

UNIVERSIDAD VERACRUZANA



FACULTAD DE INGENIERÍA

RECONOCIMIENTO DE GESTOS CORPORALES A
PARTIR DE VISIÓN ARTIFICIAL PARA LA
INTERACCIÓN HOMBRE-MÁQUINA HACIENDO
USO DE UN SENSOR RGB-D

T E S I S

QUE PARA OBTENER EL GRADO DE:
INGENIERO EN INFORMÁTICA

PRESENTA:
JOEL BARRANCO JIMÉNEZ

ASESOR DE TESIS:
DR. LUIS FELIPE MARÍN URIAS



BOCA DEL RIO, VERACRUZ

JUNIO, 2016

**Reconocimiento de Gestos Corporales a partir de
Visión Artificial para la interacción
Hombre-Máquina haciendo uso de un sensor
RGB-D**

por

Joel Barranco Jiménez

Tesis presentada para obtener el grado de

Ingeniero en Informática

en el

FACULTAD DE INGENIERÍA

UNIVERSIDAD VERACRUZANA

Boca del Rio, Veracruz. Junio, 2016

*Dedicado a mis padres
por su constante apoyo y consejos a lo largo de toda mi vida.
Gracias*

Agradecimientos

Gracias a mi familia por su apoyo y confianza a lo largo de mi carrera. En especial a mi hermano Carlos que me ayudó en la ortografía de este documento.

A mi asesor de tesis el Dr. Luis Felipe Marin Urias por haberme apoyado e impulsado a lo largo de la tesis y demás clases impartidas, por compartir su conocimiento e inspirar en mi una gran admiración.

A mis compañeros de carrera que hicieron mas amena mi estancia en la universidad

A mis colegas del laboratorio de robótica, Jesus Viveros y Carlos Nuñez por su amistad y compañerismo.

A todas aquellas personas que de alguna u otra manera me brindaron su amistad y apoyo en momentos difíciles.

Y a mi persona, por todo el esfuerzo, esmero y perseverancia en alcanzar este logro; aun apesar de todos los obstáculos y tropiezos en el camino.

Joel Barranco Jiménez

“All things change in a dynamic environment.
Your effort to remain what you are is what limits you.”
-Ghost in the shell

**Reconocimiento de Gestos Corporales a partir de Visión
Artificial para la interacción Hombre-Máquina haciendo uso
de un sensor RGB-D**

por

Joel Barranco Jiménez

Resumen

Con la reciente invención de sensores de profundidad, el reconocimiento de gestos corporales ha ido ganando significativamente interés en el área de la visión artificial y la interacción humano-robot. Es por eso que en este trabajo de tesis, nuestro objetivo es el reconocimiento de gestos (e.g. saludo con la mano levantada) realizadas por un individuo en una secuencia de imágenes. A pesar de que existen varias propuestas para el reconocimiento de gestos corporales, la gran mayoría es susceptible a el entorno donde se efectúan las pruebas y dependen de la posición y número de las cámaras, entre otros muchos factores. Es por eso que sugerimos reducir el problema ocasionado por estos elementos aplicando un algoritmo general que sea capaz de reconocer gestos corporales independientemente del ambiente observado y limitando el uso de cámaras.

Se propone un método de reconocimiento basado en aprendizaje. En medida de realizar esto, se utilizará primeramente un algoritmo de clasificación basado en el agrupamiento de vecinos (k -nearest) para la primera clasificación de posiciones corporales. Esto con el fin de crear un conjunto de entrenamiento, a partir de un video obtenido de una cámara RGB-D, que posteriormente será ingresado a una red neuronal que será la encargada de gestionar estas posiciones para su correcta interpretación y nos permita un acercamiento mas adecuado a la identificación de gestos corporales.

**Reconocimiento de Gestos Corporales a partir de Visión
Artificial para la interacción Hombre-Máquina haciendo uso
de un sensor RGB-D**

by

Joel Barranco Jiménez

Abstract

With the recent invention of depth sensors, recognition of body gestures has gained significant interest in the field of artificial vision and human-robot interaction. That's why in this thesis, we aim gesture recognition (e.g. greeting with raised hand) made by an individual in an image sequence. Although there are several proposals for the body gestures recognition, most of them are susceptible to the environment and depend on the position and number of cameras, among many other factors. That's why we suggest reducing the problems caused by these elements by applying a general algorithm that will be able to recognize body gestures observed regardless of the environment and the number of cameras.

We propose a recognition algorithm based on learning. To do this, first we'll use a classification algorithm based on k-Nearest Neighbors(kNN) in order to classify postures. Aiming to create a training set, from a video obtained from a RGB-D camera, which then will be entered into a neural network that will be responsible for managing these positions for proper interpretation and lead to a more appropriate approach to the identification of body gestures.

Índice general

Índice general	viii
Índice de figuras	xii
Índice de cuadros	xvi
1. Introducción	1
1.1. Planteamiento del Problema	3
1.2. Justificación	4
1.3. Hipótesis	5
1.4. Objetivos Generales	5
1.4.1. Objetivos Específicos	5
2. Marco Teórico	6
2.1. Antecedentes	6
2.1.1. Visión Artificial	7
2.1.2. Aplicaciones	9
2.1.2.1. Interfaces de Usuario	9
2.1.2.2. Video Vigilancia	10
2.1.2.3. Diagnóstico e Identificación en Base a Movimiento .	10
2.1.3. Máquinas de aprendizaje	11
2.1.3.1. Algoritmo de Clasificación kNN	12
2.1.3.2. Redes Neuronales	14
2.2. Estado del Arte	18
2.3. Detección	19
2.3.1. Modelos basados en Segmentación por Movimiento	19

2.3.1.1. Eliminación de Fondo	19
2.3.1.2. Métodos Probabilísticos	21
2.3.1.3. Flujo Visual	21
2.4. Seguimiento	22
2.4.1. Seguimiento basado en modelo	23
2.4.1.1. Estructura Ósea	24
2.4.1.2. Contorno 2D	24
2.4.1.3. Proyecciones 3D	25
2.5. Análisis del movimiento	26
2.5.1. Modelos Ocultos de Markov	27
2.5.2. Comparación de Modelos	28
3. Diseño	29
3.1. Diseño General del Sistema	29
3.1.1. Módulo de Detección y Seguimiento (MDSC)	31
3.1.2. Módulo de Identificación de Posturas (MIDP)	32
3.1.3. Módulo de Reconocimiento de Gestos (MRGC)	33
3.1.4. Sistema Basado en Profundidad	34
3.1.5. Sistema Basado en la detección del Esqueleto Humano	37
3.1.6. Modelo de la Red Neuronal	38
3.2. Análisis de Requerimientos	43
3.2.1. Sistema Operativo	43
3.2.1.1. Arquitectura General	43
3.2.2. Kinect	44
3.2.3. OpenCV	46
3.2.4. Libfreenect	46
3.2.5. OpenNI y NITE	47
3.2.6. Python	48
3.2.7. Nimblenet	49
4. Implementación y Resultados	50
4.1. Implementación del sistema de reconocimiento de gestos corporales .	50

4.2. Detección de Características	50
4.2.1. Máscara de detección basada en profundidad	51
4.2.1.1. Regilla de Ocupación	53
4.2.2. Modelo del Esqueleto Humano	57
4.2.2.1. Distancia y ángulos entre Articulaciones	58
4.3. Identificación de Posiciones Corporales	61
4.3.1. Clasificador de Posiciones con kNN	61
4.3.1.1. Resultados de Posturas con kNN	64
4.3.2. Clasificador de Posiciones con Redes Neuronales	66
4.3.2.1. Resultados de Posturas con Redes Neuronales	72
4.4. Interpretación de Gestos Corporales	75
4.4.1. Reconocimiento de Gestos Corporales por kNN	75
5. Conclusiones	81
5.1. Trabajo Futuro	82
Bibliografía	84

Índice de figuras

2-1. Ejemplos de aplicaciones de la visión artificial.	8
2-2. Ejemplo de Reconocimiento de Personas para Video vigilancia haciendo uso de filtros de color para detectar los cambios de intensidad de la escena original	10
2-3. Funcionamiento General del Algoritmo kNN	13
2-4. Estructura básica de una red neuronal	16
2-5. Procesamiento de la información de una neurona.	16
2-6. Perceptrón.	17
2-7. Ejemplificación del funcionamiento de una red neuronal backpropagation.	18
2-8. Procedimiento básico para la extracción del fondo, donde podemos observar la escena de entrada la cual se obtienen las diferencias a partir del modelo de fondo para generar una máscara de detección,[E5]	20
2-9. Ejecución de un método probabilístico (a) La imagen actual, (b) imagen compuesta por métodos de probabilidad Gaussiana en el modelo de fondo, (c) pixeles en primer plano, (d) la imagen actual con la información de seguimiento superpuesta.	22
2-10. Ejemplos de vectores de flujo. (a) Muestra una imagen con la que la cámara se acercara al objeto, (c) muestra los vectores de flujo de acuerdo al movimiento del objeto respecto a la cámara. (b) Secuencia de imágenes que representa un objeto girando, (d) el flujo visual obtenido de b.	23
2-11. Modelo de un esqueleto humano con 10 partes del cuerpo, torso, cabeza, brazos y piernas.	24

2-12. Modelo de persona a base de cuadros. Las extremidades de las personas estan representada por “parches” planos.	25
2-13. Modelos del cuerpo humano (a) imagen original, (b) Modelo con 13 puntos, (c) modelo 3D.	26
2-14. Modelo Basico de un Modelo Oculto de Markov. x = Estados, y = Salidas Observables, a = Probabilidad de transición, b = probabilidad de salida	27
2-15. Ejemplo de la cuadricula 2D reconocer el movimiento humano. Cada ciclo de movimientos e dividió en seis tiempos, cada escena muestra la distribución espacial del movimiento en una rejilla de 4x4.	28
3-1. Diagrama que muestra las etapas que conforman nuestro Sistema de Reconocimiento de Gestos Corporales	30
3-2. Diseño del Módulo de Detección. En ambos casos se extraerán las características relevantes para su posterior clasificación.	32
3-3. Diseño del módulo de Identificación de posturas. Previa captura y guardado de las posturas a identificar, se aplicaran dos métodos de clasificación diferentes para identificar los nuevos datos obtenidos por la fase de detección cuando en sistema se encuentre en funcionamiento.	33
3-4. Diseño del módulo de Reconocimiento de Gestos. A partir de la captura de poses, se crea un set de entrenamiento que contiene una secuencia de posturas que representa un gesto corporal. Este set de entrenamiento es ingresado a dos clasificadores para su posterior etiquetado.	34
3-5. Diseño General del Sistema basado en la detección por Profundidad. .	36
3-6. Diseño General del Sistema basado en la detección del Esqueleto Humano.	38
3-7. Modelo de la primera red Neuronal para los datos obtenidos por la región de Ocupación.	39
3-8. Modelo de la segunda red Neuronal para los datos obtenidos por la región de Ocupación con una mayor cantidad de neuronas en la capa oculta.	40

3-9. Modelo de la red neuronal para la detección de posiciones usando las características obtenidas a partir de puntos 3D en un esqueleto.	41
3-10. Modelo de la segunda red neuronal en base a las características del esqueleto aumentando el número de nodos en la capa oculta.	42
3-11. Modelo de la red neuronal para la detección de gestos corporales a partir de una secuencia de posiciones.	42
3-12. Configuración del hardware dentro del Kinect, Ejemplos de las imágenes obtenidas por la cámara RGB y el sensor de profundidad respectivamente,[E2]	45
3-13. Imagen adquirida mediante la librería Libfreenect que representa mediante colores la distancia de los objetos respecto al kinect.	47
3-14. Ejemplo de detección del esqueleto humano haciendo uso de la librería OpenNI,[E3]	48
4-1. Imágenes obtenidas mediante el Kinect. (a) Imagen RGB. (b) Imagen de Profundidad a escala de grises. (c) Aplicación del Filtro de Profundidad.	51
4-2. (a) y (b) Detección de dos objetos a diferentes profundidades. (c) y (d) Dos objetos detectados a la misma profundidad ocasionando la obtención de mas de una silueta.	52
4-3. Ejemplificación de cómo se dividió la escena capturada en celdas. . . .	53
4-4. (a) Imagen original. (b) Imagen con el filtro de profundidad y la representación de las celdas.	54
4-5. Ejemplos de porcentaje de ocupación. (a) 80 %, (b) 50 %, (c) 25 % y (d) 20 %.	54
4-6. Ejemplificación de la lista generada con los porcentajes de ocupación.	54
4-7. Algunas de las poses capturadas utilizando el umbral de profundidad y la rejilla de ocupación	55
4-8. Ejemplo de dos posturas similares.	56
4-9. (a) Detección de 15 articulaciones detectadas por el Kinect. (b) Rojos: Manos, Codos y Hombros. Verde: Cuello y torso. Puntos descartados: Cabeza, caderas, rodillas y pies	57

4-10. (a) Ángulos entre las articulaciones de los brazos. (b) Distancia relativa de las extremidades respecto al cuerpo. (c) Distancias normalizadas con la escala obtenida del torso.	58
4-11. Datos de interés que se obtendrán a partir del modelo óseo	59
4-12. Algunas de las poses capturadas implementando la librería OpenNI para identificar el esqueleto humano.	60
4-13. Correcta clasificación de posiciones.	62
4-14. Falsos Positivos al clasificar.	63
4-15. Correcta clasificación de posiciones con kNN y el modelo óseo.	64
4-16. Gráfica que muestra el número de iteraciones y como fue disminuyendo el valor del error para la red neuronal de 5 nodos en la capa oculta.	67
4-17. Gráfica que muestra el número de iteraciones y cómo fue disminuyendo el valor del error para la red neuronal de 96 nodos en la capa oculta.	68
4-18. Gráfica que muestra como el error se mantuvo constante con un valor alto.	70
4-19. Gráfica que muestra un ajuste en el error después de haber cambiado los parámetros de la red neuronal.	71
4-20. Correcta clasificación de posiciones con redes neuronales y el filtro de profundidad.	73
4-21. Clasificación errónea de posiciones con redes neuronales y el filtro de profundidad.	74
4-22. Clasificación mediante redes neuronales. (a) y (b) Muestran una correcta identificación de las posiciones. (c) y (d) Falsos positivos con una errónea identificación de la pose.	75
4-23. Ejemplo de una secuencia de posiciones que juntos representan el gesto de “Saludo”.	76
4-24. Secuencias de posiciones que representan los gestos corporales “Saludo”, “Detenerse” y “Acercarse”	77

Índice de cuadros

4-1. Tabla de Resultados para celdas de ocupación usando kNN con k=9 .	65
4-2. Tabla de Resultados para detección mediante esqueleto usando kNN con k=5	65
4-3. Tabla de Resultados para celdas de ocupación usando 5 nodos en la capa oculta	72
4-4. Tabla de Resultados para celdas de ocupación usando 96 nodos en la capa oculta	73
4-5. Tabla de Resultados para detección mediante esqueleto a partir de la red neuronal entrenada con 5 nodos en la capa oculta.	74
4-6. Cuadro con los gestos reconocidos una vez implementado la secuencia de posturas.	79

Capítulo 1

Introducción

La humanidad siempre ha buscado la manera de evolucionar día a día en todos los ámbitos de la sociedad, así en mejoras en la industria, como en la salud y el hogar por mencionar algunos. Esta evolución va de la mano con el gran avance tecnológico que se ha desarrollado en las últimas décadas, y la mejora de calidad de vida para las personas es evidentemente el principal motor de este progreso.

Dentro de todo este avance existe una disciplina en particular que ha crecido exponencialmente en un periodo bastante corto de tiempo; se trata de las ciencias de la computación. Si observamos en la actualidad todo lo que nos rodea, podremos ver que sin duda está presente en todo momento de nuestra vida, desde nuestros teléfonos celulares hasta en el cajero automático en donde retiramos efectivo, o en el gps de nuestro automóvil. Pero dentro de esta misma ciencia, se encuentra una rama en particular que ha ganado fuerza en los últimos años. Hablamos precisamente de la inteligencia artificial. El avance de esta rama ha sido tal, que lo que se creía ciencia ficción hace algunas décadas, es ahora una realidad. La creación de asistentes personales que se adaptan a nuestras necesidades, redes sociales que aprenden de nuestros gustos, reconocimiento de patrones, planificación automática de tareas, entre muchos más.

Pero, ¿qué es la inteligencia artificial? Quizás el primer personaje que introdujo el término Inteligencia Artificial (IA) fue el inglés Alan Turing en su artículo “Maquinaria computacional e Inteligencia” en 1950, en el cual cuestionaba si una máquina era capaz de pensar y propuso una serie de evaluaciones para corroborar este hecho

y que hoy conocemos como la prueba de Turing. A pesar de este hecho, no fue hasta 1955 que el matemático John McCarthy acuñó el concepto de inteligencia artificial por primera vez como “la ciencia e ingeniería de crear máquinas inteligentes” [1]. A partir de esto el área de la inteligencia artificial ha ido evolucionando constantemente en su estudio logrando que la mayoría de los investigadores y libros de texto definen el campo como “el estudio y diseño de agentes inteligentes” [2]. Un agente es algo que razona (agente viene del latín *agere*, hacer) [2]. Por consiguiente la meta primordial es crear un elemento inteligente, pero para poder diferenciarlo del resto, este elemento debe contar con ciertas características que lo distingan del resto, como sistemas autónomos que perciban el entorno que les rodea, que se adapten a cambios en su ambiente y que sean capaces de lograr objetivos. Y siendo el ser humano, el modelo a seguir.

Para alcanzar este propósito, la IA se divide principalmente en las siguientes áreas: Procesamiento de Lenguaje Natural, Representación del conocimiento, Razonamiento automático, Aprendizaje automático, Robótica y Visión artificial. Las primeras son las encargadas del tratamiento, almacenamiento y análisis información. Las cuales se obtienen por las últimas dos; la Robótica por su parte se ocupa de la manipulación de objetos y la Visión Artificial con la percepción de estos y su entorno. En sus inicios, la Inteligencia Artificial se enfocaba principalmente en las primeras cuatro áreas, ya que la tecnología aún no se encontraba lo suficientemente desarrollada para que un agente inteligente pudiera interactuar físicamente con su ambiente de forma adecuada dadas las limitantes de la época en cuestión de sensores y cámaras, como también en la creación de sistemas mecánicos para la construcción de robots. La Visión por Computadora se ha convertido es una de las partes más importantes en la inteligencia artificial, y es que hoy en día los investigadores en esta área han desarrollado, en conjunto con métodos matemáticos, técnicas que nos permiten la reconstrucción en 3D de objetos, el seguimientos de personas en una escena compleja e incluso avances en el reconocimiento de personas en tiempo real. A pesar de este gran avance, aún parece imposible que una máquina sea capaz de tener la misma interpretación visual de una persona. Y es que el dilema con la visión artificial, se trata de un problema inverso en el que se busca describir el mundo que

observamos en una o más imágenes y reconstruirlo en propiedades como siluetas, iluminación y distribución de color [3].

En el siguiente trabajo de tesis se pretende dar una propuesta de solución a una de las muchas problemáticas enfrentadas por este campo, como lo es el reconocimiento de gestos corporales, para la interacción entre un humano y un robot.

1.1. Planteamiento del Problema

Las técnicas empleadas en el área de la inteligencia artificial son considerables, y muchas veces la solución a ciertos problemas utilizando estos métodos, son principalmente implementados a la industria. Esto era cierto hasta hace algunos años; el desarrollo de nuevos algoritmos y tecnologías, sumado al esfuerzo realizado por los programadores e ingenieros, ha expandido el área de aplicación hacia los hogares. El objetivo de esto, es sin duda contar en un futuro con robots que puedan ayudarnos en distintas tareas de la vida diaria como cocinar, limpiar, ayudar con la mudanza o como una nueva forma de recreación. Para que esto sea posible el robot necesita ser capaz de interactuar de forma inherente con el ser humano. El estudio del impacto ocasionado por el desarrollo de la tecnología se ha venido estudiando desde los años 60s, pero el área de la robótica como bien mencionaba se ha enfocado más en la integración de software cada vez más inteligente que en la evolución de hardware. Y es que a pesar de que se han construido robots humanoides con asombrosas capacidades, no es suficiente para completar verdaderas tareas en el mundo real. La interacción Humano-Robot, HRI por sus siglas en inglés, estudia el diseño, entendimiento, y evaluación de sistemas robóticos, que involucran humanos y robots interactuando a través de su comunicación [4], analizar a los humanos para mejorar los robots [5]. Es por eso que debemos entender cómo nos comunicamos entre nosotros para intentar replicar este proceso en los robots.

Pongamos el siguiente ejemplo: ¿Qué sucede cuando quieres entrenar a tu perro, y le dices que gire de “esta” forma? En un principio pueda no entender lo que pasa. Algo que tal vez para nosotros es tan sencillo, él no puede interpretarlo. Y no es porque no sea suficientemente inteligente para hacerlo, es que no podemos “decirle” directamente que hacer y esperar que entienda inmediatamente. Los humanos en

cambio, encontramos esto muy fácil de entender, con solo el hecho de leer la pregunta anterior, probablemente lo primero que nos viene a la mente es el de una persona con su perro haciéndole señas con la mano moviéndola en círculos con el dedo apuntando hacia él. Estos gestos son totalmente naturales para nosotros, incluso los niños pequeños pueden entender a cierta edad este tipo de gestos. La comunicación entre las personas no solo implica transmisión de información de manera verbal, sino que hacemos uso de los gestos corporales mas de lo que pensamos. Pero para que exista un entendimiento natural de los gestos en la comunicación entre los humanos, los participantes deben poseer conocimiento de dichos gestos y contar con la habilidad cultural de aprender e imitar para crear y transmitir el significado [6].

Ahora bien, una vez señalado esto, podemos entender que la forma más natural para comunicarse con un robot es a través de gestos corporales, y no es sorpresa, que el análisis visual del movimiento de las personas es uno de los tópicos más activos actualmente en Visión Artificial. Y representa la detección, seguimiento y reconocimiento de personas, y generalmente, el entendimiento del comportamiento humano, a partir de una secuencia de imágenes [7, 8].

1.2. Justificación

Hemos alcanzado el punto donde somos capaces de construir robots que puedan actuar como compañeros, maestros, asistentes domésticos, rehabilitación, etc. Es decir, ya no vemos a los robots totalmente como herramientas, sino que hemos ido innovando en su desarrollo, y se espera que puedan actuar y reaccionar de diferentes formas como los humanos. Para que esto ocurra, los robots de servicio deben tener como eje central en su construcción una forma adecuada que les permita obtener información de su entorno, y principalmente de las personas a su alrededor, ya que con ellos interactuará mayormente y se necesita un sistema de control capaz de entender las necesidades básicas, así como órdenes de forma sencilla para un flujo de interacción cómodo entre el usuario y el robot. Es por eso que en este estudio se propone la resolución a este problema mediante la identificación de posturas corporales aplicando métodos utilizados en visión artificial, para una posterior interpretación y ser capaces de reconocer gestos corporales comunes de la vida diaria. Actualmente,

a diferencia de los inicios en la visión por computadora, contamos con cámaras de mayor precisión y calidad, capaces de simular el funcionamiento del ojo humano, implementando además sensores de distintos tipos. Quizás uno de los dispositivos más populares en nuestros días, sea el kinect. Una cámara RGB con un sensor de profundidad desarrollado por Microsoft. Originalmente creado para la interacción con su consola de videojuegos Xbox 360, se volvió muy popular en las instituciones dedicadas a la investigación en visión artificial debido a su accesible costo y adquisición.

1.3. Hipótesis

Es posible interpretar los gestos corporales de una persona de forma correcta con el uso de visión artificial.

1.4. Objetivos Generales

Interpretar los gestos corporales de una persona a partir de la identificación de su estructura ósea general, haciendo uso de un sensor RGB-D y crear de esta forma un vínculo de comunicación entre un robot y un humano.

1.4.1. Objetivos Específicos

- Estudiar las propuestas sobre análisis del movimiento humano ya establecidas en el campo.
- Diseñar un algoritmo de detección de personas que permita obtener datos significativos de poses corporales y ser almacenadas en una base de datos.
- Implementar un método automatizado de reconocimiento de posturas y/o gestos a partir de las secuencias guardadas.

Capítulo 2

Marco Teórico

Antes de iniciar con la propuesta de este trabajo de tesis, es necesario establecer primero ciertas terminologías utilizadas a lo largo de esta investigación para una mejor comprensión de esta. Además de los problemas a enfrentar en la tarea de reconocimiento de gestos corporales y los trabajos relacionados en el tema.

2.1. Antecedentes

Recientemente muchos investigadores se han interesado en desarrollar sistemas automáticos de reconocimiento del comportamiento humano. Un reconocimiento adecuado de personas puede aumentar el alcance de ciertas aplicaciones: desde detectar el comportamiento sospechoso de un individuo hasta el seguimiento y análisis de pacientes con deficiencia motriz. Así también como la interacción humano-robot (HCI) que es el objetivo principal de este estudio, facilitando la creación de un ambiente más natural en la construcción de un hogar u oficina inteligente con la ayuda de robots de servicio.

En seguida hablaremos acerca de los antecedentes relacionados con el análisis del comportamiento humano además de los términos y métodos utilizados en el diseño de nuestro sistema.

2.1.1. Visión Artificial

Todos los seres vivos tienen una habilidad innata de interactuar y manipular su ambiente de una forma apropiada. Esta correlación es lograda a través de la interacción inteligente entre la percepción y el control de movimiento; la percepción visual es básicamente la fundamental en la mayoría de los animales. Los seres humanos, percibimos la estructura tridimensional del mundo que nos rodea con bastante facilidad. Observemos cualquier objeto que tengamos a la vista, podemos decir la forma y el color así como los patrones de luz y sombra sin ningún esfuerzo aparente. En un retrato, podemos contar fácilmente y nombrar todas las personas en la imagen e incluso adivinar sus emociones que muestran sus rostros. Es por eso que la visión podría definirse de forma sencilla como “conocer el mundo que observamos” [9], lo que nos llevaría a definir la visión artificial exactamente de la misma forma exceptuando que el medio por el que conocemos el entorno es ahora mediante un instrumento computacional.

Son muchos los campos en los que investigadores en visión por computadora han estado participando, técnicas matemáticas para obtener la forma tridimensional y apariencia de los objetos en las imágenes. Técnicas fiables para calcular con precisión un modelo 3D parcial de un entorno a partir de una serie fotografías (Figura 2-1a). Dado un amplio conjunto de puntos de vistas de un objeto o de una cara en particular, se pueden crear modelos de superficies en 3D (Figura 2-1b). Realizar un seguimiento de una persona en movimiento contra un fondo complejo (Figura 2-1c). Incluso, con un éxito moderado, encontrar y nombrar todas las personas en una fotografía utilizando una combinación de la detección y reconocimiento de la cara, la ropa, y el pelo (Figura 2-1c).

De manera que estas y otros posibles usos han provocado el crecimiento de este campo, y el uso de la visión por computadora ha comenzado a emplearse en más aplicaciones sobretodo en el área de la inteligencia artificial como hemos estado mencionando a inicio de este trabajo.

Ahora que tenemos una mejor noción en lo que consiste la visión artificial, haremos acerca de los trabajos existentes acerca del análisis y reconocimiento del movimiento humano.

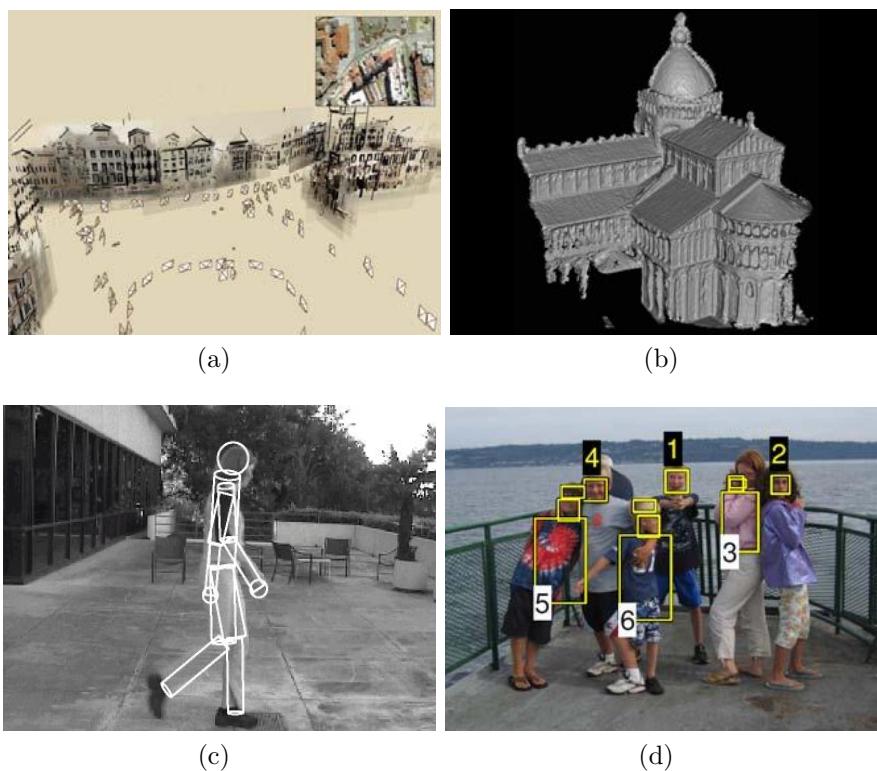


Figura 2-1: Ejemplos de aplicaciones de la visión artificial.

