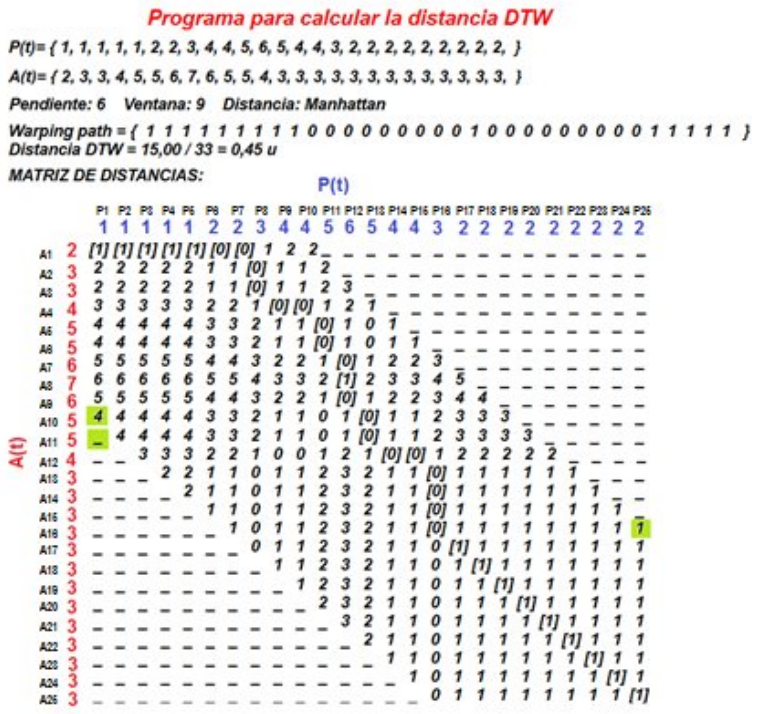


Figura 2.23 Ejemplo de la restricción de ventana entre las series A(t) y F(t)

Para el punto (P_1, A_{11}) su ubicación en la matriz es (1,11) y la ventana se obtiene así:

$$|1 - 11| \leq 9$$

$$10 \leq 9$$

Como la restricción de ventana no se cumple para el elemento (P_1, A_{11}) entonces este elemento no existe y su valor es igual a infinito representado por un guion en la figura 2.23.

Por último, para el punto (P_{25}, A_{16}) su ubicación en la matriz es (25,16) y la ventana se obtiene así:

$$|25 - 16| \leq 9$$

$$9 \leq 9$$

Como la restricción de ventana se cumple, entonces el elemento (P_{25}, A_{16}) si existe.

1.6.5 RESTRICCIÓN DE PENDIENTE

Esta restricción se aplica en el "*warping path*" con el objetivo de evitar que pequeñas partes de una serie coincidan con largas partes de otra, es decir, el "*warping path*" no debe tener muchos pasos ni en "i" (sentido horizontal), ni en "j" (sentido vertical) lo cual permite analizar las dos series de manera uniforme como se ilustra en la figura 2.24

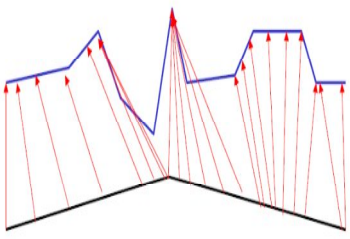


Figura 2.24 Ejemplo de la restricción de pendiente, tomado de [17]

Esta restricción se aplica cuando se encuentra el "*warping path*" mediante la fórmula:

$$min(c_{i-1,j-1}, c_{i-1,j}, c_{i,j-1})$$

Cada vez que un elemento del "*warping path*" recae en el elemento $c_{i-1,j}$ se empieza a contabilizar el número de veces que se repite esta posición para controlar que no supere el valor que se haya dado a la pendiente en "i".

De igual manera, cada vez que un elemento del "*warping path*" recae en el elemento $c_{i,j-1}$ se empieza a contabilizar el número de veces que se repite esta posición para controlar que no supere el valor que se haya dado a la pendiente en "j".

Usualmente, el valor de la pendiente para "i" y "j" es el mismo.

Para visualizar esta restricción se analiza la matriz de costo y el "*warping path*" entre las series B(t) y P(t):

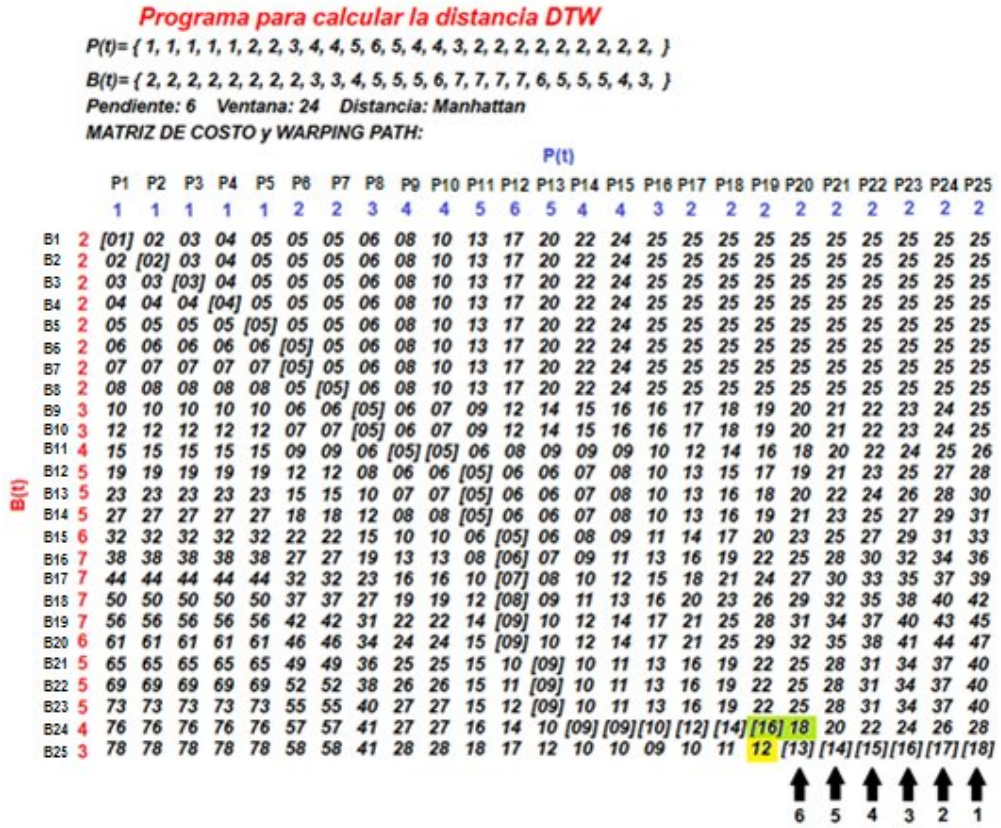
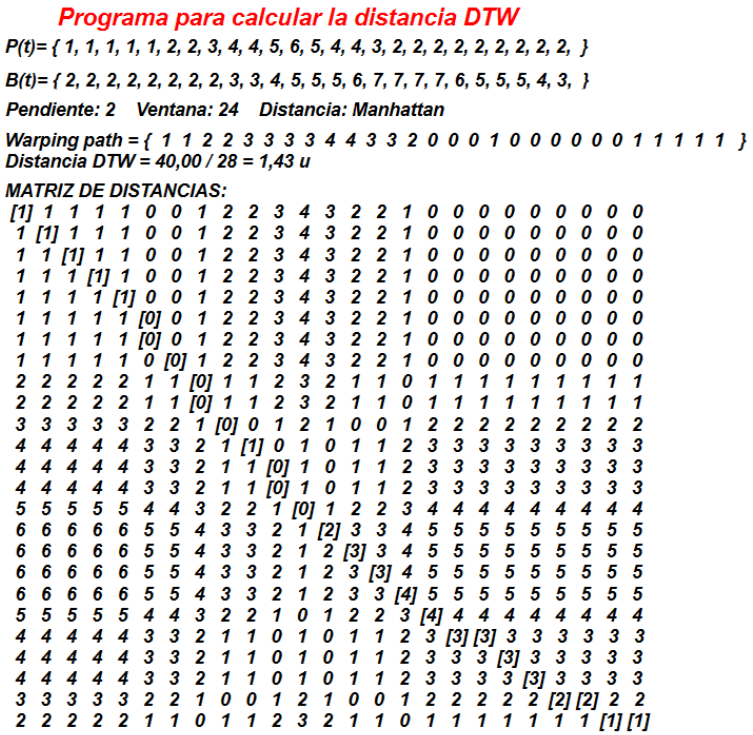


Figura 2.25 Restricción de pendiente entre las series B(t) y P(t)

El valor de la pendiente es 6 en la figura 2.25, esto significa que en el "*warping path*" sólo se pueden dar máximo 6 pasos consecutivos en sentido horizontal o vertical como ocurre de (P_{25}, B_{25}) hasta $(P_{20}, B_{25}) = 13\,[cm]$ en el que se

contabilizan 6 pasos consecutivos en sentido horizontal por lo que a pesar que el siguiente elemento del "*warping path*" debería ser $(P_{19}, B_{25}) = 12\ [cm]$ de acuerdo con la fórmula del "*warping path*" la restricción de pendiente hace que el siguiente elemento sea $(P_{19}, B_{24}) = 16\ [cm]$ para hacer que el "*warping path*" no se deforme tanto.

Ahora se ve lo que sucede con el "*warping path*" cuando el valor de la pendiente es de 2 entre las series B(t) y P(t).



Programa para calcular la distancia DTW

P(t)= { 1, 1, 1, 1, 1, 2, 2, 3, 4, 4, 5, 6, 5, 4, 4, 3, 2, 2, 2, 2, 2, 2, 2, 2, }

B(t)= { 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 4, 5, 5, 5, 6, 7, 7, 7, 7, 6, 5, 5, 5, 4, 3, }

Pendiente: 2 Ventana: 24 Distancia: Manhattan

Warping path = { 1 1 2 2 3 3 3 3 4 4 4 3 3 2 0 0 0 1 0 0 0 0 0 0 1 1 1 1 1 }
Distancia DTW = 40,00 / 28 = 1,43 u

MATRIZ DE DISTANCIAS:

[1]	1	1	1	0	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0	0
1	[1]	1	1	1	0	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0
1	1	[1]	1	1	0	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0
1	1	1	[1]	1	0	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0
1	1	1	1	[1]	0	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0
1	1	1	1	1	[0]	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0
1	1	1	1	1	[0]	0	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0
1	1	1	1	1	0	[0]	1	2	2	3	4	3	2	2	1	0	0	0	0	0	0	0
2	2	2	2	2	1	1	[0]	1	1	2	3	2	1	1	0	1	1	1	1	1	1	1
2	2	2	2	2	1	1	[0]	1	1	2	3	2	1	1	0	1	1	1	1	1	1	1
3	3	3	3	3	2	2	1	[0]	0	1	2	1	0	0	1	2	2	2	2	2	2	2
4	4	4	4	4	3	3	2	1	[1]	0	1	0	1	1	2	3	3	3	3	3	3	3
4	4	4	4	4	3	3	2	1	1	[0]	1	0	1	1	2	3	3	3	3	3	3	3
4	4	4	4	4	3	3	2	1	1	[0]	1	0	1	1	2	3	3	3	3	3	3	3
5	5	5	5	5	4	4	3	2	2	1	[0]	1	2	2	3	4	4	4	4	4	4	4
6	6	6	6	6	5	5	4	3	3	2	1	[2]	3	3	4	5	5	5	5	5	5	5
6	6	6	6	6	5	5	4	3	3	2	1	2	[3]	3	4	5	5	5	5	5	5	5
6	6	6	6	6	5	5	4	3	3	2	1	2	3	[3]	4	5	5	5	5	5	5	5
6	6	6	6	6	5	5	4	3	3	2	1	2	3	3	[4]	5	5	5	5	5	5	5
5	5	5	5	5	4	4	3	2	2	1	0	1	2	2	3	[4]	4	4	4	4	4	4
4	4	4	4	4	3	3	2	1	1	0	1	0	1	1	2	3	[3]	[3]	3	3	3	3
4	4	4	4	4	3	3	2	1	1	0	1	0	1	1	2	3	3	3	[3]	3	3	3
4	4	4	4	4	3	3	2	1	1	0	1	0	1	1	2	3	3	3	3	[3]	3	3
3	3	3	3	3	2	2	1	0	0	1	2	1	0	0	1	2	2	2	2	2	[2]	[2]	2	2
2	2	2	2	2	1	1	0	1	1	2	3	2	1	1	0	1	1	1	1	1	1	1	[1]	[1]

Figura 2.26 Restricción de pendiente de valor 2 entre las series B(t) y P(t)

Las series deben ser más parecidas a medida que el valor de la pendiente se reduce ya que se obliga a que el "*warping path*" tome la forma de la diagonal que va de (P_n, B_m) a (P_1, B_1), por tal motivo la distancia DTW entre las series B(t) y P(t) es mayor como se aprecia en la figura 2.26. En otras palabras, esta restricción pone límites a la semejanza entre las series analizadas.

Ahora se ve en la figura 2.27 el resultado de aumentar el valor de la pendiente entre las series B(t) y P(t).

Como era de esperarse la distancia DTW disminuyó al aumentar el valor de la pendiente.

Todas estas restricciones ayudan a que no se cometan errores al momento de reconocer patrones y deben calibrarse de acuerdo a la semejanza que se requiera.

Programa para calcular la distancia DTW

```
P(t)= { 1, 1, 1, 1, 1, 2, 2, 3, 4, 4, 5, 6, 5, 4, 4, 3, 2, 2, 2, 2, 2, 2, 2, 2, 2,  }

B(t)= { 2, 2, 2, 2, 2, 2, 2, 2, 3, 3, 4, 5, 5, 5, 6, 7, 7, 7, 7, 6, 5, 5, 5, 4, 3,  }

Pendiente: 10    Ventana: 24    Distancia: Manhattan

Warping path = {  1  1  1  1  1  1  1  1  1  0  0  0  0  0  0  0  1  1  1  1  0  0  0  0  0  0  0  0  0  0  0  1  1  1  1  1  }
Distancia DTW = 18,00 / 36 = 0,50 u

MATRIZ DE DISTANCIAS:
[1]  1   1   1   1   0   0   1   2   2   3   4   3   2   2   1   0   0   0   0   0   0   0   0   0
 1  [1]  1   1   1   0   0   1   2   2   3   4   3   2   2   1   0   0   0   0   0   0   0   0   0
 1   1  [1]  1   1   0   0   1   2   2   3   4   3   2   2   1   0   0   0   0   0   0   0   0   0
 1   1   1  [1]  1   0   0   1   2   2   3   4   3   2   2   1   0   0   0   0   0   0   0   0   0
 1   1   1   1  [1]  0   0   1   2   2   3   4   3   2   2   1   0   0   0   0   0   0   0   0   0
 1   1   1   1   1  [0]  0   1   2   2   3   4   3   2   2   1   0   0   0   0   0   0   0   0   0
 1   1   1   1   1  [0]  0   1   2   2   3   4   3   2   2   1   0   0   0   0   0   0   0   0   0
 1   1   1   1   1   0  [0]  1   2   2   3   4   3   2   2   1   0   0   0   0   0   0   0   0   0
 2   2   2   2   2   1   1  [0]  1   1   2   3   2   1   1   0   1   1   1   1   1   1   1   1   1
 2   2   2   2   2   1   1  [0]  1   1   2   3   2   1   1   0   1   1   1   1   1   1   1   1   1
 3   3   3   3   3   2   2   1  [0] [0]  1   2   1   0   0   1   2   2   2   2   2   2   2   2   2
 4   4   4   4   4   3   3   2   1   1  [0]  1   0   1   1   2   3   3   3   3   3   3   3   3   3
 4   4   4   4   4   3   3   2   1   1  [0]  1   0   1   1   2   3   3   3   3   3   3   3   3   3
 4   4   4   4   4   3   3   2   1   1  [0]  1   0   1   1   2   3   3   3   3   3   3   3   3   3
 5   5   5   5   5   4   4   3   2   2   1  [0]  1   2   2   3   4   4   4   4   4   4   4   4   4
 6   6   6   6   6   5   5   4   3   3   2  [1]  2   3   3   4   5   5   5   5   5   5   5   5   5
 6   6   6   6   6   5   5   4   3   3   2  [1]  2   3   3   4   5   5   5   5   5   5   5   5   5
 6   6   6   6   6   5   5   4   3   3   2  [1]  2   3   3   4   5   5   5   5   5   5   5   5   5
 6   6   6   6   6   5   5   4   3   3   2  [1]  2   3   3   4   5   5   5   5   5   5   5   5   5
 5   5   5   5   5   4   4   3   2   2   1  [0]  1   2   2   3   4   4   4   4   4   4   4   4   4
 4   4   4   4   4   3   3   2   1   1   0   1  [0]  1   1   2   3   3   3   3   3   3   3   3   3
 4   4   4   4   4   3   3   2   1   1   0   1  [0]  1   1   2   3   3   3   3   3   3   3   3   3
 4   4   4   4   4   3   3   2   1   1   0   1  [0]  1   1   2   3   3   3   3   3   3   3   3   3
 3   3   3   3   3   2   2   1   0   0   1   2   1  [0] [0]  1   2   2   2   2   2   2   2   2   2
 2   2   2   2   2   1   1   0   1   1   2   3   2   1   1  [0] [1] [1] [1] [1] [1] [1] [1] [1] [1]
```

Figura 2.27 Restricción de pendiente de valor 10 entre las series B(t) y P(t)

1.7 APLICACIONES DEL DTW

Vídeo, audio, gráficos y todos aquellos datos que se puedan convertir en una representación lineal pueden ser analizados por el DTW. Una aplicación bien conocida ha sido el reconocimiento de voz, para hacer frente a diferentes velocidades del habla. Otras aplicaciones incluyen el reconocimiento de la firma de una persona y detección de similitudes en los patrones de caminata, incluso si en un vídeo una persona camina lento y en otro lo hace más rápido, tomando en cuenta además si hubo aceleraciones y desaceleraciones durante el transcurso de una observación. [18]

CAPÍTULO 3

DESARROLLO E IMPLEMENTACIÓN DEL SOFTWARE

1.8 INTRODUCCIÓN

En este capítulo se explica el funcionamiento del *software* de reconocimiento de patrones mediante la presentación de la interfaz gráfica del programa y la aplicación del algoritmo del DTW con el objetivo de controlar diapositivas.

