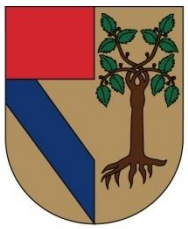


UNIVERSIDAD
PANAMERICANA

ESCUELA DE INGENIERÍA

RNA para el Reconocimiento de Gestos

TESIS

QUE PRESENTA

Blanca Miriam Lee Cosío

PARA OBTENER EL GRADO DE

MAESTRÍA EN CIENCIAS

CON RECONOCIMIENTO DE VALIDEZ OFICIAL DE ESTUDIOS DE LA SECRETARÍA DE EDUCACIÓN PÚBLICA, DE ACUERDO CON EL N° 2007574 DE FECHA 29 DE JUNIO DE 2007.

DIRECTOR DE TÉSIS

Dr. Carlos Delgado Mata

Aguascalientes, Ags., Abril 2012

Tabla de Contenido

1 Introducción

El ámbito del reconocimiento de patrones ha sido ampliamente estudiado con diferentes técnicas y aplicaciones.

Desde la creación de los dispositivos de interfaz con la computadora (HID, por sus siglas en inglés, *Human Interface Device*), se ha enfocado la investigación en el área de reconocimiento de patrones y se han abierto nuevos campos de aplicación. Es así que en la actualidad existen diversos dispositivos que son capaces de proporcionar información relevante respecto a diferentes fenómenos de interés, sin embargo cada uno entrega la información codificada y por tanto se requiere de un algoritmo de procesamiento para cada uno.

Las aplicaciones más comunes de reconocimiento de patrones han sido desarrolladas para el procesamiento de voz y reconocimiento de escritura donde los métodos probabilísticos y estadísticos fueron los más utilizados por sus buenos resultados.

Trabajos similares al presentado en este documento se han desarrollado para la evaluación y caracterización de trayectorias y movimientos a partir de la información obtenía por acelerómetros.

En particular, técnicas de filtrado con la Transformada Rápida de Fourier (FFT) y algoritmos de K-medias se han utilizado para la evaluación de información. Paralelamente, Modelos Ocultos de Markov (HMM, por sus siglas del inglés, *Hidden Markov Model*), Redes Neuronales Artificiales (RNA), y algoritmos de clasificación con redes bayesianas se han utilizado para el entrenamiento y caracterización de modelos.

Los sistemas existentes de reconocimiento de patrones son complejos y costosos, y necesitan de una infraestructura especial no siempre accesible. Hasta ahora los mayores avances sobre el reconocimiento de patrones se pueden encontrar en videojuegos, los cuales no son adaptables para beneficio del individuo y son limitados a cierto uso, es decir su adaptabilidad es casi nula para otros fines.

En esta investigación se evaluaron diversas formas de utilizar la información obtenida de dichas interfaces electrónicas, para reconocer patrones. Es decir, se analizaron diferentes técnicas de reconocimiento de patrones para la clasificación y caracterización de patrones de movimiento y trayectorias. Para esto se utilizó el control remoto de la consola de videojuegos Wii, ya que, por sus características de costo, diseño y capacidades tecnológicas, es ideal para el desarrollo de una aplicación adaptable y genérica.

Para lograr los objetivos, hay que tratar con patrones variables en espacio y tiempo, lo cual requiere un respaldo teórico para complacer estos requerimientos.

1.1 Objetivos

Los objetivos que se buscan en este proyecto son:

- Ofrecer un sistema de reconocimiento de gestos arbitrarios generados por el control remoto del Wii, que es un control basado en acelerómetros.
- Implementar algoritmos de clasificación de patrones y por otro lado de entrenamiento y de aprendizaje de redes neuronales para compararlos y obtener el mejor resultado.

2 Marco Teórico

2.1 Inteligencia Artificial

Desde tiempo remoto los humanos hemos querido entender como pensamos; como percibimos, comprendemos, manipulamos el mundo a nuestro alrededor. La Inteligencia Artificial (IA) no solamente trata de hacer esto, sino también de reproducirlo y construirlo.

Una definición aceptada es que IA es "*el arte de crear máquinas que realicen funciones que requieren de inteligencia cuando son realizadas por personas*" [1]. Aunque cualquiera de sus definiciones se puede agrupar en alguna de estas cuatro categorías:

a) Sistemas que piensan como humanos
b) Sistemas que actúan como humanos
c) Sistemas que piensan racionalmente
d) Sistemas que actúan racionalmente

Donde se considera que un sistema es racional si hace lo correcto según lo que sabe.