El análisis del movimiento humano no es algo nuevo y se ha venido estudiando, como una de las áreas principales de la visión por computadora, desde hace ya muchos años en muchos proyectos de investigación notables. Por ejemplo, La Agencia de Investigación de Proyectos de Defensa Avanzada(DARPA) fundó una institución en Video Vigilancia y Monitoreo(VSAM) [10], que tenía como propósito el desarrollar tecnología capaz de comprender de manera automática mediante video actividades humanas en áreas complejas tales como campos de batalla y escenarios civiles. El sistema en tiempo real de vigilancia visual W^4 [11] utilizaba una combinación del análisis y seguimiento de siluetas, y generaba modelos a partir de la apariencia de personas lo que le permitía detectar y seguir múltiples sujetos así como monitorear sus actividades. Además de esto, compañías como IBM y Microsoft se han visto envueltas en la investigación del movimiento humano [12, 13].

En los últimos años, el análisis del movimiento humano ha destacado en múltiples artículos a nivel internacional como Artículos Internacionales de Visión por Computadora(IJCV), Visión por Computadora y Comprensión de Imágenes(CVIU), IEEE Operaciones en el Reconocimiento de Patrones y Máquinas Inteligentes(PAMI), e

Imagen y Visión por Computadora(IVC). Todas las actividades antes mencionadas han demostrado un creciente interés en el análisis del movimiento humano a partir del reconocimiento de patrones. A continuación se presentan las distintas técnicas e investigaciones existentes así como el progreso obtenido en ellas.

2.1.2. Aplicaciones

El análisis de la actividad humana tiene un vasto número de posibles aplicaciones tales como vigilancia inteligente, interfaces de usuario avanzadas, diagnósticos basados en el movimiento, por nombrar algunos [7,8,14].

2.1.2.1. Interfaces de Usuario

Quizás una de las aplicaciones con más potencial sea la creación de interfaces que procesen la información recibida por distintos sensores para un correcto análisis e interpretación del movimiento humano para proveer control y mando. Generalmente, la comunicación entre personas se genera principalmente mediante el uso del habla. Es por eso que esta área, ya ha sido estudiada en gran medida e implementada en versiones tempranas de interfaces humano-máquina. Sin embargo, este método se encuentra restringido principalmente por el ruido generado en el ambiente así como la distancia entre el usuario y el receptor. Por su parte la visión es una herramienta muy importante capaz de complementar y mejorar las deficiencias en la comunicación auditiva y comprender el lenguaje natural de los humanos y obtener una forma más natural e inteligente de interacción entre las máquinas y los humanos. Es decir, las señales más detalladas se pueden obtener por medio de gestos, posiciones corporales, expresiones faciales, etc. [15–21]. Es por eso, que el desarrollo de futuros robots debe tener la capacidad de localizarse dentro de su entorno, detectar la presencia de personas e interpretar sus comportamientos. Algunos otros ejemplos dentro del área de aplicación de interfaces incluyen la traducción del lenguaje de señas, control mediante gestos o tele-operación [22], y la señalización de espacios muy ruidosos como fábricas o aeropuertos [23].

2.1.2.2. Video Vigilancia

Existe una gran necesidad de contar con sistemas de vigilancia inteligentes [24–26] en aquellos lugares en los que se debe contar con un alto índice de seguridad como lo son bancos, aeropuertos, tiendas departamentales, etc. Las cámaras de seguridad ya son empleadas en estos sitios, no obstante normalmente su uso se enfoca en guardar las escenas capturadas en alguna base de datos, para su posterior revisión en caso de algún incidente. Esto en ocasiones resulta bastante problemático en circunstancias de emergencias donde se necesita la inmediata acción de las personas encargadas de la seguridad. Es por eso que se necesitan sistemas inteligentes capaces de alertar a oficiales de seguridad en este tipo de situaciones para una eficaz intervención. Actualmente, las técnicas de seguimiento y reconocimiento de rostro [27–31] se han enfocado principalmente en cuestiones de control de acceso. Algunos otros trabajos [32, 33], se concentran en la detección de movimiento para detectar personas a partir de cambios repentinos en el ambiente y la iluminación para crear módulos de vigilancia en el hogar (Figura 2-2).

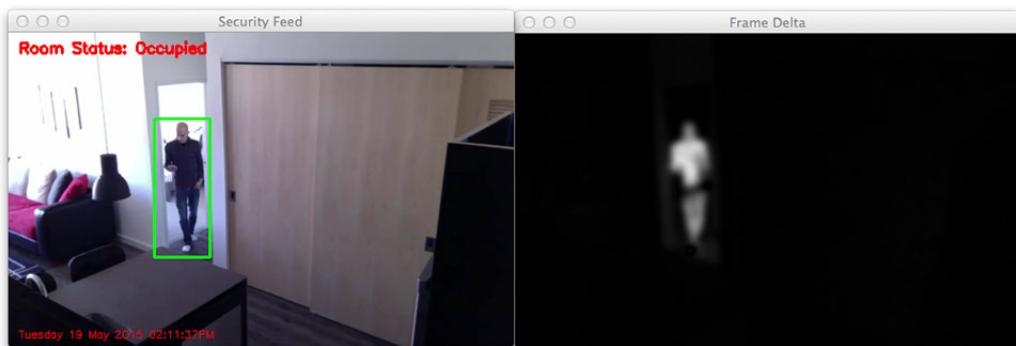


Figura 2-2: Ejemplo de Reconocimiento de Personas para Video vigilancia haciendo uso de filtros de color para detectar los cambios de intensidad de la escena original

2.1.2.3. Diagnóstico e Identificación en Base a Movimiento

El seguir el movimiento de las articulaciones de una persona a través de una secuencia de imágenes, y recuperar la estructura general del cuerpo en 3D para su posterior análisis puede resultar muy beneficioso en algunos casos. El análisis tradicional de pasos [34–36] tiene como propósito ayudar con el diagnóstico y tratamiento

médico de pacientes. Aunque algunos estudios han demostrado que el estilo de caminar de las personas puede ser una herramienta para la identificación personal [37–39], otras aplicaciones de este análisis se dirige a sistemas de tratamientos personalizados para distintos deportes, diagnósticos médicos de pacientes ortopédicos e incluso coreografías de baile.

2.1.3. Máquinas de aprendizaje

El aprendizaje automático o aprendizaje de máquinas (Machine Learning) es una de las tecnologías recientes más sorprendentes actualmente. Muy probablemente has usado algoritmos de aprendizaje de máquinas en tu vida diaria sin saberlo. Cada vez que usas algún buscador como Google en internet, una de las razones de que funcionen tan bien y te sugieran resultados aun antes de terminar de escribir lo que buscas, es debido a máquinas de aprendizaje, que han aprendido como clasificar las páginas web.

El principal propósito y objetivo de la IA, como ya mencionábamos en un principio, es el sueño de que algún día seamos capaces de construir computadoras o robots tan inteligentes como cualquier ser humano. Y aunque la meta parezca aún distante, muchos investigadores en el campo creen que la mejor forma de conseguirlo es a través de los algoritmos de aprendizaje que simulen cómo el cerebro del ser humano aprende.

Pero, ¿por qué se hace más frecuente el uso de algoritmos de aprendizaje de máquinas hoy en día? Esto se debe a que el aprendizaje de máquinas es uno de los campos que ha ido ganando fuerza dentro de la inteligencia artificial. Se desean crear computadoras inteligentes y como resultado hemos logrado realizar ciertas tareas como el encontrar el camino más corto entre dos puntos. Pero para problemas más complejos, tales como un gestor de búsqueda más avanzado o reconocer patrones en visión artificial, no teníamos el conocimiento suficiente para escribir programas inteligentes capaces de resolver estos y más problemas. Debido a esto, se comprendió que la única forma de hacer estas cosas era que una máquina aprendiera hacerla por sí misma. Así que el aprendizaje automático se desarrolló como una nueva capacidad para las computadoras y hoy se encuentra en muchos sectores de la industria y la

ciencia.

2.1.3.1. Algoritmo de Clasificación kNN

kNN (k-Nearest-Neighbour) es un método de clasificación basado en las distancias existentes entre los valores de un conjunto de entrenamiento. kNN clasifica los nuevos objetos agrupándolos con los casos más similares. KNN es un tipo de aprendizaje basado en la instancia o de aprendizaje perezoso, donde la función sólo es aproximada a nivel local y todos los cálculos se aplazan hasta la clasificación

- El algoritmo kNN es posiblemente una de las máquinas de aprendizaje más sencillas y fáciles de entender dentro de todos los algoritmos de aprendizaje de máquinas. El método kNN puede ser usado para predecir las etiquetas de cualquier tipo en un conjunto de datos. Un objeto se clasifica por el voto de la mayoría de sus vecinos, con el objeto que se asigna a la clase más común entre sus k vecinos más próximos, donde k es un entero positivo.
- Es altamente adaptable a la información local. KNN algoritmo utiliza los puntos de datos más cercanos para la estimación, por lo tanto es capaz de sacar el máximo provecho de la información local y formar límites de decisión altamente no lineales y altamente adaptables para cada punto de datos.
- Se implementa fácilmente en paralelo. Porque se basa ejemplos, para cada punto de datos que se anotó, el algoritmo compara contra la tabla de entrenamiento para el k vecino más cercano. Dado que cada punto de datos es independiente de los demás, la ejecución de la búsqueda y la puntuación puede llevarse a cabo en paralelo, reduciendo así el tiempo de clasificación.

Los datos de entrenamiento están descritos por atributos numéricos de n dimensiones. Estos se almacenan en un espacio de n dimensiones. Cuando se recibe una muestra de ejemplo (sin etiquetar), el clasificador kNN busca dentro de los datos de entrenamiento k (número de vecinos más cercanos) que estén más cerca de la muestra desconocida. La proximidad entre datos se describe generalmente en términos de distancia Euclídea. La distancia Euclídea entre dos puntos $P(p_1, p_2, \dots, p_n)$ y $Q(q_1, q_2, \dots, q_n)$ está representada por la siguiente expresión (ecuación 2-1).

$$d(P, Q) = \sum_{i=1}^n (P_i - Q_i)^2 \quad (2-1)$$

Para un mejor entendimiento del funcionamiento del algoritmo kNN, este puede dividirse en tres principales pasos (Figura 2-3):

1. Calcula la distancia entre dos puntos cualquiera.
2. Encuentra el vecino más cercano basado en las distancias entre parejas.
3. Clasifica el nuevo objeto de acuerdo a la etiqueta definida basado en la lista de vecinos más cercanos.

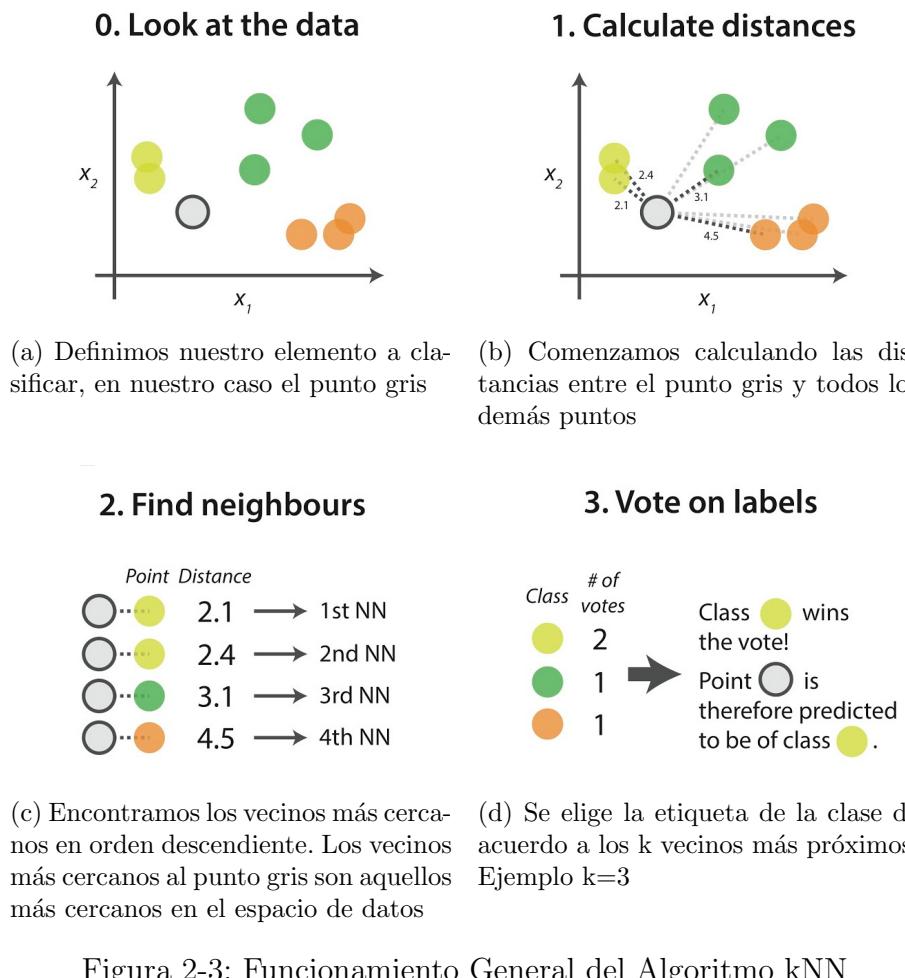


Figura 2-3: Funcionamiento General del Algoritmo kNN

Cuando los datos de entrada del algoritmo son demasiado largos para procesar, estos se convierten en un conjunto de entrenamiento con características reducidas

(también conocido como vector de características). A esta acción se le conoce como extracción de características. Si las características extraídas son cuidadosamente seleccionadas, se espera que el conjunto de características que van a extraer la información de los datos de entrada sean relevantes, con el fin de realizar la tarea deseada usando este nuevo conjunto reducido en lugar de la entradas de tamaño completo. La extracción de características se realiza sobre los datos originales antes de aplicar el algoritmo k-NN sobre los datos transformados en el espacio de características.

Un ejemplo típico de este caso se encuentra dentro de la visión artificial en el reconocimiento facial haciendo uso del algoritmo kNN al extraer características y reducir los datos de entrenamiento.

Aun con las ventajas que cuenta kNN, también cuenta con ciertas desventajas que hay que considerar:

- Alto costo de procesamiento, debido a que se necesita calcular las distancias entre los datos de entrenamiento cada vez que se requiere clasificar un objeto nuevo.
- Requiere de gran capacidad de memoria de acuerdo al tamaño del conjunto de entrenamiento.
- Baja exactitud en conjuntos de entrenamiento multidimensionales con características irrelevantes
- No existe una regla para determinar k

2.1.3.2. Redes Neuronales

Existen ciertas categorías de problemas que no pueden ser formulados simplemente con un algoritmo. Problemas que dependen de factores muy sutiles que una computadora no puede interpretar, pero para que cualquier persona resultaría fácil de resolver. La gran diferencia entre una computadora y un ser humano, es que los humanos “aprendemos”. Contamos con un cerebro que puede aprender debido a la experiencia. Las computadoras por su parte, cuentan con unidades de procesamiento y memoria. Esto les permite ejecutar cálculos numéricos complejos en un periodo de tiempo muy corto, pero no son adaptivos.

Las Redes Neuronales Artificiales, también conocidas como Redes Neuronales, ANN por sus siglas en inglés (Artificial Neuronal Network), son herramientas computacionales modelo de la interconexión de las neuronas en el sistema nervioso humano. Las redes neuronales biológicas son el equivalente natural de las ANN. Ambos sistemas de redes construidas a partir de componentes atómicos conocidos como “neuronas” [40]. Las redes neuronales artificiales son muy diferentes de las redes biológicas, aunque muchos de los conceptos y características de los sistemas biológicos se intentan reproducir en los sistemas artificiales. Las redes neuronales artificiales son un tipo de sistema de procesamiento no lineal que es ideal para una amplia gama de tareas, en especial tareas en las que no hay algoritmo existente para la realización de estas. Las ANN pueden ser entrenadas para resolver ciertos problemas utilizando un método de enseñanza y un conjunto de entrenamiento. Con el entrenamiento adecuado las ANN son capaces, en la mayoría de los casos, de reconocer similitudes entre los diferentes patrones de entrada, especialmente en aquellos patrones afectados por alguna clase de ruido.

Las redes neuronales están organizadas en capas. Estas capas están formadas a partir de un número de interconexiones llamados “nodos”, o como antes mencionábamos neuronas, que contienen una función de activación. Los patrones o datos de entrenamiento son presentados a la red neuronal por medio de la “capa de entrada”, la cual se comunica con una o más “capas ocultas” que es donde todo el proceso de la red es realizado mediante un sistema de conexiones ponderadas. Las capas ocultas a continuación enlazan a la “capa de salida”, donde la respuesta se emite [41].

La característica principal de las redes neuronales es su habilidad de “aprender”. Una red neuronal no solamente es un sistema complejo, sino también un sistema adaptivo, que puede cambiar su estructura interna basada en la información que fluye a través de ella. Normalmente esto se logra mediante el ajuste de “pesos”. En la figura anterior (Figura 2-4), cada línea representa una conexión entre dos neuronas e indica el camino en el que fluye la información. Cada una de estas conexiones cuenta con un peso, un número que controla el flujo de la información entre las dos neuronas. Si la red genera una salida favorable, entonces no hay necesidad de

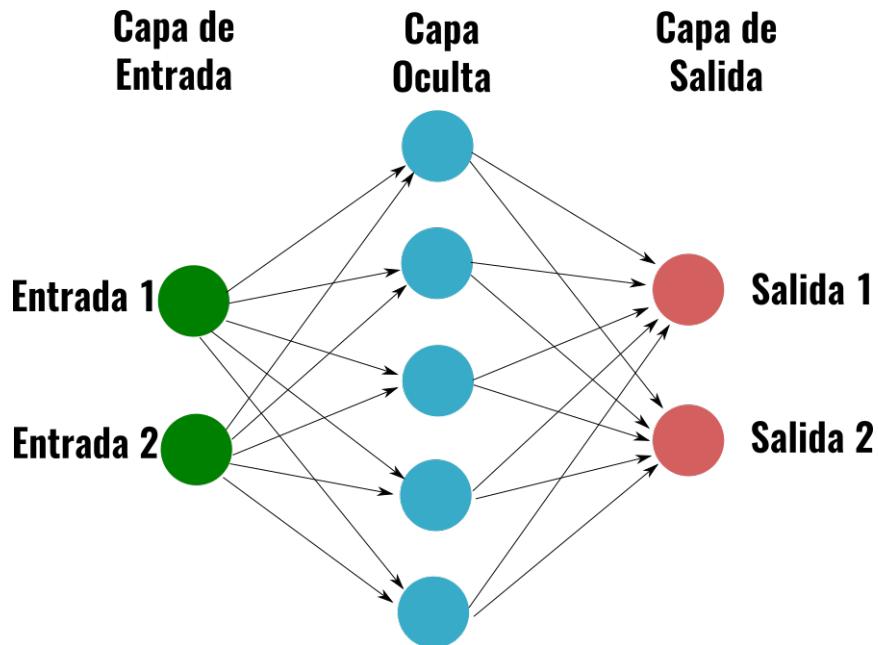


Figura 2-4: Estructura básica de una red neuronal

ajustar los pesos. No obstante, si la red genera un resultado no deseado, el sistema se adaptará, alterando los pesos con el fin de mejorar los resultados posteriores. (Figura 2-5)

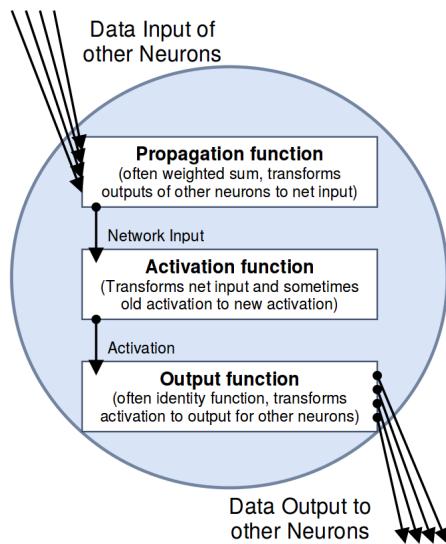


Figura 2-5: Procesamiento de la información de una neurona.

Perceptrones inventados en 1957 por Frank Rosenblatt, un perceptrón es la red neuronal más simple posible: un modelo computacional de una sola neurona. Un perceptrón consiste en uno o más datos de entradas, que son procesados, con un sólo

resultado.

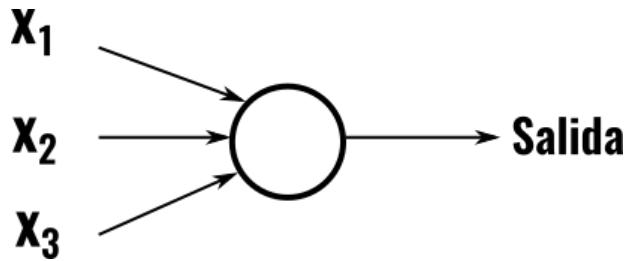


Figura 2-6: Perceptrón.

En el ejemplo anterior (Figura 2-6), el perceptrón tiene tres entradas, x_1, x_2, x_3 . Rosenblatt propuso una regla muy simple para calcular la salida. Él introducía pesos, w_1, w_2, \dots , números reales que expresaban la importancia de las respectivas entradas a la salida. El resultado final de la neurona, 0 o 1, se determinaba por la suma ponderada $\sum_j w_j x_j$ que es menor o mayor al valor de un umbral determinado. Al igual que los pesos, el umbral es un número real que es un parámetro de la neurona (ecuación 2-2).

$$\text{output} = \begin{cases} 0 & \text{si } \sum_j w_j x_j \leq \text{umbral} \\ 1 & \text{si } \sum_j w_j x_j \geq \text{umbral} \end{cases} \quad (2-2)$$

Aun cuando no existe una regla precisa y existen muchas variantes de las reglas de aprendizaje usadas por las redes neuronales, la más frecuente es la regla delta. La regla delta es gradiente decente empleada para actualizar los pesos de los datos de entrada en las redes neuronales. Se utiliza en las redes neuronales de propagación inversa (backpropagation en inglés). Este tipo de redes neuronales en el que las salidas fluyen hacia adelante y la propagación de errores hacia atrás como un ajuste de los pesos. Es decir, cuando la red neuronal se inicializa con una muestra realiza una conjectura al azar. A continuación, ve que tan lejos su respuesta fue de la verdadera y hace el ajuste correspondiente de sus pesos en la conexión (Figura 2-7).

Existen muchas ventajas y limitaciones en el análisis y desarrollo de las distintas tipos de redes neuronales. En cuanto a las redes neuronales del tipo de propagación inversa existen ciertos inconvenientes que se deben tomar en cuenta []:

- La redes neuronales de propagación inversa (y cualquier otro tipo de red) son en esencia “cajas negras”. Dejando a un lado la definición general de la

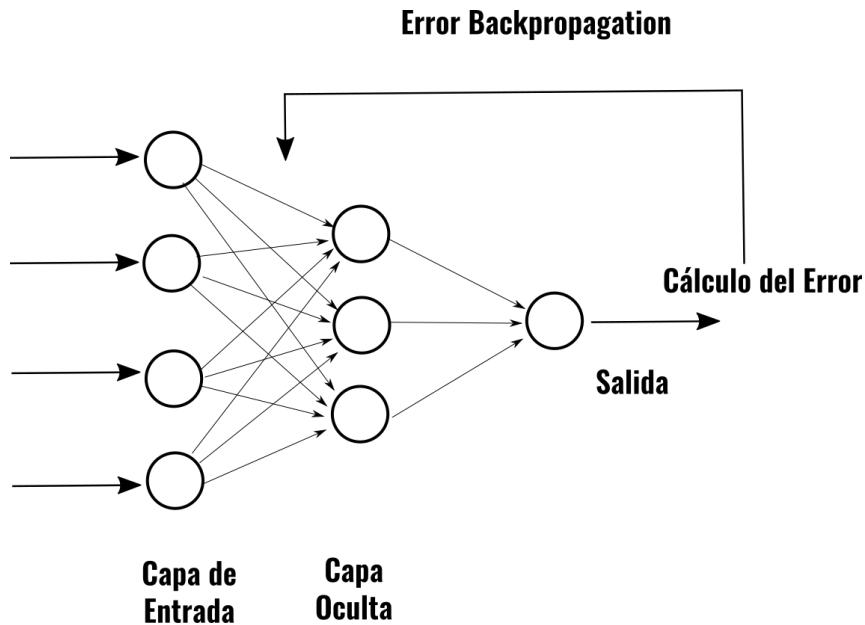


Figura 2-7: Ejemplificación del funcionamiento de una red neuronal backpropagation.

arquitectura de la red, y la inicialización de datos de entrada, no se tiene otro medio posible en el que pueda modificar el funcionamiento de la misma, y solo pasamos a ser observadores del entrenamiento y la salida de datos. En ocasiones se ha dicho que con la propagación inversa “casi no sabes lo que estás haciendo”.

- Este tipo de redes suelen tener un entrenamiento más lento que los de otro tipo y en ocasiones necesitan cientos de iteraciones. En computadoras de sistemas paralelos no suele ser un problema, pero si se simula en una computadora estándar, el entrenamiento puede tomar un tiempo. Esto se debe a que el CPU debe calcular la función de cada nodo y conexión de forma separada, lo que podría ocasionar problemas en una red neuronal extensa con una gran cantidad de datos. Sin embargo, la velocidad de la mayoría de las máquinas actuales es tal, que normalmente ya no es un impedimento.

2.2. Estado del Arte

Una de las áreas más investigadas dentro de la visión artificial es sin duda el análisis de la locomoción humana que intenta detectar, seguir e identificar personas, y generalmente, tratar de interpretar el comportamiento humano a través de una

serie de imágenes en las que se encuentra involucrado alguna persona. El análisis del movimiento humano ha atraído el interés de muchos investigadores del área de visión por computadora debido a las prometedores campos de aplicación tales como vigilancia, el uso como interfaz de usuario, almacenamiento y recuperación de contenido basado en imágenes, realidad virtual entre otras.

La interpretación del movimiento humano se divide en tres partes esenciales: La detección de la persona, el seguimiento de esta, ya por último el análisis de adecuado de las 2 fases anteriores para su correcto procesamiento.

A continuación hablaremos acerca de los estudios existentes que se relacionan con este trabajo de investigación sobre la detección del movimiento humano y la interpretación de sus actividades, junto con los métodos utilizados en ellos.

2.3. Detección

Como ya hemos estado comentando a lo largo de este capítulo, todo sistema de visión basada en el movimiento de los humanos requiere principalmente detectar a la persona en cuestión. La detección de personas significa segmentar regiones de la imagen que correspondan a personas y los separen o identifiquen del resto. Es una parte vital en sistemas basados en el análisis del movimiento humano debido a que los procesos subsecuentes de la detección, como son el seguimiento y reconocimiento de acciones dependen fundamentalmente de esto.

2.3.1. Modelos basados en Segmentación por Movimiento

La segmentación por movimiento es utilizada para detectar regiones correspondientes a elementos en movimiento que pudieran ser posibles objetivos en alguna escena con el fin de proveer un foco de atención para futuros procesos como el seguimiento y análisis de actividades [26].

2.3.1.1. Eliminación de Fondo

La eliminación del fondo es uno de los métodos más empleados para la segmentación de movimiento, especialmente en situaciones en las que nos encontramos con

fondos relativamente estáticos, y la cámara se mantiene inmóvil. La idea detrás de esta técnica se refiere a detectar objetos en movimiento a partir de la diferencia pixel por pixel entre la escena actual y una escena de referencia previamente analizada. Generalmente la escena de referencia es conocida como “imagen de fondo” o “modelo de fondo”, y debe ser una representación de la escena a analizar en cuestión, sin objetos en movimiento, y debe mantenerse actualizada regularmente con el fin de adaptarse a las condiciones de cambios de luminosidad y configuraciones geométricas (calibración de la cámara) [42].