El *software* ofrece al usuario las herramientas necesarias para grabar patrones, asignarles un nombre y un comando de control de diapositivas, además el usuario puede editar, archivar y recuperar las listas de patrones creadas.

El *software* también permite configurar parámetros del algoritmo del DTW que intervienen en el reconocimiento de patrones y mostrar en pantalla los cálculos realizados cada vez que un patrón es reconocido.

1.9 PROCESO DE RECONOCIMIENTO DE PATRONES DE MOVIMIENTO

El proceso de reconocimiento de patrones empieza cuando el sensor kinect envía información al programa sobre la ubicación del usuario en la escena. La información es acondicionada y almacenada en un *buffer* de datos para luego ser analizada por el algoritmo del DTW y verificar si se ha realizado algún patrón para finalmente proceder a controlar diapositivas como se muestra en la figura 3.1

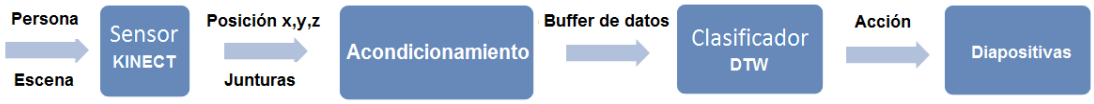


Figura 3.1 Proceso de reconocimiento de patrones de movimiento

La información que el kinect genera debe ser acondicionada por dos motivos: para evitar repetición en la lectura de datos y para cambiar el origen del eje de referencia. Cuando el kinect detecta a una persona empieza a generar continuamente información de la ubicación de sus articulaciones sin importar si el usuario ha cambiado de posición en la escena por lo que es necesario evitar que el *buffer* de datos se llene de datos repetidos. También, es necesario cambiar el

origen del eje de referencia del kinect a uno que sea relativo a la ubicación de la persona para que al momento de repetir los movimientos no importe que el usuario cambie de posición en la escena.

El kinect tiene la capacidad de detectar hasta 6 personas en una escena, pero el programa ha sido diseñado para leer únicamente los movimientos de la persona más cercana al kinect, es decir, sólo una persona interactúa con el programa y cuando esta sale de la escena se escoge a la persona más cercana entre las que quedan como se aprecia en la figura 3.2.

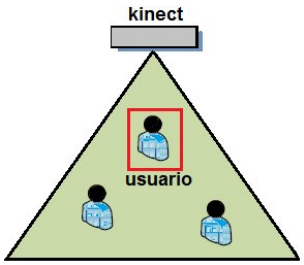


Figura 3.2 Elección del usuario más cercano al kinect

1.10 DESCRIPCIÓN DE LA INTERFAZ GRÁFICA DEL PROGRAMA

El HMI del programa ha sido desarrollado en el marco de *Windows Presentation Foundation* (WPF) el cual permite crear aplicaciones con una amplia gama de elementos como botones, etiquetas, cuadros de texto, listas, imágenes, tablas, etc. los que se enlazan de manera sencilla con la programación en C#.

El programa presenta herramientas que permiten grabar, editar y configurar patrones así como visualizar el algoritmo del DTW.

En la figura 3.3 se puede apreciar la pantalla principal del programa que contiene los siguientes elementos: 1) Una “barra de menú” que contiene todas las herramientas que ofrece el programa, 2) Un “*Canvas*” en el que se dibuja el esqueleto de la persona que realiza los movimientos y también se muestra el listado de patrones que se van reconociendo junto con algunos datos de *status*, 3) Un “*Tab*” con varias pestañas las que permiten alternar entre las ventanas del programa, 4) Un “*ComboBox*” que contiene la lista de patrones grabados, 5) Un “*ProgressBar*” para visualizar cómo se llena el *buffer* de datos cuando se graba un patrón y para ver el porcentaje de semejanza entre el patrón y el movimiento

realizado por un usuario, 6) Un “*Slider*” para regular el ángulo de elevación del kinect, 7) Un “*CheckBox*” para habilitar el control de diapositivas y 8) Un “*Button*” para grabar nuevos patrones.

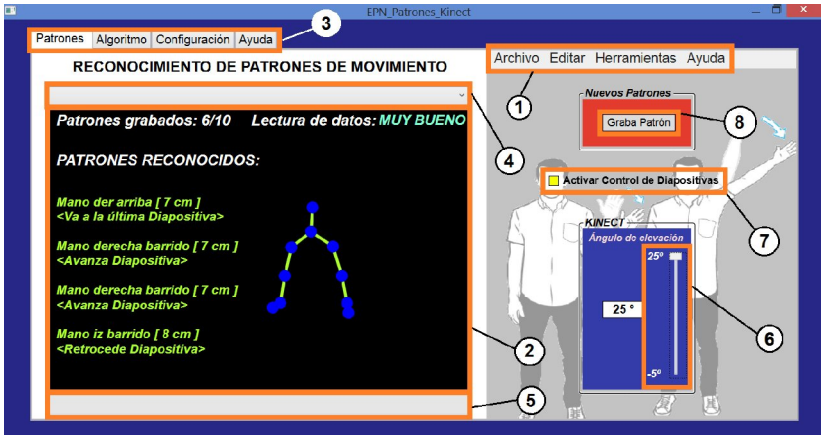


Figura 3.3 Descripción de la pantalla principal del programa

Para el detalle de cada uno de los elementos que conforman el programa referirse al manual de usuario en el anexo A.

1.11 DESARROLLO E IMPLEMENTACIÓN DEL ALGORITMO DTW

El programa funciona en base a “eventos” los cuales permiten llevar a cabo las siguientes tareas: recoger datos desde el sensor kinect, implementar el algoritmo del DTW y controlar diapositivas.

En el diagrama de flujo de la figura 3.4 se describe el funcionamiento general del programa. Al ejecutar el programa, los primeros pasos que se realizan son configurar el sensor kinect, declarar “eventos” e inicializar variables. Luego, el programa espera a que el kinect detecte algún usuario en la escena para lanzar el evento “*SkeletonFrameReady*” en el que se recibe y procesa la información de la posición de las articulaciones del usuario. Cada vez que el evento “*SkeletonFrameReady*” se ejecuta lanza otro evento llamado “ReconocePatrones” en el que se aplica el algoritmo del DTW el cual lanza a su vez el evento “ActivaDiapositiva” para controlar diapositivas.

Como se aprecia, la estructura del programa se halla basada en “eventos” los cuales ejecutan tareas cada vez que son llamados.

Por ejemplo, si en el HMI un usuario da un *click* sobre un botón el programa lanza el evento “*BottonClick*” y ejecuta el código escrito dentro del mismo, de igual manera si un usuario da un *click* sobre cualquier otro elemento del HMI se genera su respectivo evento para llevar a cabo una tarea. Cualquier evento lanzado desde el HMI puede ser llamado sin necesidad de declararlo, pero para crear eventos que son lanzados por la lógica del programa como los que se muestran en la figura 3.4 es necesario crearlos al inicio del programa como cualquier otra variable. Por ejemplo, el evento “*SkeletonFrameReady*” es lanzado por el sensor kinect cada vez que detecta a un usuario en la escena.

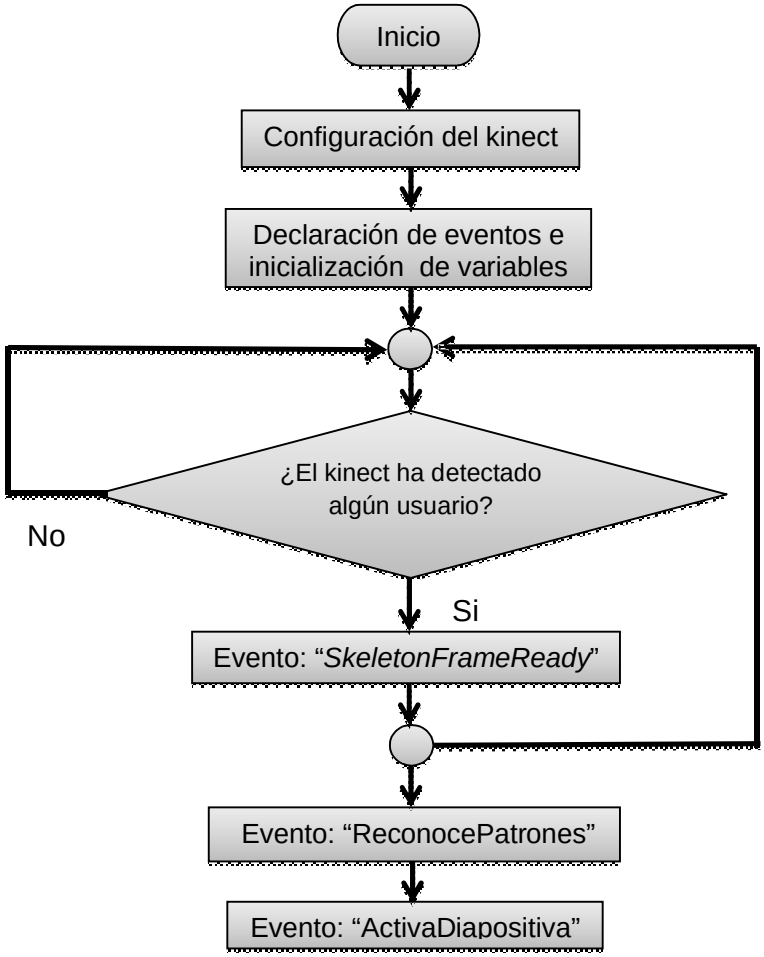


Figura 3.4 Diagrama de flujo general

Un evento genera información útil para otro, por ejemplo, el evento "*SkeletonFrameReady*" lee y acondiciona información del kinect para enviarla al evento "ReconocePatrones" y este envía información al evento "AcitivaDiapositiva", la importancia del uso de eventos radica en que todas estas tareas se concatenan a pesar que cada evento se ejecute a su propia velocidad ya que no es posible realizar todas estas tareas dentro de un solo evento porque no es posible interrumpir al sensor kinect pues su única función es la de leer datos a una velocidad de 30 [fps] y no podría esperar hasta aplicar el algoritmo del DTW.

El sensor kinect ofrece varios tipos de información al usuario por lo que se debe configurar exclusivamente las funciones útiles para el reconocimiento de patrones, en la figura 3.6 se describe de manera detallada cómo se inicializa el programa. Cuando se ejecuta el programa primero se verifica si existe algún sensor kinect conectado al puerto USB del computador, luego se activa la función "*skeleton stream*" en modo "sentado" que permite obtener información de las articulaciones de las extremidades superiores del cuerpo humano, luego se inicializan las variables del programa y por último se declaran tres eventos: "*SkeletonFrameReady*", "ReconocePatrones" y "ActivaDiapositiva".

Todo el proceso de reconocimiento de patrones se lleva a cabo dentro de los nuevos eventos creados. En el evento "*SkeletonFrameReady*" se receptan y acondicionan los datos generados por el sensor kinect los cuales son enviados al siguiente evento "ReconocePatrones" en el que se crea un *buffer* de datos y se aplica el algoritmo del DTW para luego enviar al evento "ActivaDiapositiva" el nombre del patrón reconocido para encontrar el correspondiente comando para controlar diapositivas en *power point* como se muestra en la figura 3.5

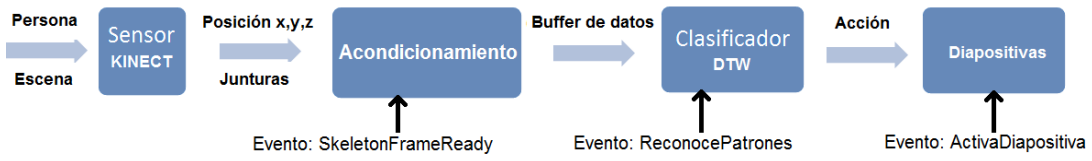


Figura 3.5 Proceso de reconocimiento de patrones ligados a "eventos"

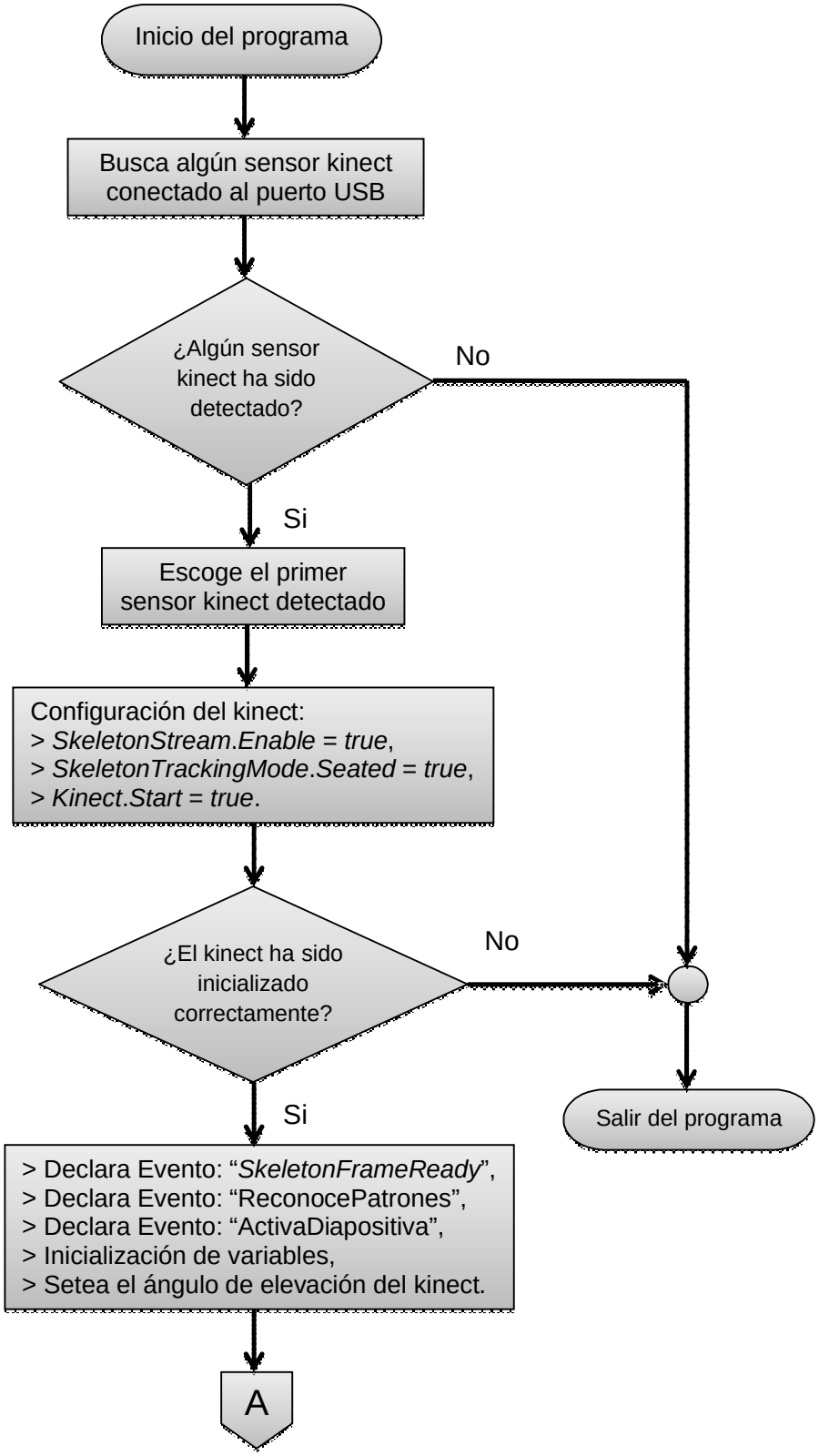


Figura 3.6 Diagrama de flujo de la inicialización del programa

Para dibujar el esqueleto del usuario en la pantalla del computador es necesario tomar la información de las articulaciones en tres dimensiones y transformarlas a dos dimensiones sin perder la perspectiva de profundidad, afortunadamente el kinect presenta una función llamada "*MapSkeletonPointToColor*" la cual permite mapear los puntos en 3D y llevarlos a 2D como se puede apreciar en la figura 3.7.

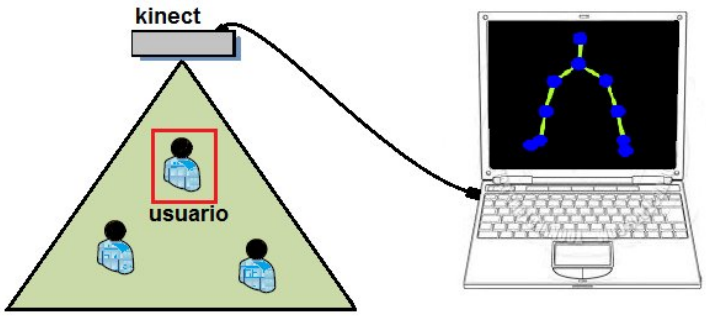


Figura 3.7 Gráfico de las extremidades superiores del esqueleto del usuario

Una vez obtenida la información de la persona más cercana al kinect es necesario cambiar el sistema de referencia del kinect a uno que sea relativo a la ubicación de la persona en la escena en el cual el origen de referencia sea una articulación perteneciente a la misma persona para que los movimientos sean siempre medidos desde un mismo punto. La articulación escogida como origen para el nuevo sistema de referencia es el "*ShoulderCenter*" como se muestra en la figura 3.8. Por ejemplo, la posición de la articulación llamada "*HandLeft*" está medida desde la articulación "*ShoulderCenter*" y no desde el sensor kinect, lo mismo ocurre con las demás articulaciones.