Estos cuatro enfoques han sido estudiados por diversas personas en diferentes aplicaciones. Hoy en día, la IA se divide en muchas ramas que van desde lo general hasta lo específico.

2.2 Reconocimiento de patrones

Es la rama de la IA encargada de la clasificación y descripción de las observaciones. El reconocimiento de patrones tiene como objetivo clasificar los datos (patrones) ya sea sobre una base de un conocimiento a priori o en la información estadística extraída de los patrones. Los patrones a clasificar son por lo general grupos de medidas u observaciones, que definen puntos en un espacio multidimensional apropiado. [2]

El área del reconocimiento de patrones ha sido ampliamente estudiada, sobre todo desde la invención de los dispositivos electrónicos, donde se han descubierto nuevas aplicaciones con gran potencial. Hoy en día existen varios dispositivos que proveen información de diferentes eventos, aunque cada uno la entrega de diferente manera, por lo que requieren de un procesamiento distinto.

El reconocimiento de patrones comprende a su vez otras disciplinas como caracterización, análisis discriminante, error de estimación, análisis de clústeres, análisis sintáctico, y otras. Entre las aplicaciones más importantes están las de análisis de imágenes, reconocimiento de caracteres, análisis de voz, diagnóstico de máquinas y personas, identificación personal e inspección industrial.

Particularmente, la problemática del reconocimiento de gestos ha sido abordado en diversos trabajos, como los de [3] y recientemente en [4], sin dejar de lado el trabajo comercial por AiLive Inc. [5]

2.2.1 Reconocimiento de Voz y Escritura

Desde los años 70's, muchos métodos y técnicas han sido explorados para el reconocimiento de voz y escritura. Recientemente, las aplicaciones con modelos ocultos de Markov han resultado las más exitosas en este campo.

Generalmente, un sistema de reconocimiento de voz relaciona la transformación de la voz con la representación almacenada.

Los sistemas de reconocimiento de voz se pueden clasificar de la siguiente forma [6]:

- Dependiente o independiente del orador
- Señal discreta o continua
- Tamaño del vocabulario
- Tasa de reconocimiento

En investigaciones previas de Bell Laboratories [7] e IBM Research [8] se han aplicado métodos probabilísticos y estadísticos por sus buenos resultados para el reconocimiento de texto manuscrito.

2.2.2 Reconocimiento de Gestos

Uno de los objetivos primordiales de la investigación en reconocimiento de gestos es la creación de un sistema que pueda identificar gestos humanos específicos y los utilice para transmitir información o controlar dispositivos.

Los gestos se han definido según el contexto en que se habla o la aplicación que se les vaya a dar.

En sociología, los gestos son utilizados con frecuencia, desde señalar a una persona para llamar su atención hasta obtener conclusiones a partir del espacio y características temporales [9]. Un gesto no es utilizado solo para enriquecer el lenguaje hablado, sino que es parte del proceso de generación e interpretación del lenguaje [10]

En la biología, "la noción de gesto abarca todo tipo de casos en que un individuo se involucra en movimientos cuya intención comunicativa es fundamental, manifiesta y reconocida abiertamente" [11].

En términos de sistemas de reconocimiento de gestos, estas definiciones varían según la tecnología utilizada para obtener los gestos y las acciones que se derivan de estos. En muchos de los casos, los gestos son interpretados para controlar los mecanismos de accionamiento, por lo que los estudios de interacción Humano-Computadora (HCI) suelen centrarse en la interfaz entrada/salida de la computadora [12].

Pero, sabiendo de antemano que los gestos sirven para comunicar información, ¿cómo obtener esta información de manera precisa para que sea útil?

Basándonos en el trabajo de Wolf [13] sobre gestos realizados con las manos, se puede analizar el control de dispositivos mediante gestos al evaluar las siguientes preguntas de generación, representación, reconocimiento y transformación:

- ¿Qué es un diccionario de gestos?
- ¿Cómo se generan los gestos?
- ¿Cómo reconoce el sistema los gestos?

- ¿Cuáles son los requerimientos de procesamiento (i.e. memoria y tiempo) para el reconocimiento de gestos?