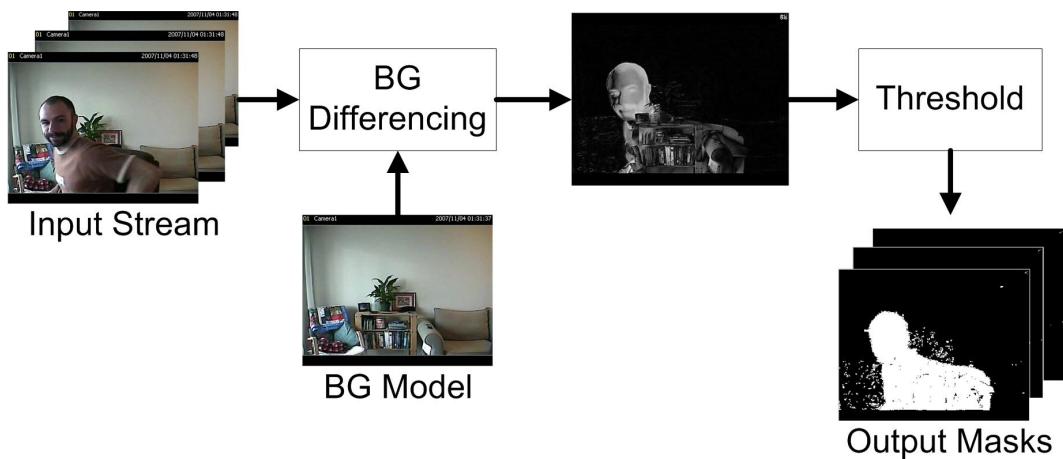


Figura 2-8: Procedimiento básico para la extracción del fondo, donde podemos observar la escena de entrada la cual se obtienen las diferencias a partir del modelo de fondo para generar una máscara de detección,[E5]

Hay muchos métodos diferentes para la eliminación de fondo propuestos en muchos artículos actuales. Todos estos métodos intentan estimar con eficacia el fondo de la secuencia de imágenes temporales. La diferencia de estos enfoques reside normalmente en el tipo del modelo de fondo y el procedimiento usado para actualizarlo. El modelo de fondo más sencillo es el basado imagen temporal promedio, la cual es una aproximación del fondo a la escena estática actual (Figura 2-8). También existentes estudios en las que las secuencias de imágenes son transformadas a escala de grises, lo que reduce en gran medida los falsos positivos al momento de detectar el fondo, además de que al usar un sólo canal para el color reduce el tamaño de los datos a analizarse y aumenta la calidad del modelo de fondo [43]. Por otra parte, otros trabajos [44] proponen el uso de un valor medio de los últimos n cuadros como su modelo de fondo. Este algoritmo puede manejar ciertos cambios de iluminación

ya que el modelo base se toma momentos antes de analizar el objeto en movimiento.

2.3.1.2. Métodos Probabilísticos

Recientemente, algunos métodos estadísticos han sido propuestos para la extracción de regios cambiantes en un fondo. Unos de los primeros en trabajar con este tipo de métodos fueron Stauffer y Grimson [45], quienes presentaron un modelo de mezcla de fondo adaptivo para el seguimiento en tiempo real, en la que cada pixel se modela como una mezcla de aproximaciones gaussianas y lineales, que se emplean para actualizar el modelo 2-9. Las ventajas de este tipo de métodos: es que en primer lugar, puede manejar múltiples objetos de fondo mediante el uso de modelo de fondo multivariable, y como segundo punto, es insensible al ruido, sombra, y el cambio de las condiciones de iluminación.

El seguimiento de personas siempre ha sido un problema desafiante, sobre todo cuando se emplean métodos de extracción del fondo, en el que el cambio de variables es considerable. Con el fin de resolver este problema, algunos investigadores presentaron un modelo de fondo no paramétrico que estima la probabilidad de los valores de intensidad de cada pixel observado basado en muestras de valores de intensidad de los píxeles individuales. Este tipo de modelos puede hacer frente a situaciones en las que el fondo de una escena es desordenado y no completamente estático, sino que contienen pequeños movimientos como las ramas de árboles, arbustos, etc. [46].

2.3.1.3. Flujo Visual

Este tipo de método se utiliza generalmente para describir el movimiento coherente de puntos o características entre cuadros de imagen. La segmentación de movimiento basado en el flujo visual [47–49] utiliza características de los vectores de flujo de objetos en movimiento en el tiempo para detectar las regiones de cambio en una secuencia de imágenes. Por ejemplo, Meyer [50] realizó operaciones que calculaban el campo de desplazamiento vectorial para inicializar un algoritmo de seguimiento basado en contorno, llamados rayos activos, para la extracción de objetos articulados que se utiliza para el análisis de la marcha (Figura 2-10).

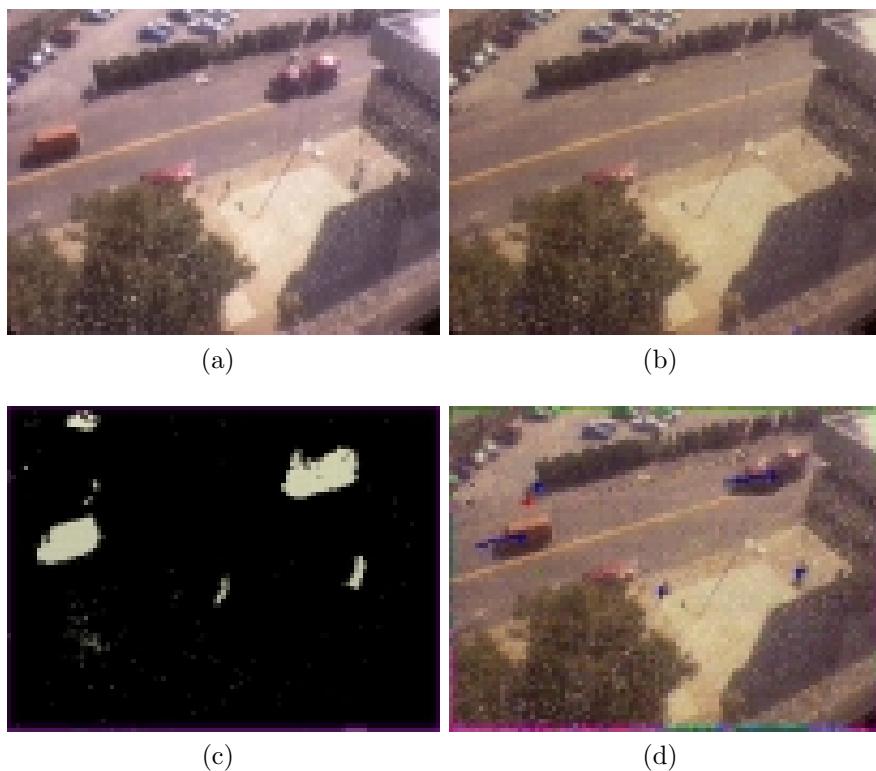


Figura 2-9: Ejecución de un método probabilístico (a) La imagen actual, (b) imagen compuesta por métodos de probabilidad Gaussiana en el modelo de fondo, (c) pixeles en primer plano, (d) la imagen actual con la información de seguimiento superpuesta.

2.4. Seguimiento

El seguimiento de objetos en videos se ha vuelto un tema muy popular en el campo de la visión artificial. El seguimiento es particularmente importante en problemas sobre el análisis del movimiento humano debido a que prepara los datos para estimación de posiciones y reconocimiento de acciones. Al contrario de la detección humana, el seguimiento humano pertenece a un nivel más alto dentro de los problemas de visión por computadora.

Sin embargo, los algoritmos de seguimiento dentro del análisis de movimiento humano por lo general tienen una considerable intersección con la segmentación de movimiento durante el procesamiento. El seguimiento a través del tiempo suele implicar objetos coincidentes en escenas consecutivas utilizando características tales como puntos, líneas o manchas. Es decir, el seguimiento puede ser considerado al equivalente de establecer relaciones coherentes de características de la imagen entre los marcos con respecto a la posición, la velocidad, forma, textura, color, etc.

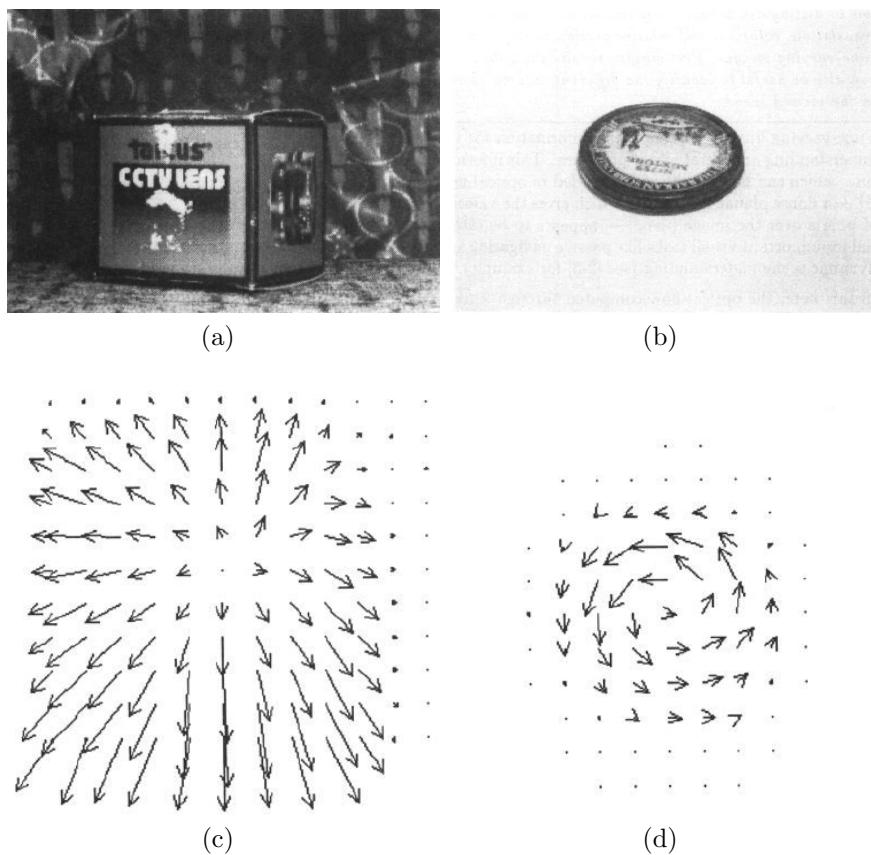


Figura 2-10: Ejemplos de vectores de flujo. (a) Muestra una imagen con la que la cámara se acerca al objeto, (c) muestra los vectores de flujo de acuerdo al movimiento del objeto respecto a la cámara. (b) Secuencia de imágenes que representa un objeto girando, (d) el flujo visual obtenido de b.

Algunas herramientas matemáticas útiles para el proceso de seguimiento incluyen filtros de Kalman [51], algoritmo de condensación [52, 53], red dinámica bayesiana [54], etc. El filtro Kalman es un método de estimación de estado basado en una distribución de Gaussiana. Por desgracia, se limita a situaciones en las que la distribución de probabilidad de los parámetros de estado es unimodal. Es decir, que es inadecuada para tratar con las distribuciones multimodales simultáneas con la presencia de oclusión, un fondo desordenado parecido a los objetos en seguimiento, etc.

2.4.1. Seguimiento basado en modelo

Normalmente, la estructura geométrica del cuerpo humano puede ser representada con un sencillo dibujo de los huesos principales, contornos del cuerpo humano

en 2D o modelos 3D [117].

2.4.1.1. Estructura Ósea

El movimiento natural del ser humano está basado por los movimientos del torso, la cabeza y las extremidades(brazos y piernas), por lo que un modelo que incorpore las principales articulaciones y huesos puede utilizarse para semejar un cuerpo humano como una combinación de líneas y uniones (Figura 2-11) [55–57]. Este modelo a base de líneas y puntos es obtenido de varias maneras, por ejemplo, por medio de la transformación del eje central o distancia transformada [58].

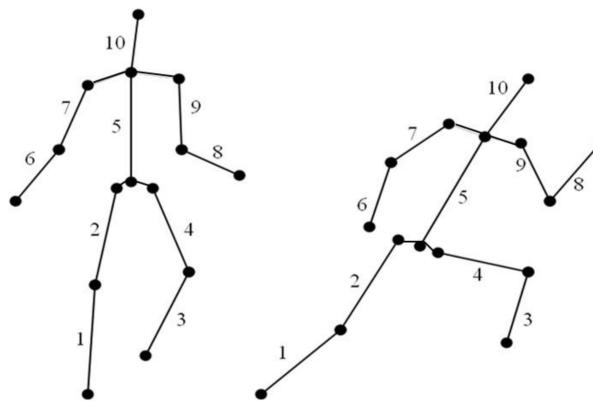


Figura 2-11: Modelo de un esqueleto humano con 10 partes del cuerpo, torso, cabeza, brazos y piernas.

El movimiento de las articulaciones proporciona una clave para la estimación de movimiento y el reconocimiento de toda figura. Por ejemplo, Guo [56] representa la estructura del cuerpo humano un modelo de figura de palos que tenía diez líneas articulados con seis articulaciones. Transformaba el problema en encontrar una figura con un mínimo de energía en un campo potencial. Además, se introdujeron la predicción y ángulo entre articulaciones para reducir la complejidad del proceso de correspondencia. Otros trabajo [55] también que utiliza este tipo de representación del cuerpo humano para construir un modelo jerárquico novedoso de la dinámica humana codificada utilizando modelos ocultos de Markov.

2.4.1.2. Contorno 2D

Este tipo de representación del cuerpo humano es importante para la proyección de una imagen plana del cuerpo humano. En dicha descripción, los segmentos

corporales humanos son análogas a las cintas 2-D o manchas [59–62].

En otros trabajos [62] se propone un modelo de personas a cuadros, en el que las extremidades de humano fueron representados por un conjunto de figuras planas conectadas (Figura 2-12). La imagen parametrizada de estos parches se contraían para forzar el movimiento articulado, y se utiliza para tratar el análisis del movimiento articulado de extremidades. En el trabajo de Leung y Yang [80], el contorno del objeto se estimó como zonas de borde representados por las cintas 2-D, que eran segmentos de borde en forma de U. El modelo de la cinta 2-D se utiliza para guiar el etiquetado de los datos de imagen.

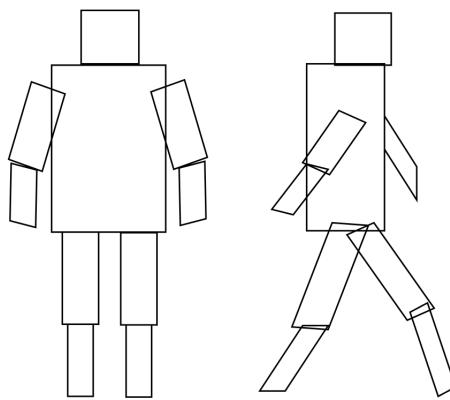


Figura 2-12: Modelo de persona a base de cuadros. Las extremidades de las personas están representada por “parches” planos.

2.4.1.3. Proyecciones 3D

Algunas de las desventajas de los modelos en 2D son las restricciones con el ángulo de la cámara, muchos investigadores han intentado recrear la estructura geométrica del cuerpo humano con más detalles haciendo uso de modelos 3D (Figura 2-13) como conos, cilindros, esferas, etc. [63–67]. Mientras más complejo sea el modelo 3D, se esperaría obtener mejores resultados pero esto requeriría un mayor cantidad de parámetros lo que ocasionaría un considerable uso de recursos para su procesamiento durante el seguimiento de la persona.

Rohr [63] creó un modelo del cuerpo humano en 3D usando 14 cilindros elípticos. El origen del sistema de coordenadas en este modelo es el eje central del torso. El autovector se aplicaba fuera de la imagen donde se encontraba la persona, donde la proyección 2D era aplicaba al modelo 3D usando una medida de distancia similar.

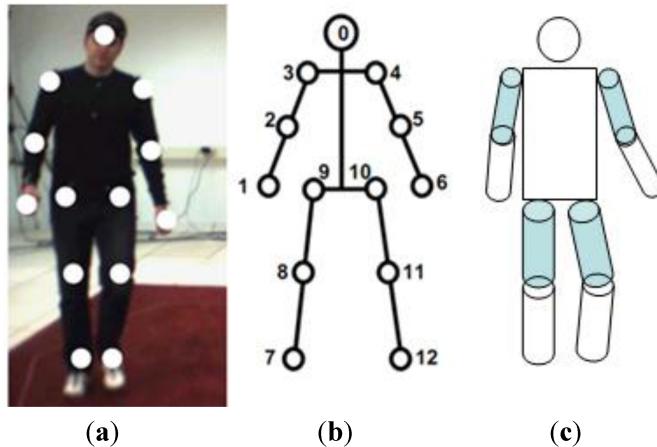


Figura 2-13: Modelos del cuerpo humano (a) imagen original, (b) Modelo con 13 puntos, (c) modelo 3D.

Con el objetivo de generar una descripción en 3D de las personas mediante un modelado, [64] recientemente se intentó establecer la correspondencia entre un modelo de cuerpo 3D de conos elípticos conectados y una secuencia de imagen real. Basados en el filtrado de Kalman, se incorporaba información tanto del borde y la región donde se encontraba la persona para determinar los grados de libertad de las articulaciones y orientaciones respecto a la cámara, obtuvieron la descripción cualitativa del movimiento humano en secuencias de imágenes obtenidas de una cámara RGB.

2.5. Análisis del movimiento

Después de que se obtiene un correcto seguimiento de la persona en una secuencia de imágenes, es momento de estudiar el comportamiento de esta. Este análisis involucra reconocimiento y descripción de acciones.

La comprensión del comportamiento humano significa analizar y reconocer los patrones del movimiento humano, y producir una descripción de alto nivel de las acciones realizadas. Y aunque parezca ser un simple problema de clasificación en los que hay que identificar un conjunto de datos obtenidos a través del seguimiento y compararlos con una base de datos previamente etiquetada, el verdadero problema es entender cómo aprender e identificar qué características de los movimientos humanos que representan un movimiento, pueden ser relevantes en un conjunto de

entrenamiento.

2.5.1. Modelos Ocultos de Markov

Los modelos ocultos de Markov (HMM en inglés) [68,69], es un tipo de máquina de estado estocástica, es decir, está sometido al azar y a un análisis estadístico. Es una técnica más sofisticada para analizar datos que cambian respecto al tiempo con variabilidad espacio-tiempo. Este modelo es la suma de la HMM y un conjunto finito de salida con distribuciones de probabilidad. El uso de las HMM está dividido en dos etapas: entrenamiento y clasificación. En la fase de entrenamiento, el número de estados de la HMM debe ser establecido, así como el correspondiente estado de transformación y la probabilidad de las salidas son optimizadas para que sean capaces de que los símbolos generados puedan corresponder con las características observadas en la imagen dentro de los ejemplos de una clase de movimiento específico. Durante la etapa de clasificación, es calculada la probabilidad de que una HMM pueda generar la etiqueta de la secuencia de prueba correspondiente a las características de la imagen observada.

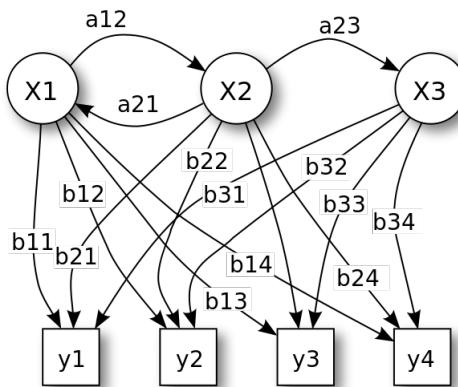


Figura 2-14: Modelo Basico de un Modelo Oculto de Markov. x = Estados, y = Salidas Observables, a = Probabilidad de transición, b = probabilidad de salida

Las HMM se han vuelto muy popular en el uso de reconocimiento de patrones en el movimiento humano [70–72], algunos han propuesto siluetas como datos de entrada [73, 74] aplicando además análisis de Fourier y Máquinas de vectores de soporte [75] para clasificar posturas corporales.

2.5.2. Comparación de Modelos

Este acercamiento basado en comparación de modelos, convierte una secuencia de imágenes en un patrón de forma estática, y luego lo compara con prototipos de acción previamente almacenados durante la fase de reconocimiento. En estudios como [76], se utilizaron características que constan de mallas 2D para reconocer el movimiento humano (Figura 2-15). En primer lugar, los campos de flujo óptico se calcularon entre cuadros sucesivos, y cada trama de flujo se descompone en una cuadrícula espacial en ambas direcciones horizontal y vertical. Entonces, la amplitud de movimiento de cada celda se acumuló para formar un vector de características de alta dimensión para el reconocimiento. Con el fin de normalizar la duración del movimiento, suponían que el movimiento humano era periódico, por lo que toda la secuencia se podría dividir en muchos procesos circulares de cierta actividad que se promediaron en divisiones respecto al tiempo. Por último, eligieron el algoritmo kNN para realizar el reconocimiento y clasificación del movimiento humano.

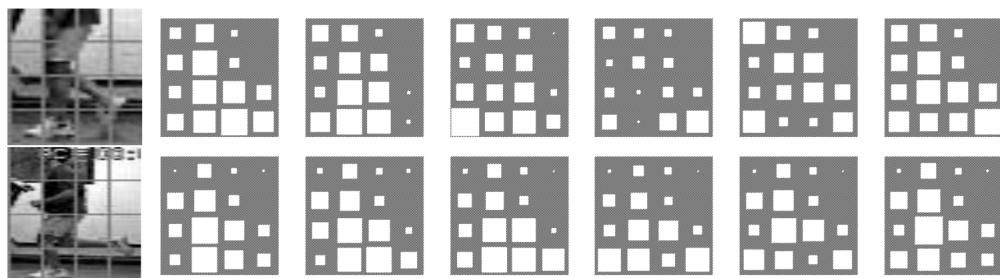


Figura 2-15: Ejemplo de la cuadricula 2D reconocer el movimiento humano. Cada ciclo de movimientos e dividió en seis tiempos, cada escena muestra la distribución espacial del movimiento en una rejilla de 4x4.

Capítulo 3

Diseño

Todo programa o sistema antes de su implementación necesita un buen diseño de su estructura. Para ello primeramente se debe analizar con detalle el problema a resolver para así poder aplicar de forma correcta una solución. En nuestro estudio sobre el reconocimiento de gestos corporales, logramos identificar tres etapas fundamentales para llegar a nuestro objetivo: Detección de la persona, Seguimiento, y Análisis posterior del movimiento realizado.

3.1. Diseño General del Sistema

Para nuestro sistema, las etapas principales se dividen en Detección y Seguimiento, Análisis de Posturas y Reconocimiento de Gestos. Cada una de ellas es subsecuente de la anterior por lo que si el módulo anterior tiene algún fallo o el procesamiento de datos no es el correcto, muy probablemente el módulo siguiente no podrá actuar de manera apropiada y por consecuente el sistema fallará o simplemente no obtendremos los resultados esperados. Al inicio de nuestro proyecto, sólo nos enfocamos en la parte de Detección de la persona y un Análisis muy básico de movimiento. Posteriormente se mejoró la fase de Detección usando filtros de profundidad y se anexó el uso de clasificadores para el reconocimiento de las posturas. En la versión final de nuestro sistema se implementó la fase de Seguimiento de personas con el uso del esqueleto humano identificando las articulaciones y extremidades principales y un mejor clasificador para interpretar una secuencia de posiciones como un

gesto corporal.

Primeramente debemos observar el panorama general de nuestro problema para crear un diseño consistente del sistema. En la Figura 3-1 se muestra el esquema general de nuestro sistema. En el cual a partir de un sensor de profundidad que en nuestro caso es el Kinect de Microsoft®, el cual detallaremos más adelante, nos proporcionará los datos con los cuales se alimentará nuestro sistema. A partir de esto nuestro primer módulo es la detección de la persona como total, una vez detectada la persona en cuestión, proseguiremos a su seguimiento dentro del cuadro de la cámara para su continua captura de datos. Posteriormente los datos capturados en las primeras etapas serán tratados dentro del módulo de Identificación de Posturas y finalmente se realizará un Reconocimiento de Gestos Corporales a partir de las posiciones identificadas.

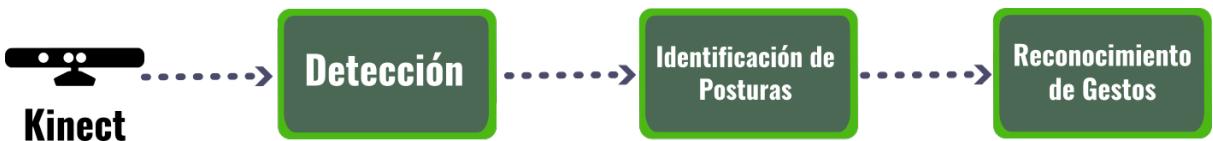


Figura 3-1: Diagrama que muestra las etapas que conforman nuestro Sistema de Reconocimiento de Gestos Corporales

Ya teniendo identificadas las etapas principales de nuestro sistema, es hora de detallar cada una de ellas. En un principio, se pretendía desarrollar un sólo sistema. Sin embargo, durante la implementación del primer diseño, a medida que se avanzaba nos enfrentábamos a ciertos problemas que no habíamos contemplado en un principio o que simplemente las herramientas empleadas en su momento no reflejaban los resultados esperados. Por lo que al indagar de forma más detallada los trabajos relacionados, así como métodos existentes para detección de personas, y clasificación de características, nuestro sistema evolucionó con la introducción de mejores técnicas y herramientas. De esta forma las dos variantes se realizaron de forma lineal y no de manera paralela. Aunque al final sólo nos quedamos con una versión, para fines prácticos se decidió continuar con las dos versiones para comparar los resultados finales de cada uno hasta la etapa de reconocimiento de posturas donde la primera versión arrojó datos poco favorables; y al ser esta fase demasiado importante para

la interpretación de los gestos corporales, no se continuó más allá en esta versión. A continuación explicaremos el diseño de cada una de las etapas:

- Módulo de Detección y Seguimiento (MDSC)
- Módulo de Identificación de Posturas (MIDP)
- Módulo de Reconocimiento de Gestos (MRGC)

3.1.1. Módulo de Detección y Seguimiento (MDSC)

La primera fase, la responsable de la detección de las características de la persona, se dividirá de la siguiente forma (Figura 3-2):

- En la primera versión, que llamaremos A, la detección de la persona se realizará aplicando un filtro de profundidad para la eliminación del fondo y se aplicará una rejilla de ocupación que se utilizará para la identificación de posturas. En este primer diseño se ignoró el seguimiento de la persona dada las limitantes del uso del filtro por profundidad. Más adelante en la implementación del sistema abordaremos y ejemplificaremos este tema.
- En la segunda variante, versión B, implementaremos una detección y seguimiento de la persona mediante la extracción de un conjunto de puntos y líneas, que representarán las articulaciones y extremidades de la persona. Posteriormente los datos obtenidos se normalizarán para un mejor manejo de los datos.

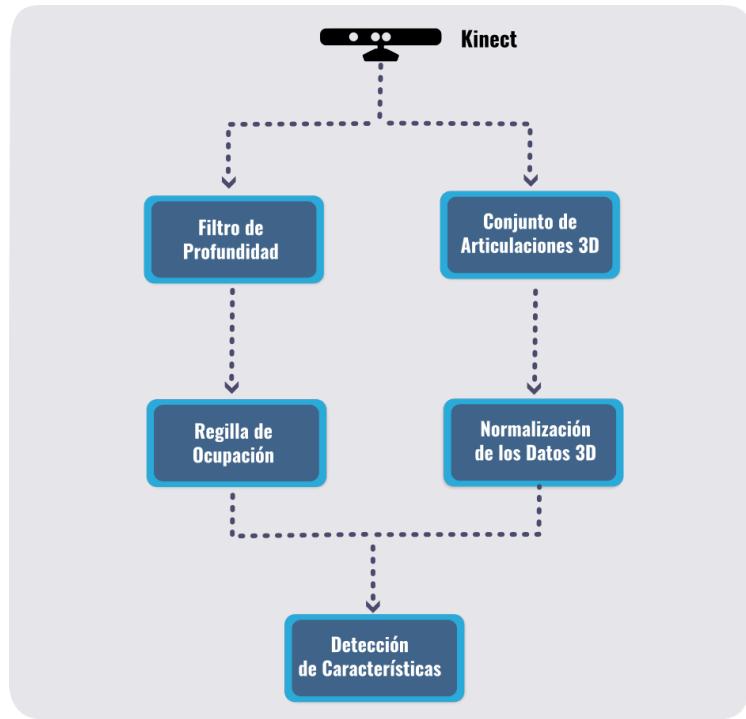


Figura 3-2: Diseño del Módulo de Detección. En ambos casos se extraerán las características relevantes para su posterior clasificación.