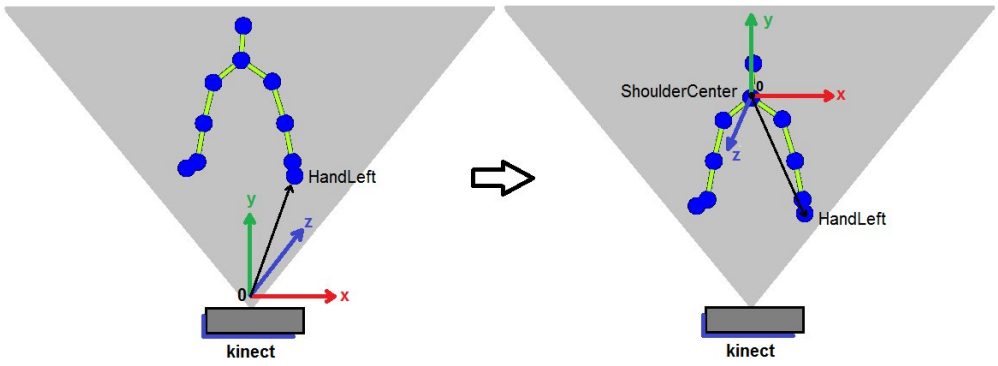


Figura 3.8 Cambio de origen del sistema de referencia

Para obtener la ubicación de las articulaciones del usuario en el nuevo sistema de referencia simplemente se resta vectores como se aprecia en la figura 3.9.

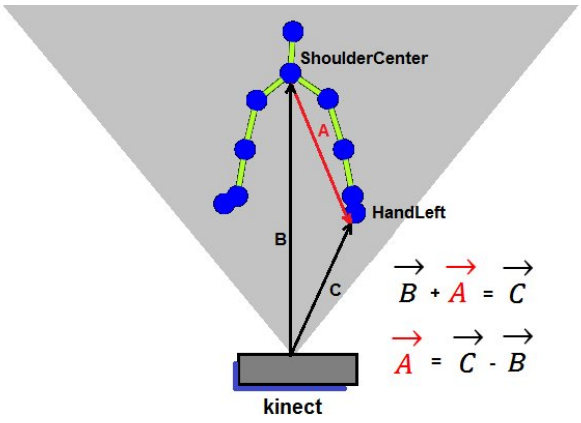


Figura 3.9 Sistema de referencia con respecto a la articulación "*ShoulderCenter*"

Una vez obtenido el nuevo sistema de referencia se llama al evento "ReconocePatrones" para enviarle la información de la posición del usuario como se muestra en el diagrama de flujo de la figura 3.10.

Cada vez que el evento "*SkeletonFrameReady*" obtiene nueva información lanza el evento "ReconocePatrones" el cual recibe los datos de las articulaciones siempre y cuando éstos sean válidos.

Un dato se considera como no válido cuando su valor es igual a "NaN" que por sus siglas en inglés significa "*not a number*" el cual aparece cuando el resultado de una operación es indefinido, por ejemplo el resultado de dividir cero por cero es "NaN" y puede aparecer cuando el usuario se ubica cerca de los límites máximos de visión del kinect, es decir, cuando parte del usuario se encuentra dentro de la escena y parte se encuentra fuera.

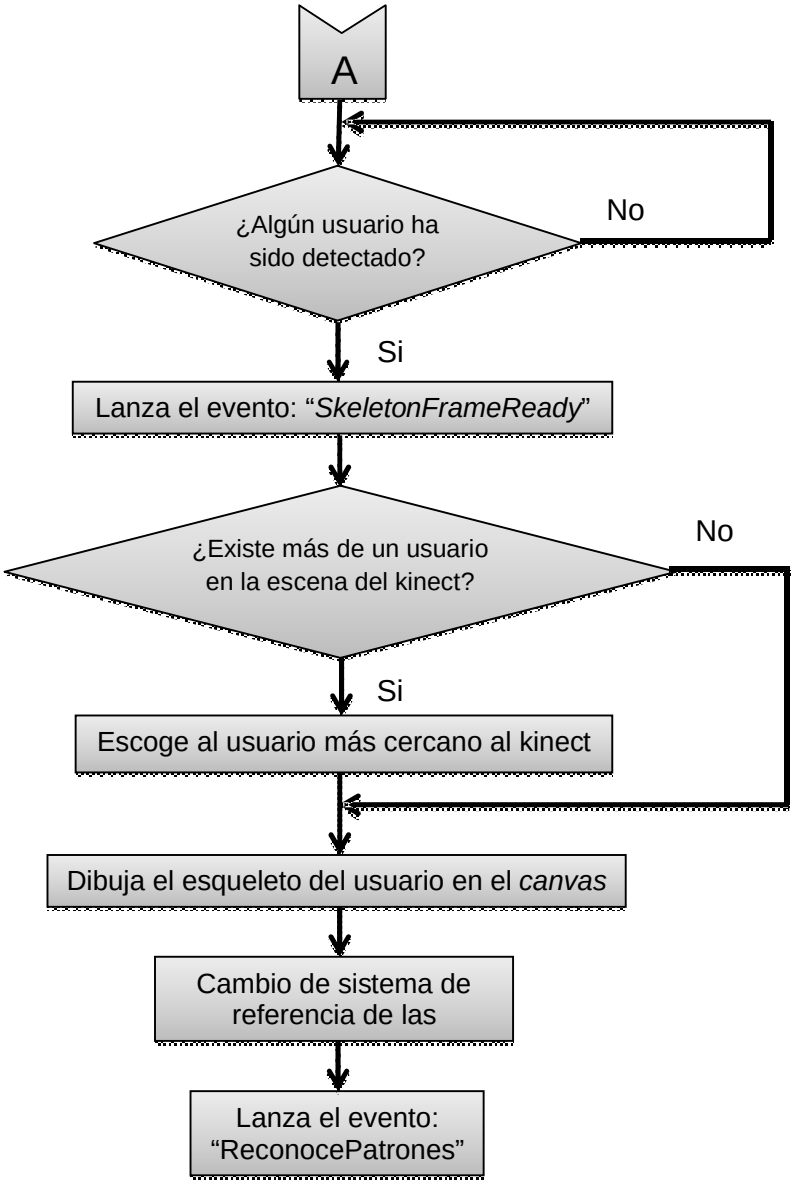


Figura 3.10 Diagrama de flujo del evento "*SkeletonFrameReady*"

En esta sección los datos de las articulaciones se agrupan en un solo arreglo llamado "*frame*". Cada *frame* se almacena en el *buffer* de datos el cual puede contener máximo 50 *frames*. Como el programa está siempre recolectando datos el *buffer* debe ir almacenando los nuevos *frames* e ir borrando los antiguos mientras el algoritmo del DTW reconoce si existe algún patrón grabado dentro del *buffer* como se muestra en la figura 3.11.

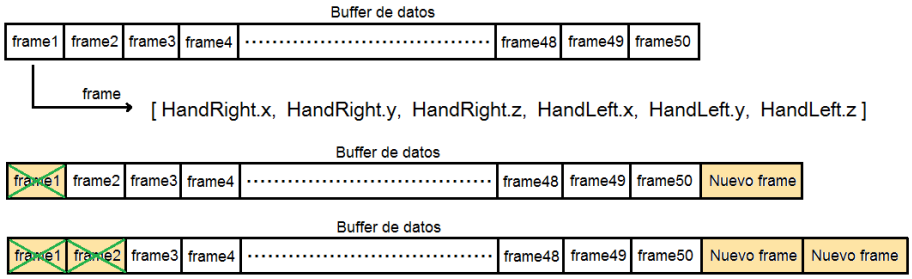


Figura 3.11 Descripción del *buffer* de datos

Antes de que un nuevo *frame* pase a formar parte del *buffer* de datos, el programa verifica si el nuevo *frame* es diferente al *frame* anterior calculando la distancia entre la posición anterior y la posición actual de las manos del usuario para saber si se ha movido, esta distancia debe ser mayor o igual a una constante que por defecto es igual a 3 [cm] la misma que puede ser modificada en la página de configuración en la variable “sensibilidad de lectura de datos”.

La distancia entre la posición anterior y la actual del usuario se calcula de forma individual para cada extremidad, por ejemplo, se extraen los datos de la mano derecha del *frame* anterior y del nuevo *frame* y luego se aplica la fórmula de la distancia entre dos puntos en el espacio, de igual manera para la mano izquierda como se aprecia en la figura 3.12.

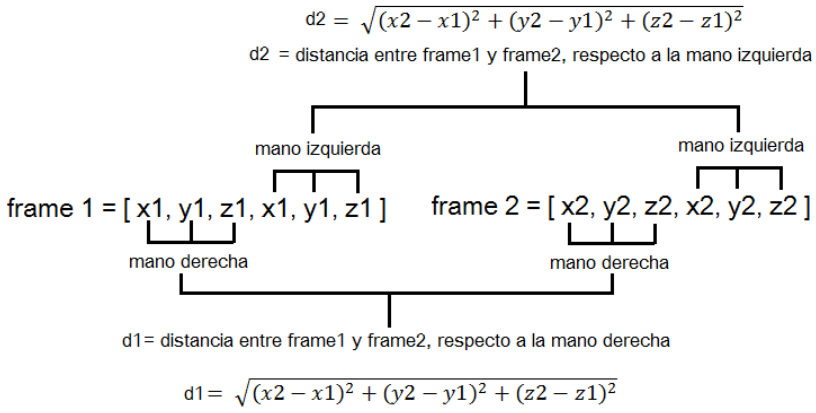


Figura 3.12 Ejemplo de cálculo de la distancia entre dos *frames*

En conclusión cada vez que el usuario mueve sus manos una cierta distancia entonces el *buffer* de datos acepta un nuevo *frame* con el objetivo de descartar datos repetidos cuando el usuario permanece inmóvil en la escena.

Dentro del evento “ReconocePatrones” el *buffer* de datos puede ser empleado para grabar o reconocer patrones. Si el usuario se encuentra grabando un patrón el *buffer* de datos es almacenado en la lista de patrones y si el programa está en modo de reconocimiento el *buffer* de datos es analizado por el algoritmo del DTW como se muestra en la figura 3.13.

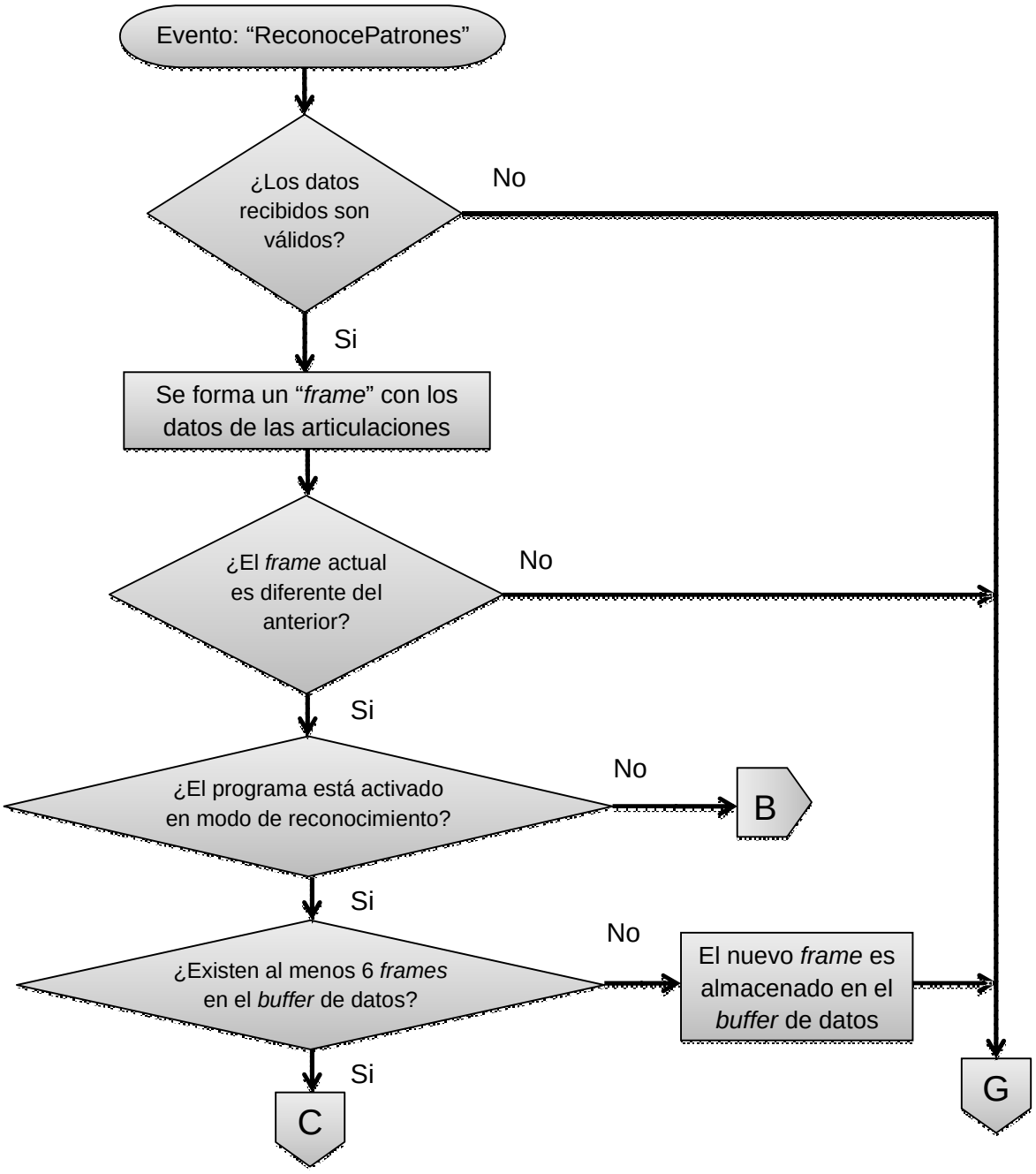


Figura 3.13 Diagrama de flujo del evento “ReconocePatrones” parte 1

Para que la subrutina del algoritmo del DTW sea ejecutada es necesario que el *buffer* de datos al menos tenga 6 elementos ya que no tendría sentido aplicar el algoritmo del DTW en un *buffer* de datos vacío.

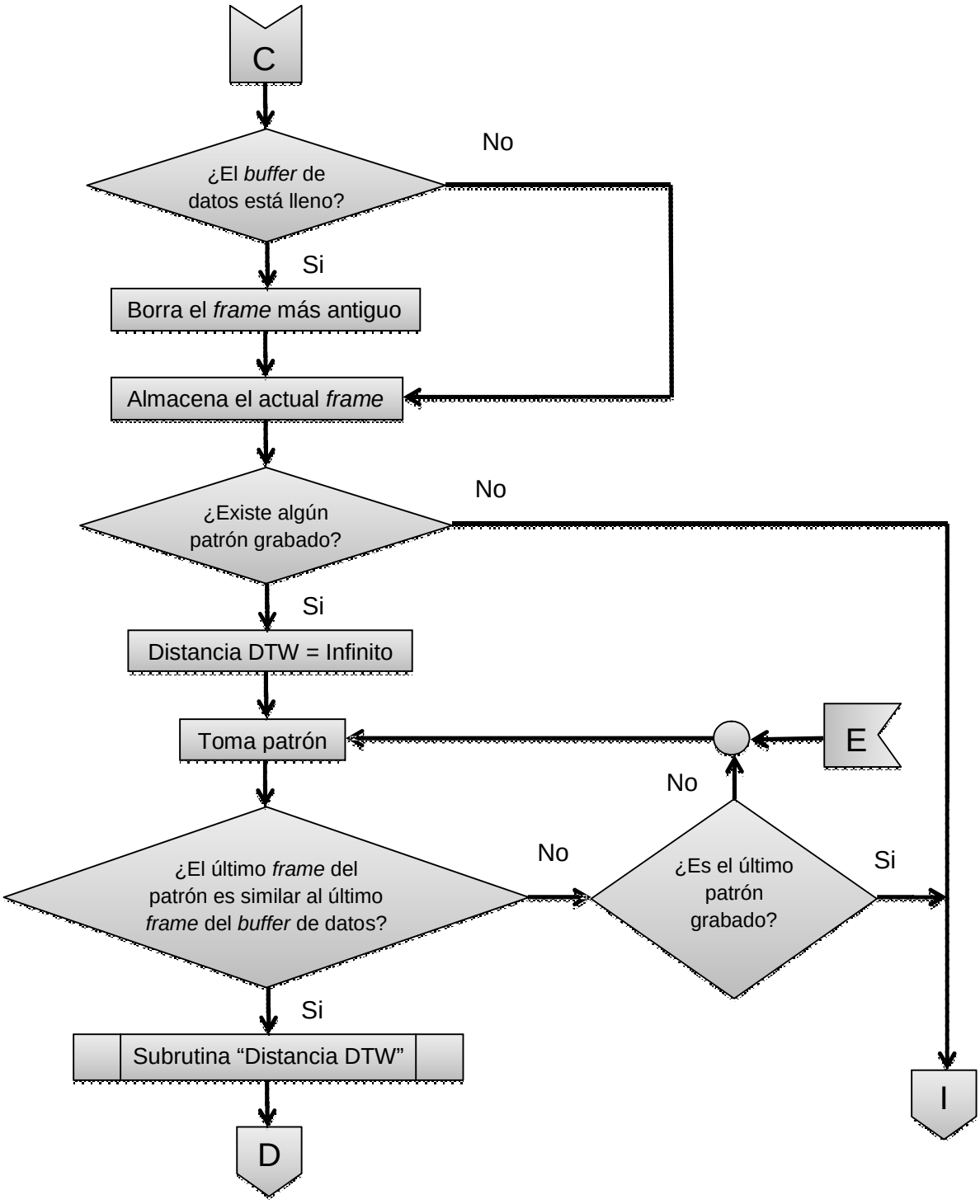


Figura 3.14 Diagrama de flujo del evento “ReconocePatrones” parte 2

El *buffer* de datos puede almacenar máximo 50 *frames* y cada vez que un nuevo *frame* es leído se borra el *frame* más antiguo, es decir, el primer *frame* del *buffer* de datos, luego se recorren todos los elementos para que se almacene el nuevo *frame* en la posición vacía que es la que contiene el *frame* más reciente y de esta manera el algoritmo del DTW puede detectar si algún patrón está presente dentro del *buffer* de datos como se aprecia en la figura 3.14.