Se han realizado varios trabajos relacionados con el reconcomiendo de gestos. En estos proyectos, los gestos fueron creados por un cuerpo o parte de un cuerpo (ej. mano) estático, o por un movimiento físico en dos o tres dimensiones, que puede ser traducido por la computadora en comandos simbólicos (ej. iniciar, detener, girar), trayectorias de movimiento, o incluso letras de un alfabeto y palabras de una lengua. Estos son representados por patrones característicos de señales de datos de entradas, que en este caso, son vectores representando la aceleración del control en las tres dimensiones.

Específicamente, se han realizado trabajos para la evaluación y caracterización de trayectorias y movimientos a partir de la información obtenida por acelerómetros. Proyectos como WiiGee [14], GesApp, parte del Google Summer of Code 2008 [15] y el de Prekopcsák [16] han investigado el reconocimiento de patrones de este tipo.

En sistemas representativos como los de Mardia [17] y Rubine [18], gestos con las manos en dos dimensiones envían a través de un dispositivo de entrada (mouse en el caso de Mardia, tableta gráfica en el caso de Rubine) a la memoria de la computadora y también aparecen en el monitor de la computadora. Estos gestos simbólicos se identifican como comandos de edición para el procesamiento de textos a través de técnicas de modelado geométrico. Las órdenes se ejecutan a continuación, modificando el documento almacenado en la memoria de la computadora. Ambos sistemas requieren que el gesto sea completado antes que el reconocimiento pueda comenzar.

Otros sistemas, como el de Murakami [19], utilizan como dispositivo de entrada un guante con sensores, y en lugar de modelarse geométricamente, una red neuronal identifica 42 símbolos realizados con dedos. El guante provee un mayor rango de gestos y movimientos que un mouse o una tableta. Sin embargo, el entrenamiento de la red neuronal es un proceso que requiere de varias horas.

En el proyecto de Expressive Footwear [20] el dispositivo con el que se captura la información multimodal de un bailarín en movimiento es un circuito impreso con sensores dinámicos (acelerómetros, giroscopios, compas magnéticos), montados a un costado de una zapatilla de danza. Los datos son filtrados para buscar características específicas, como zapatazos o giros, que son utilizados para generar música al instante. Fue exitoso en su aplicación, sin embargo no se puede generalizar ya que los sensores y la distribución utilizada de éstos fueron muy específicos para el zapato y los movimientos del bailarín, además de que la tarjeta electrónica era muy grande para otras aplicaciones.

Hoy en día, hay sistemas que reconocen gestos y movimientos de cuerpo completo. El trabajo comercial de AiLive II [5] realiza esta función mediante el procesamiento de imágenes para controlar 'entes virtuales' en un 'entorno virtual' en la memoria de la computadora. Este fue el primer trabajo que utilizó la información cinética de un gesto como parte de un comando de control. Es decir, cada gesto se traducía en una acción dentro del entorno virtual.

Sistemas como el de Starner y Pentland [21] utilizan modelos ocultos de Markov para reconocer 40 señas del lenguaje americano. Los vectores característicos adquiridos pasan a través de todos los conjuntos posibles de enunciados de cinco palabras. Después se determina la probabilidad de cada flujo de información generada por el modelo. El sistema elige el modelo que tenga la más alta probabilidad de generar el flujo de datos. Los estados consisten en la distribución de la probabilidad de las características de la mano.

2.3 Wiimote

Para hacer que los videojuegos fueran más accesibles a todas las personas de prácticamente todas las edades y habilidades, Nintendo desarrolló un dispositivo que fuera tan atractivo como sofisticado. El resultado fue el Wiimote. Nintendo combinó la familiaridad de un control remoto con la moderna tecnología de detección de movimiento para llegar a un dispositivo de entrada para prácticamente todas las edades. Con el tamaño de un control remoto tradicional, el Wiimote es un dispositivo

multifuncional, desde una raqueta de tenis hasta el arma con que se apunta a un enemigo. La lista de usos potenciales es extensa.

Principalmente, el Wiimote fue diseñado para la medición de aceleraciones en un espacio euclidiano. Tiene seis grados de libertad; es decir, se puede mover a lo largo de los ejes X, Y, Z. Igualmente es capaz de rotar sobre estos mismos ejes, lo cual también se conoce como cabeceo, alabeo, y guiñada, como se muestra en la Figura 2.1.