3.1.2. Módulo de Identificación de Posturas (MIDP)

En el módulo anterior, en ambos modelos, se obtiene un conjunto de características que hay que analizar para poder identificar las posturas que realice el usuario. Para poder realizar la clasificación de las posturas, primeramente necesitamos un conjunto de entrenamiento con el que podamos comparar los movimientos del usuario cuando el sistema se ejecute. Para esto, primero deberemos capturar manualmente cada una de las posiciones y asignarles una etiqueta que representará a qué pose pertenece. Este conjunto de entrenamiento será almacenado en una base de datos para su posterior consultado por los métodos de clasificación que implementemos. Para ello de la misma forma que la fase previa, se realizaron dos formas de clasificación de los datos. (Figura 3-3)

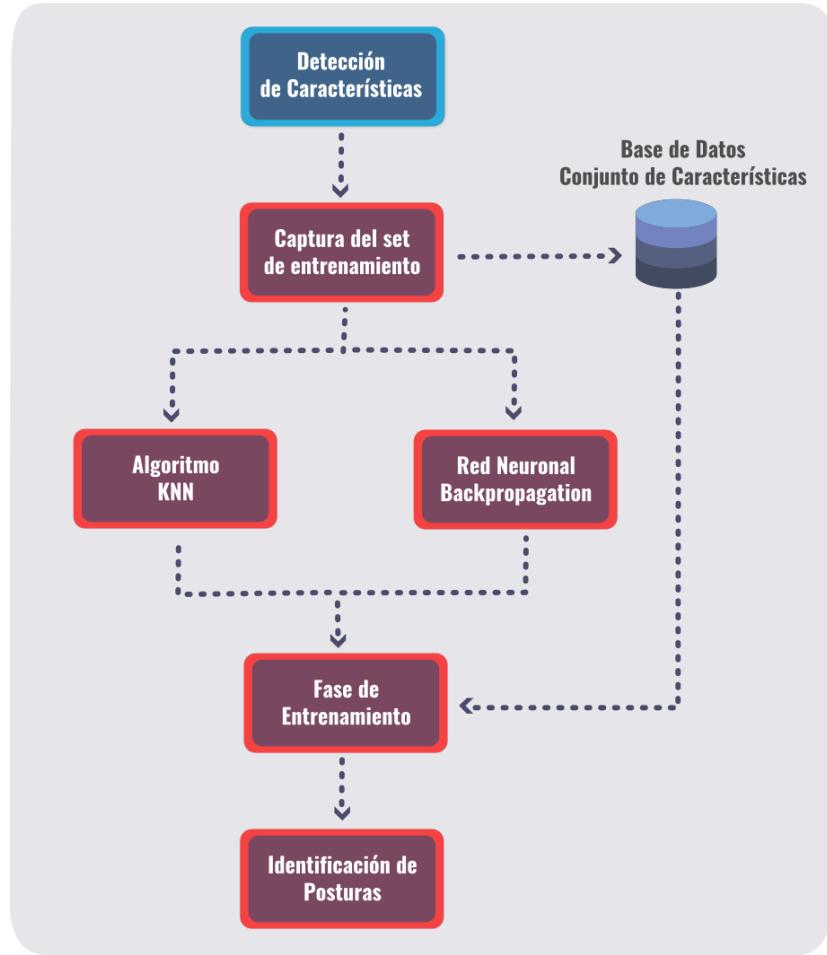


Figura 3-3: Diseño del módulo de Identificación de posturas. Previa captura y guardado de las posturas a identificar, se aplicaran dos métodos de clasificación diferentes para identificar los nuevos datos obtenidos por la fase de detección cuando en sistema se encuentre en funcionamiento.

- Primeramente se hizo uso del algoritmo de clasificación kNN, k-vecinos más cercanos, para etiquetar los datos con su correspondiente posición.
- Para la segunda versión, se optó por crear una red neuronal usando la técnica de backpropagation como algoritmo de aprendizaje, que sería entrenada a partir de los datos obtenidos en la fase de detección.

3.1.3. Módulo de Reconocimiento de Gestos (MRGC)

Después de haber clasificado las posiciones con los métodos, knn y redes neuronales, es hora de interpretarlos para reconocer los gestos corporales. Para ello un gesto corporal lo interpretaremos como una secuencia de posiciones, en nuestro caso

los gestos que vayamos a interpretar estarán compuestos por una secuencia de cuatro posiciones. Esta sucesión de poses serán ingresadas nuevamente como datos en entrada para una segunda clasificación usando kNN y una red neuronal. Con esto se espera una favorablemente identificación con cada uno de los gestos realizados por el usuario. (Figura 3-4)

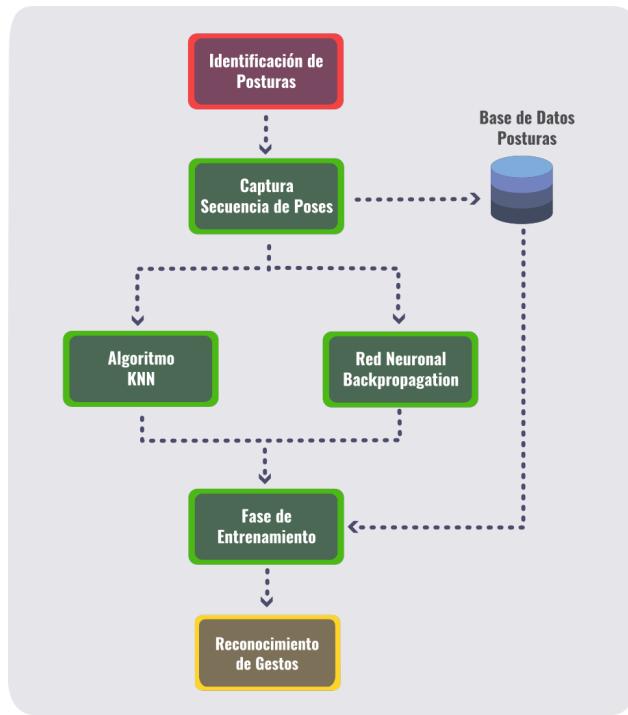


Figura 3-4: Diseño del módulo de Reconocimiento de Gestos. A partir de la captura de poses, se crea un set de entrenamiento que contiene una secuencia de posturas que representa un gesto corporal. Este set de entrenamiento es ingresado a dos clasificadores para su posterior etiquetado.

3.1.4. Sistema Basado en Profundidad

Una vez que distinguimos el funcionamiento básico para cada módulo, podemos detallar el flujo de trabajo entre cada una de las fases que se emplearán en el Diseño General del sistema, esto nos ayudará a tener una mejor perspectiva del funcionamiento completo una vez que nuestro sistema esté completamente implementado.

Como ya mencionamos anteriormente, nuestro sistema cuenta con dos versiones. Una que utiliza una máscara basada en profundidad que nos permitirá la eliminación del fondo y que a una distancia predeterminada seremos capaces de detectar el área de ocupación que genera el usuario respecto a la cámara. A continuación se describirá

con mayor detalle el funcionamiento de este sistema. (Figura 3-5)

1. En la etapa de detección de características se optó por usar las ventajas que proporciona el sensor de profundidad dentro del Kinect. Se pretende establecer una distancia predeterminada, a partir de la cual se generaría un umbral que nos permitiera detectar a las personas dentro del rango establecido.
2. Una vez establecido el umbral de profundidad, generamos una cuadrícula dentro del rango de visión de la cámara del Kinect.
3. Posteriormente se reconoce el porcentaje de ocupación generado por las distintas posiciones del usuario. Estos datos de ocupación se guardarán en una base de datos con su correspondiente etiqueta de la posición a la cual pertenece cada conjunto de datos.
4. Después de la captura de datos, al detectarse un nuevo conjunto de datos, se procede a la clasificación de este usando la base de datos que generamos anteriormente. Es aquí donde se deriva de nuevo en dos partes el sistema. Por un lado se utiliza el algoritmo de clasificación kNN, como ya hemos ido refiriendo, y una red neuronal. La diferencia entre ambos clasificadores es que el kNN lo podemos utilizar inmediatamente después de generar nuestra base de datos, ya que hace las comparaciones en tiempo real. Para poder implementar la red neuronal, hace falta entrenarla con anticipación, ya que dependiendo de la estructura de esta, el entrenamiento puede llevar menor o mayor tiempo.
5. Cuando los dos métodos estén listos para clasificar, identificaremos las posiciones que realice el usuario. Estas posiciones se guardarán en una lista que representará un gesto una secuencia de posiciones. Posteriormente se guardarán en una base de datos al igual que las características anteriores. Esto nos servirá para una segunda clasificación.
6. Esta segunda clasificación tendrá como entrada un vector con una secuencia de posiciones para poder catalogar gestos como saludo o comandos de dirección.

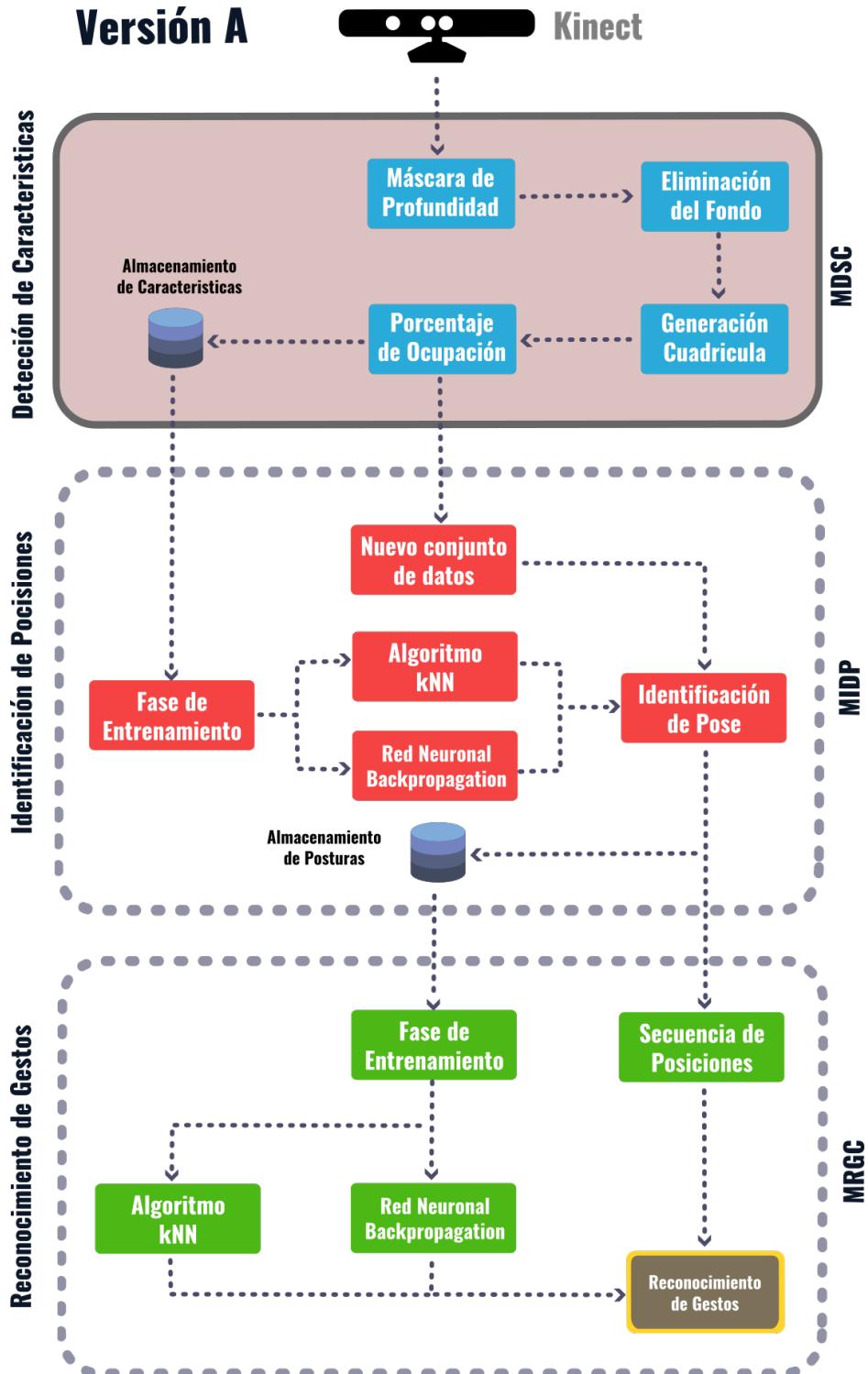


Figura 3-5: Diseño General del Sistema basado en la detección por Profundidad.

3.1.5. Sistema Basado en la detección del Esqueleto Humano

Para la segunda versión de nuestro sistema (Figura 3-6), se pretende mejorar la forma en que detectamos a las personas, para contar con un seguimiento de ellas y evitar así, depender de la posición en la que se encuentre la cámara y demás situaciones que puedan ocurrir con el método anterior. El funcionamiento de este sistemas es básicamente el mismo que el anterior, sólo cambiamos el módulo de detección por profundidad por la de detección de articulaciones y extremidades.

1. Se pretende detectar las articulaciones y extremidades del cuerpo mediante el kinect, obteniendo sus valores de posición respecto a x y y , además de la profundidad en la que se encuentran cada una de ellas, z .
2. Cuando el esqueleto sea detectado de forma correcta, normalizaremos los datos obtenidos para poder realizar un seguimiento adecuado.
3. Y finalmente obtendremos nuestro conjunto de datos como serían los ángulos y/o distancias entre uno o más articulaciones para su posterior en los métodos de clasificación.

Para las demás etapas, Identificación de Posiciones y Reconocimiento de Gestos, se mantendrán con la misma estructura que las del sistema anterior a fin de comparar los datos obtenidos de ambos.

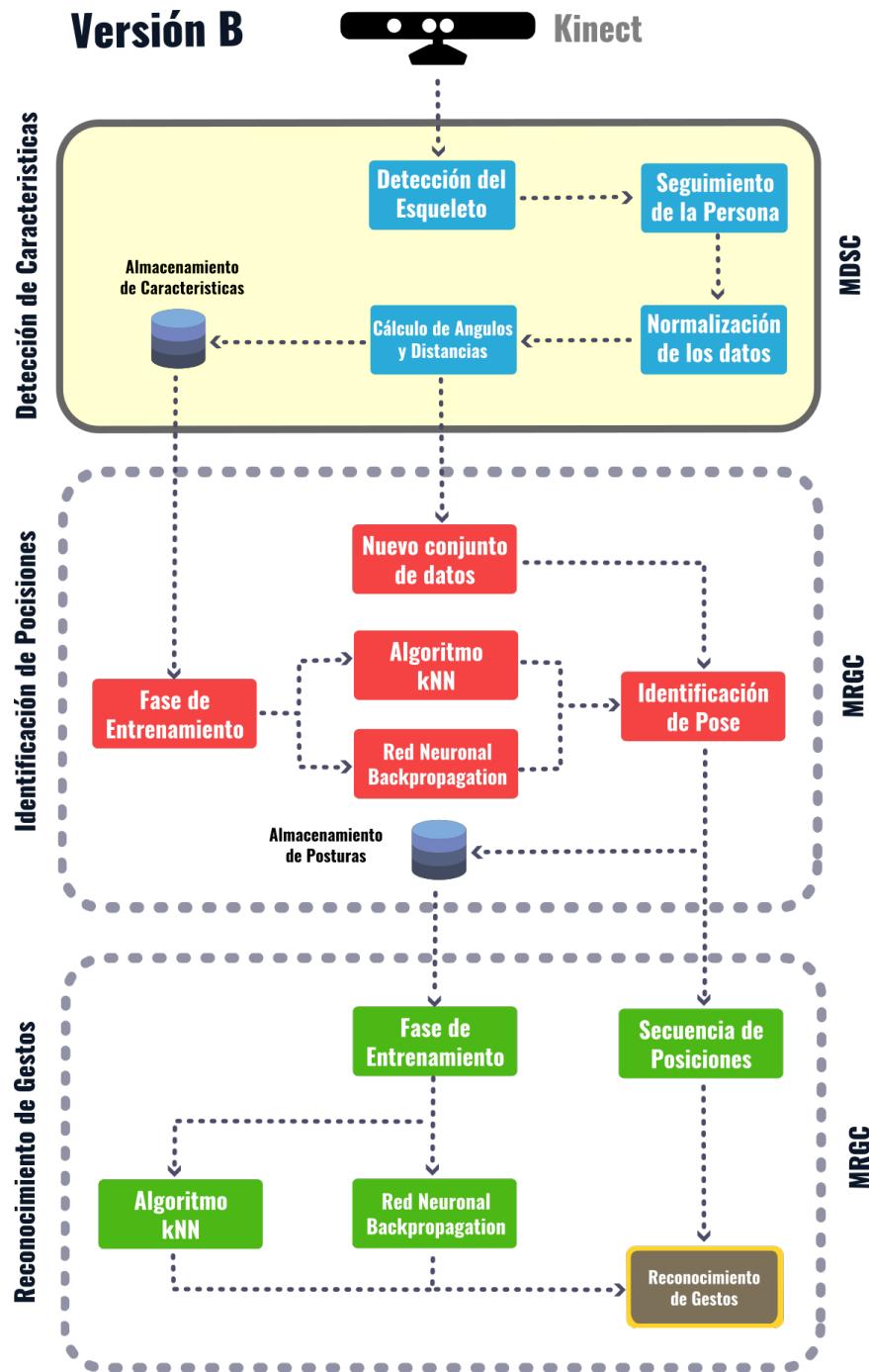


Figura 3-6: Diseño General del Sistema basado en la detección del Esqueleto Humano.

3.1.6. Modelo de la Red Neuronal

En las últimas dos fases de nuestro sistema, donde identificamos las posturas e interpretamos posteriormente una secuencia de estas, señalábamos el uso de redes neuronales como método de clasificación. Para ello se hicieron 2 propuestas en el

modelo de las redes. La primera red neuronal para el sistema basado en mascara de profundidad (Figura 3-7). Tendremos un Capa de entrada de 48 neuronas, estas 48 neuronas (6×8) representan el numero de cuadros en las que se dividirá la escena captura por el Kinect. Cada neurona recibirá un valor entre 0 y 100 correspondiente al porcentaje que ocupa el área generada por el usuario después de haberse aplicado el umbral de profundidad. Dentro de la capa oculta de la red, utilizaremos 5 neuronas. Y finalmente para la capa de salida utilizaremos 4 neuronas. Estas 4 neuronas representaran el numero de la posición correspondiente en binario (e.g. Posición número 4 será representado como [0,1,0,0]).

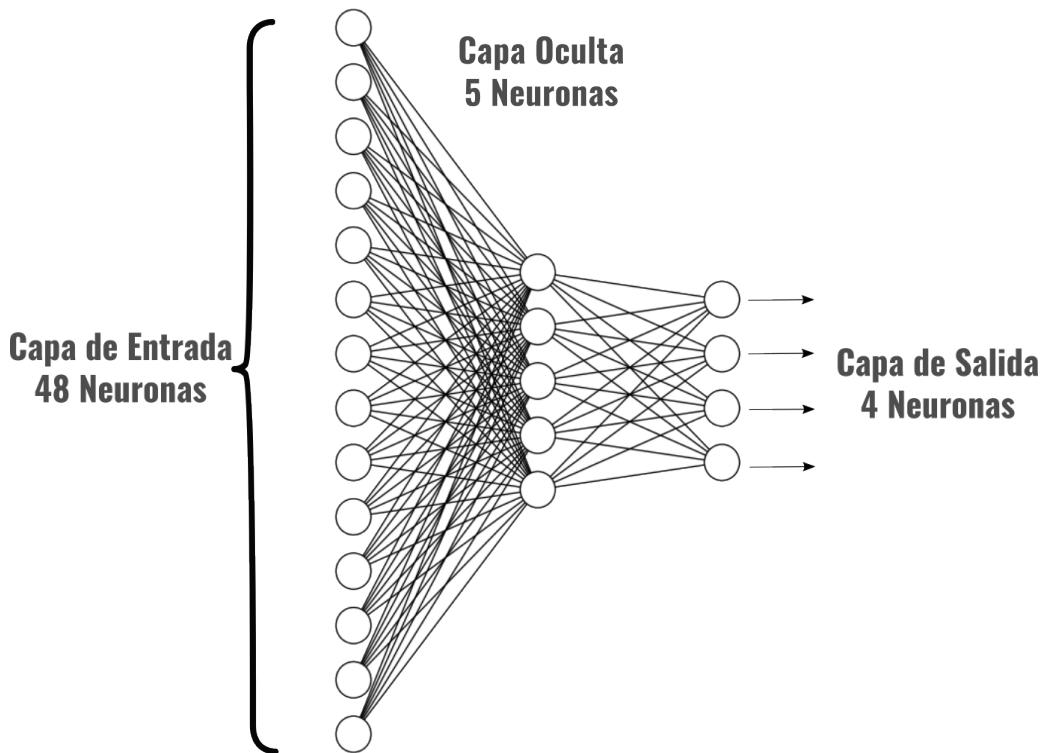


Figura 3-7: Modelo de la primera red Neuronal para los datos obtenidos por la región de Ocupación.

No existe ninguna regla general que especifique el número de capas que una red neuronal deba tener. En este primer modelo optamos por una red neuronal piramidal, en el que hay una mayor cantidad de entradas y va disminuyendo conforme se avanza por las capas (48-5-4). Esto en teoría debería ser suficiente para resolver nuestro problema. Sin embargo para tener más seguridad en las respuestas esperadas, se

propone un segundo modelo (Figura 3-8) donde aumentamos el número de nodos dentro de la capa oculta (96). Dado que en esta capa es donde ocurre el “aprendizaje” de la red neuronal, se infiere que este modelo puedan ser mejor que el anterior, sin embargo también hay que tomar en cuenta que un número mayor de neuronas dentro de la capa oculta aumenta el tiempo de entrenamiento y en ocasiones puede provocar que el gradiente se vuelva inestable provocando que la red colapse o se entrene de manera errónea.

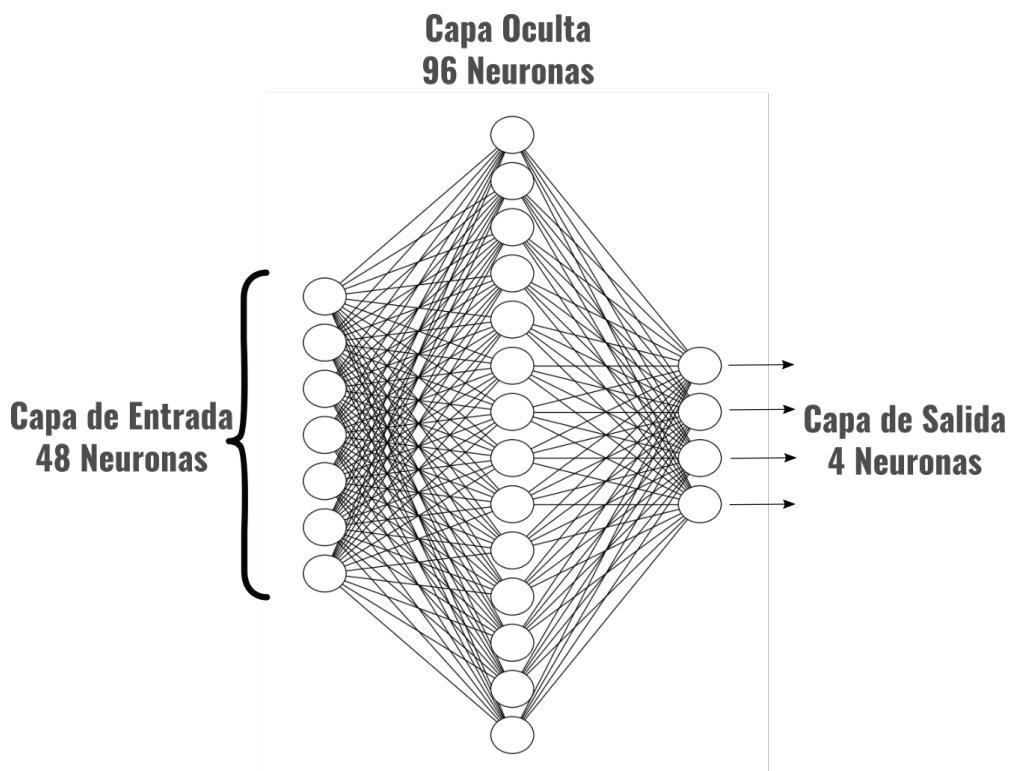


Figura 3-8: Modelo de la segunda red Neuronal para los datos obtenidos por la región de Ocupación con una mayor cantidad de neuronas en la capa oculta.

Para nuestro segundo sistema en el que utilizamos la detección mediante la generación de un modelo óseo del cuerpo, se propone un modelo de 14-5-4 (Figura 3-9), para las capas de entrada, oculta y salida respectivamente. Más adelante en la sección de implementación de este trabajo de tesis, se hablará con mayor detalle cuales son los datos de entrada para esta red neuronal.

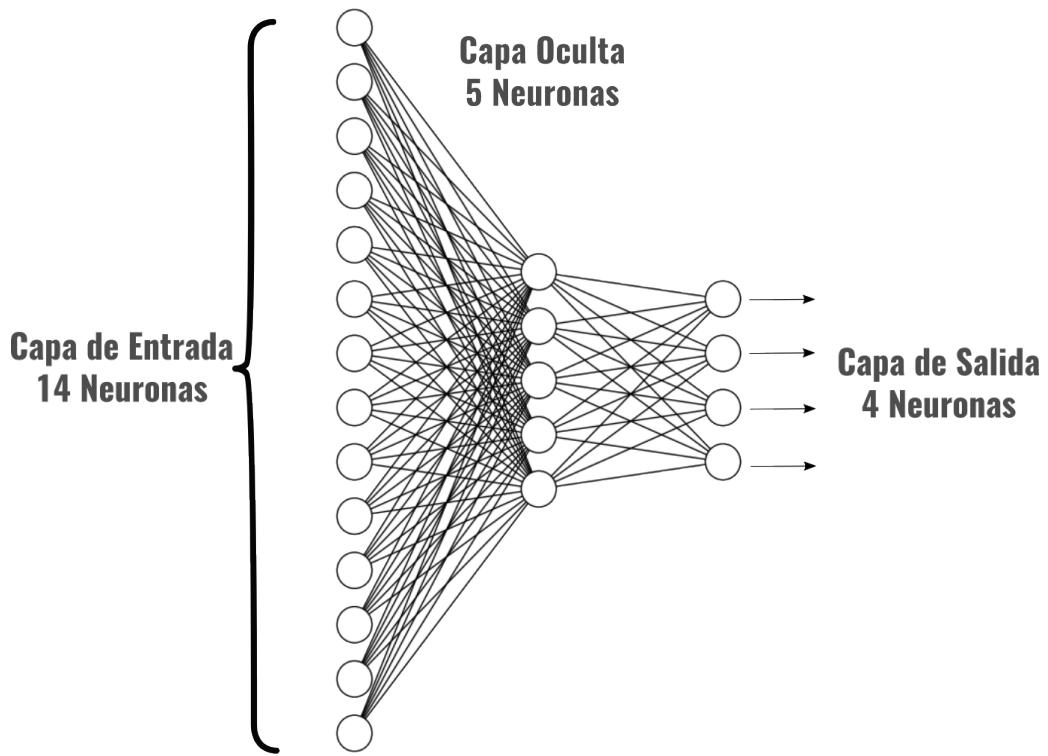


Figura 3-9: Modelo de la red neuronal para la detección de posiciones usando las características obtenidas a partir de puntos 3D en un esqueleto.

De la misma forma que las redes anteriores, se propone un segundo modelo de red neuronal (Figura 3-10) en la que la capa oculta cuenta con un mayor número de nodos que las capas de entrada y salida.

Y finalmente, después de la detección de posiciones y su correcta clasificación, es tiempo de identificar gestos corporales. Para ello, ocuparemos nuevamente una red neuronal. Conformada por 4 entradas, 5 nodos en la capa oculta y 2 salidas. Las 4 entradas serán una secuencia de posiciones que juntas generan un gesto. Es decir, supongamos que tenemos la combinación de la posiciones 1, 2, 3, 1. Esto representaría el gesto número 1 ([0,1]) Que representaría el gesto de mover la mano hacia enfrente y atraerla al cuerpo: “Ven”.

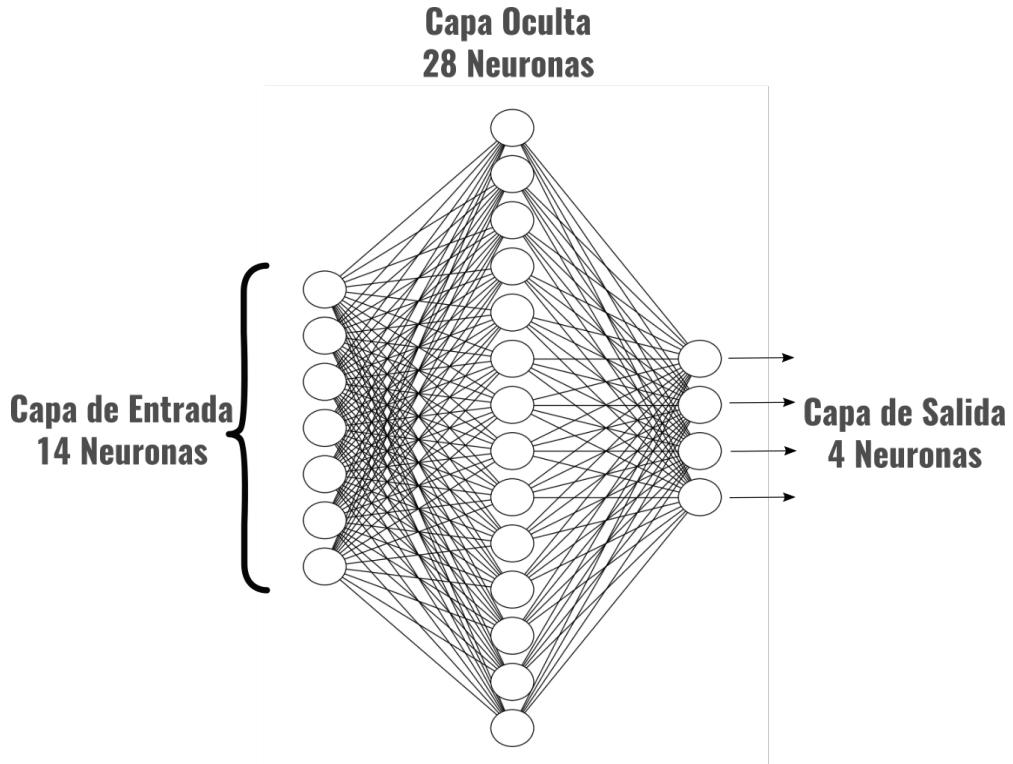


Figura 3-10: Modelo de la segunda red neuronal en base a las características del esqueleto aumentando el número de nodos en la capa oculta.