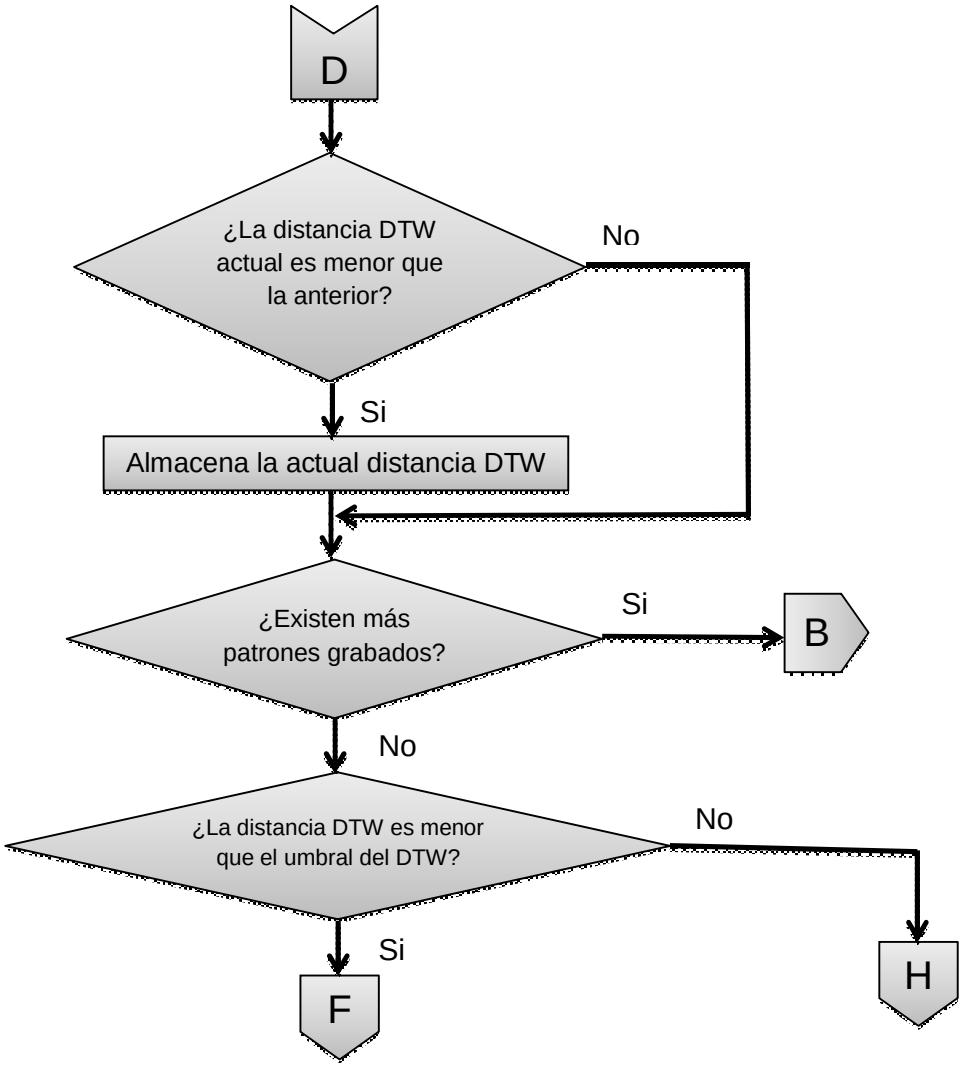


Figura 3.15 Diagrama de flujo del evento “ReconocePatrones” parte 3

Cada vez que el *buffer* de datos almacena un nuevo *frame* la subrutina del algoritmo del DTW se ejecuta tantas veces como patrones grabados existan, es decir, si existen 5 patrones grabados, el algoritmo del DTW se aplica 5 veces entre el *buffer* de datos y cada uno de los 5 patrones grabados por lo que se

obtiene como resultado 5 distancias DTW de las cuales se escoge la menor y si está por debajo del umbral del DTW el cual es obtenido en la fase de pruebas entonces el patrón con aquella distancia es reconocido por el programa. De igual manera este proceso se repite cada vez que un nuevo *frame* ingresa al *buffer* de datos lo cual significa que el programa está continuamente procesando información mediante el algoritmo del DTW como se muestra en la figura 3.15.

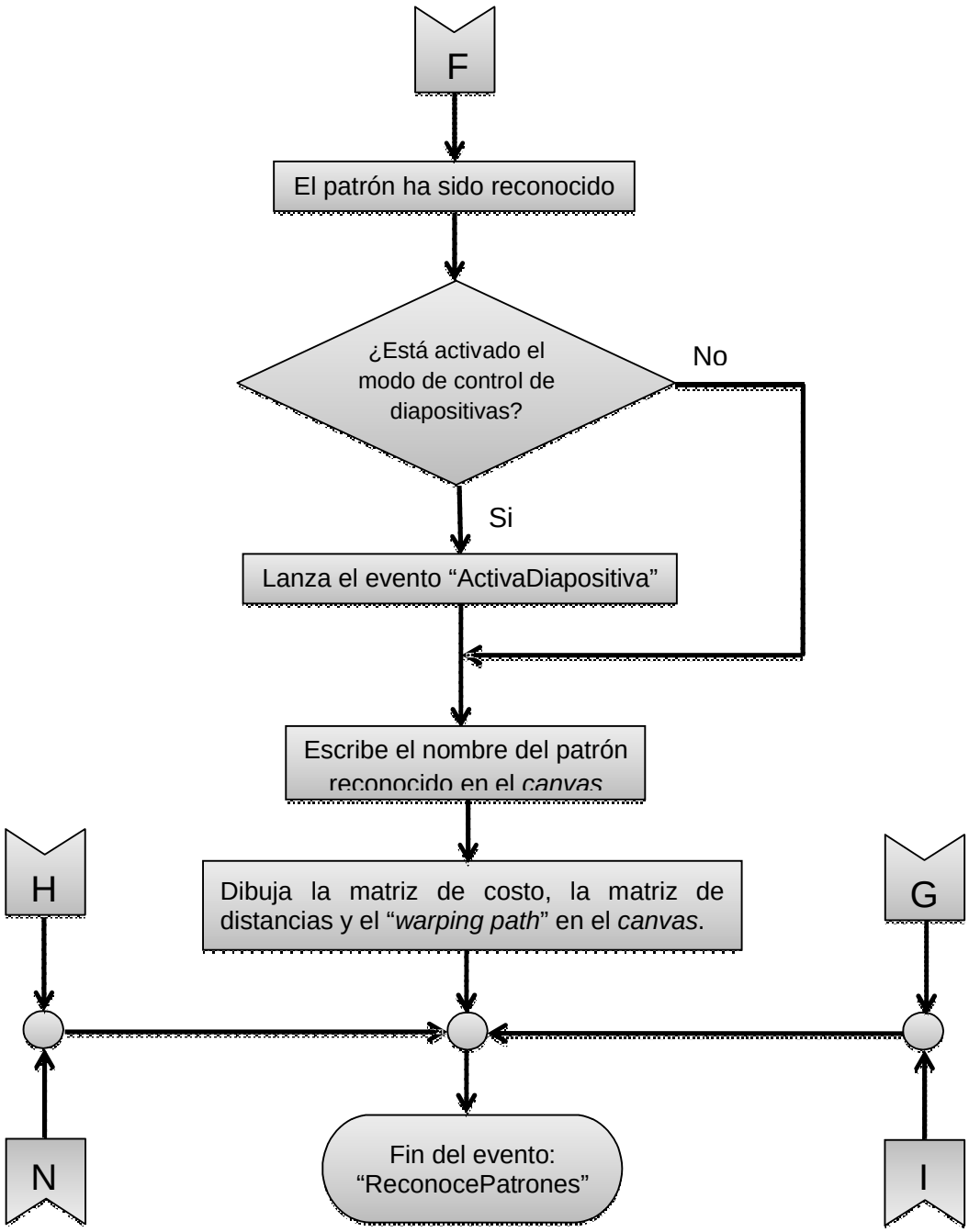


Figura 3.16 Diagrama de flujo del evento “ReconocePatrones” parte 4

Si el patrón es reconocido y está activado el control de diapositivas en el HMI se lanza el evento “ActivaDiapositiva”. También se escribe el nombre del patrón reconocido, la matriz de costo, la matriz de distancias y el “*warping path*” en el *canvas*. De este modo finaliza el evento “Reconoce patrones” como se muestra en la figura 3.16.

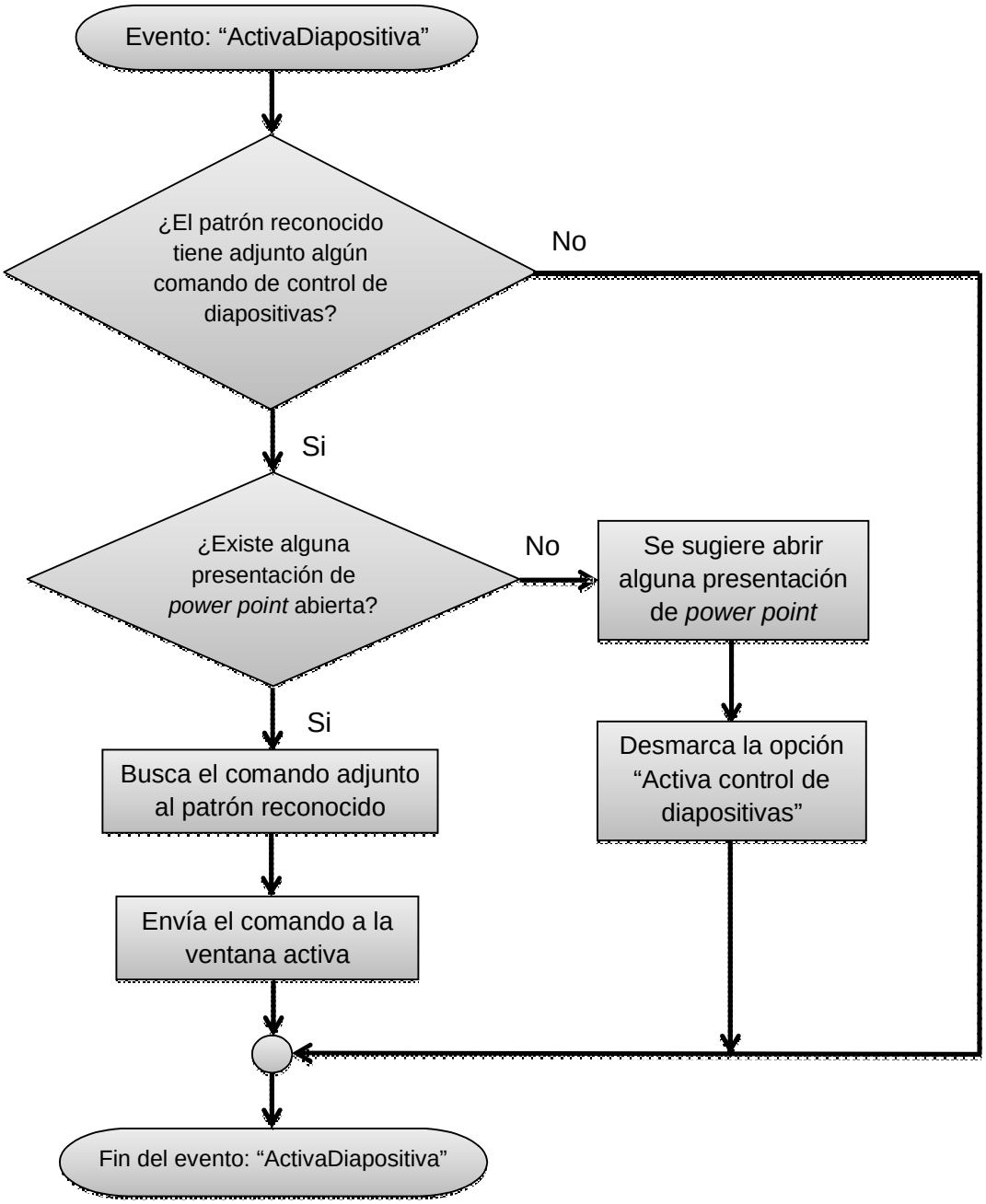


Figura 3.17 Diagrama de flujo del evento “ActivaDiapositiva”

Cuando el evento “ActivaDiapositiva” es lanzado se verifica si el patrón reconocido tiene algún comando de control de diapositivas adjunto el cual es asignado al momento de grabar patrones, si existe algún comando se procede a verificar si está abierta alguna ventana de *power point* para enviarle el comando correspondiente caso contrario se sugiere al usuario abrir alguna presentación como se muestra en la figura 3.17.

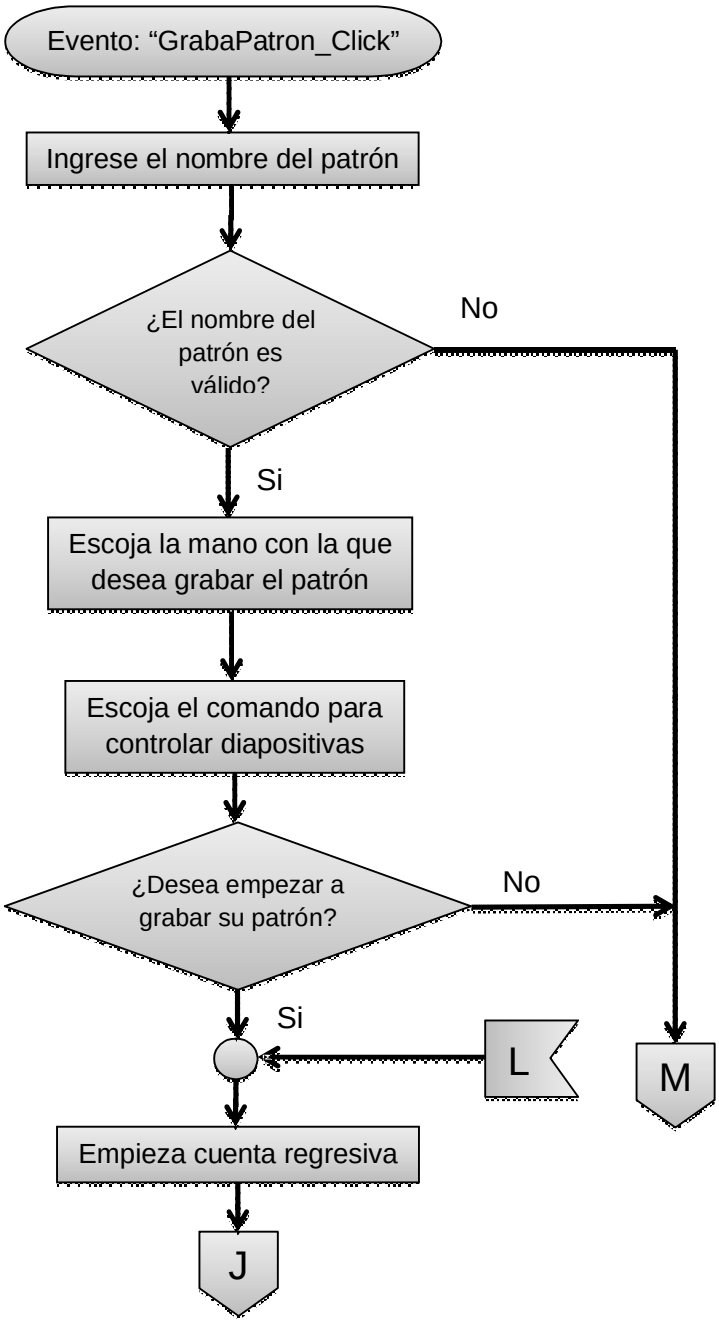


Figura 3.18 Diagrama de flujo del evento “GrabaPatron_Click” parte 1

Cuando el usuario da un *click* en el botón “Graba Patrón” se lanza el evento “GrabaPatron_Click” el cual permite grabar patrones como se aprecia en la figura 3.18. El programa permite al usuario grabar hasta 10 patrones de movimiento.

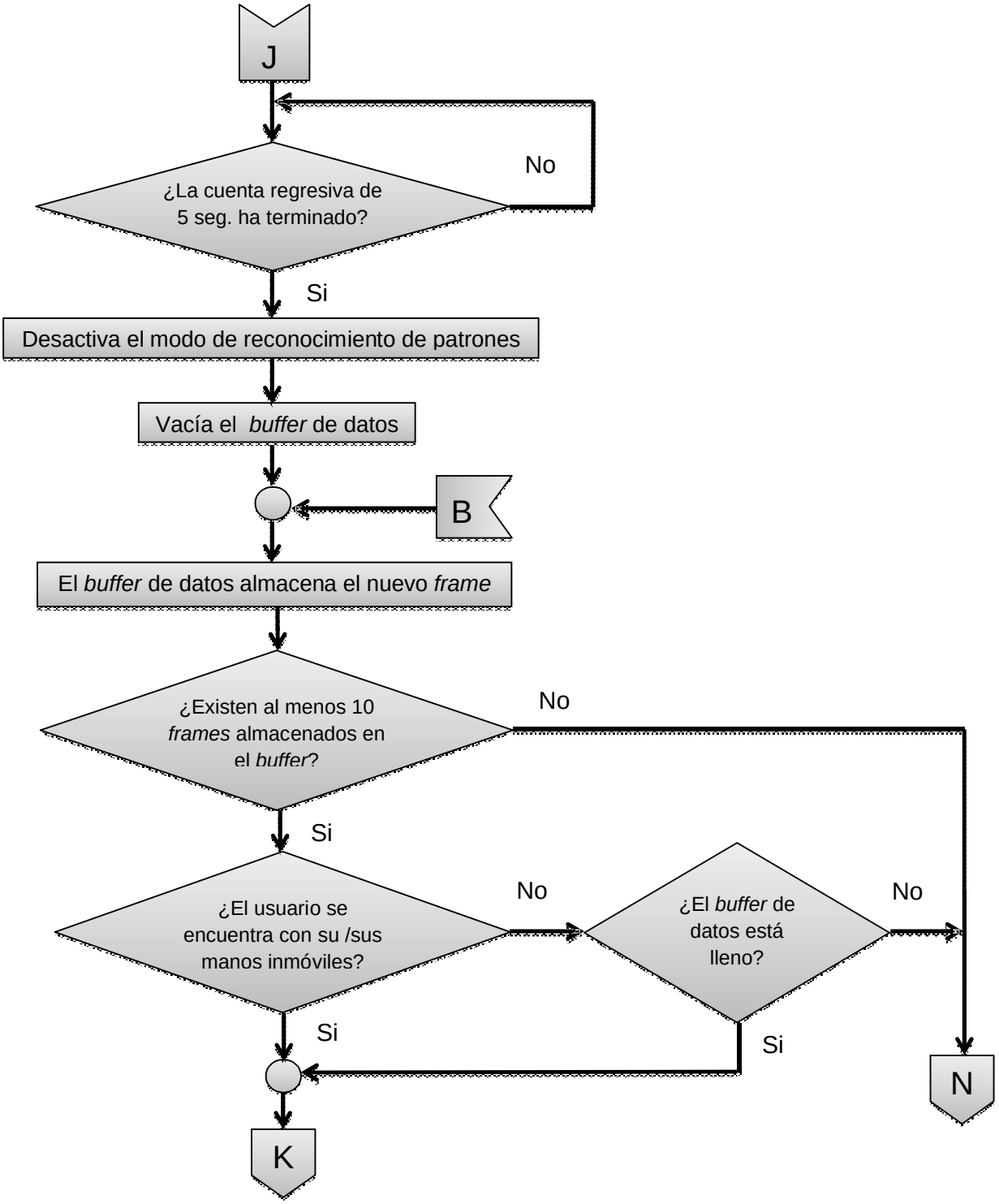


Figura 3.19 Diagrama de flujo del evento “GrabaPatron_Click” parte 2

Una vez que el usuario ha escogido un nombre, un comando y la extremidad con la que grabará el patrón entonces empieza la cuenta regresiva de 5 segundos tiempo en el cual el usuario debe ubicarse en la posición inicial del patrón que desea grabar. Cuando la cuenta regresiva termina el *buffer* de datos es vaciado para que el usuario empiece a grabar sus movimientos desde cero. El usuario puede grabar su patrón hasta llenar el *buffer* de datos en su totalidad o sólo parte de él, el programa automáticamente deja de grabar cuando el *buffer* se llena totalmente, pero cuando se desea grabar patrones más cortos el usuario debe permanecer inmóvil en la última posición del patrón por 3 segundos para que el programa deje de grabar y de este modo el *buffer* se llena sólo parcialmente, el llenado del *buffer* se puede visualizar gráficamente mediante una barra de progreso en el HMI. El *buffer* puede almacenar máximo 50 posiciones, es decir, 1.5[m] si la variable llamada “sensibilidad de lectura de datos” tiene un valor de 3[cm], esta variable sirve para leer una nueva posición sólo cada n centímetros y se explica con más detalle en el siguiente capítulo, para impedir que se graben patrones muy cortos el *buffer* tiene un límite mínimo de 10 posiciones, es decir, 30[cm] si la variable “sensibilidad de lectura de datos” es de 3[cm], todo esto se puede apreciar en la figura 3.19.

Cuando el usuario termina de grabar el patrón el programa le pregunta si desea grabarlo de nuevo o le advierte si el patrón contiene datos con lecturas de mala calidad para que proceda a grabarlo de nuevo. Un dato se considera de mala calidad o inferido cuando el kinect no puede leer una articulación directamente sino que la ubica en base a las articulaciones adyacentes y cuando el *buffer* contiene muchos datos inferidos el patrón se ha grabado incorrectamente.

Si el usuario desea volver a grabar el patrón el proceso de grabado se repite otra vez, es decir, se vacía el *buffer* de datos y empieza la cuenta regresiva. Una vez que el usuario está conforme con el patrón grabado el *buffer* de datos se guarda en la lista de patrones grabados y finaliza el evento “GrabaPatron_Click” como se aprecia en la figura 3.20.

El evento “GrabaPatron_Click” hace uso del evento “ReconocePatrones” en el que el *buffer* de datos almacena los datos provenientes del sensor kinect, pero en este caso el *buffer* es usado para grabar patrones.

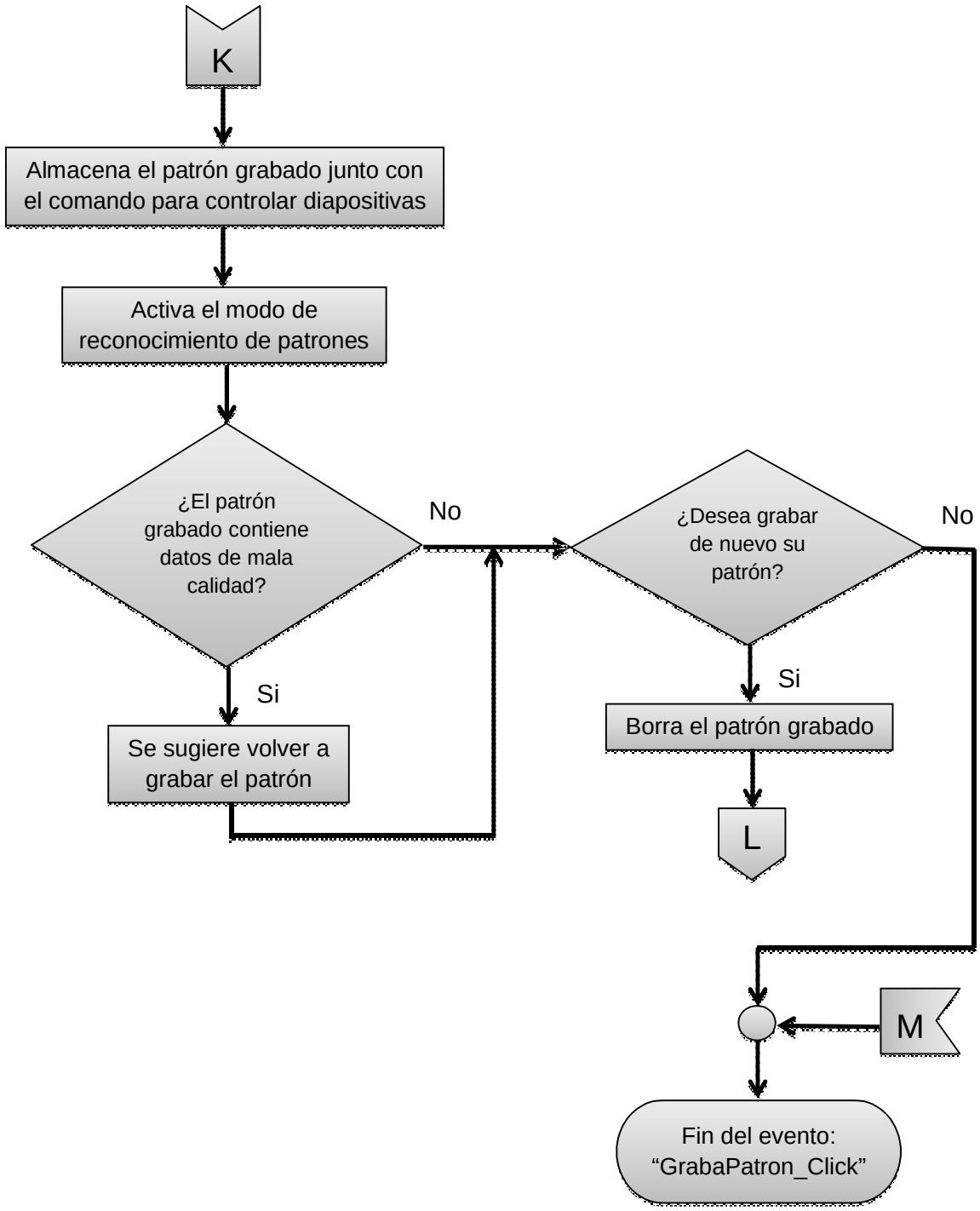


Figura 3.20 Diagrama de flujo del evento “GrabaPatron_Click” parte 3

Dentro del evento “ReconocePatrones” se encuentra la subrutina “Distancia DTW” en la que se aplica el algoritmo del DTW como se muestra en la figura 3.21.

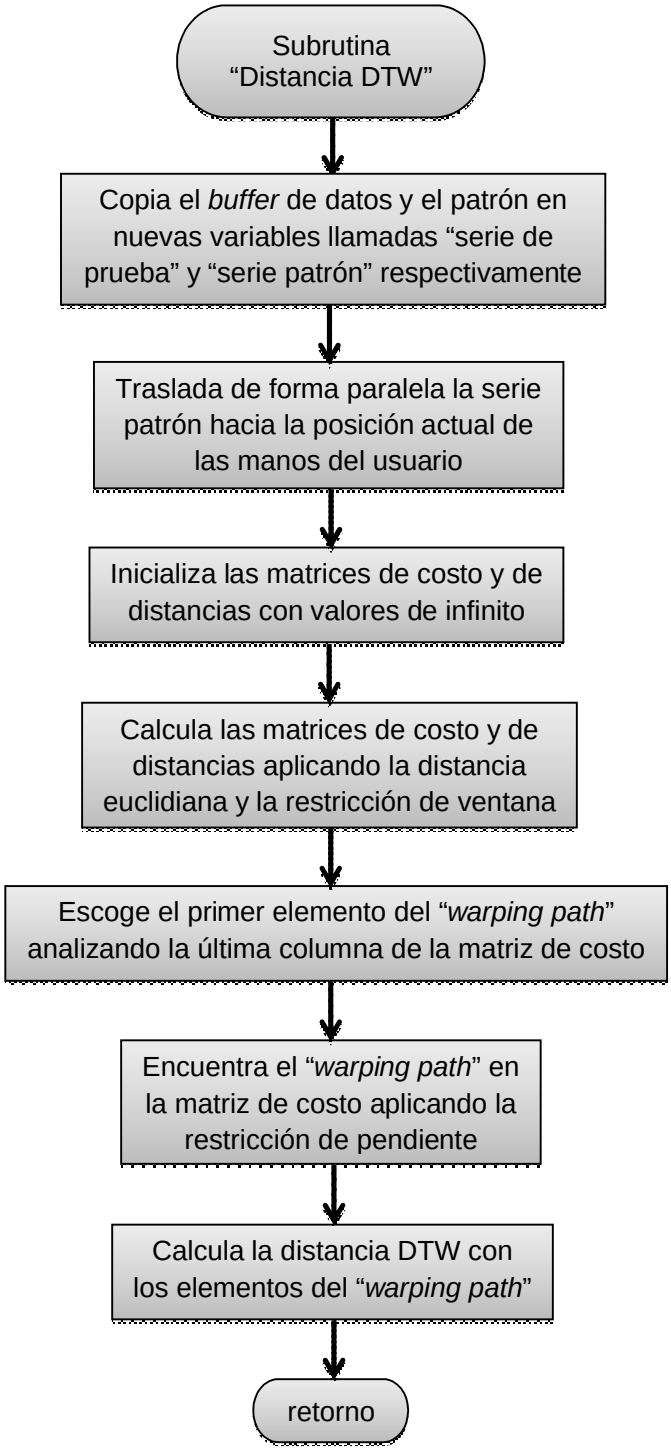


Figura 3.21 Diagrama de flujo de la subrutina “Distancia DTW”

Cada vez que se ejecuta la subrutina “Distancia DTW” se aplica el algoritmo del DTW entre dos series: el *buffer* de datos y un patrón grabado. Luego, la serie correspondiente al patrón se trasladada de forma paralela hacia la posición actual

de las manos del usuario que corresponde al último elemento del *buffer* de datos como se muestra en la figura 3.21.

Una persona graba su patrón en una posición determinada, pero en la fase de reconocimiento podría repetirlo en una posición diferente y aun así el sistema debe reconocerlo ya que se trata del mismo patrón solo que es realizado en una posición diferente, por este motivo el patrón es llevado de forma paralela hacia al posición actual de las manos del usuario.

Por ejemplo, cuando una persona graba un patrón con una mano moviéndola de izquierda a derecha en sentido horizontal a cierta altura, en la fase de reconocimiento podría repetir el mismo patrón, pero a diferente altura y el programa aun así debe reconocer el movimiento.

La utilidad de trasladar de forma paralela el patrón toma mayor importancia cuando el programa debe reconocer patrones realizados por personas de diferente estatura. Ya sea que los patrones hayan sido grabados por una persona de baja estatura y en la fase de reconocimiento sean realizados por una persona más alta o viceversa el sistema debe reconocer los patrones grabados.

Una vez que se ha trasladado paralelamente el patrón se procede a aplicar el algoritmo del DTW empezando por declarar e inicializar las matrices de costo y de distancias con valores de infinito, luego se hallan los valores de cada elemento de la matriz de distancias aplicando la distancia euclidiana, al mismo tiempo se halla la matriz de costo en base a la matriz de distancias empleando programación dinámica como se aprecia en la figura 3.21. La restricción de ventana se aplica a medida que se van hallando los elementos de la matriz de costo y de distancias.

El tamaño de las matrices de costo y de distancias está en función de la longitud del patrón y del *buffer* de datos los cuales pueden almacenar hasta 50 elementos cada uno por lo que las matrices pueden alcanzar un tamaño de 50x50 cada una, es decir, se tendría que aplicar 2500 veces la distancia euclidiana para hallar cada elemento de la matriz de distancias y al mismo tiempo obtener los elementos de la matriz de costo, es decir, el tamaño de las series influye en el tiempo de procesamiento del DTW.

Al aplicar la restricción de ventana se acorta el tamaño de las matrices de costo y de distancias y por ende se agilita el procesamiento del algoritmo del DTW especialmente si las series son largas.

Por el *buffer* de datos pasan todos los movimientos que realiza un usuario y cuando está lleno borra los datos más antiguos y almacena los nuevos movimientos manteniendo su tamaño constante, es decir, 50 movimientos. En el *buffer* de datos pueden encontrarse patrones de diferente longitud. Por ejemplo, si el usuario repite un patrón cuyo tamaño es de 30 movimientos, en el *buffer* de datos aparecerá el patrón con los 30 movimientos, pero todavía existirán 20 movimientos que no tienen nada que ver con el patrón realizado y que son restos de movimientos previos realizados antes de repetir el patrón ya que el *buffer* de datos es más amplio que el patrón que se quiere reconocer. Por este motivo, no toda la matriz de costo es analizada sino únicamente la sección que corresponde al tamaño del patrón, esto se logra aprovechando ciertas características presentes en la matriz de costo como se muestra en la figura 3.22.

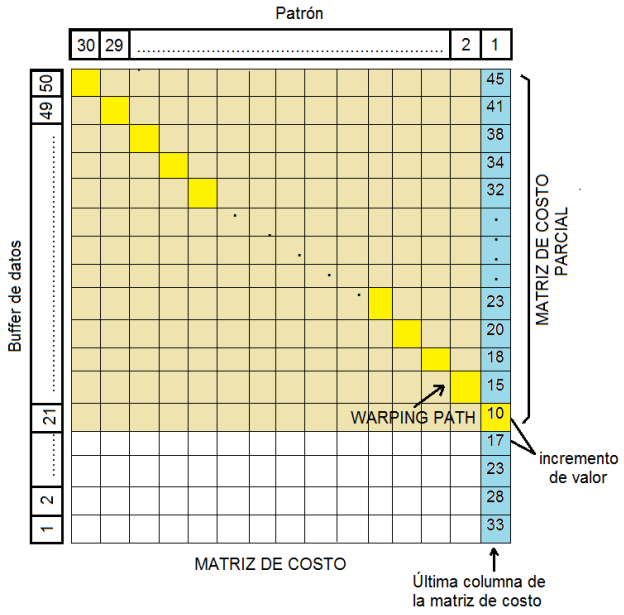


Figura 3.22 Búsqueda del "*warping path*" en la matriz de costo

En la figura 3.22 se puede apreciar que no se analiza toda la matriz de costo para hallar el "*warping path*" y la distancia DTW. Cuando se analiza los elementos de la última columna de la matriz de costo de arriba hacia abajo se nota que los valores

disminuyen hasta cierto punto y luego se incrementan. El rango en el que los valores disminuyen es en el que se aplica el “*warping path*” para verificar si en realidad existe un patrón grabado en el *buffer* de datos. Esto sucede no solamente cuando un patrón corto está presente dentro del *buffer* de datos el cual es más amplio sino también cuando el usuario repite un patrón a diferentes velocidades, por ejemplo, si el usuario ha grabado un patrón corto y si lo repite lentamente el patrón aparecerá en el *buffer* de datos con una longitud mayor y si lo repite rápidamente el patrón aparecerá en el *buffer* con una longitud menor. De esta manera, analizando la última columna de la matriz de costo se puede escoger qué parte del *buffer* de datos tiende a parecerse a un patrón grabado lo cual es verificado posteriormente mediante la aplicación del “*warping path*” y la distancia DTW.

Dado que ya se conoce qué parte del *buffer* de datos tiende a parecerse al patrón se procede a encontrar el “*warping path*” para hallar la distancia DTW. Mientras se halla el “*warping path*” se aplica la restricción de pendiente la cual garantiza que pequeñas partes de una secuencia no coincidan con largas partes de otra.

La distancia DTW se calcula sacando el promedio entre los elementos que conforman el “*warping path*” y de esta manera termina la subrutina para hallar la distancia DTW.

CAPÍTULO 4

PRUEBAS Y RESULTADOS

1.12 INTRODUCCIÓN

En el presente capítulo se muestran los procedimientos y resultados experimentales de la implementación del sistema de reconocimiento de patrones.