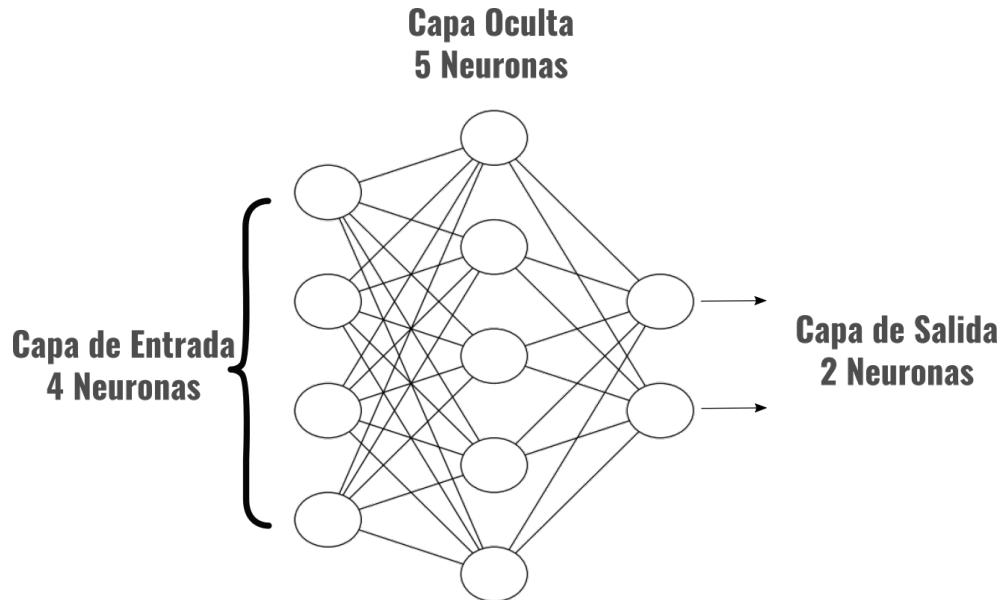


Figura 3-11: Modelo de la red neuronal para la detección de gestos corporales a partir de una secuencia de posiciones.

Para la clasificación mediante el algoritmo kNN se utilizaron las mismas entradas que las redes neuronales. Como en nuestro caso hicimos uso del método incluido en

la librería OpenCV no es necesario crear un modelo del algoritmo, simplemente se utilizará para comparar con los datos resultantes de las redes neuronales.

3.2. Análisis de Requerimientos

Una vez que hemos completado el diseño con el cual desarrollaremos el sistema, debemos especificar todos los elementos y herramientas utilizadas para desarrollar el sistema de detección de gestos corporales presentado en este trabajo de tesis para su posterior implementación.

3.2.1. Sistema Operativo

El sistema de reconocimiento se desarrolló sobre el sistema operativo Ubuntu 14.0.3. Ubuntu es un sistema operativo basado en GNU/Linux y que se distribuye como software libre, el cual incluye su propio entorno de escritorio [página ubuntu]. Además de contar con el intérprete de python 2.7 preinstalado que es el lenguaje de programación utilizado en este proyecto.

La versión en específico de Ubuntu 14.04.3 se utilizó debido a que una de las librerías utilizadas en el sistema, de la cual platicaremos más adelante, se encuentra actualmente sin soporte, y en versiones más recientes del sistema, tiende a fallar la instalación de dicha librería.

Los requisitos mínimos esenciales para ejecutar Ubuntu 14.04 son los siguientes:

- Procesador Intel x86 o compatible a 1000 Mhz o 1 GHz.
- Mínimo 512 Mb de memoria RAM.
- 10 GB de disco duro (swap incluida).
- Tarjeta de gráfica VGA y monitor compatibles con una resolución de 1024 x 768

3.2.1.1. Arquitectura General

En nuestro trabajo de Tesis se utilizaron dos equipos. Uno en el cual corre el sistema ya en funcionamiento y otro con mayor capacidad donde se realizó el entre-

namiento de las redes neuronales. El equipo utilizado en el cual se implementó el prototipo del sistema de detección de gestos corporales contaba con las siguientes características:

- Procesador Intel Core i3 2.53GHz x 4.
- 4 GB de memoria RAM.
- Una partición de 80 GB de disco duro (swap incluida).
- Tarjeta gráfica VGA y monitor compatibles con una resolución de 1024 x 768

Cabe aclarar que dada la naturaleza de los sistemas basados en visión artificial, estos tienden a consumir bastantes recursos del computador. Por lo que es muy probable que con los requerimientos mínimos solicitados para la instalación del sistema operativo no sean los suficientes, provocando que el sistema de reconocimiento no funcione adecuadamente. Es por eso que se recomienda utilizar un equipo que cuente mínimamente con un procesador Intel i3 o equivalente y/o mayor para una correcta ejecución del sistema.

Para el entrenamiento de las redes neuronales uso un computador de mayor capacidad que la del sistema debido a que el entrenamiento de las redes neuronales demanda una gran cantidad de procesamiento. De esta forma podíamos disminuir los tiempos de entrenamiento permitiéndonos así probar distintos parámetros.

- Procesador Intel Core i7 3.5GHz x 8
- 8 GB de memoria RAM.
- Una partición de 200 GB de disco duro (swap incluida).
- Tarjeta gráfica NVIDIA GeForce GTX 960M 4GB

3.2.2. Kinect

Otro de los elementos indispensables empleados en este trabajo es el Kinect para Xbox 360, o simplemente Kinect. Es un controlador de juego libre y entretenimiento creado por Alex Kipman, desarrollado por Microsoft ©para la videoconsola Xbox

360. Es un sensor RGB-D que provee una imagen sincronizada de color y profundidad. Con un algoritmo de captura de movimiento humano en 3D, es capaz de crear una interacción entre el usuario y un juego sin la necesidad de utilizar algún control.



Figura 3-12: Configuración del hardware dentro del Kinect, Ejemplos de las imágenes obtenidas por la cámara RGB y el sensor de profundidad respectivamente,[E2]

Recientemente, el área de visión artificial descubrió el gran potencial de este dispositivo con la tecnología de percepción de profundidad que proporciona para ser aplicado más allá de un controlador de videojuegos, aunado al hecho de su fácil adquisición y bajo costo comparado con otros sensores de profundidad. Es precisamente por estas últimas dos características, y la ventaja de contar con un sensor de profundidad, que ocupamos el Kinect como mecanismo principal para la obtención de los datos necesarios para la detección de personas.

El Kinect (Figura 3-12) cuenta una cámara RGB con resolución de 640 x 480 pixeles. Asimismo como ya mencionábamos, con un sensor de profundidad que está compuesto por una cámara y proyector infrarrojo. El proyector infrarrojo es el responsable de disparar rayos infrarrojos al ambiente. Los ángulos de distorsión de cada uno de los rayos proyectados son usados para calcular un mapa de profundidad en el que cada pixel representa la distancia específica desde el sensor.

El Kinect además para su correcto funcionamiento junto con la computadora necesita de un adaptador con una salida USB para conectarse a esta. Este mismo adaptador provee una conexión a corriente para suministrar los 12v necesarios para ponerlo en marcha ya que la computadora mediante la interfaz USB sólo provee 5v.

3.2.3. OpenCV

Quizás una de las herramientas más importantes para este proyecto de investigación sea sin duda OpenCV (Open Source Computer Vision Library). La cual es una librería de código abierto para visión artificial y máquinas de aprendizaje desarrollada por Intel. La versión utilizada para el proyecto fue la 2.4.12. OpenCV que ofrece una gran cantidad de métodos y funciones para el tratamiento de imágenes, y es sin duda una de las librerías más utilizadas en trabajos relacionados con visión artificial. Al contar con una licencia BSD, se permite su libre uso para trabajos comerciales o de investigación.

El uso principal dado en este trabajo de investigación, fue el de procesar los datos obtenidos mediante el Kinect, para su correcta interpretación, y posterior procesamiento, haciendo uso de varios filtros para la limpieza de los datos, sistema de coordenadas y métodos de clasificación incluidos dentro de la misma librería como lo son el kNN que fue uno de los algoritmos de clasificación que usamos en el sistema.

3.2.4. Libfreenect

Como ya mencionábamos, nuestro principal canal de datos será a través del Kinect, es por eso que se necesita un controlador capaz de interactuar con el mismo. LibFreenect o también conocida como OpenKinect es una librería de código libre desarrollada por una comunidad de personas interesadas en hacer uso del hardware del Kinect de Xbox con computadoras u otros dispositivos.

Libfreenect nos proporciona las herramientas básicas para el manejo adecuado del Kinect, como son la obtención de la imagen RGB y de profundidad (Figura 3-13).



Figura 3-13: Imagen adquirida mediante la librería Libfreenect que representa mediante colores la distancia de los objetos respecto al kinect.

3.2.5. OpenNI y NITE

OpenNI, (Open Natural Interaction), es una organización sin ánimo de lucro impulsada por la industria centrada en la certificación y mejora de la interoperabilidad de la interfaz natural de usuario y la interfaz orgánica de usuario para dispositivos de interacción natural.

OpenNI fue creado para ayudar a desarrollar aplicaciones que utilicen la visión en 3D como principal información base (e.g. control a través del reconocimiento del cuerpo). Uno de los principales contribuyentes a este programa fue PrimeSense, que fue la empresa que creó la tecnología utilizada en el Kinect. Sin embargo, tiempo después PrimeSense liberó sus propios drivers de código abierto junto con un middleware de detección de movimiento denominado NITE[referencia nite]. Aun a pesar del éxito y progreso que fue ganando estas librerías para la detección del cuerpo humano, tras la adquisición de PrimeSense por Apple, la página principal de OpenNi fue dada de baja, y el mantenimiento de la librería fue descontinuado. Aun a pesar de eso algunas otras organizaciones conservaron la documentación y códigos fuentes para su uso posterior.

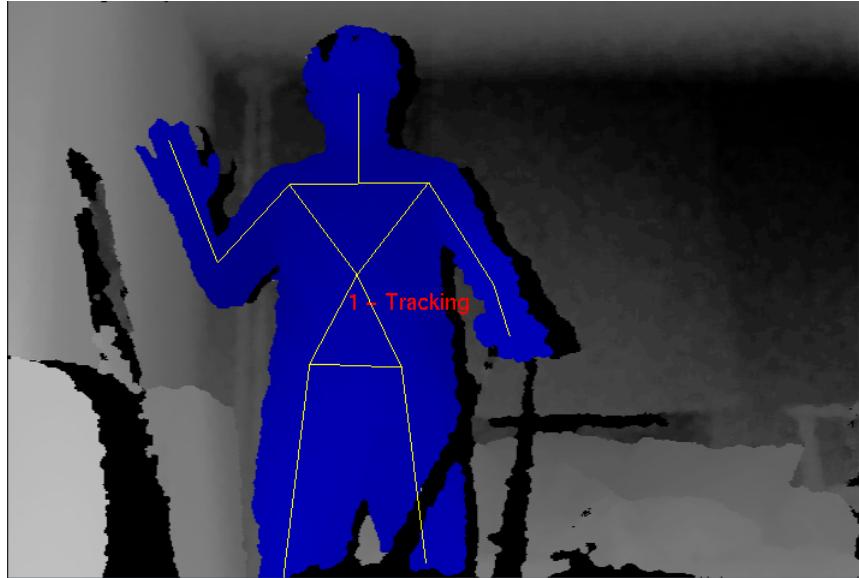


Figura 3-14: Ejemplo de detección del esqueleto humano haciendo uso de la librería OpenNI,[E3]

OpenNI junto con NITE fueron utilizados para la segunda parte del diseño de nuestro sistema de reconocimiento de gestos, ya que nos brinda una detección más clara y precisa de la anatomía corporal en entornos que podrían ser inconvenientes si se aplicaran otros métodos de detección, como los que se abordan en la primera fase de diseño.

Las versiones utilizadas en la integración del sistema son v1.5.4.0 y v1.5.2.21 para OpenNI y NITE respectivamente. Además, se utilizó de igual forma la librería PyOpenNI que integra y proporciona portabilidad al lenguaje de programación python.

3.2.6. Python

Python es un lenguaje de programación interpretado y multiparadigma, que soporta la programación orientada a objetos, programación imperativa y programación funcional. Pese a que fue creado a finales de los ochenta por Guido van Rossum, no fue hasta hace algunos años que comenzó a ganar bastante popularidad, esto debido a su filosofía de fácil lectura debido a la sintaxis que maneja, y por ser un lenguaje de sencillo entendimiento e implementación. Permitiendo de esta forma un desarrollo más rápido y sencillo a la hora de la creación de algoritmos.

3.2.7. Nimblenet

Nimblenet es una librería desarrollada por Jorgen Grimnes en Python usando la librería Numpy para el procesamiento de datos, que implementa una red neuronal lista para su uso y que utiliza distintos algoritmos de aprendizaje como backpropagation, resilient backpropagation y scaled conjugate gradient. La librería permite su uso y modificación siempre y cuando se haga mención del autor y de la licencia de Copyright.

Capítulo 4

Implementación y Resultados

4.1. Implementación del sistema de reconocimiento de gestos corporales

Después de haber expuesto los requerimientos y herramientas necesarias para la realización de este sistema de reconocimiento de gestos corporales, así como el diseño del mismo, es momento de pasar a su implementación. En el capítulo anterior definimos el diseño del sistema, sin embargo este no fue el planteamiento inicial ya que tanto diseño como implementación se fueron trabajando en conjunto haciendo las modificaciones necesarias. Con cada nueva iteración del sistema, obteníamos datos que nos ayudaban a mejorarlo, esto junto con una investigación más profunda de trabajos relacionados nos daban más herramientas e ideas para hacer evolucionar el sistema. La implantación del sistema está dividido en dos partes principales de acuerdo a los cambios significativos entre una versión y la otra. A continuación explicaré con más detalle el desarrollo del sistema y cuáles fueron las mejoras utilizadas en cada una de las partes así como los resultados obtenidos en cada uno de ellos.

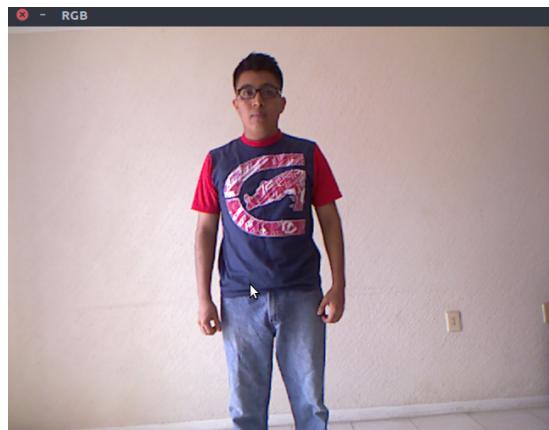
4.2. Detección de Características

Para poder identificar de manera correcta las posturas y gestos que realicé de una persona, es necesario una buena captura de datos, ya que a partir de estos se generará el análisis correspondiente, y si los datos obtenidos no son los adecuados,

muy posiblemente nuestro sistema de reconocimiento no funcionará adecuadamente.

4.2.1. Máscara de detección basada en profundidad

Como ya planteábamos en nuestro diseño, la primera aproximación que tuvimos para la detección de características que se pudieran traducir como posiciones, fue la de crear un umbral de profundidad que nos permitiera calcular el área que ocupa un individuo respecto a la cámara. Aprovechando los sensores con los que cuenta el Kinect, se obtiene una imagen de profundidad (Figura 4-1b) a la cual, después de definir una distancia prudente a partir del Kinect, aplicamos el filtro de profundidad a la imagen de profundidad obtenida, teniendo como resultado una silueta (Figura 4-1c).



(a) Imagen RGB



(b) Imagen de Profundidad



(c) Imagen Obtenida al aplicar el filtro de profundidad

Figura 4-1: Imágenes obtenidas mediante el Kinect. (a) Imagen RGB. (b) Imagen de Profundidad a escala de grises. (c) Aplicación del Filtro de Profundidad.

Como se pudo observar la silueta obtenida después de aplicar el umbral de profundidad es bastante aceptable. El usar este filtro nos beneficia en gran medida, debido a que no dependemos de la imagen RGB que podría proporcionar cualquier cámara de video. Una imagen RGB es complicada de tratar, existen muchas variables como la iluminación, que afecta la tonalidad de los colores dependiendo de esta y puede resultar difícil de normalizar en ciertos casos. Cuando delimitamos la distancia en la que nuestro Kinect comienza a detectar, eliminamos todos los problemas que dependen de una imagen normal a colores, ya que gracias al sensor del Kinect, detectamos los objetos respecto a la distancia que se encuentran, y que favorece para eliminar elementos innecesarios en la escena (Figura 4-2b).

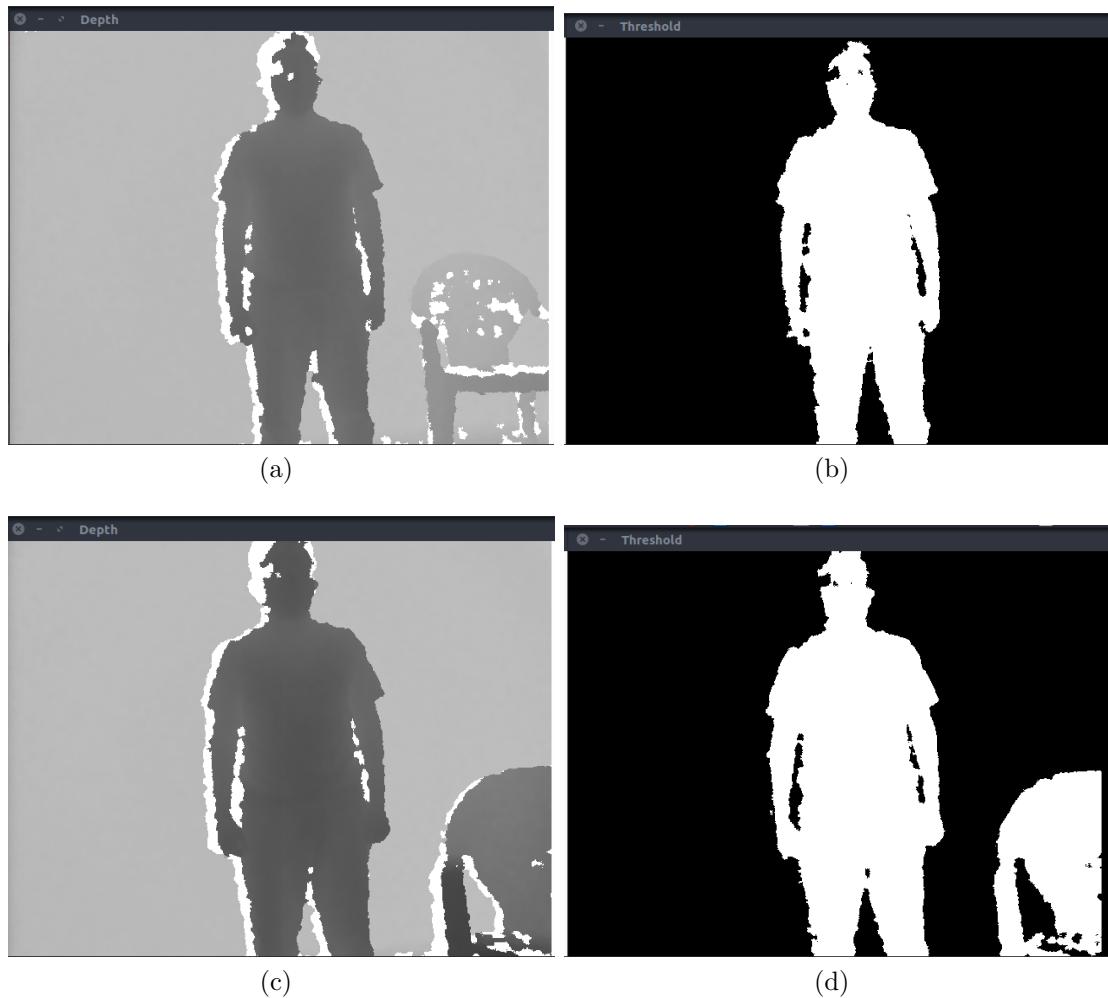


Figura 4-2: (a) y (b) Detección de dos objetos a diferentes profundidades. (c) y (d) Dos objetos detectados a la misma profundidad occasionando la obtención de mas de una silueta.

Aun cuando la eliminación del fondo nos ayuda sustancialmente, existe un inconveniente en el cual si un objeto, u otra persona entra del rango de detección, este afectará los datos que obtengamos ya que se modificará la silueta obtenida afectando en gran medida de los datos obtenidos (Figura 4-2d). Es por eso que al usar este método supondremos el hecho de que solamente el usuario se encontrará dentro del rango de detección.

4.2.1.1. Regilla de Ocupación

Una vez que implementamos nuestro algoritmo de eliminación de fondo basado en profundidad, es tiempo de analizar los datos obtenidos mediante la silueta generada. Para ello, la imagen obtenida que tiene por medidas 640 x 480 pixeles, se dividió en 48 celdas, 6 filas y 8 columnas. Cada celda tiene una medida de 80px de cada lado (Figura 4-3).

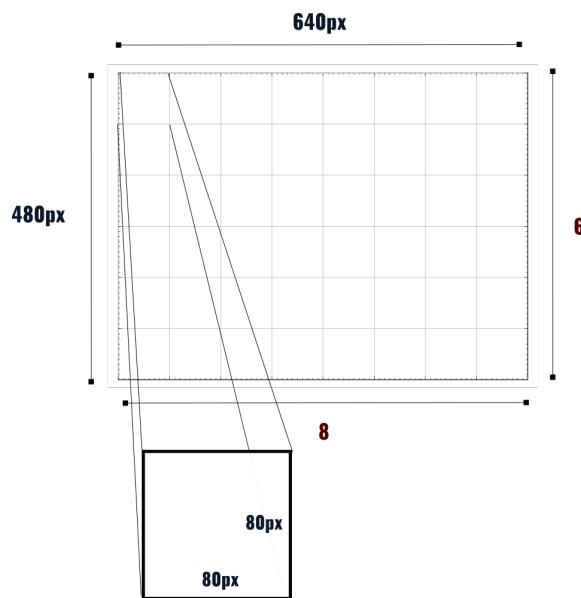


Figura 4-3: Ejemplificación de cómo se dividió la escena capturada en celdas.

Se decidió utilizar estas proporciones ya que la pantalla quedaba dividida en un buen número de celdas. Un mayor número de celdas no aseguraba un mejor reconocimiento de las características, pero si disminuíamos la cantidad de estas, claramente se observaba una deficiencia considerable en los datos conseguidos.

El resultado de la anterior división se puede observar en la Figura 4-4b.

Cada una de estas celdas contendría un número de pixeles blancos y negros,

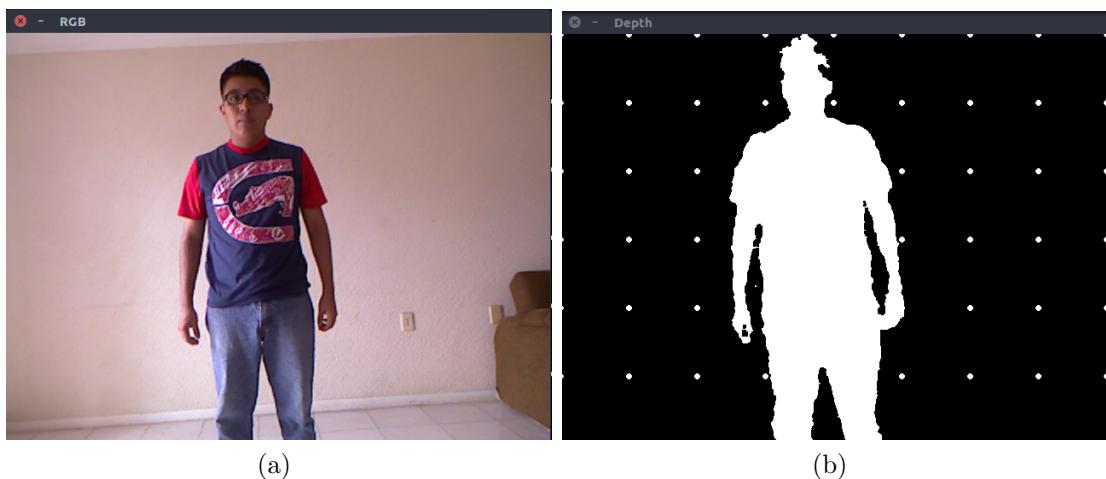


Figura 4-4: (a) Imagen original. (b) Imagen con el filtro de profundidad y la representación de las celdas.

donde blanco es el área detectada y el negro sin detectar. Para cada celda creada, se calculará el porcentaje de ocupación de acuerdo al área que se encuentre detectada respecto al área total de cada celda (Figura 4-5). Se contabilizarán los píxeles de color blanco y se dividirán entre el número total de píxeles por cada celda. Los valores comprenderán un valor desde 0 a 100.

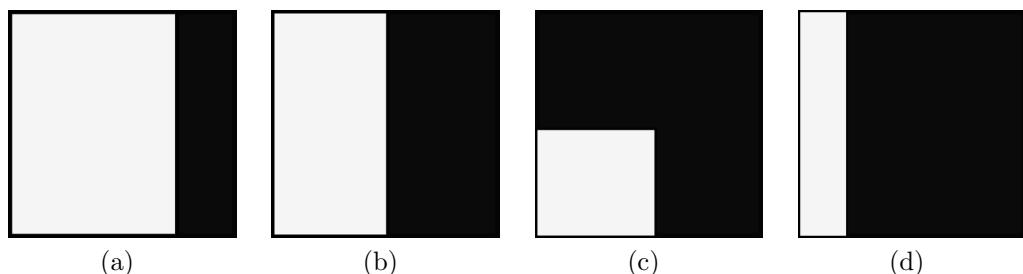


Figura 4-5: Ejemplos de porcentaje de ocupación. (a) 80 %, (b) 50 %, (c) 25 % y (d) 20 %.

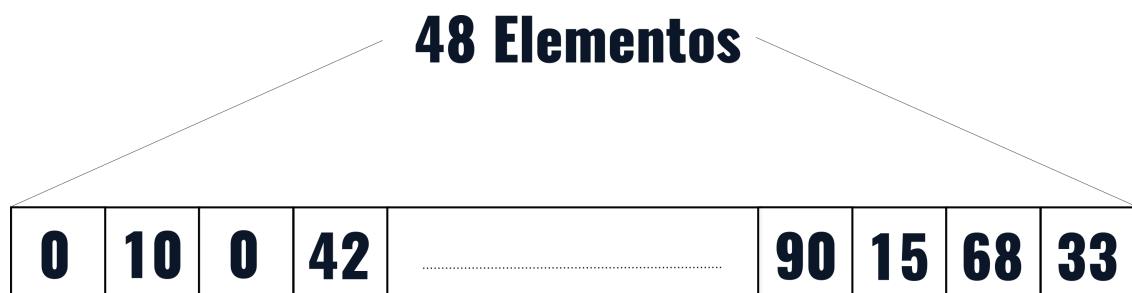


Figura 4-6: Ejemplificación de la lista generada con los porcentajes de ocupación.

Nuestro primer algoritmo, recorrerá en tiempo real cada una de estas celdas, que en visión artificial las conocemos como Región de Interés (ROI Region of interest en inglés), en un ciclo que nos permita calcular cuánto es el porcentaje que ocupa el área detectada en cada una de estas celdas. Estos valores se introducirán en una lista para su posterior análisis y almacenamiento (Figura 4-6).