Se muestra los pasos realizados para realizar la calibración de parámetros que intervienen en el algoritmo del DTW para permitir que el sistema obtenga el mayor número de aciertos al momento de reconocer patrones.

Se presentan cuadros comparativos que muestran el porcentaje de efectividad del sistema para reconocer patrones frente a varios escenarios posibles y también se presentan los costos del proyecto.

1.13 CALIBRACIÓN DE PARÁMETROS DEL DTW

Los parámetros que intervienen en el algoritmo del DTW deben ser calibrados de tal manera que cada vez que un usuario repita un patrón éste sea reconocido por el programa.

Dado que el sistema permite controlar diapositivas en una presentación de *power point* los parámetros que intervienen en el reconocimiento deben permitir que usuarios tanto de diferente estatura y que repitan los patrones a diferente velocidad puedan controlar fácilmente una presentación con diapositivas.

Los parámetros calibrados son: umbral del DTW, restricción de pendiente, restricción de ventana y sensibilidad de lectura de datos.

Umbral del DTW

El umbral del DTW es la distancia mínima necesaria para que un patrón sea reconocido, es decir, si la distancia DTW entre el patrón y el movimiento realizado es menor o igual al valor del umbral del DTW entonces el patrón es reconocido por tal motivo es el parámetro más importante a ser calibrado.

Mientras más se parece un movimiento a un patrón menor es la distancia DTW por lo tanto si el valor del umbral del DTW es bajo significa que el usuario debe

ser más estricto al momento de repetir el patrón, por el contrario si el valor del umbral del DTW es alto el sistema reconocerá más fácilmente un patrón con el riesgo de que si su valor es muy alto el sistema empieza a confundir un patrón con otro o empieza a reconocer cualquier tipo de movimiento que no tiene nada que ver con los patrones grabados. El umbral del DTW puede ser calibrado en un rango de 7 a 15 [cm] en la página de configuración de parámetros del programa el cual ha sido obtenido mediante heurística en las tablas 4.1, 4.2 y 4.3.

Restricción de pendiente

La restricción de pendiente hace que el "*warping path*" no tenga muchos elementos en sentido vertical u horizontal garantizando que pequeñas partes de una secuencia no coincidan con grandes partes de otra, es decir, con este parámetro se evita que movimientos cortos sean reconocidos como patrones.

Por ejemplo, si se ha grabado un patrón largo y el sistema reconoce el patrón con tan sólo repetir una pequeña parte de él significa que el valor de este parámetro es muy alto. Por este motivo mientras más pequeño sea el valor de la restricción de pendiente el usuario deberá realizar todo el patrón para que sea reconocido. Este parámetro varía de 0 a 20 [u] en la página de configuración del programa.

Restricción de ventana

La restricción de ventana permite recortar la matriz de costo y de distancias. Si el valor de este parámetro es pequeño más serán recortadas las matrices y por ende el programa solo reconocerá los movimientos que más se parezcan al patrón. Su valor varía de 0 a 49 [u] en la página de configuración del programa.

Sensibilidad de lectura de datos.

Este parámetro no tiene que ver directamente con el algoritmo del DTW sino con la forma de leer los datos por parte del programa. Por ejemplo si su valor es de 3 [cm] esto significa que el programa lee un nuevo dato cada vez que el usuario mueva sus manos 3 [cm] con el objetivo de evitar leer datos repetidos de una misma posición. Su valor puede variar de 3 a 6 [cm] en la página de configuración del programa.

Para calibrar los parámetros del DTW se procede a repetir un patrón grabado hasta que sea reconocido el mayor número de veces sin errores.

Cada usuario puede grabar su propia lista de patrones hasta alcanzar un límite máximo de 10 patrones. Para realizar las pruebas es necesario emplear patrones *standard* por lo que se utilizará una lista de 6 patrones modelo incorporados en el programa los cuales se hallan descritos en detalle en el manual de usuario en el anexo A. Para calibrar los parámetros del DTW se escoge el patrón “Mano derecha barrido a la derecha” el cual consiste en mover la mano derecha horizontalmente de izquierda a derecha y variar los parámetros del DTW hasta conseguir la mejor calibración posible como se aprecia en las tablas 4.1, 4.2 y 4.3.

Cada vez que un patrón es reconocido aumenta el “porcentaje de aciertos” el cual puede ser realizado con “movimientos precisos y deformados”. Los “aciertos con movimientos precisos” son aquellos en los que el usuario trata de repetir los patrones tal como éstos fueron grabados y los “aciertos con movimientos deformados” son aquellos en los que el usuario repite los patrones un tanto desviados con respecto a los patrones originales. En este sentido el objetivo es lograr un alto porcentaje de “aciertos con movimientos precisos” y un bajo porcentaje de “aciertos con movimientos deformados”. Para obtener el “porcentaje de aciertos con movimientos precisos” se repite 10 veces cada patrón y de igual manera para obtener el “porcentaje de aciertos con movimientos deformados”.

Tabla 4.1 Calibración de parámetros del DTW parte 1

Velocidad a la que se realiza el patrón: **media**					
Patrón realizado: **Mano derecha barrido a la derecha**					
Paso	**Variables**			**Resultados**	
	Umbral DTW [cm]	**Restricción de pendiente [u]**	**Restricción de ventana [u]**	**% aciertos mov. precisos**	**% aciertos mov. deformados**
1	7	15	30	20	0
2	8	15	30	80	10
3	9	15	30	80	10
4	10	15	30	100	10
5	11	15	30	100	10
6	10	10	20	100	10
7	10	6	20	100	0
8	11	6	20	100	20

De la tabla 4.1 se concluye que la mejor calibración se logra en el paso 7 ya que se obtiene un porcentaje de acierto del 100% con movimientos precisos y 0% con movimientos no precisos, es decir, el programa reconoce muy bien los patrones y no los confunde con otros.

Ahora se realizará el mismo procedimiento, pero repitiendo el patrón a diferentes velocidades.

Tabla 4.2 Calibración de parámetros del DTW parte 2

Velocidad a la que se realiza el patrón: **muy rápido**					
Patrón realizado: **Mano derecha barrido a la derecha**					
Paso	**Variables**			**Resultados**	
	Umbral DTW [cm]	**Restricción de pendiente [u]**	**Restricción de ventana [u]**	**% aciertos mov. precisos**	**% aciertos mov. deformados**
1	10	6	20	80	0
2	10	10	20	80	0
3	11	6	20	80	10
4	10	15	30	80	10

En la tabla 4.2 se observa que se obtiene un 80% de efectividad en el reconocimiento cuando se repite el patrón muy rápido en todas las pruebas, aunque se puede mejorar el "porcentaje de aciertos con movimientos precisos" aumentando el valor de los parámetros también se incrementaría el "porcentaje de aciertos con movimientos deformados".

Tabla 4.3 Calibración de parámetros del DTW parte 3

Velocidad a la que se realiza el patrón: **muy lento**					
Patrón realizado: **Mano derecha barrido a la derecha**					
Paso	**Variables**			**Resultados**	
	Umbral del DTW [cm]	**Restricción de pendiente [u]**	**Restricción de ventana [u]**	**% aciertos mov. precisos**	**% aciertos mov. deformados**
1	10	6	20	70	0
2	10	15	30	70	10
3	11	6	20	70	0
4	11	15	30	70	10

En la tabla 4.3 se obtiene un 70% de efectividad en el reconocimiento cuando se repite el patrón muy lentamente debido a que el *buffer* se satura de datos similares lo que incrementa la distancia DTW.

En conclusión se obtiene que los parámetros más idóneos para el reconocimiento de patrones son: umbral del DTW = 10 [cm], restricción de pendiente = 6 [u], restricción de ventana = 20 [u]. Todas las prueban han sido realizadas con una sensibilidad de lectura de datos igual a 3 [cm].

1.14 PRUEBAS Y RESULTADOS FINALES

Para obtener las pruebas y resultados finales se utilizarán los 6 patrones modelo los cuales serán realizados por tres sujetos de prueba a diferentes velocidades.

Tabla 4.4 Pruebas y resultados 1 correspondientes al sujeto A

Sujeto: **A**	Estatura: **1.82 m**	
Velocidad a la que se realizan los patrones: **media**		
Patrón realizado	**% aciertos mov. precisos**	**% aciertos mov. deformados**
Mano derecha barrido a la derecha	100	0
Mano izquierda barrido a la izquierda	100	0
Mano derecha arriba	100	0
Mano izquierda arriba	100	0
Dos manos cierra	100	0
Dos manos arriba	90	10

Tabla 4.5 Pruebas y resultados 2 correspondientes al sujeto A

Sujeto: **A**	Estatura: **1.82 m**	
Velocidad a la que se realizan los patrones: **rápida**		
Patrón realizado	**% aciertos mov. precisos**	**% aciertos mov. deformados**
Mano derecha barrido a la derecha	90	0
Mano izquierda barrido a la izquierda	100	0
Mano derecha arriba	90	0
Mano izquierda arriba	100	0
Dos manos cierra	90	10
Dos manos arriba	90	10

Tabla 4.6 Pruebas y resultados 3 correspondientes al sujeto A

Sujeto: **A**	Estatura: **1.82 m**	
Velocidad a la que se realizan los patrones: **lento**		
Patrón realizado	**% aciertos mov. precisos**	**% aciertos mov. deformados**
Mano derecha barrido a la derecha	80	0
Mano izquierda barrido a la izquierda	100	0
Mano derecha arriba	70	0
Mano izquierda arriba	70	0
Dos manos cierra	70	0
Dos manos arriba	70	0

Los resultados de las tablas 4.4, 4.5 y 4.6 muestran que el sistema logra más aciertos cuando el usuario repite los patrones a velocidad media.

Ahora se procede a realizar las respectivas pruebas con el sujeto B.

Tabla 4.7 Pruebas y resultados 1 correspondientes al sujeto B

Sujeto: **B**	Estatura: **1.68 m**	
Velocidad a la que se realizan los patrones: **media**		
Patrón realizado	**% aciertos mov. precisos**	**% aciertos mov. deformados**
Mano derecha barrido a la derecha	80	0
Mano izquierda barrido a la izquierda	90	0
Mano derecha arriba	100	0
Mano izquierda arriba	100	0
Dos manos cierra	80	0
Dos manos arriba	90	0

Tabla 4.8 Pruebas y resultados 2 correspondientes al sujeto B

Sujeto: **B**	Estatura: **1.68 m**	
Velocidad a la que se realizan los patrones: **rápida**		
Patrón realizado	**% aciertos mov. precisos**	**% aciertos mov. deformados**
Mano derecha barrido a la derecha	80	0
Mano izquierda barrido a la izquierda	90	0
Mano derecha arriba	90	0
Mano izquierda arriba	90	0
Dos manos cierra	80	10
Dos manos arriba	90	10

Tabla 4.9 Pruebas y resultados 3 correspondientes al sujeto B

Sujeto: **B**	Estatura: **1.68 m**	
Velocidad a la que se realizan los patrones: **lento**		
Patrón realizado	**% aciertos mov. precisos**	**% aciertos mov. deformados**
Mano derecha barrido a la derecha	70	0
Mano izquierda barrido a la izquierda	70	0
Mano derecha arriba	90	0
Mano izquierda arriba	90	0
Dos manos cierra	70	10
Dos manos arriba	80	0

Ahora se procede a realizar las pruebas con el sujeto C.

Tabla 4.10 Pruebas y resultados 1 correspondientes al sujeto C

Sujeto: **C**	Estatura: **1.56 m**	
Velocidad a la que se realizan los patrones: **media**		
Patrón realizado	**% aciertos mov. precisos**	**% aciertos mov. deformados**
Mano derecha barrido a la derecha	70	0
Mano izquierda barrido a la izquierda	70	0
Mano derecha arriba	100	0
Mano izquierda arriba	100	0
Dos manos cierra	70	0
Dos manos arriba	80	0

Tabla 4.11 Pruebas y resultados 2 correspondientes al sujeto C

Sujeto: **C**	Estatura: **1.56 m**	
Velocidad a la que se realizan los patrones: **rápida**		
Patrón realizado	**% aciertos mov. precisos**	**% aciertos mov. deformados**
Mano derecha barrido a la derecha	70	0
Mano izquierda barrido a la izquierda	70	0
Mano derecha arriba	90	0
Mano izquierda arriba	90	0
Dos manos cierra	70	10
Dos manos arriba	70	10

Tabla 4.12 Pruebas y resultados 3 correspondientes al sujeto C

Sujeto: **C**	Estatura: **1.56 m**	
Velocidad a la que se realizan los patrones: **lento**		
Patrón realizado	**% aciertos mov. precisos**	**% aciertos mov. deformados**
Mano derecha barrido a la derecha	60	0
Mano izquierda barrido a la izquierda	60	0
Mano derecha arriba	90	0
Mano izquierda arriba	90	0
Dos manos cierra	70	10
Dos manos arriba	80	0

En promedio se obtiene que el “porcentaje de acierto con movimientos precisos” para el sujeto A es del 89.44%, para el sujeto B es del 85% y para el sujeto C es del 77.77%.

Hay que tener en cuenta varios factores que intervienen al momento de realizar las pruebas, por ejemplo, que el kinect esté adecuadamente ubicado, que los lentes del kinect estén limpios, que no haya mucha luz solar y principalmente hay que percatarse si el kinect está reconociendo adecuadamente al usuario observando el estado del *buffer* de datos y el gráfico del esqueleto del usuario.

Analizando los resultados finales se puede notar que el sistema reconoce de mejor manera los movimientos realizados por el sujeto A debido a que los

patrones modelo fueron grabados por una persona de 1.80 m de estatura y mientras más semejante sea la persona que realiza los movimientos a la persona que grabó los patrones menos errores se cometen por tal motivo el porcentaje de reconocimiento del sujeto C es menor, aunque también hay que tener en cuenta que los usuarios pueden cometer errores al momento de repetir los patrones, al menos hasta que se familiaricen con ellos.

En general los resultados fueron satisfactorios ya que el porcentaje de aciertos obtenido en promedio es del 84,07% realizado con sujetos de diferente estatura y repitiendo los patrones a diferente velocidad.

1.15 COSTOS DEL PROYECTO

A continuación se muestra los costos del proyecto en la tabla 4.13.

Tabla 4.13 Costos del proyecto

Elemento	**Unitario $**	**Cantidad**	**Total $**
PC *Intel core* i7, 1.90 GHz x64, 8 Gb	800	1	800
Sensor "*KINECT for Windows*"	150	1	150
Costo total de elementos			**950**
Costos de Ingeniería			
Actividad	**Costo $/h**	**Horas**	**Total**
Estudio del sensor Kinect	7	160	1.120
Estudio del algoritmo del Alineamiento Temporal Dinámico	7	320	2.240
Implementación del algoritmo del DTW en lenguaje C#	7	240	1.680
Programación del *software* de reconocimiento de patrones	7	160	1.120
Costo total de Ingeniería			**6.160**
Costo total del proyecto			**7.110**

CAPÍTULO 5

CONCLUSIONES Y RECOMENDACIONES

1.16 CONCLUSIONES

Analizando los alcances y objetivos planteados en el proyecto, se determina que estos han sido cumplidos en su totalidad, concluyendo lo siguiente:

- El sensor Kinect se muestra como una magnífica herramienta de bajo costo para la identificación y seguimiento del cuerpo humano lo que permite llevar a cabo innovadores proyectos de ingeniería orientados a permitir que robots interactúen de mejor manera con los seres humanos en ambientes cotidianos.
- El DTW ofrece gran versatilidad al momento de reconocer patrones ya que permite configurar los parámetros de reconocimiento de acuerdo a las necesidades del entorno de operación. Una de las mayores ventajas que ofrece el DTW es el alto porcentaje de reconocimiento de patrones de movimiento cuando éstos son realizados a diferente velocidad.
- La velocidad de ejecución del algoritmo del DTW depende fundamentalmente de dos factores: de la longitud de los patrones y de la cantidad de patrones grabados. Cuando la longitud de los patrones es muy grande el número de cálculos necesarios para obtener la distancia DTW es mayor. Cuando la lista de patrones grabados es muy extensa el número de veces que se aplica el algoritmo del DTW se incrementa ya que es necesario calcular la distancia entre el movimiento y cada uno de los patrones de la lista.
- El porcentaje de aciertos en el reconocimiento de patrones puede verse afectado por una mala lectura de datos por parte del sensor kinect, este problema es más frecuente en personas de baja estatura.
- El 84,07% de aciertos obtenidos en el reconocimiento de patrones durante la fase de pruebas es un resultado satisfactorio tomando en cuenta que las pruebas fueron realizadas por personas de diferente estatura y los patrones fueron realizados a diferente velocidad con el objetivo de permitir

que el mayor número de personas pueda manejar diapositivas en una presentación de *Power Point*.

1.17 RECOMENDACIONES

- Antes de proceder a utilizar el programa de reconocimiento de patrones se recomienda verificar que las lentes del kinect estén limpias, que en el lugar no haya demasiada luz solar, que el kinect esté ubicado a una altura adecuada y sobre una superficie estable y sin desniveles. También es importante que el usuario cuente con el espacio suficiente para realizar sus movimientos con toda libertad y no esté ubicado demasiado cerca del kinect.
- Si el usuario no es reconocido inmediatamente cuando entra en la escena del kinect se recomienda que mueva sus extremidades hasta que sea reconocido y luego verifique que el kinect esté leyendo adecuadamente sus movimientos observando al gráfico del esqueleto en la pantalla del programa. También se recomienda que el usuario se ubique enfrente al kinect, sin girar, con el objetivo de no ocultar las articulaciones a la vista del kinect.
- Se recomienda que cuando un usuario desee grabar un patrón lo realice de manera fluida y que al final el usuario mantenga inmóviles sus extremidades para que el programa deje de grabar el patrón y evitar que el programa acumule datos repetidos de la última posición del patrón lo que causaría errores al momento del reconocimiento.
- Se recomienda que los patrones se graben a una velocidad media para que cuando el usuario los repita a mayor o menor velocidad el DTW los pueda reconocer con mayor facilidad.
- Se recomienda emplear el algoritmo del DTW cuando se requiera un alto porcentaje de acierto en el reconocimiento y cuando la cantidad de patrones grabados no sea demasiado extensa.
- Se recomienda que cuando se graben varios patrones éstos sean diferentes unos de otros ya que si un patrón pequeño es parte de otro patrón más grande, lo más probable es que sólo el patrón pequeño sea reconocido.