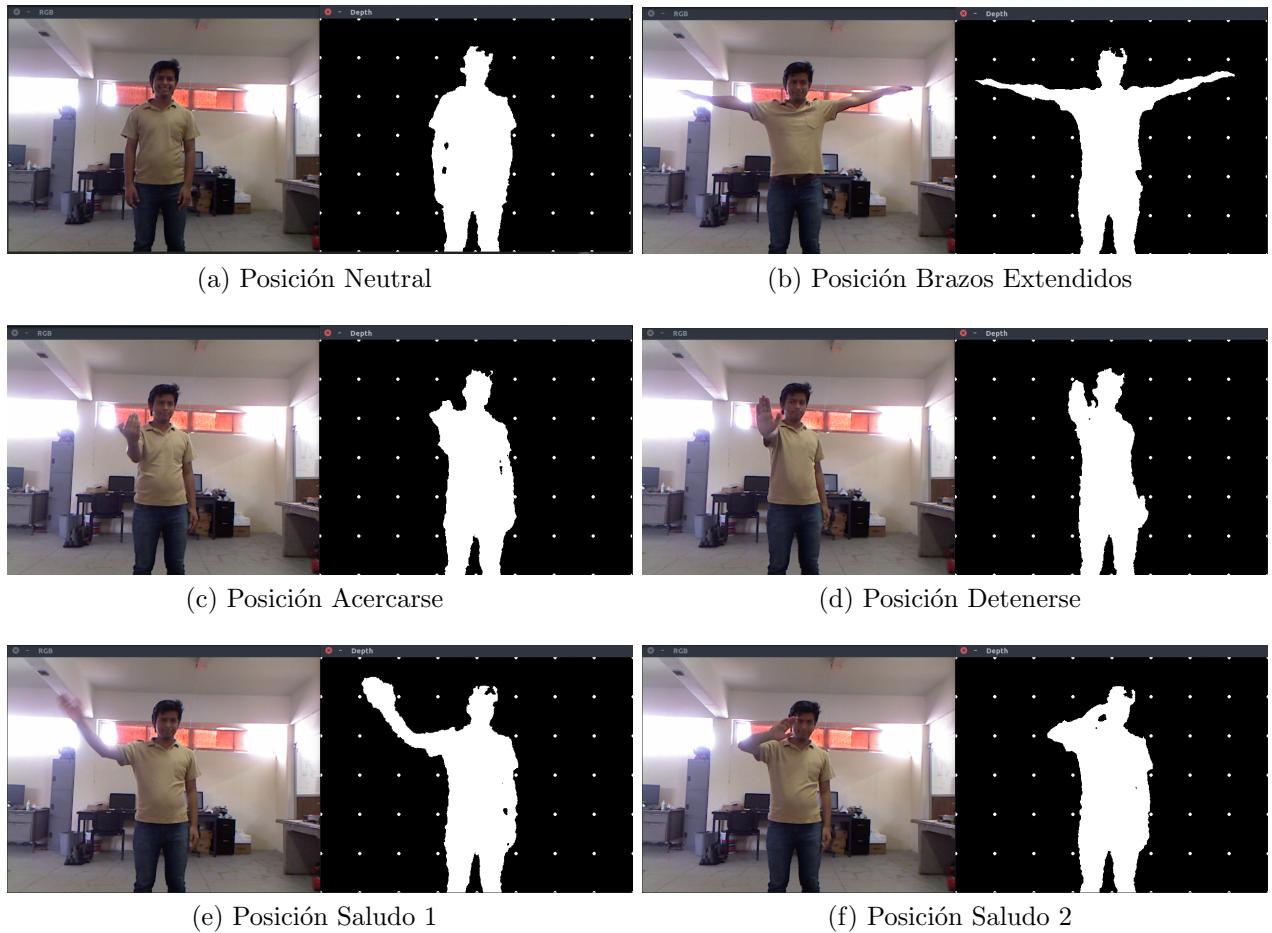


Figura 4-7: Algunas de las poses capturadas utilizando el umbral de profundidad y la rejilla de ocupación

Estos conjuntos de datos, serán capturados antes de comenzar con la clasificación ya que para que esto sea posible necesitamos tener un conjunto de entrenamiento con el cual posteriormente podamos comparar e identificar las posiciones que el usuario realice. Para ello se realizó la captura de 10 personas distintas realizando distintas posiciones que deseamos identificar (Figura 4-7), generando así 900 conjuntos de datos que se almacenaron con su respectiva etiqueta en un archivo para la futura consulta de los métodos de clasificación que más adelante emplearemos.

A pesar de que esta propuesta pueda parecer efectiva, ya que como se observa las áreas ocupadas entre posiciones varían respecto una de otra, existen posiciones que incluso nosotros si sólo vemos las siluetas generadas, podríamos confundir una de otra. En el ejemplo de la Figura 4-8 observamos dos posiciones realizadas por un usuario. Estas dos siluetas generadas por nuestro filtro de profundidad pueden parecer ambiguas, ya que si bien en las dos imágenes podemos apreciar que se encuentra una mano levantada, no podemos diferenciar si representa a la posición de alto, acercarse o inclusive un saludo.

No obstante los resultados de los clasificadores determinarán si es viable o no utilizar este método de detección.

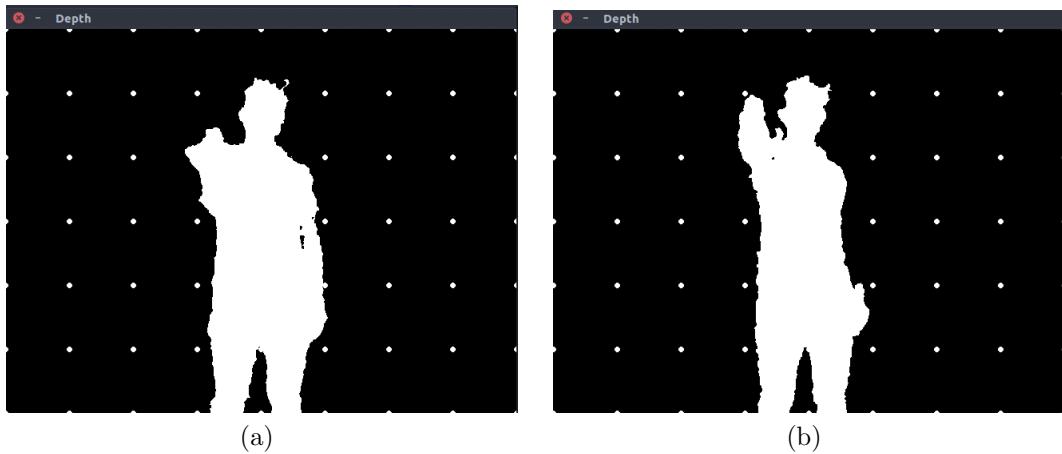


Figura 4-8: Ejemplo de dos posturas similares.

4.2.2. Modelo del Esqueleto Humano

Durante esta primera etapa que consiste en identificar características de interés, necesarias para comprender las actividades o gestos que realice el usuario, necesitamos seguir el movimiento de las partes del cuerpo que intervienen en la ejecución de estos movimientos. Con el anterior método esto es ignorado, ya que sólo nos centramos en la forma general de cada postura, y no identificamos qué partes del cuerpo intervienen para ello. Por consiguiente para describir el movimiento humano con más detalle, se decidió emplear un seguimiento del esqueleto humano haciendo énfasis en las articulaciones y extremidades más importantes como lo son la cabeza, cuello, torso, brazos y piernas. Estas diferentes partes del cuerpo serán modeladas como segmentos de líneas conectadas entre sí por puntos que representarán la articulaciones, y que representarán al cuerpo en un espacio tridimensional (Figura 4-9).

Para obtener estos puntos 3D, utilizaremos nuevamente el sensor del Kinect en conjunto con la librería OpenNI que nos proporcionará una interfaz capaz de brindarnos la información necesaria para la construcción del modelo óseo ofreciéndonos información en x , y y z .

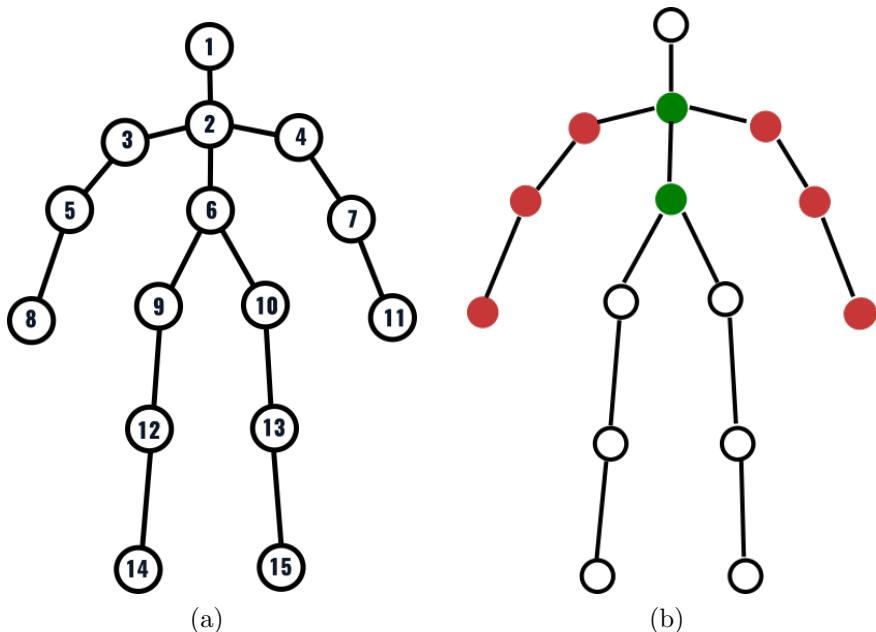


Figura 4-9: (a) Detección de 15 articulaciones detectadas por el Kinect. (b) Rojos: Manos, Codos y Hombros. Verde: Cuello y torso. Puntos descartados: Cabeza, caderas, rodillas y pies

4.2.2.1. Distancia y ángulos entre Articulaciones

Tan pronto hayamos obtenidos los datos de todas las articulaciones, es momento de definir cuáles de estos son relevantes para nuestro propósito. De las posiciones que hemos elegido reconocer, podemos darnos cuenta que en su totalidad dependen del movimiento ocasionado de los brazos. Tomando en cuenta esta situación, podemos resolver que son datos relevantes para nuestro estudio. De igual forma, sucede con los puntos representados para la cabeza y las caderas, ya que se mantienen inmóviles ante cualquier tipo de movimiento. Quedando entonces como puntos importantes los que comprenden los brazos y la parte media del cuerpo (Figura 4-9b).

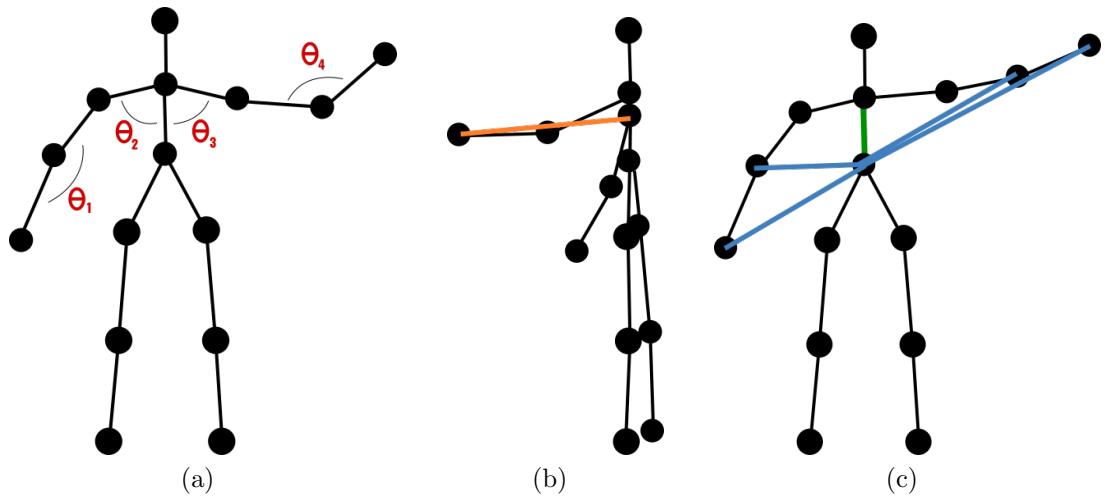


Figura 4-10: (a) Ángulos entre las articulaciones de los brazos. (b) Distancia relativa de las extremidades respecto al cuerpo. (c) Distancias normalizadas con la escala obtenida del torso.

Debido además de que la apariencia del esqueleto depende de muchos factores, entre ellos, la distancia entre el usuario y el sensor, los datos necesitan ser normalizados. Los datos obtenidos x , y y z donde x es la posición respecto a la cámara en el eje horizontal y y el eje vertical varían dependiendo de qué tan alejados estemos del Kinect. Si bien OpenNI nos proporciona un seguimiento de la persona por toda la escena, los datos en bruto no son de utilidad si quisiéramos, por ejemplo, medir la distancia entre dos articulaciones como característica de una pose. De manera que para conseguir una medida que se mantenga constante sin depender en dónde se encuentre la persona, podemos recurrir a la distancia entre el cuello y el torso. Esta distancia puede ser calculada en tiempo real para que se convierta en nuestra escala

de referencia, generando así nuestra propia unidad de medida que podemos utilizar en todo momento para medir la distancia existente entre el torso y las articulaciones de los brazos (Figura 4-10c). Otra característica importante que consideramos es el ángulo que generan las articulaciones de los brazos al mover los brazos y realizar una pose. Consideramos cuatro ángulos principales: los dos ángulos generados en el codo al mover el antebrazo respecto al brazo, y los ángulos generados entre el torso y el brazo (Figura 4-10a). Por último, cuantificamos otro dato que podría considerarse de los más importantes entre todos, nos referimos a la profundidad de cada punto (Figura 4-10b). Si queremos diferenciar la pose de saludo cuando la mano se encuentra frente al pecho y la pose de alto, el dato de profundidad de cada punto resulta bastante relevante para ignorarlo. En la primera posición (Saludo), la mano se encuentra a pocos centímetros del pecho, mientras que para la pose de alto, la mano se encuentra al frente de nosotros a casi un metro de distancia. Sin embargo, al igual que ocurría con los datos en bruto proporcionados en x y y , para z los datos originales varían respecto a la cámara y el valor varía en miles debido a que se nos da la distancia en milímetros. Para ello normalizamos estos datos utilizando como punto de referencia absoluto a la profundidad del punto correspondiente al torso. De esta forma para calcular la profundidad de las articulaciones correspondientes a los brazos, restamos el valor actual con el valor generado por el torso, obteniendo así una distancia relativa que inclusive determinaría si los brazos se encuentran detrás del cuerpo si el valor final es negativo. La organización de estos datos se muestra en la Figura 4-11



Figura 4-11: Datos de interés que se obtendrán a partir del modelo óseo

De la misma forma que en el método del filtro de profundidad, una vez que determinamos los datos que debemos capturar, es hora de crear un conjunto de entrenamiento. Se les solicitó ayuda a 10 personas distintas para la captura de estos datos, generando de esta forma un set de entrenamiento con 1400 datos incluyendo 8 posturas distintas (Figura 4-12)

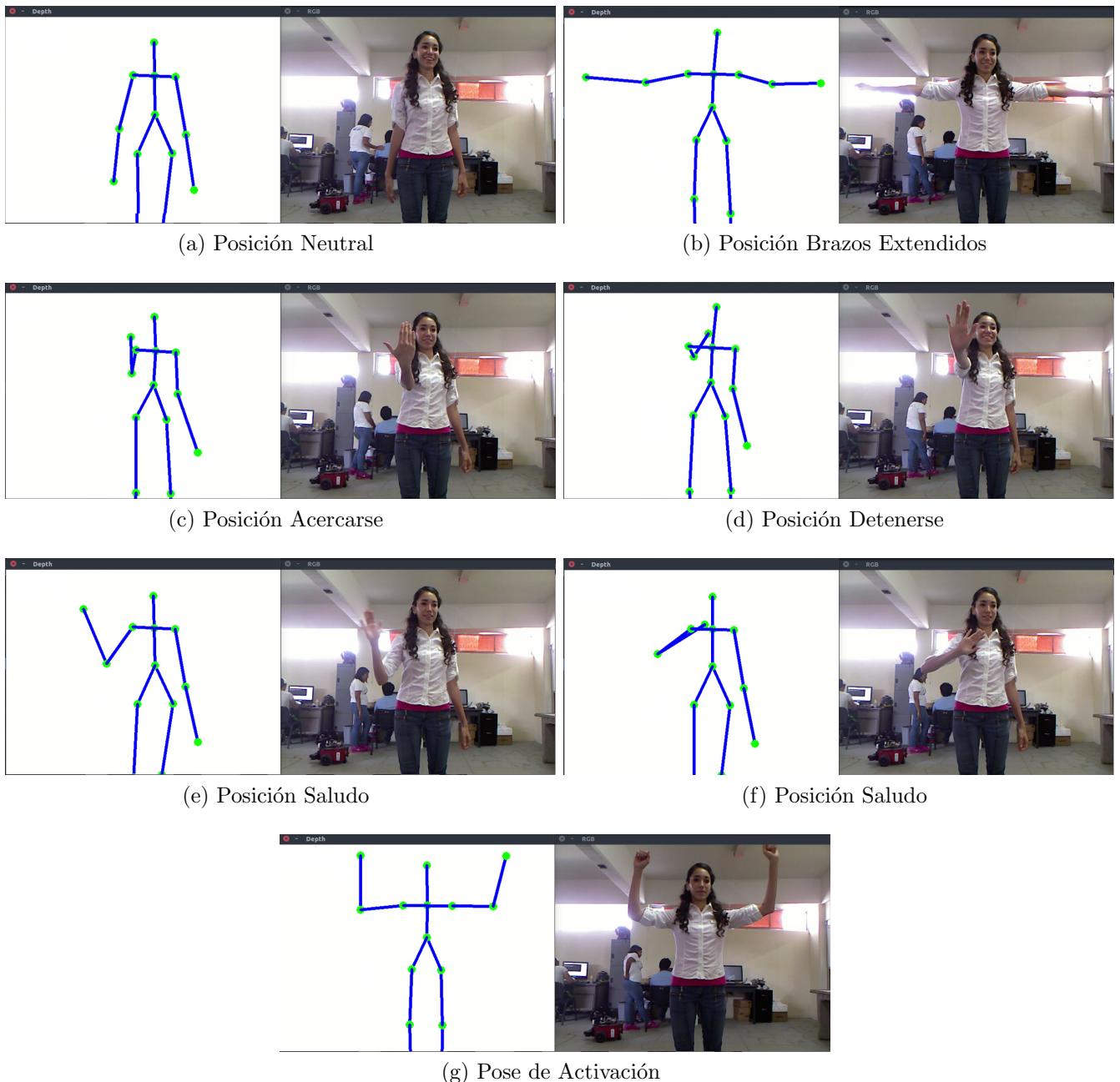


Figura 4-12: Algunas de las poses capturadas implementando la librería OpenNI para identificar el esqueleto humano.

4.3. Identificación de Posiciones Corporales

Tan pronto como hemos recolectado los datos correspondientes a cada posición, es momento de clasificar los datos que se adquieran cuando el sistema se encuentre en ejecución y etiquetar en alguna categoría la nueva posición que se encuentre realizando el usuario al momento de su uso.

Como ya mencionamos en la parte de diseño para ambos casos, la cuadrícula de ocupación y el modelo óseo, se utilizaron dos métodos de clasificación, por medio de k-Nearest Neighbours y Redes Neuronales, el fin de esto es comparar el resultado de ambas y a partir de ello elegir el método más adecuado para la identificación de posturas.

4.3.1. Clasificador de Posiciones con kNN

Para la clasificación de posturas comenzamos con el algoritmo kNN el cual ya hemos abordado sobre su funcionamiento con anterioridad. Para ello, haremos uso de los datos recolectados y usaremos el método knn() existente ya en la librería de OpenCV.

El algoritmo que se presenta enseguida (Algoritmo 1) intenta clasificar las nuevas posturas utilizando los datos obtenidos mediante el umbral de profundidad usando kNN, y suponemos que ya se han capturados los datos con anterioridad. El kNN al ser un clasificador en tiempo real, no existe un entrenamiento previo de los datos para su uso.

En la Figura 4-13 podemos apreciar la correcta clasificación de las posturas. Sin embargo también aparecieron falsos positivos, en los que una postura era clasificada erróneamente (Figura 4-14).

Algorithm 1 Algoritmo para la identificación de posturas usando kNN

Data: Conjunto de ocupación, Set de entrenamiento

Result: Identificación de Postura

Inicialización: Cámara, Rejilla, Set de entrenamiento

mientras Kinect Activo **hacer**

 | Obtener profundidad

 | Aplicar umbral

 | Dividir en celdas

 | Obtener porcentajes de ocupación **if** hay datos de entrenamiento **then**

 | Aplicar kNN

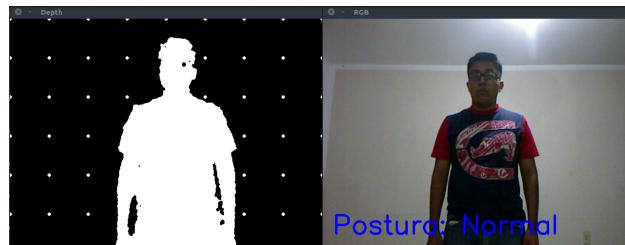
 | Mostrar predicción

else

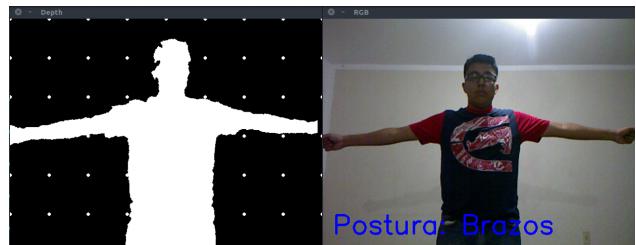
 | Capturar datos

end

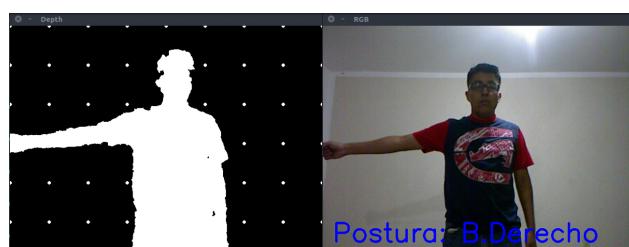
fin



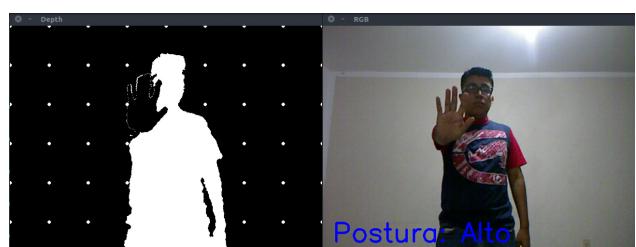
(a) Posición Neutral



(b) Posición Brazos Extendidos



(c) Posición Brazo Derecho



(d) Posición Detenerse

Figura 4-13: Correcta clasificación de posiciones.

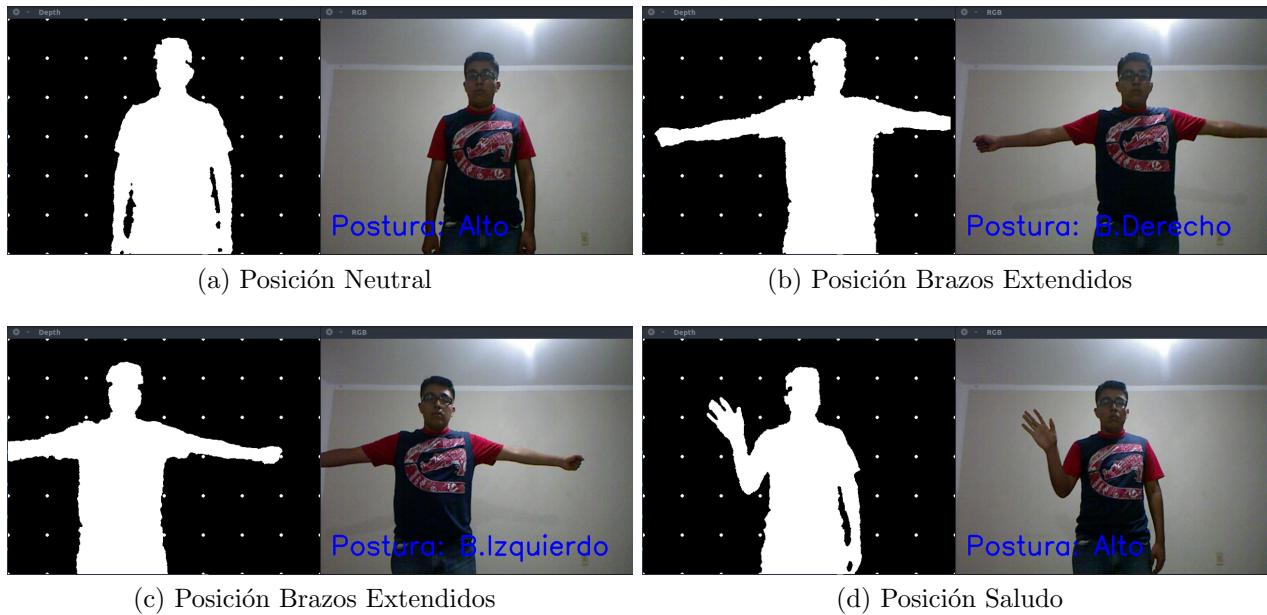


Figura 4-14: Falsos Positivos al clasificar.

Con la detección de personas basado en el esqueleto humano, aplicaremos de igual forma el algoritmo kNN (Algoritmo 2). Esta vez utilizando los datos de interés establecidos es la sección anterior. En la Figura 4-15) podemos apreciar la correcta clasificación de las posturas utilizando la detección de las articulaciones del cuerpo.

Algorithm 2 Algoritmo para la identificación de posturas usando kNN con los puntos obtenidos del modelo óseo.

Data: Conjunto de ocupación, Set de entrenamiento

Result: Identificación de Postura

Inicialización: Cámara, Set de entrenamiento

mientras Kinect Activo **hacer**

 | Obtener puntos 3D

 | Calcular - ángulos, profundidades y distancias

 | **if** hay datos de entrenamiento **then**

 | | Aplicar kNN

 | | Mostrar predicción

 | **else**

 | | Capturar datos

 | **end**

fin

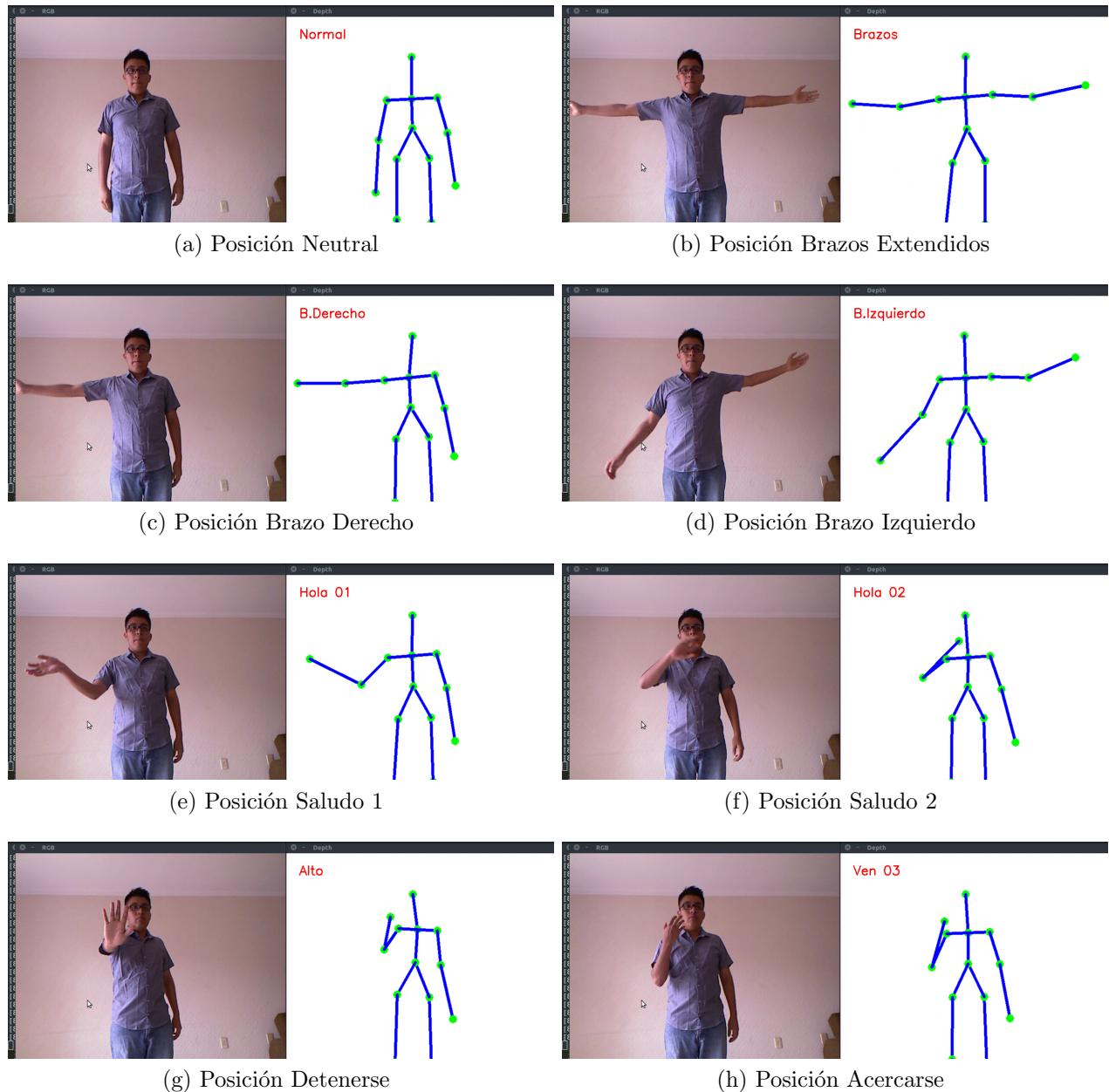


Figura 4-15: Correcta clasificación de posiciones con kNN y el modelo óseo.

4.3.1.1. Resultados de Posturas con kNN

Después de varias pruebas realizadas con distintos valores de $k = 3, 5, 9, 15, 25$. Obtuvimos la siguiente tabla donde $k = 9$ y es cuando obtenemos un mayor acierto de clasificación obtuvimos.

En la Tabla 4-1 se muestra un total de 59 muestras. Cada una de ellas con la cantidad de veces que el algoritmo kNN con $k = 9$ las clasificó en cada una de

Cuadro 4-1: Tabla de Resultados para celdas de ocupación usando kNN con k=9

Posturas	Normal	B. Izquierdo	B. Derecho	Brazos	Saludo	Detenerse	Acercarse
Normal	6	0	0	0	0	5	1
B. Izquierdo	0	3	0	1	1	0	0
B. Derecho	1	0	5	2	0	3	0
Brazos	0	1	3	4	0	0	0
Saludo	0	0	2	0	6	4	3
Detenerse	0	0	0	0	0	3	0
Acercarse	0	0	0	0	1	2	2

las categorías existentes. Si contabilizamos las ocasiones en las que él se clasificó de manera correcta contra las veces que lo hizo de forma errónea, podemos apreciar que el porcentaje de aceptación es del 49 %. Este número puede parecer alarmante ya que esto significa que solo la mitad de las veces se pudo detectar de forma correcta la posición deseada. Sin embargo, este resultado no depende en su totalidad del algoritmo de clasificación, depende también de la posición de la cámara y de la persona respecto a esta. Por ejemplo, en la Figura 4-14b donde el sujeto extiende ambos brazos y se identifica como la pose de brazo derecho, se puede observar que el sujeto en cuestión se encuentra posicionado más a la derecha por lo que el brazo izquierdo no aparece por completo en la pantalla, provocando que se interprete como la de un solo brazo.

Cuadro 4-2: Tabla de Resultados para detección mediante esqueleto usando kNN con k=5

Posturas	Normal	B. Izquierdo	B. Derecho	Brazos	Saludo 1	Saludo 2	Detenerse	Acercarse 1	Acercarse 2	Aciertos
Normal	10	0	0	0	0	0	0	0	0	100 %
B. Izquierdo	0	10	0	0	0	0	0	0	0	100 %
B. Derecho	0	0	10	0	0	0	0	0	0	100 %
Brazos	0	0	0	10	0	0	0	0	0	100 %
Saludo 1	0	0	0	0	7	2	0	0	1	70 %
Saludo 2	0	0	0	0	0	9	0	0	1	90 %
Detenerse	0	0	0	0	0	0	7	2	1	70 %
Acercarse 1	0	0	0	0	0	2	1	5	2	50 %
Acercarse 2	0	0	0	0	0	3	1	0	6	60 %

En esta segunda Tabla 4-2 perteneciente a los datos obtenidos usando la detección del esqueleto humano, se obtuvo una muestra 44 poses realizadas por el usuario. 35 veces se clasificó de manera correcta por lo que contamos con una tasa de acierto

del 80% tomando en cuenta todas las poses realizadas. Comparado con la tabla anterior, se aprecia que la proporción de acierto aumento sustancialmente.

Las poses básicas (normal y brazos) se clasificaron sin ningún problema, obtuvieron un rango reconocimiento del 100%. En donde se generaron más errores fue en las posiciones de acercarse y detenerse, esto debido a que los movimientos del brazo derecho son bastante similares en ambas poses provocando una confusión en la clasificación por parte del programa.

4.3.2. Clasificador de Posiciones con Redes Neuronales

Para la clasificación por medio de redes neuronales se implementaron los dos diseños propuestos en la etapa de diseño esperando obtener mejores resultados que los obtenidos mediante el kNN. El Algoritmo 3 muestra el flujo de trabajo para el entrenamiento y clasificación mediante redes neuronales con los datos obtenidos a través del filtro de profundidad.

Algorithm 3 Algoritmo para la identificación de posturas usando Redes Neuronales con los datos obtenidos mediante las celdas de ocupación

Data: Conjunto de ocupación, Set de entrenamiento

Result: Identificación de Postura

Inicialización: Cámara, Rejilla, Pesos, Set de entrenamiento

mientras *Kinect Activo* **hacer**

 Obtener profundidad, aplicar umbral

 Dividir en celdas, obtener porcentajes de ocupación

if *Red Entrenada* **then**

 Predecir nuevo conjunto

 Mostrar predicción

else

 | Entrenar Red

end

fin

El primer modelo de la red neuronal propuesto está compuesto por una entrada de 48 neuronas, 5 nodos en la capa oculta y para la salida 3 nodos. A continuación

se muestran los demás parámetros aplicados a la red neuronal utilizando la librería nimblenet.

- **Pesos iniciales:** Valor mínimo = -0.1, Valor máximo = 0.1
- **Rango de aprendizaje:** 0.3
- **Valor del momentum:** 0.5

Con estos parámetros establecidos se realizó el entrenamiento para los 900 sets de entrenamientos en la base de datos, el cual duró 9 minutos y 16 segundos, con un total de 8327 iteraciones y un error final de 0.00008. (Figura 4-17)

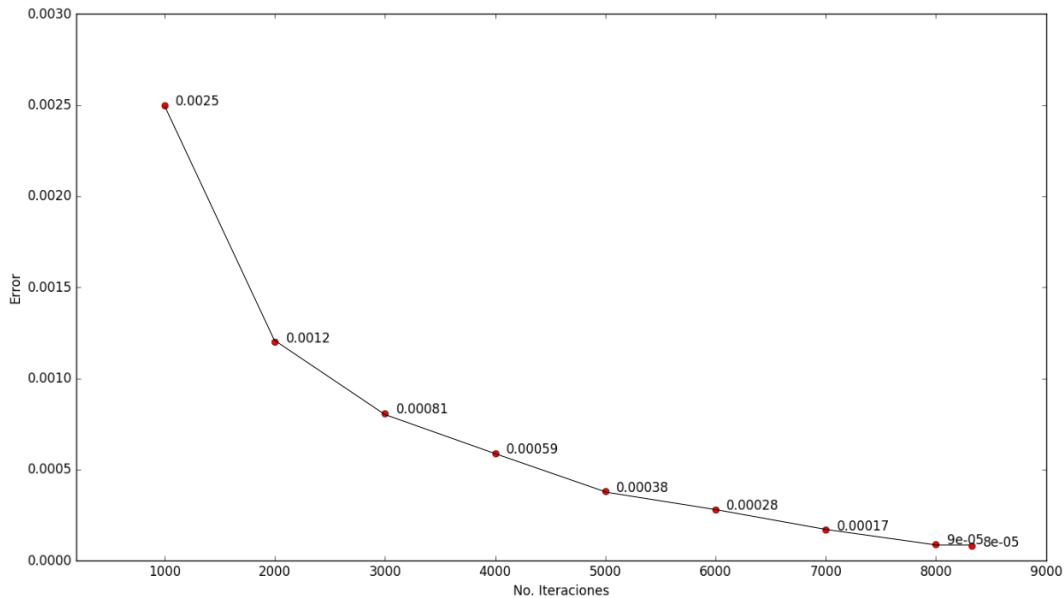


Figura 4-16: Gráfica que muestra el número de iteraciones y como fue disminuyendo el valor del error para la red neuronal de 5 nodos en la capa oculta.

El segundo modelo de la red neuronal que consta con una capa oculta de 96 nodos algunas variables fueron cambiadas ya que con los pesos establecidos causaban un mal cálculo del gradiente ocasionando que el entrenamiento fallara. Para llegar a estos valores, en un principio se inicializaban de forma aleatoria y guardábamos estos valores generados con el fin de encontrar aquellos con los cuales el gradiente fuera calculado de forma correcta y el error producido en las primeras iteraciones no fuera demasiado grande. Hubo ocasiones en que el valor del error generado era de 8.1

por poner un ejemplo y por cada iteración simplemente no disminuía lo suficiente como para obtener un entrenamiento adecuado.

- **Pesos iniciales:** Valor mínimo = -0.1, Valor máximo = 0.1
- **Rango de aprendizaje:** 0.1
- **Valor del momentum:** 0.1

Con estos parámetros establecidos se realizó el entrenamiento para los 900 sets de entrenamientos en la base de datos, el cual duró 17 minutos y 37 segundos, con un total de 6544 iteraciones y un error final de 0.000009. (Figura 4-17). Como podemos apreciar, el tiempo del entrenamiento aumento 3 veces que el anterior. Esto se debe a que aumentamos en gran cantidad el número de nodos en la capa oculta.

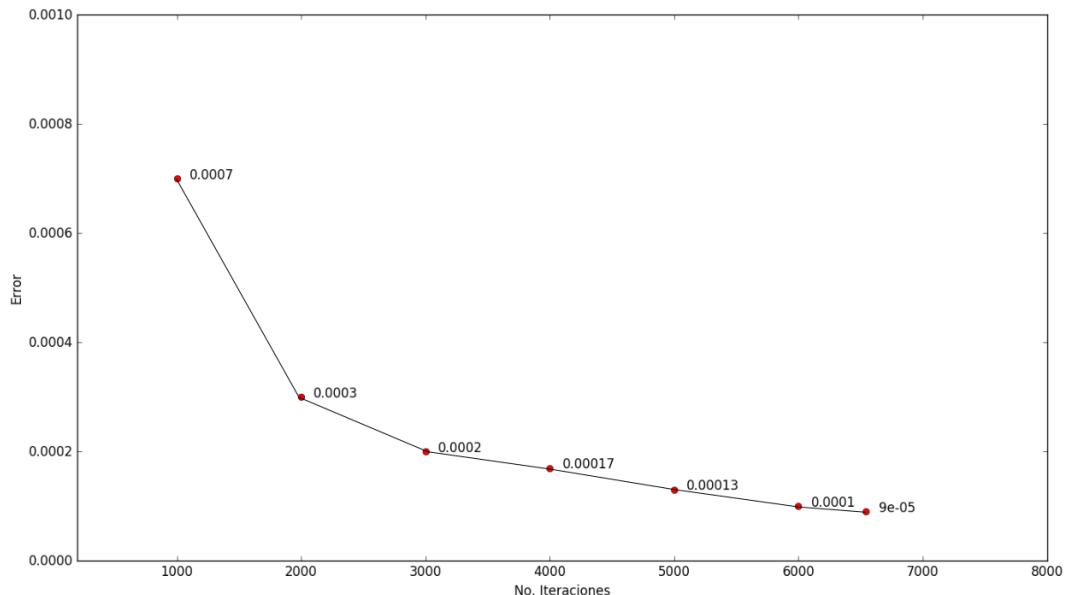


Figura 4-17: Gráfica que muestra el número de iteraciones y cómo fue disminuyendo el valor del error para la red neuronal de 96 nodos en la capa oculta.

Quizás pueda parecer extraño para algunos, que los tiempos expresados se hablen de sólo unos minutos, pero debemos recordar que comentamos que las redes neuronales fueron entrenadas en una computadora de última generación con un procesador de 8 núcleos por lo que los tiempos se reducen drásticamente en comparación a otras

computadoras. Otro punto importante para la reducción de tiempo en entrenamiento es debido a que utilizamos una librería ya existente para la implementación de redes neuronales y que ya ha sido optimizada para su uso. Por último, se probaron distintos parámetros de entradas para los pesos, rangos de aprendizaje y momentum; y si observábamos que el entrenamiento después de 5000 iteraciones no comenzaba a arrojar datos con un buen rango de error, se cancelaba el entrenamiento e ingresábamos nuevos parámetros hasta obtener los esperados.

Una vez entrenadas las redes pertenecientes a los datos conseguidos por el procedimiento del filtro de profundidad, es turno de entrenar los datos adquiridos a partir del esqueleto humano generado. En el algoritmo 4 se muestra el esquema general de la detección de posiciones con redes neuronales junto con los valores calculados de los puntos 3D del modelo óseo.

Algorithm 4 Algoritmo para la identificación de posturas usando Redes Neuronales con los datos obtenidos a partir del modelo óseo.

Data: Conjunto de características, Set de entrenamiento

Result: Identificación de Postura

Inicialización: Cámara, Pesos, Set de entrenamiento

mientras *Kinect Activo* **hacer**

 Obtener puntos 3D

 Calcular ángulos, profundidades y distancias

if *Red Entrenada* **then**

 Predecir nuevo conjunto

 Mostrar predicción

else

 | Entrenar Red

end

fin

En un primer modelo de la red neuronal se sugirió una entrada de 14 neuronas, 5 nodos en la capa oculta y para la salida 4 nodos. A continuación se muestran los demás parámetros aplicados en este primer prototipo.

- **Pesos iniciales:** Valor mínimo = -0.2, Valor máximo = 0.6

- **Rango de aprendizaje:** 0.5

- **Valor del momentum:** 0.9

Con estos parámetros el entrenamiento de las 1400 muestras de entrenamientos después de 5min con 48sg y 10000 iteraciones obtuvimos un error de 1.621 (Figura 4-18). El error generado por este entrenamiento no resulta satisfactorio debido a que el error es demasiado alto como para poder obtener buenos resultados.

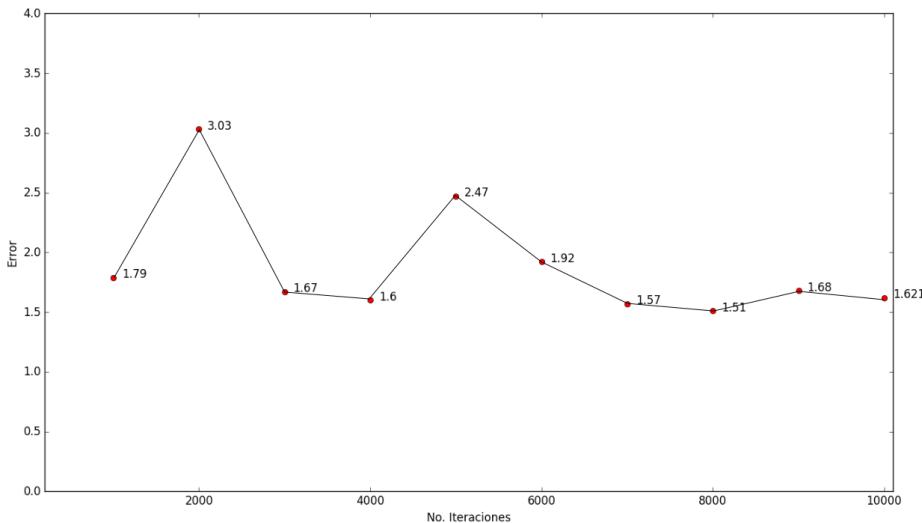


Figura 4-18: Gráfica que muestra como el error se mantuvo constante con un valor alto.

Intentamos de nuevo el entrenamiento pero ahora con estos parámetros de entrada:

- **Pesos iniciales:** Valor mínimo = -0.2, Valor máximo = 0.4
- **Rango de aprendizaje:** 0.1
- **Valor del momentum:** 0.9

El error obtenido se muestra en la Figura 4-19.

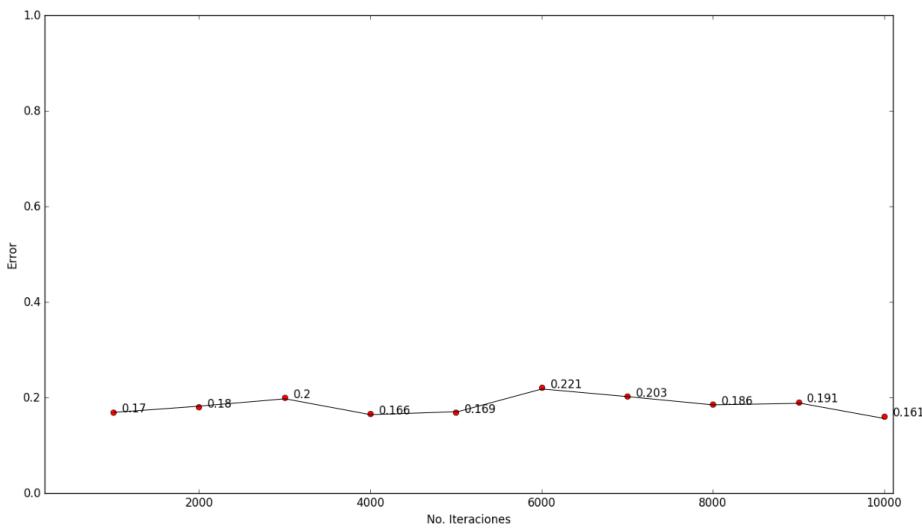


Figura 4-19: Gráfica que muestra un ajuste en el error después de haber cambiado los parámetros de la red neuronal.

Aun cuando el error disminuyó a valores por debajo de 0.3, el error se mantenía alto para nuestros estándares. Intentamos nuevamente cambiar los parámetros y en esta ocasión conseguimos un excelente resultado con sólo **1713 iteraciones** y **3 minutos con 30 segundos** de tiempo de entrenamiento.

- **Pesos iniciales:** Valor mínimo = -0.2, Valor máximo = 0.4
- **Rango de aprendizaje:** 0.3
- **Valor del momentum:** 0.5

Con estos datos se logró conseguir un error de 0.000178 lo que resulta ser un excelente valor. En esta ocasión no pudimos graficar el cambio del error debido a que fueron muy pocas iteraciones realizadas en un lapso de tiempo reducido.

Para terminar el entrenamiento de las redes neuronales, toca el momento de implementar el segundo y último modelo el cual consta, si recordamos, de 14 entradas, 28 nodos de la capa oculta y 4 de salida. Los valores asignados fueron los siguientes:

- **Pesos iniciales:** Valor mínimo = -0.2, Valor máximo = 0.4
- **Rango de aprendizaje:** 0.1

- **Valor del momentum:** 0.9

De la misma manera que la red neuronal anterior, se obtuvo un excelente resultado para el error que fue de **0.00009** en sólo **132 iteraciones en 26 segundos** de entrenamiento.

4.3.2.1. Resultados de Posturas con Redes Neuronales

Una vez entrenadas todas nuestras redes neuronales es hora de clasificar las posturas nuevas que realice el usuario y calcular el rango de acierto, con el fin de determinar cuál de las dos clasificaciones es mejor en este problema.

Cuadro 4-3: Tabla de Resultados para celdas de ocupación usando 5 nodos en la capa oculta

Posturas	Normal	B. Izquierdo	B. Derecho	Brazos	Saludo 1	Saludo 2	Detenerse	Acercarse
Normal	8	2	0	0	0	1	3	0
B. Izquierdo	3	3	0	0	0	0	2	0
B. Derecho	2	4	3	5	5	0	1	2
Brazos	5	5	0	4	2	0	2	0
Saludo 1	3	3	0	0	5	0	0	0
Saludo 2	2	2	1	1	0	0	2	1
Detenerse	0	3	0	0	0	0	4	0
Acercarse	4	2	0	0	0	0	3	0

Como se puede observar en la Tabla 4-3, de 98 muestras en total, solo el 27% fueron clasificadas de forma correcta con la red neuronal de 5 nodos en la capa oculta. Si lo comparamos con los resultados de obtenidos con el kNN, estos resultados fueron muy inferiores.

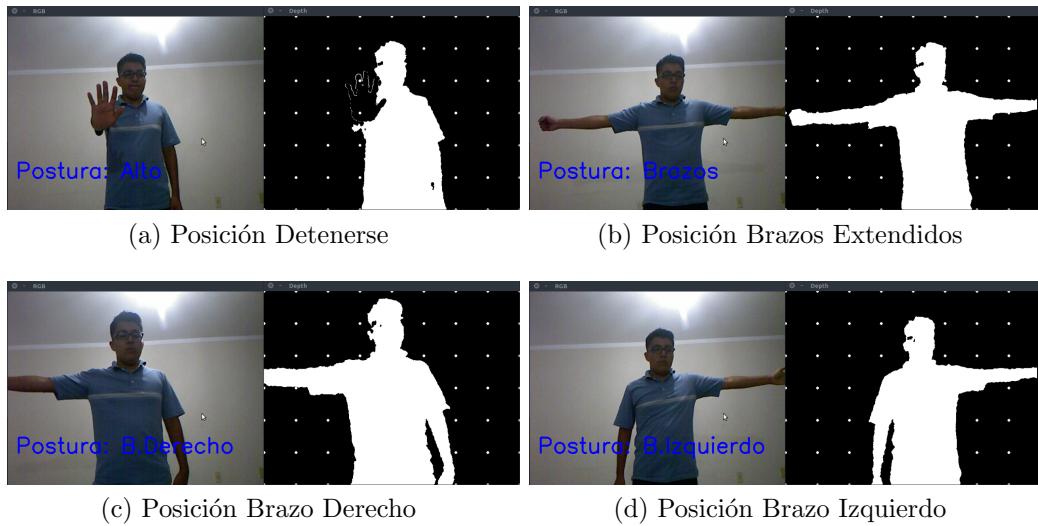


Figura 4-20: Correcta clasificación de posiciones con redes neuronales y el filtro de profundidad.

Cuadro 4-4: Tabla de Resultados para celdas de ocupación usando 96 nodos en la capa oculta

Posturas	Normal	B. Izquierdo	B. Derecho	Brazos	Saludo 1	Saludo 2	Detenerse	Acercarse
Normal	4	0	0	0	0	0	0	0
B. Izquierdo	1	3	1	0	0	0	0	0
B. Derecho	1	2	3	0	4	0	0	0
Brazos	0	2	0	3	0	0	2	0
Saludo 1	0	1	0	0	4	0	0	0
Saludo 2	1	2	0	0	1	0	0	0
Detenerse	0	0	0	0	0	0	4	0
Acercarse	2	0	0	0	2	0	0	1

Con la red neuronal con capa oculta de 96 neuronas, los resultados mostrados en la Tabla 4-4 mejoraron sustancialmente en comparación a la anterior. Esta red neuronal tuvo un acierto de clasificación correcta del 51 %. Este porcentaje solo varía por dos puntos del resultado obtenido por la clasificación mediante kNN. Este dato en particular no ayudará a llegar a una conclusión al final de este trabajo.

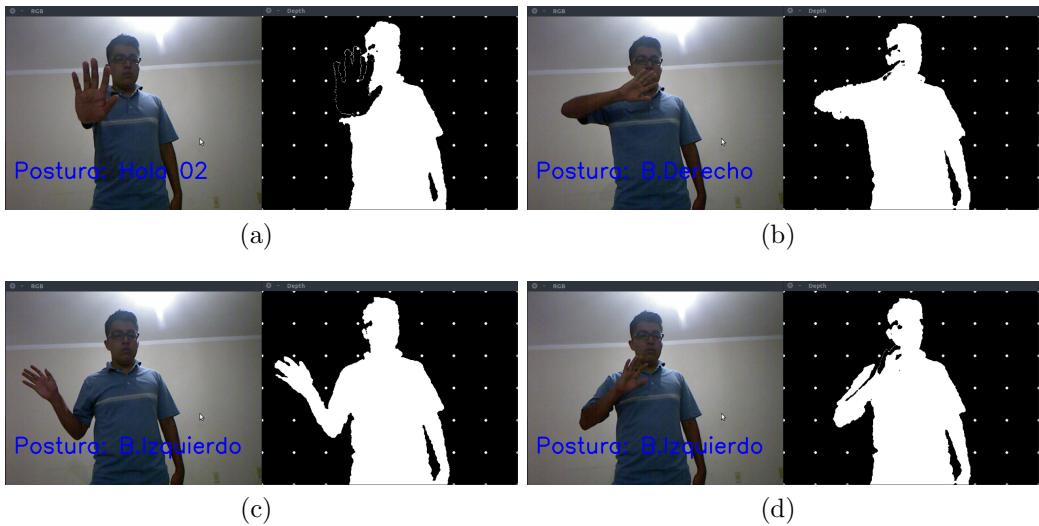


Figura 4-21: Clasificación errónea de posiciones con redes neuronales y el filtro de profundidad.

Cuadro 4-5: Tabla de Resultados para detección mediante esqueleto a partir de la red neuronal entrenada con 5 nodos en la capa oculta.

Posturas	Normal	B. Izquierdo	B. Derecho	Brazos	Saludo 1	Saludo 2	Detenerse	Acercarse 1	Acercarse 2	Aciertos
Normal	10	0	0	0	0	0	0	0	0	100 %
B. Izquierdo	0	10	0	0	0	0	0	0	0	100 %
B. Derecho	0	0	10	0	0	0	0	0	0	100 %
Brazos	0	0	0	10	0	0	0	0	0	100 %
Saludo 1	0	0	0	0	7	2	0	0	1	70 %
Saludo 2	0	0	0	0	0	8	0	0	0	100 %
Detenerse	0	0	0	0	0	0	7	1	2	70 %
Acercarse 1	0	0	0	0	0	1	2	6	1	60 %
Acercarse 2	0	0	0	0	0	3	1	0	6	60 %

Por último, en la Tabla 4-5 se muestran los resultados obtenidos de identificar las posiciones con la red neuronal a partir de los datos obtenidos por el modelo óseo. El porcentaje de clasificaciones identificadas correctamente fue de 84 %, bastante similar al obtenido con el algoritmo kNN. Asimismo hay que examinar con detenimiento que el otro 16 % que corresponde a las posiciones mal clasificadas, procede de las poses de “Saludos” y “Acercarse”. Estos resultados nos demuestran que habría que cambiar o inclusive aumentar las características a detectar que sean relevantes y diferencien estas posiciones unas de otras para mejorar el sistema de detección de poses.

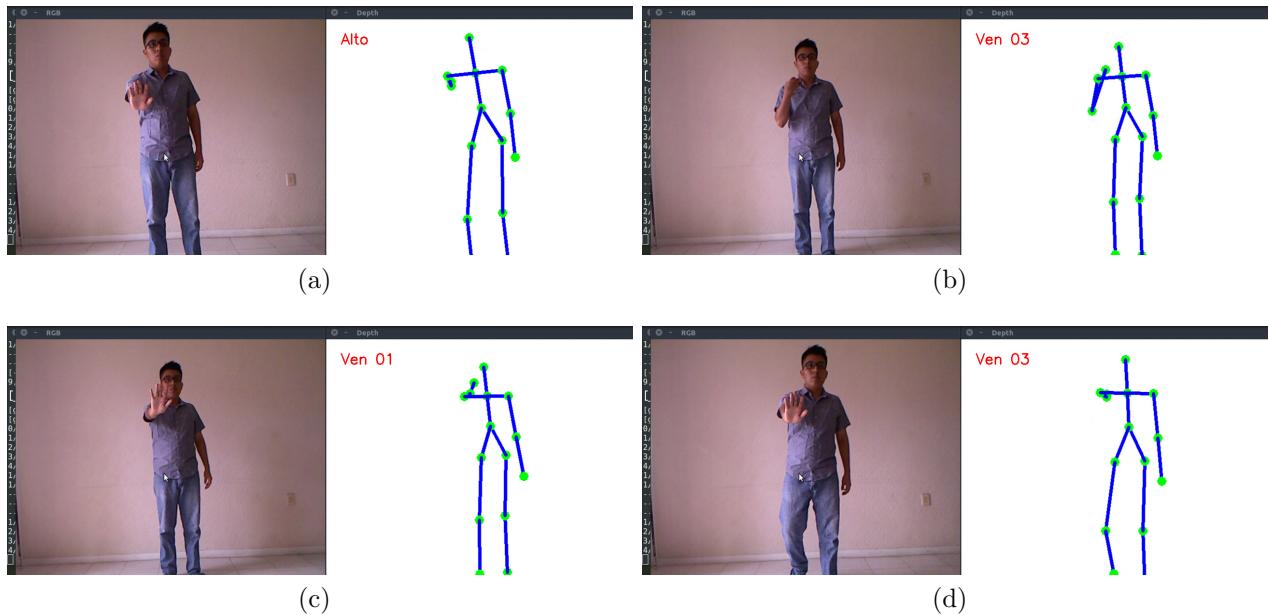


Figura 4-22: Clasificación mediante redes neuronales. (a) y (b) Muestran una correcta identificación de las posiciones. (c) y (d) Falsos positivos con una errónea identificación de la pose.