TRABAJOS FUTUROS

- En base al programa de reconocimiento desarrollado en el presente trabajo se podría crear una pizarra virtual en la que los patrones grabados sean letras del alfabeto las cuales sean reconocidas cuando el usuario las dibuje en el aire frente al kinect y de esta manera pueda escribir un texto en el computador.
- Utilizando la más reciente versión del sensor kinect con el que es posible detectar los dedos de las manos se abre la posibilidad de crear un sistema de reconocimiento de lenguaje de señas.
- En base al presente trabajo se podría ampliar el reconocimiento de patrones no sólo de las extremidades superiores del cuerpo humano sino de todo el cuerpo.
- Con el sensor kinect se puede reconstruir espacios en 3D por lo que se podría crear una plataforma de reconocimiento de objetos tridimensionales.
- El sensor kinect puede ser utilizado por cualquier dispositivo electrónico móvil para mejorar su desplazamiento en el espacio, por ejemplo, para sortear obstáculos.

REFERENCIAS BIBLIOGRÁFICAS

[1] Lopategui Corsino Edgar. (2012). “Patrones de Movimiento Fundamentales” [Online]. Disponible en: http://www.saludmed.com/Entrena_II/Presentaciones/Entrenamiento_Funcional_PATRONES-MOV-FUND.pdf pp. 3 - 25.

[2] Maleny MSP, “Dentro de la más reciente versión del Kinect para Windows SDK” [Online]. Disponible en: http://malenyabrego.wordpress.com/category/kinect/

[3] Kramer Jeff, Burrus Nicolas, Echtler Florian, Herrera C. Daniel, Parker Matt, “Hacking the kinect”, Apress, 2012. pp. 11 – 12.

[4] SolidQ, “Controla PowerPoint mediante gestos con Kinect” [Online]. Disponible en: http://blogs.solidq.com/cuevanet/post.aspx?id=40

[5] ANIMUS, “Una interfaz diferente…” [Online]. Disponible en: http://animusproject.wix.com/web/apps/blog/una-interfaz-diferente-1

[6] Miles Rob, “Start Here! Learn the kinect API”, O’Reilly Media, Inc., 1005 Gravenstein Highway North Sebastopol California 95472, 2012. pp. 47 – 83 – 85 – 124 – 125.

[7] Abhijit’s World of .NET, “Development With Kinect .NET SDK (Part II) – Using NUI APIs with Camera” [Online]. Disponible en: http://abhijitjana.net/2011/09/17/development-with-kinect-net-sdk-part-ii-using-nui-apis-with-camera/

[8] Microsoft Developer Network, “Skeletal Tracking” [Online]. Disponible en: http://msdn.microsoft.com/en-us/library/hh973074.aspx

[9] Webb Jarrett, Ashley James, “Beginning Kinect Programming with the Microsoft Kinect SDK”, Apress, 2012. pp. 97.

[10] PrimeSense, “3D Sensors” [Online]. Disponible en: http://www.primesense.com/solutions/3d-sensor/

[11] Wikipedia, “Reconocimiento de Patrones” [Online]. Disponible en: http://es.wikipedia.org/wiki/Reconocimiento_de_patrones

[12] Furtuna Titus Felix, Academy of Economic Studies, “Dynamic Programming Algorithms in Speech Recognition”, Bucharest, 2008, [Online]. Disponible en: http://revistaie.ase.ro/content/46/s%20-%20furtuna.pdf pp. 1.

[13] Niels Ralph, Department of Artificial Intelligence, Radboud University Nijmegen, “Dynamic Time Warping, An intuitive way of handwriting recognition?”, 2004, [Online]. Disponible en: http://www.ralphniels.nl/pubs/niels-dtw.pdf

[14] Rath Toni M. y Manmatha R, Multi-Media Indexing and Retrieval Group Center for Intelligent Information Retrieval University of Massachusetts Amherst, MA USA, “Word Image Matching Using Dynamic Time Warping”, 2003, [Online]. Disponible en: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=1227662&queryText%3Drath+toni+dtw

[15] Tsiporkova Elena, GenTXWarper, “Dynamic Time Warping Algorithm for Gene Expression Time Series” [Online]. Disponible en: http://www.psb.ugent.be/cbd/papers/gentxwarper/DTWalgorithm.htm Descarga disponible en: http://www.psb.ugent.be/cbd/papers/gentxwarper/DTWAlgorithm.ppt

[16] Ramos Andrés, Universidad Pontificia COMILLAS MADRID, “Programación Dinámica DP” [Online]. Disponible en: http://www.iit.upcomillas.es/aramos/simio/transpa/t_dp_ar.pdf pp. 5

[17] Computational Biology, “GenTxWarper, DTW algorithm” [Online]. Disponible en: http://www.psb.ugent.be/cbd/papers/gentxwarper/DTWalgorithm.htm

[18] Wikipedia, “Dynamic time warping” [Online]. Disponible en: http://en.wikipedia.org/wiki/Dynamic_time_warping

ANEXO A

MANUAL DE USUARIO

1) Requisitos de funcionamiento

a. La computadora que va a ser utilizada debe cumplir con los siguientes requerimientos:

Sistema operativo

- *Microsoft Windows* 7 o *Microsoft Windows* 8

Requerimientos de hardware

- Computadora de 32 bits (x86) o 64 bits (x64) con un procesador de doble núcleo de 2.66 GHz.
- Mínimo 2 GB de RAM
- Puerto USB 2.0 o más, dedicado.
- Sensor "*KINECT for Windows*".

Requerimientos de software

- *Microsoft Visual Studio*® 2010 *Ultimate* u otra edición de *Visual Studio* 2010.
- *Microsoft* .NET *Framework* 4 (instalado con *Visual Studio* 2010)

b. Para que el *software* de reconocimiento de patrones pueda funcionar primero se debe instalar el *driver* del sensor kinect almacenado en el CD del *software* con el nombre "KinectSDK-v1.5-Setup.exe" o puede descargarse desde la dirección http://kinectforwindows.org asegurándose de escoger la versión 1.5. Este *driver* permite que el sensor sea reconocido por el computador y pueda comunicarse con el programa de reconocimiento de patrones. Si el *driver* ha sido instalado correctamente el *led* de *status* del sensor se mostrará de color verde. Cabe señalar que este *driver* se instala si el sistema operativo es *Windows* 7 o superior.

c. Para instalar el programa de reconocimiento de patrones se debe abrir la carpeta "Patrones Movimiento EPN" ubicado en el CD del *software* en ella

se debe abrir la carpeta “Debug” y se encontrará el instalador “setup.exe” el cual debe ser ejecutado y se debe seguir las instrucciones de instalación.

d. Si se desea ver el código fuente del programa se debe abrir la carpeta “Código fuente” presente en el CD del *software*, ahí se encontrará la carpeta llamada “EPN_Patrones_Kinect 1.0” esta carpeta es la que contiene todos los archivos fuente del programa de reconocimiento de patrones. Si se requiere ver el código de programación del *software* de reconocimiento se recomienda copiar esta carpeta en el computador. Cabe mencionar que para que éstos archivos se ejecuten se debe tener instalado cualquier versión de *Visual Studio* 2010. Para que el código se muestre se debe ejecutar el archivo llamado “EPN_Patrones_Kinect.sln” ubicado dentro de la carpeta copiada “EPN_Patrones_Kinect 1.0” de esta manera se abrirá *Visual Studio* mostrando el código fuente.

Si al momento de compilar los archivos éstos no se ejecutan satisfactoriamente puede ser que sea necesario agregar manualmente las librerías del kinect en las referencias del programa.

En la figura A.1 se muestra los pasos para agregar las librerías, primero se hace *click* derecho en la pestaña de “referencias” ubicada en el explorador de soluciones y luego se escoge “Agregar referencia”, luego aparece un cuadro de diálogo en el que debe buscar el archivo “Microsoft.Kinect.dll” en la carpeta donde se instaló el *driver* del kinect, si el archivo ha sido correctamente agregado aparecerá dentro de la pestaña de referencias del programa.

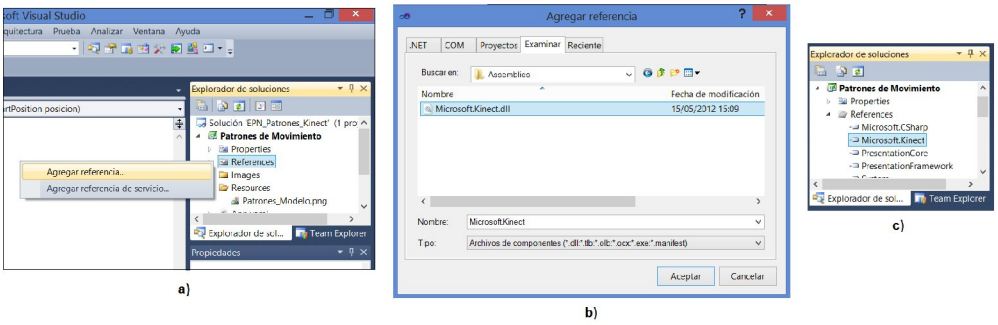


Figura A.1 Agregar librerías del kinect a las referencias del programa

2) Utilización del sensor kinect

a. Para utilizar el sensor kinect es necesario conectarlo a una fuente de energía de 110 Vac y al puerto USB del computador.

Para que el sensor kinect pueda capturar la escena de mejor manera es recomendable tomar en cuenta los siguientes detalles:

- Verificar que las lentes del kinect estén limpias.
- Verificar que al sensor no le llegue demasiada luz solar.
- Asegurarse que el kinect esté ubicado a una altura adecuada sobre una superficie estable y sin desniveles.
- Procurar que el usuario cuente con el espacio suficiente para realizar sus movimientos con toda libertad.
- No ubicarse demasiado cerca del kinect.
- Cuando un usuario entra en la escena y no es reconocido por el kinect debe mover sus extremidades hasta que aparezca el dibujo de su esqueleto en pantalla.
- El usuario debe procurar no usar atuendos o artículos que cambien drásticamente la forma del cuerpo humano.
- Evitar compartir el puerto USB que utiliza el kinect con otros dispositivos.

3) Uso del programa de reconocimiento de patrones de movimiento

Cuando el programa “EPN_Patrones_Kinect” se ejecuta correctamente aparece la ventana principal del programa mostrada en la figura A.2.

Figura A.2 Pantalla principal del programa

Cuando el programa no detecta ningún sensor kinect aparece una ventana de diálogo advirtiendo al usuario que no se ha detectado ningún sensor y luego el programa se cierra. En este caso es recomendable verificar si el sensor ha sido correctamente detectado por el puerto USB del computador y que el *led* de *status* del kinect esté de color verde.

Desde la pantalla principal del programa se puede acceder a todas la herramientas que el usuario dispone para el reconocimiento de patrones las cuales se muestran en la figura A.3.

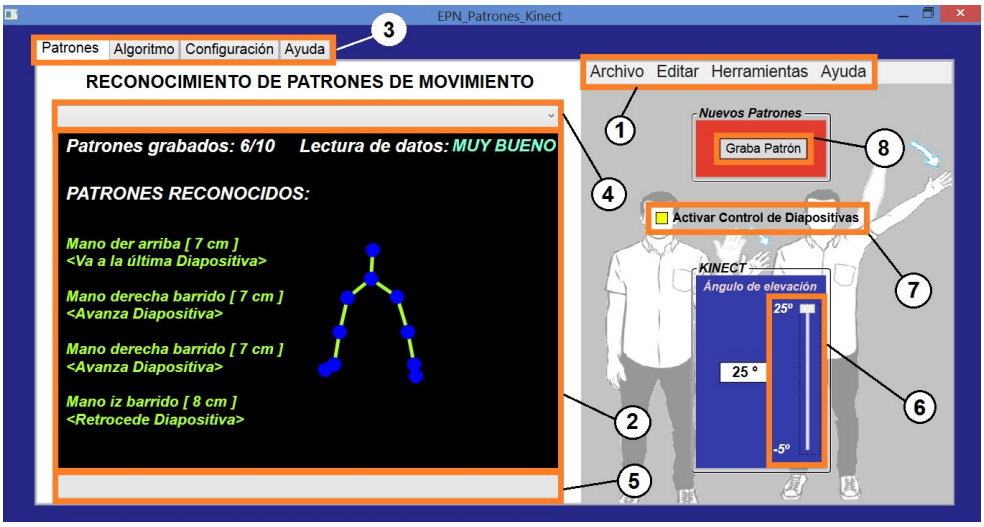


Figura A.3 Herramientas del programa.

A continuación se describe detalladamente cada una de las herramientas del programa que aparecen en la figura A.3.

1. Barra de Menú.

La “barra de menú” contiene los siguientes elementos: Archivo, Editar, Herramientas y Ayuda.

En la opción de “Archivo” se puede elegir entre: Nuevo, Abrir, Guardar y Salir como se aprecia en la figura A.4 a.

La opción “Nuevo” permite grabar patrones desde cero, es decir, mediante esta opción se borran todos los patrones existentes. Antes de borrar los patrones

aparece un cuadro de diálogo preguntando al usuario si desea guardar sus patrones en caso que no lo haya hecho.

La opción “Abrir” permite recuperar patrones previamente guardados en un *file* tipo txt. Cuando se elige esta opción aparece un cuadro de diálogo mediante el cual se puede buscar y escoger el archivo con los patrones grabados. Si los patrones se han cargado correctamente aparece un cuadro de diálogo con el número de patrones encontrados y si el archivo no contiene patrones aparece un cuadro de diálogo señalando que no se ha encontrado ningún patrón grabado.

La opción “Guardar” permite guardar los patrones en un *file* tipo txt. Cuando se escoge esta opción se abre un cuadro de diálogo que permite dar un nombre al archivo y escoger dónde guardarlo.

La opción “Salir” permite terminar con la ejecución del programa. Antes de abandonar el programa se pregunta al usuario si desea guardar los patrones en caso que no lo haya hecho. Otra forma de salir del programa es presionando sobre el botón “cerrar” que aparece en la parte superior derecha del programa adyacente a los botones de minimizar y maximizar.

Todas estas opciones se despliegan al hacer *click* en la opción “Archivo” como se visualiza en la figura A.4 a.

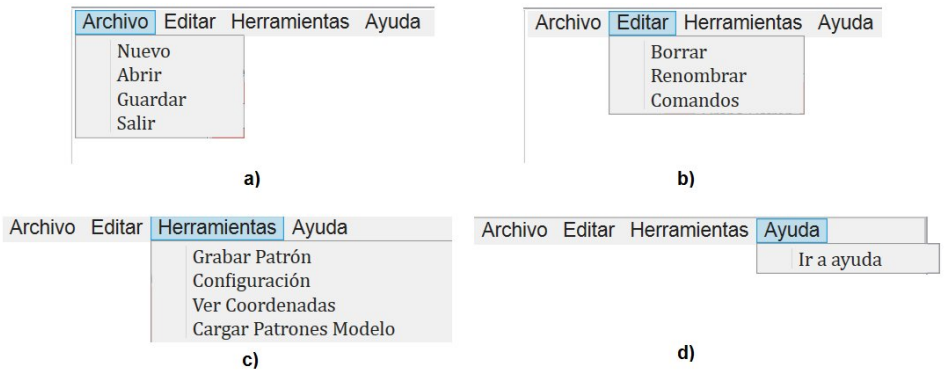


Figura A.4 Elementos desplegados en la barra de menú del programa

En la opción “Editar” se puede elegir las siguientes opciones: Borrar, Renombrar y Comandos como se puede apreciar en la figura A.4 b.

La opción “Borrar” sirve para escoger un patrón de la lista de patrones y así proceder a borrarlo.

La opción “Renombrar” permite cambiar el nombre del patrón. Cuando se elige esta opción aparece una ventana con todos los patrones, se escoge uno y se procede a escribir el nuevo nombre que se quiere dar al patrón.

La opción “Comandos” permite modificar los comandos que están asociados a cada movimiento. Cada vez que se graba un nuevo patrón se puede asociar uno de los siguientes comandos para controlar diapositivas: avanza diapositiva, retrocede diapositiva, va a la última diapositiva, va a la primera diapositiva, finaliza la presentación, inicia la presentación y ningún comando. Cuando se escoge esta opción aparece una ventana con la lista de patrones y sus comandos actuales, se selecciona el nombre de un patrón y se procede a elegir el nuevo comando desde la lista de comandos y se aplican los cambios como se aprecia en la figura A.5 de esta manera el usuario puede editar los comandos asociados a cada movimiento.

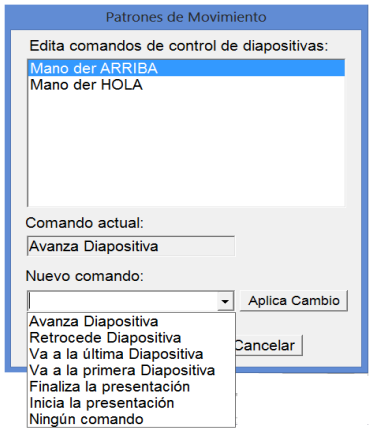


Figura A.5 Ventana de edición de comandos de control de diapositivas

En la opción “Herramientas” se tiene las siguientes opciones: Grabar patrón, Configuración, Ver coordenadas y Cargar patrones modelo, como se aprecia en la figura A.4 c.

La opción “Grabar Patrón” permite grabar un nuevo patrón. Al hacer *click* en esta opción aparece una ventana con un cuadro de texto que permite ingresar un nombre al patrón como se muestra en la figura A.6, el programa no permite que

se ingrese nombres de patrones que ya existen o que el nombre de un patrón quede en blanco y la longitud máxima permisible para el nombre de un patrón no debe superar los 22 caracteres. Luego aparece una ventana que permite escoger si el patrón se lo realizará con los dos brazos, con el brazo derecho o con el brazo izquierdo. Luego aparece una ventana que permite asociar un comando para controlar diapositivas con el patrón. Por último aparece un cuadro de diálogo con las instrucciones para grabar un patrón y al final se pregunta al usuario si desea empezar a grabar el patrón, si la respuesta es “si” se empieza a realizar la cuenta regresiva de 5 segundos la misma que se puede visualizar en el *canvas* y que sirve para esperar hasta que el usuario se ubique en la posición inicial frente al kinect, este proceso se puede visualizar en la figura A.6.

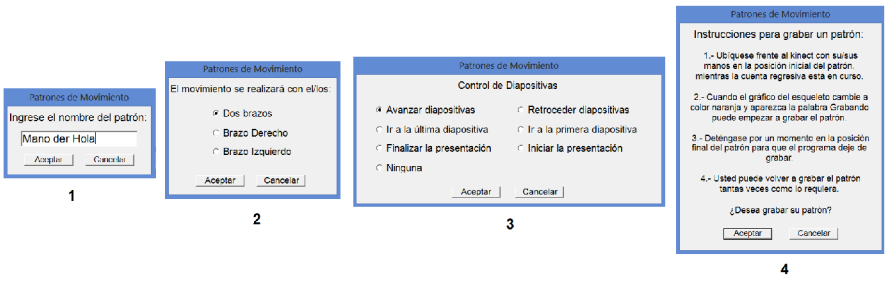


Figura A.6 Proceso para grabar un patrón

Luego del conteo aparece la palabra “grabando” y el color de las articulaciones del esqueleto se vuelven de color naranja lo que significa que el usuario puede empezar a grabar el patrón. El usuario puede ver cómo se llenan los datos en el *buffer* mediante el *ProgressBar* el cual se va llenando de color verde a medida que el usuario graba el patrón como se muestra en la figura A.7.

Figura A.7 Usuario grabando un patrón

El usuario puede grabar patrones de diferente longitud, cuando el usuario desea grabar un patrón corto debe empezar a moverse para empezar a grabar el patrón y debe detenerse por un momento para que el programa sepa que el patrón ha finalizado y deje de grabar, cuando el patrón es extenso el programa simplemente graba hasta que se llena el *buffer* de datos el cual puede almacenar de 10 a 50 posiciones, es decir, si la variable llamada “sensibilidad de lectura de datos” en la página de configuración está en 3[cm] entonces el usuario podrá grabar patrones de 30[cm] a 1.5[m] de longitud. Una vez grabado el patrón aparece una ventana preguntándole al usuario si desea grabar de nuevo el patrón o finalizar con el proceso de grabado, es decir, el usuario puede grabar un patrón una y otra vez hasta que esté satisfecho. Cada vez que se lee un dato de mala calidad las articulaciones del gráfico del esqueleto cambian a color turquesa y después de grabar un patrón en el que el *buffer* contiene muchos datos de mala calidad aparece una ventana advirtiendo que el patrón grabado podría estar corrompido y le sugiere que se vuelva a grabar el patrón. Un dato se considera de mala calidad cuando es inferido, es decir, cuando el kinect no puede detectar una articulación directamente y la ubica calculando su posición en base a articulaciones adyacentes.

Para que un patrón sea grabado correctamente el usuario debe procurar realizarlo de manera uniforme, es decir, grabarlo a un solo ritmo para evitar la acumulación de datos en determinadas partes del patrón lo que causa que ciertos segmentos del patrón tengan más peso que otros lo que provoca que el algoritmo del DTW trate de encontrar alguna similitud entre el patrón y cualquier movimiento que pase por algún punto de mayor peso. Por esta razón es importante tener cuidado al momento de iniciar y terminar de grabar un patrón ya que si las extremidades no están del todo inmóviles podrían acumularse datos en la posición inicial y final del patrón.

La opción de “Configuración” permite abrir una ventana en la que se puede modificar parámetros del algoritmo del DTW y del programa.

La opción “Ver Coordenadas” permite visualizar las coordenadas de las manos del usuario en el esqueleto dibujado en el *canvas*.

La opción “Cargar patrones modelo” permite cargar 6 patrones modelo, es decir, patrones previamente grabados que se utilizan a manera de demostrativo. Los patrones modelo grabados son: brazo derecho barrido a la derecha, brazo izquierdo barrido a la izquierda, brazo derecho arriba, brazo izquierdo arriba, dos brazos abre y dos brazos arriba como se aprecia en la figura A.8.

Figura A.8 Patrones Modelo

El patrón “Brazo derecho barrido a la derecha” permite mover diapositivas a la derecha, el patrón “Brazo izquierdo barrido a la izquierda” permite mover diapositivas a la izquierda, el patrón “Brazo derecho arriba” permite ir a la última diapositiva en una presentación, el patrón “Brazo izquierdo arriba” permite ir a la primera diapositiva en una presentación, el patrón “Dos brazos cierra” permite finalizar una presentación y el patrón “Dos brazos arriba” permite iniciar una presentación.

Por último en la barra de menú tenemos la opción “Ayuda” en la que se encuentra la opción “Ir a ayuda” la cual permite ir a una ventana de ayuda en la que se encuentran instrucciones de uso del programa como se puede apreciar en la figura A.4 d.

2. *Canvas*

El *canvas* es un elemento dentro del que se puede dibujar objetos incluyendo texto. En este espacio se dibuja el esqueleto de la persona más cercana al kinect, se enlista los patrones reconocidos, se muestra el número de patrones reconocidos y el *status* de la lectura de datos como se puede apreciar en la figura A.3. El programa trabaja con el kinect seteado en modo “sentado” por lo que

únicamente se dibujan las 10 articulaciones pertenecientes a las extremidades superiores de la persona.

Cada vez que un patrón es reconocido aparece su nombre en el *canvas* junto con la distancia DTW obtenida al aplicar el algoritmo del DTW y en la siguiente línea aparece el respectivo comando de control de diapositivas. El programa puede almacenar 10 patrones por *file* lo que permite que el usuario pueda grabar el número de patrones que desee, pero archivados en diferentes *files*.

Existen 3 tipos de lectura de datos: muy buena, buena y mala, el estado de la lectura de datos depende si el *buffer* se llena o no de datos de mala calidad lo que influye en el reconocimiento de patrones.

3. *Tab*

Un *tab* permite alternar entre diferentes ventanas. En la pantalla principal aparece un *tab* que permite escoger entre las siguientes ventanas: patrones, algoritmo, configuración y ayuda como se muestra en la figura A.3.

La pestaña “Patrones” permite ir a la pantalla principal.

La pestaña “Algoritmo” tiene un *tab* con dos pestañas adicionales, una para ver la matriz de distancias junto con el “*warping path*” y otra para ver la matriz de costo con el objetivo de mostrar cómo se obtiene la distancia DTW como se aprecia en la figura A.9. Las matrices mostradas en estas ventanas pueden tener diferente tamaño ya que los patrones pueden tener diferente longitud.

El algoritmo del DTW procesa toda la información en metros, pero el resultado final, es decir, la distancia DTW se presenta en centímetros para facilitar la lectura y presentación de resultados. La matriz de distancias que aparece en la figura A.9 no aparece completa porque se ha aplicado la restricción de ventana seteada con un valor de 20 [u] y la restricción de pendiente igual a 4 [u] en el algoritmo del DTW.

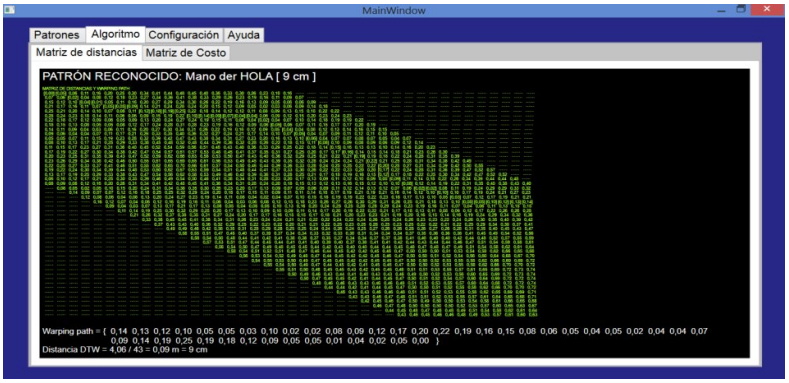


Figura A.9 Ventana para visualizar el algoritmo del DTW

La pestaña de “Configuración” permite ir a una ventana en la que se puede configurar algunos parámetros del algoritmo del DTW como se puede apreciar en la figura A.10.

El primer parámetro permite regular el umbral de la distancia DTW, es decir, para que un patrón sea reconocido la distancia DTW entre la serie de prueba y el patrón debe ser menor o igual al valor seteado en esta sección que por defecto tiene el valor de 10 [cm] el cual fue obtenido como la mejor calibración en el capítulo de pruebas y resultados.

El segundo parámetro es la restricción de pendiente del DTW el cual garantiza que pequeñas partes de una serie no coincidan con largas partes de otra, es decir, mediante este parámetro se controla la deformación del “*warping path*”. Mientras más pequeño es el valor de la pendiente más parecidos deben ser el patrón y la serie de prueba para ser reconocidos ya que no se permite que el “*warping path*” se deforme mucho.

El tercer parámetro es la restricción de ventana la cual permite poner límites al “*warping path*” recortando las matrices de costo y distancias ya que si la serie de prueba se parece al patrón entonces el “*warping path*” no debería alejarse mucho de la diagonal que va del primer al último elemento de la matriz de distancias. Mientras menor es el valor de la ventana más se recortan las matrices.

El cuarto parámetro permite controlar la sensibilidad de la lectura de datos, es decir, el valor seteado en esta sección indica cada cuantos centímetros se lee una nueva posición respecto de la anterior con el objetivo de evitar que el *buffer* se

llene de datos repetidos ya que el kinect siempre está leyendo la posición de la persona en la escena ya sea que esta se mueva o esté inmóvil.

Por último existe un botón de “Configuración Inicial” el cual sirve para retornar a los valores iniciales de configuración, es decir, el umbral del DTW en 10 [cm], la restricción de pendiente en 6 [u], la restricción de ventana en 20 [u] y la sensibilidad de lectura de datos en 3 [cm] como se muestra en la figura A.10.

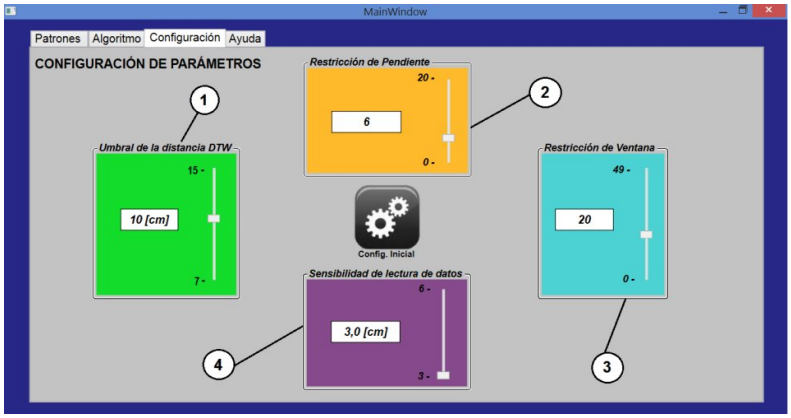


Figura A.10 Ventana de configuración de parámetros del algoritmo del DTW

La pestaña “Ayuda” permite guiar al usuario acerca del funcionamiento del programa, especialmente en la forma de grabar patrones como se aprecia en la figura A.11.

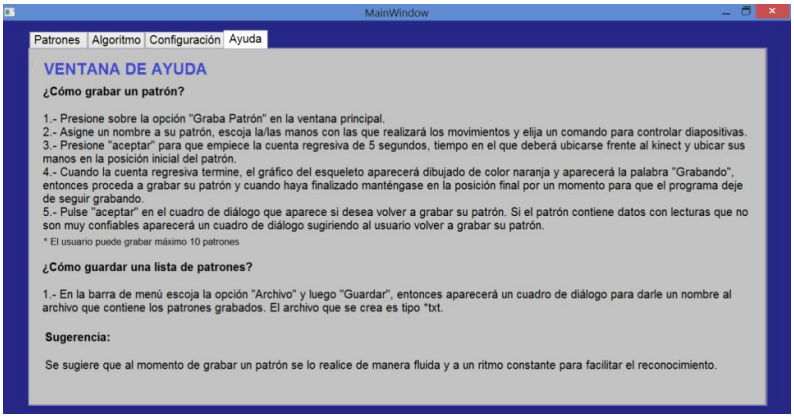


Figura A.11 Ventana de ayuda

4. *ComboBox*

Un *combobox* permite desplegar listas de elementos. El programa presenta un *combobox* en el que se enlistan los nombres de los patrones que se han grabado como se aprecia en la figura A.12. Cada vez que se graba un patrón o cuando se abre un archivo con patrones grabados sus nombres aparecen en el *combobox* ubicado en la pantalla principal del programa como se muestra en la figura A.3.

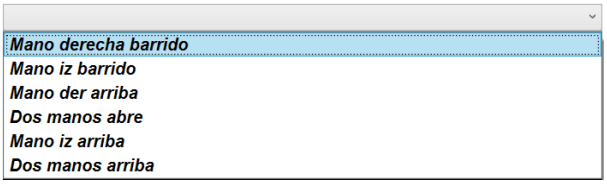


Figura A.12 *Combobox* desplegado mostrando una lista de patrones grabados

5. *Progressbar*

Un *progressbar* es una barra que permite visualizar el estado de avance de una acción. Mientras el usuario graba un patrón esta barra se llena de color verde para indicar el avance de llenado del *buffer* de datos como se aprecia en la figura A.7. El *progressbar* también es usado para visualizar el porcentaje con el que es reconocido un patrón, cuando un patrón es reconocido el color del *progressbar* es verde y cuando un patrón no es reconocido su color es rojo como se muestra en la figura A.13.

Figura A.13 *Progressbar* mostrando la detección de un patrón

6. *Slider*

Un *slider* permite cambiar el valor de una variable dentro de un límite máximo y mínimo. El kinect tiene la capacidad de regular el ángulo de visión mediante el

movimiento vertical del panel de sensores para enfocar de mejor manera la escena que el kinect captura, mediante un *slider* ubicado en la pantalla principal se puede regular el ángulo de elevación del kinect en un rango de -5º a 25º como se aprecia en la figura A.3.

7. *CheckBox*

Un *checkbox* permite activar o desactivar funciones del programa. Para habilitar el control de diapositivas es necesario activar el *checkbox* llamado “Activar Control de Diapositivas” ubicado en la pantalla principal como se muestra en la figura A.3. Una vez activada esta opción se puede controlar cualquier presentación de diapositivas hecha en *power point*.

8. *Button*

Un *button* permite ejecutar una acción al hacer *click* sobre él. En la pantalla principal del programa existe un *button* para grabar patrones el cual cumple con la misma función que la opción “Grabar Patrón” ubicado en la sección “Herramientas” en la barra de menú como se aprecia en la figura A.3. Su ubicación permite grabar patrones de forma directa.

4) Uso de los patrones modelo del programa

El programa permite al usuario usar 6 patrones modelo desde la “barra de menú” en la opción “herramientas” como se aprecia en la figura A.14.

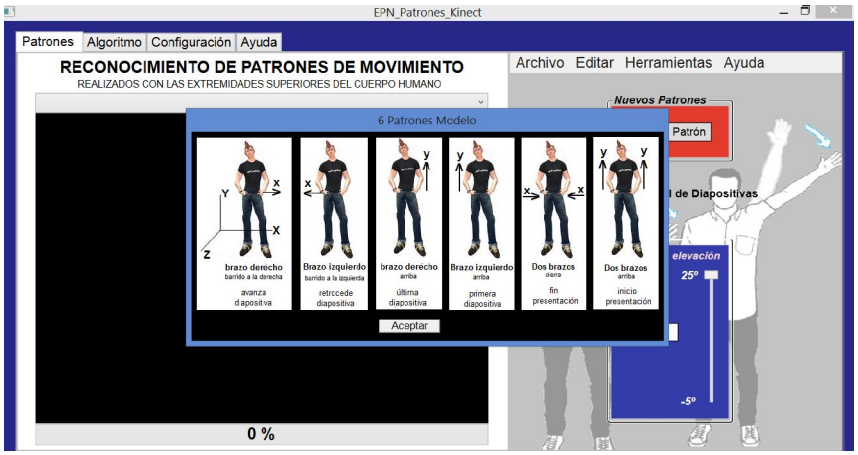


Figura A.14 Patrones modelo en la pantalla principal

A continuación se describen los patrones modelo y su uso.

1. **Brazo derecho barrido a la derecha**

   Para que el programa reconozca este patrón el usuario debe mover su brazo derecho horizontalmente (paralelo al eje “X”) de izquierda a derecha como se aprecia en la figura A.14. Con este patrón se puede avanzar diapositivas.

2. **Brazo izquierdo barrido a la izquierda**

   Este patrón consiste en mover el brazo izquierdo horizontalmente de derecha a izquierda y con este patrón se puede retroceder diapositivas.

3. **Brazo derecho arriba**

   El programa reconoce este patrón cuando se mueve el brazo derecho hacia arriba en sentido vertical (paralelo al eje “Y”) como se aprecia en la figura A.14. Con este patrón se puede ir a la última diapositiva.

4. **Brazo izquierdo arriba**

   Consiste en mover el brazo izquierdo hacia arriba en sentido vertical. Con este patrón se puede ir a la primera diapositiva.

5. **Dos brazos cierra**

   Para repetir este patrón se empieza con los brazos abiertos y luego se los mueve simultáneamente de forma horizontal hasta que los brazos se encuentren. Con este patrón se puede finalizar una presentación.

6. **Dos brazos arriba**

   Este patrón consiste en mover los dos brazos a la vez hacia arriba en sentido vertical. Con este patrón se puede iniciar una presentación.