4.4. Interpretación de Gestos Corporales

Hemos llegado a la parte culminante en la implementación de nuestro sistema de reconocimiento de gestos corporales. Gracias a los resultados obtenidos de la parte de detección de características mediante el uso de los datos obtenidos de las articulaciones del cuerpo humano y la correcta identificación de las posturas con nuestros métodos de clasificación, podemos proceder a interpretar estas posiciones en gestos.

El planteamiento para lograr esto, es que los gestos están compuestos por una secuencia de posiciones. Al poder identificar en un 82 % de las veces de manera correcta una postura, podemos crear un segundo clasificador que reciba como datos de entrada una secuencia de posturas, y con esto identificar de qué gesto se trata. Los gestos que intentaremos predecir son Saludo, Alto, y Acercarse.

4.4.1. Reconocimiento de Gestos Corporales por kNN

Gracias a los resultados proporcionados por las pruebas anteriores, podemos concluir que para este sistema de identificación y los datos en nuestro conjunto de

entrenamiento, la mejor opción como clasificador es el algoritmo kNN. No existe una diferencia significativa en el resultado entre la red neuronal y kNN, y usar kNN nos beneficia debido a que no necesitamos de un entrenamiento de los datos, sino que la comparación se realiza en tiempo real, y hasta el momento esto no ha afectado en el rendimiento del sistema a la hora de clasificar las posturas. El método utilizado para la detección de poses será, evidentemente, con el uso del modelo óseo que fue el que mejor resultados nos dió.

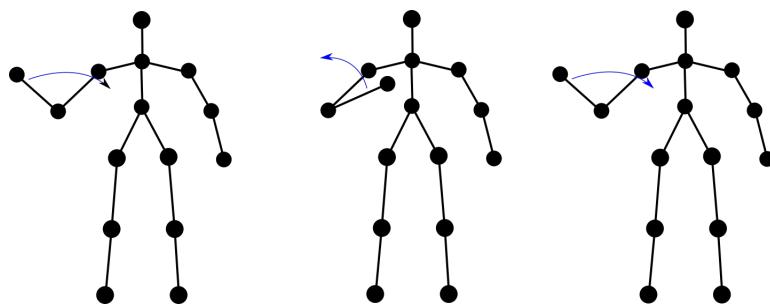


Figura 4-23: Ejemplo de una secuencia de posiciones que juntos representan el gesto de “Saludo”.

En la figura 4-23 se ejemplifica la idea de reconocer el gesto de “Saludo” utilizando las posiciones de Saludo1 y Saludo2 que identificábamos en el modulo anterior, de esta forma una secuencia compuesta por: Saludo1-Saludo2-Saludo1, nos debería indicar que el gesto realizado fue el del usuario saludando. Si ya reconocemos las posiciones de Saludo1 y Saludo2 correctante en la mayoría de los casos, ¿es necesario clasificar una secuencia de posiciones para identificar que el usuario saludó? Para responder esto regresemos a los resultados obtenidos por el identificador de posturas. Usando el esqueleto humano como base para la identificación de poses obtuvimos muy buenos resultados, sin embargo nuestro sistema de identificación en ocasiones clasificaba alguna posición de manera incorrecta. Por ejemplo, la acción de alto en ocasiones era indentificado como la pose etiquetada como “ven01”. Esta pose en realidad es aquella en la que el brazo está a medio camino cuando alzamos el brazo para decir alto. Esta misma pose se realiza cuando efectuamos el gesto de acercarse. Lo que determinaría cuál gesto fue realizado, es la combinación de todas estas, pasando de sólo detectar una pose, a una serie de estas como un gesto.

Para nuestro módulo de interpretación de gestos, se aumentaron dos poses más a nuestra base de datos que llamaremos “ven02” y “ven04”. En la Figura 4-24 se

muestran cada una de las poses que utilizaremos para reconocer los tres gestos propuestos. Cada gesto estará compuesto por una secuencia de tres poses, si alguna combinación que nosotros determinemos como uno de los gestos es realizada por el usuario, entonces podremos determinar mediante la clasificación por kNN que efectivamente la persona realizó alguno de esos gestos.

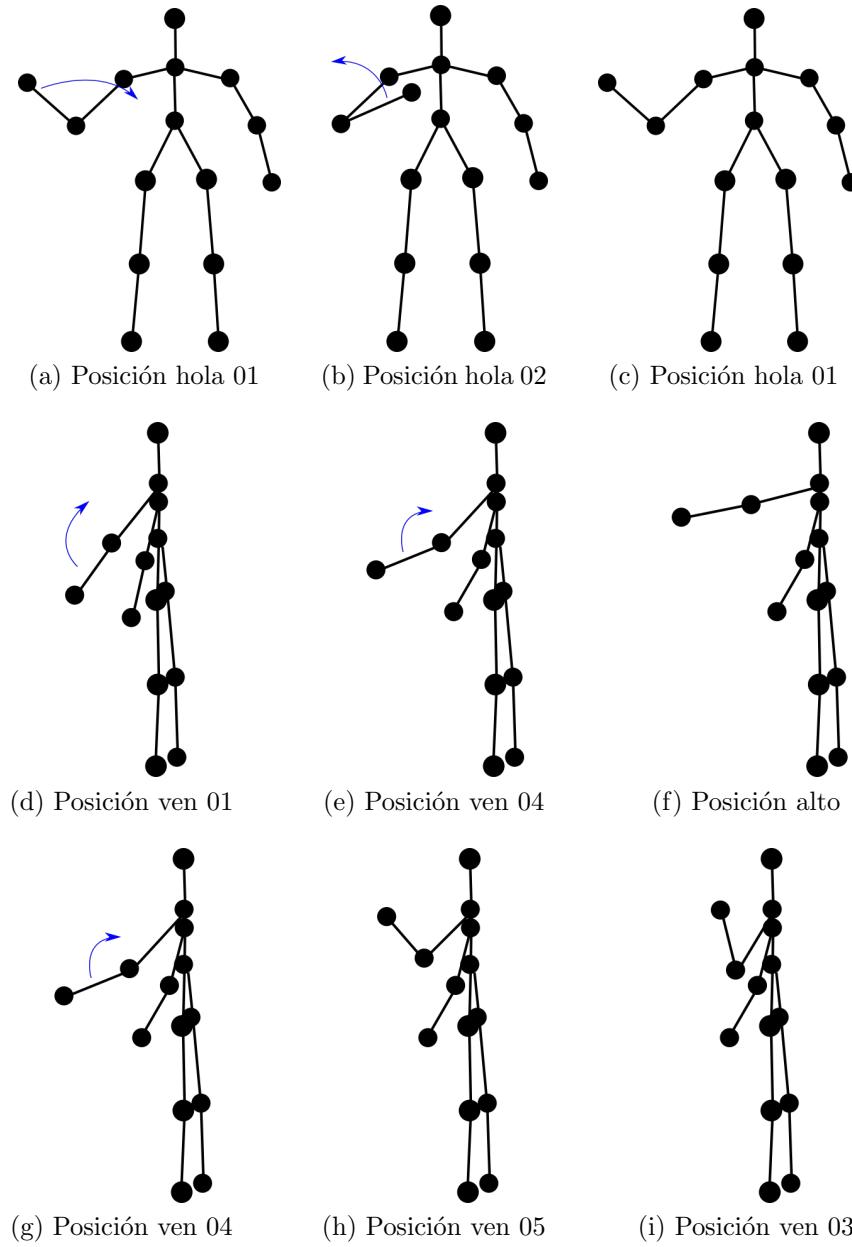


Figura 4-24: Secuencias de posiciones que representan los gestos corporales “Saludo”, “Detenerse” y “Acercarse”

Cuando detectamos una postura mediante kNN, este nos genera una salida de un número entero correspondiente a la posición como nosotros las etiquetamos. Por ejemplo la posición hola01 es un 3 y la posición hola02 es un 4. Entonces si queremos reconocer el gesto de saludo debemos identificar una secuencia de números de esta forma: 3-4-3. Esta combinación de números será guardada como un conjunto de muestra en una base de datos de igual forma que lo hicimos para detectar las posiciones. De este modo el kNN comparará estos datos con las nuevas secuencias de posiciones que nosotros detectemos para obtener un gesto como respuesta.

Cabe aclarar, que no sólo existe una secuencia posible para detectar un gesto. El mismo gesto de saludo puede ser interpretado como 4-3-4 suponiendo que el usuario empiece primero con la segunda posición de saludo. Esta secuencia de números será guardada en un array que inmediatamente después será ingresada al kNN para su clasificación.

En el algoritmo 5 se muestra el funcionamiento general del módulo de reconocimiento de gestos corporales. Despues de la detección de posturas, la posición detectada es ingresada en el array de secuencia, siempre y cuando no sea la postura 0 (la postura normal o neutra) y la postura anterior no sea la misma, eso para evitar que la misma postura se repita en toda la lista de secuencia. Una vez lleno el array, procedemos a clasificar el gesto para su correcta identificación.

Una vez implementado el algoritmo, se procedió a la realización de pruebas dando como resultado la Tabla 4-6.

Algorithm 5 Algoritmo el reconocimiento de gestos usando kNN**Data:** Secuencia de Posturas**Result:** Reconocimiento de Gestos**Inicialización:** Cámara, array de tamaño 3, Set de entrenamiento**mientras** Kinect Activo **hacer**

Obtener puntos 3D

Calcular - ángulos, profundidades y distancias

if hay datos de entrenamiento **then**

postura = kNN

if postura != 0 and postura != postura anterior **then** insertar postura en array **if** array lleno **then**

gesto = kNN

mostrar gesto

else

| pass

end **else**

| pass

end **else**

| Capturar datos

end**fin**

Cuadro 4-6: Cuadro con los gestos reconocidos una vez implementado la secuencia de posturas.

Gesto	Saludo	Acerca	Detenerse	Acierto
Saludo	10	0	0	100 %
Acerca	0	6	4	60 %
Detenerse	0	2	8	80 %

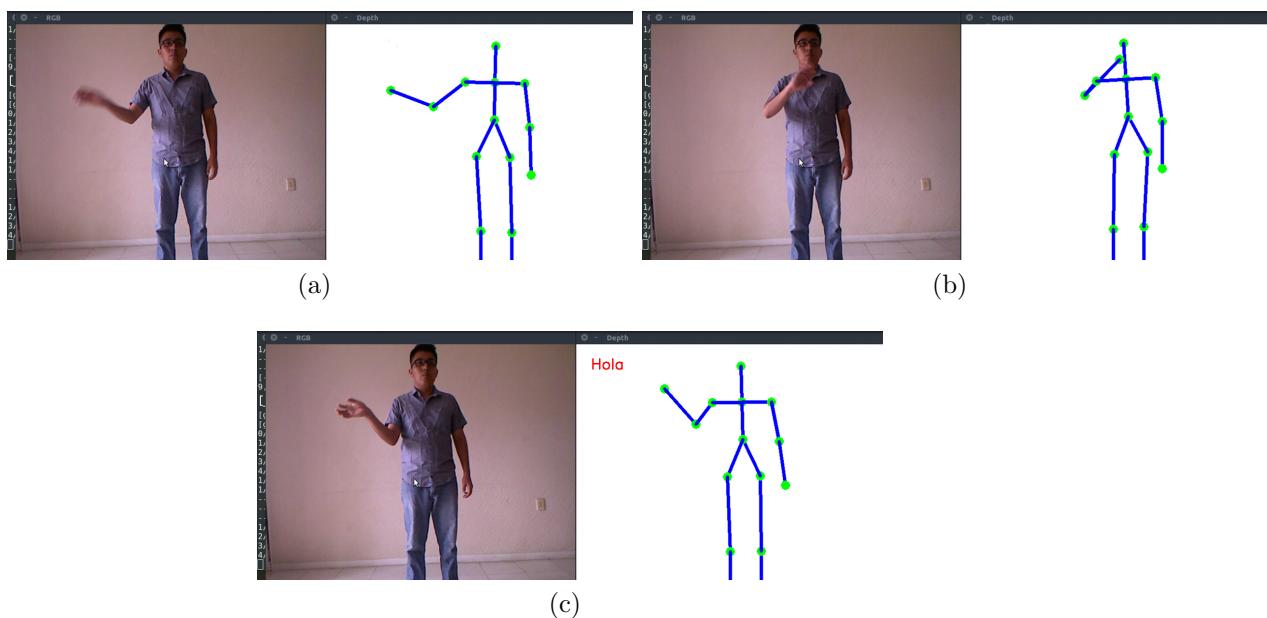


Figura 4-25: Secuencia de imágenes con las que se reconoce el gesto de “Saludo” a partir de tres posturas.

Capítulo 5

Conclusiones

Este proyecto nace con el objetivo principal de desarrollar una herramienta capaz de reconocer e interpretar los gestos corporales realizados por un usuario como medio de interacción con un robot. Una vez desarrollado el programa, realizando pruebas de funcionamiento y recolección de datos es momento de obtener las conclusiones pertinentes.

En el desarrollo del presente trabajo de tesis se han alcanzado los objetivos inicialmente planteados:

- Crear un algoritmo de detección de personas que sea capaz de capturar información acerca de posturas corporales realizadas por el usuario. Primeramente se necesitaba elegir un método de detección que fuera eficaz en la captura de datos, para ellos se implementaron dos propuestas basadas en datos de profundidad obtenidas por el sensor Kinect. El primero que analizaba las áreas de ocupación de cada una de las posturas respecto a la cámara. Y un segundo algoritmo, que mediante los datos obtenidos por la librería OpenNI construimos un modelo óseo que detectaba las articulaciones principales del individuo, permitiéndonos así llevar un seguimiento apropiado de las personas y de sus extremidades, quedando este último como el procedimiento elegido para la detección de posiciones.
- Diseñar un sistema de reconocimiento de posturas. Gracias a los datos obtenidos con el modelo óseo, pudimos detectar las características principales que determinaban cada una de las posturas que pretendíamos a identificar. Una

vez capturados estos datos y almacenados de forma correspondiente a cada una de las posturas, procedimos a clasificar dichos datos para poder predecir nuevas posiciones realizadas por el usuario. Los dos métodos de clasificación utilizados, kNN y redes neuronales, fueron implementados y probados con todos los datos almacenados para tener una comparativa clara de sus resultados pudiendo decidir cuál de los dos era más apropiado para la problemática que enfrentábamos. Dado que los dos métodos de clasificación obtuvieron datos muy parecidos, se determinó que la mejor opción era el algoritmo kNN, esto debido a que no necesita de un entrenamiento previo y la comparación de cerca de 1500 conjuntos de muestra no afectó en lo absoluto el rendimiento del sistema.

- Finalmente, la capacidad del sistema de reconocer gestos corporales. Habiendo logrado una detección de posturas con un buen porcentaje de acierto, pudimos pasar a la siguiente fase que consistía en interpretar una serie de posiciones y definir qué gesto era realizado por el usuario. De la misma forma que resolvimos al detección de posiciones, para la interpretación de gestos corporales se aplicó nuevamente el algoritmo kNN para la clasificación de la combinación de posturas. En esta parte obtuvimos resultados variados, en ocasiones muy buenos y en otros no tanto. El problema radicaba en la detección de ciertas posturas que eran compartidas entre gestos o que simplemente eran muy similares, provocando que el sistema lo interpretara de otra forma produciendo que la detección del gesto no fuera la adecuada.

5.1. Trabajo Futuro

Es evidente que pese al buen rendimiento mostrado por el sistema desarrollado en este trabajo, esto no supone una solución definitiva al problema de detección de posturas e interpretación de gestos corporales ya que aún existen detalles que se pueden mejorar y así conseguir un comportamiento más fiable del programa.

La principal mejora que se podría aplicar al sistema, sería la identificación de más características relevantes en las posiciones corporales. En nuestro primer acercamien-

to al problema de la detección de posturas; se definieron 14 datos que consideramos relevantes en todas las posiciones a detectar. Sin embargo, en posiciones como el de saludo y acercarse, la posición de la mano era tan parecida en ambas, que el sistema no podía definir adecuadamente de cuál posición se trataba ocasionado una mala detección de la postura. Si aplicamos un mayor numero de características a identificar, es mmuy probable que el sistema mejore en su fase de detección.

Otro posible cambio que ayudaría en la mejora del sistema, sería una mayor captura de datos de muestra. En nuestro caso sólo obtuvimos la muestra de 10 personas diferentes, cada una realizaba las posturas solicitadas de acuerdo a como normalmente las realizaban, y al igual que la escritura, no todas las personas realizan gestos corporales con los mismos movimientos. Por lo que tener un muestreo mayor, aumentaría la capacidad del sistema sustancialmente.

Bibliografía

- [1] John McCarthy. What is artificial intelligence? *Computer Science Department, Stanford University*, 2007.
- [2] P. Russell, S.J. y Norving. *Inteligencia Artificial. Un Enfoque Moderno*. PEARSON EDUCACIÓN, S.A., 2004.
- [3] Szeliski Richard. *Computer Vision: Algorithms and Applications*. Springer, 2010.
- [4] M. A. Goodrich and A. C. Schultz. Human-robot interaction: A survey. *Foundations and Trends in Human-Computer Interaction*, 1:203–275, 2007.
- [5] Erika Rogers. Human-robot interaction. *Berkshire Encyclopedia of Human-Computer Interaction*, pages 328–332, 2004.
- [6] Michael Tomasello. Origins of human communication. *The MIT Press. Cambridge, Massachusetts*, 2008.
- [7] Honghai Liu. Xiaofei Ji. Advances in view-invariant human motion analysis: A review. *IEEE Transactions on Systems, Man and Cybernetics -Part C: Applications and Reviews*, 2010.
- [8] Q.Cai. J. K. Aggarwal. Human motion analysis: A review. *Computer and Vision Research Center Department of Electrical and Computer Engineering The University of Texas at Austin*, 1997.
- [9] David Vernon. *Machine Vision: Automated Visual Inspection and Robot Vision*. Prentice Hall, 1991.

- [10] Robert T. Collins et al. A system for video surveillance and monitoring: Vsam. *CMU-RI-TR-00-12, Technical Report*, The Robotics Institute, Carnegie Mellon University, Pittsburgh PA, 2000.
- [11] L.S. Davis I. Haritaoglu, D. Harwood. W4 : real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22, Aug 2000.
- [12] B. Kammerer C. Maggioni. Gesture computer: history, design, and applications. In *Computer Vision for Human–Machine Interaction*. Cambridge University Press, Cambridge, MA, 1998.
- [13] C. Weissman W. Freeman. Television control by hand gestures. *Proceedings of the International Conference on Automatic Face and Gesture Recognition*, pages 179–183, 1995.
- [14] D.M. Gavrila. The visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, 73(1):82–98, 1999.
- [15] Hanqing Lu Yi Li, Songde Ma. Human posture recognition using multi-scale morphological method and kalman motion estimation. *Proceedings of the IEEE International Conference on Pattern Recognition*, 1:175–177, 1998.
- [16] Taishi Noro et al. Estimation method of operating force using an extended kalman filter based on ground reaction force and human behavior. *Proceedings of the IEEE International Conference on Pattern Recognition*, pages 1–4, 2014.
- [17] J. Ohya K. Takahashi, T. Sakaguchi. Remarks on a real-time 3d human body posture estimation method using trinocular images. *Proceedings of the IEEE International Conference on Pattern Recognition*, 4:693–697, 2000.
- [18] S. Kumar J. Segen. Shadow gestures: 3d hand pose estimation using a single camera. *Computer Vision and Pattern Recognition IEEE*, 1, 1999.
- [19] M. Stohr J. Dunker, G. Hartmann. Single view recognition and pose estimation of 3d objects using sets of prototypical views and spatially tolerant contour

- representations. *Proceedings of the IEEE International Conference on Pattern Recognition*, 4:14–18, 1996.
- [20] N. Ahuja Ming-Hsuan Yang. Recognizing hand gesture using motion trajectories. *Computer Vision and Pattern Recognition IEEE*, 1:468–472, 1999.
- [21] J. J. Weng Y. Cui. Hand segmentation using learning-based prediction and veriacation for hand sign recognition. *Computer Vision and Pattern Recognition IEEE*, pages 88–93, 1996.
- [22] Siddharta S. Srinivasa Anca D. Dragan. Online customization on teleoperation interfaces. *IEEE International Symposium on Robot and Human Interactive Communication.*, 2012.
- [23] M. Turk. Visual interaction with lifelike characters. *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, pages 368–373, 1996.
- [24] T. Kanade R.T. Collins, A.J. Lipton. Introduction to the special section on video surveillance. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 22:745–746, 2000.
- [25] AnandhaKumar. P Sowmiya. D, Saithevakunjari. P. Human detection in video surveillance using mbcca: Macro block connected component algorithm. *International Conference on Advanced Computing (ICoAC)*, pages 551–561, 2013.
- [26] Weiming Hu et al. A survey on visual surveillance of object motion and behaviors. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 34:334–352, 2004.
- [27] H. Neven J. Steffens, E. Elagin. Person spotter-fast and robust system for human detection, tracking and recognition. *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, pages 516–521, 1998.
- [28] A. Waibel J. Yang. A real-time face tracker. *Proceedings of the IEEE CS Workshop on Applications of Computer Vision*, pages 142–147, 1996.

- [29] A. Pentland B. Moghaddam, W. Wahid. Beyond eigenfaces: Probabilistic matching for face recognition. *IEEE International Conference on Automatic Face and Gesture Recognition*, pages 30–35, 1998.
- [30] Vinod Chandran Moh Edi Wibowo, Dian Tjondronegoro. Probabilistic matching of image sets for video-based face recognition. *Conference Digital Image Computing Techniques and Applications*, pages 1–6, 2012.
- [31] M. S. Brandstein Ce Wang. A hybrid real-time face tracking system. *Proceedings of IEEE International Conference Acoustics, Speech and Signal Processing*, 6:3737–3740, 1998.
- [32] Jianguang Lou et al. An illumination invariant change detection algorithm. *Asian Conference on Computer Vision*, pages 23–25, 2002.
- [33] Desire Sidibe Bishesh Khanal. Efficient skin detection under severe illumination changes and shadows. *ICIRA 2011 - 4th International Conference on Intelligent Robotics and Applications*, pages 1–8, 2011.
- [34] H.M. Lakany et al. Human walking: tracking and analysis. *Proceedings of the IEE Colloquium on Motion Analysis and Tracking*, pages 5/1–514, 1999.
- [35] J. Kastner M. Kohle, D. Merkl. Clinical gait analysis by neural networks: issues and experiences. *Proceedings of Computer-Based Medical Systems*, pages 138–143, 1997.
- [36] H. Niemann D. Meyer, J. Denzler. Model based extraction of articulated objects in image sequences for gait analysis. *International Conference on Proceedings Image Processing*, 3:78–81, 1997.
- [37] J.E. Boyd J.J. Little. Recognizing people by their gait: the shape of motion. *Journal of Computer Vision Research*, 2, 1998.
- [38] C.J. Harris J.D. Shutler, M.S. Nixon. Statistical gait recognition via velocity moments. *IEE Colloquium on Visual Biometrics*, pages 10/1 – 10/5, 2000.

- [39] J. Kastner M. Kohle, D. Merkl. Clinical gait analysis by neural networks: issues and experiences. *Proceedings of Computer-Based Medical Systems*, pages 138–143, 1997.
- [40] Wikibooks. Artificial neural networks — wikibooks, the free textbook project, 2015. [Online; accessed 15-May-2016].
- [41] Daniel Shiffman. The nature of code, 2012. [Online; accessed 16-May-2016].
- [42] M. Piccardi. Background subtraction techniques: A review. *Conference on Systems, Man and Cybernetics, IEEE International*, 4:3099–3104, 2004.
- [43] Sergey A. Kuchuk Rauf Kh. Sadykhov. Background substraction in grayscale images algorythm. *International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS)*, pages 425–428, 2013.
- [44] S. A. Velastin B. P. L. Lo. Automatic congestion detection system for underground platforms. *International Symposium on Intelligent Multimedia, Video and Speech Processing*, pages 158–161, 2001.
- [45] W.E.L Grimson Chris Stauffer. Adaptive background mixture models for real-time tracking. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2, 1999.
- [46] A. Brandt K.P. Karmann. Moving object recognition using an adaptive background memory. *Time-Varying Image Processing and Moving Object Recognition*, 2, 1990.
- [47] E. DeMicheli A. Verri, S. Uras. Motion segmentation from optical flow. *Proceedings of the Fifth Alvey Vision Conference*, pages 209 – 210, 1989.
- [48] M. Thonnat A. Meygret. Segmentation of optical flow and 3d data for the interpretation of mobile objects. *Proceedings of the International Conference on Computer Vision*, 1990.
- [49] S. Beauchemin J. Barron, D. Fleet. Performance of optical flow techniques. *Computer Vision and Pattern Recognition*, pages 236 – 242, 1992.

- [50] J. Denzler D. Meyer and H. Niemann. Model based extraction of articulated objects in image sequences for gait analysis. *Proceedings of the IEEE International Conference on Image Processing*, pages 78–81, 1997.
- [51] G. Bishop G. Welch. An introduction to the kalman filter. *University of North Carolina at Chapel Hill Department of Computer Science*, 2001.
- [52] A. Blake M. Isard. Condensation—conditional density propagation for visual tracking. *International Journal of Computer Vision*, pages 5–28, 1998.
- [53] D.J. Fleet H. Sidenbladh, M.J. Black. Stochastic tracking of 3d human agures using 2d image motion. *Proceedings of the European Conference on Computer Vision*, 2000.
- [54] V. Pavlović et al. A dynamic bayesian network approach to agure tracking using learned dynamic models. *Proceedings of the International Conference on Computer Vision*, pages 94–101, 1999.
- [55] A.D. Marshall I.A. Karaulova, P.M. Hall. A hierarchical model of dynamics for tracking people with a single video camera. *British Machine Vision Conference*, pages 352–361, 2000.
- [56] S. Tsuji Y. Guo, G. Xu. Tracking human body motion based on a stick figure model. *Journal of Visual Communication and Image Representation*, 5:1–9, 1994.
- [57] S. Tsuji Y. Guo, G. Xu. Understanding human motion patterns. *Proceedings of the International Conference on Pattern Recognition*, pages 325–329, 1994.
- [58] S. Iwasawa et al. Real-time estimation of human body posture from monocular thermal images. *Proceedings of the IEEE CS Conference on Computer Vision and Pattern Recognition*, 1997.
- [59] Y.H. Yang M.K. Leung. First sight: a human body outline labeling system. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:359–377, 1995.

- [60] C.-L. Huang I.-C. Chang. Ribbon-based motion analysis of human body movements. *Proceedings of the International Conference on Pattern Recognition*, 3:436–440, 1996.
- [61] E.H. Adelson S.A. Niyogi. Analyzing and recognizing walking figures in xyt. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 469–474, 1994.
- [62] Y. Yacob S. Ju, M. Black. Cardboard people: a parameterized model of articulated image motion. *International Conference on Automatic Face and Gesture Recognition*, pages 38–44, 1996.
- [63] K. Rohr. Towards model-based recognition of human movements in image sequences. *CVGIP: Image Understanding*, 59:94–115, 1994.
- [64] H.-H. Nagel S. Wachter. Tracking persons in monocular image sequences. *Non-rigid and Articulated Motion Workshop*, pages 2–9, 1997.
- [65] T. Kanade J.M. Rehg. Model-based tracking of self-occluding articulated objects. *Proceedings of International Conference on Computer Vision*, pages 612–617, 1995.
- [66] D. Metaxas I.A. Kakadiaris. Model-based estimation of 3-d human motion with occlusion based on active multi-viewpoint selection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 81–87, 1996.
- [67] N. Goddard. Incremental model-based discrimination of articulated movement from motion features. *Proceedings of the IEEE Workshop on Motion of Non-Rigid and Articulated Objects*, pages 89–94, 1994.
- [68] A.B. Poritz. Hidden markov models: a guided tour. *Proceedings of the International Conference on Acoustic Speech and Signal Processing*, pages 7–13, 1988.
- [69] Donald O. Tanguay. Hidden markov models for gesture recognition. *Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA*, 1993.

- [70] A. Pentland T. Starner. Real-time american sign language recognition from video using hidden markov models. *Proceedings of the International Symposium on Computer Vision*, pages 265–270, 1995.
- [71] K. Ishii J. Yamato, J. Ohya. Recognizing human action in time-sequential images using hidden markov model. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 379–385, 1992.
- [72] M. Morana S. Gaglio, G. Lo Re. Human activity recognition process using 3-d posture data. *IEEE Transactions on Human-MACHINE Systems*, 45:586–597, 2015.
- [73] K. Ishii J. Yamato, J. Ohya. Recognizing human action in time-sequential images using hidden markov model. *Conference on Computer Vision and Pattern Recognition*, pages 379–385, 1992.
- [74] J. Heikkila M. P. V. Kellokumpu. Human activity recognition using sequences of postures. *Conference on Machine Vision Applications*, pages 570–573, 2005.
- [75] A. J. Smola B. Scholkopf. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2001.
- [76] R. Nelson R. Polana. Low level recognition of human motion. *Proceedings of the IEEE CS Workshop on Motion of Non-Rigid and Articulated Objects*, pages 77–82, 1994.