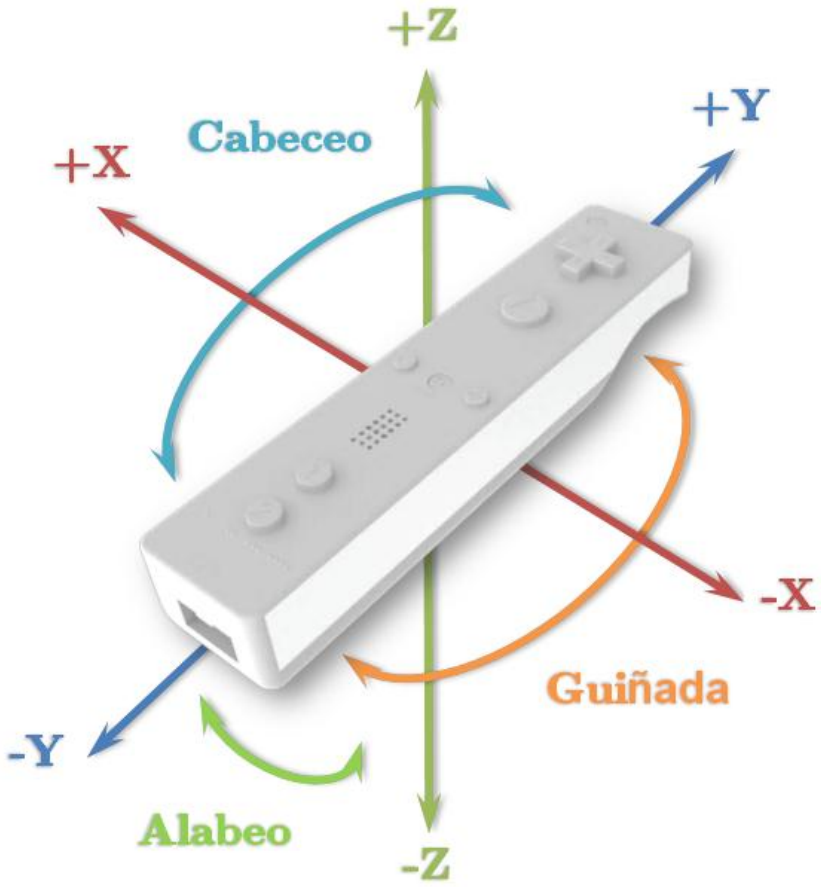


Figura 2.1 - Grados de Libertad y ejes del Wiimote

2.3.1 Bluetooth

Para su comunicación con otros dispositivos, el Wiimote cuenta con un Bluetooth integrado Broadcom BCM2042 y utiliza el protocolo estándar Bluetooth HID. [22]

El Bluetooth© es un protocolo de red inalámbrica desarrollado con el principal objetivo de conectar diversos dispositivos a corta distancia sin necesidad de cables y de bajo consumo.

El núcleo del sistema Bluetooth consiste en un transmisor de radio, una banda base y una pila de protocolos. El sistema permite la conexión entre dispositivos y el intercambio de datos entre ellos [23]. Las principales características de esta tecnología son robustez, bajo costo y bajo consumo de energía.

Para la comunicación con la computadora con el bluetooth del Wiimote se utilizaron varias herramientas:

- ***BlueZ en Linux.*** BlueZ es un protocol official de Linux. Es un proyecto de código abierto distribuido bajo licencia GNU General Public License (GPL). El kernel de BlueZ es parte del kernel oficial de Linux desde la versión 2.4.6.
- ***32feet.NET In the Hand en Windows.*** En Windows, de igual manera, se utilizó una librería open-source que permite la comunicación desde la PC con dispositivos a través de Bluetooth llamada 32feet.NET. [24]

2.3.2 Acelerómetro

Un acelerómetro es un dispositivo electromecánico que se utiliza para medir las fuerzas de aceleración. Estas fuerzas pueden ser estáticas (como la fuerza de la gravedad), o dinámicas (producidas por el movimiento o aceleración del acelerómetro). El sensor acelerómetro genera una muestra de Δx, Δy, y Δz que indican los cambios en aceleración de cada eje. [25]

El Wiimote contiene un circuito integrado ADXL330, que es un acelerómetro linear de tres ejes y fabricado por Analog Devices©. Tiene la capacidad de medir aceleraciones en el rango de $\pm 3g$ con 10% de resolución. Este dispositivo tiene una frecuencia de muestreo preestablecida de 100Hz, es decir, hasta 100 valores por segundo para cada una de la aceleraciones en X, Y y Z. [15]

Esto significa que si el Wiimote estuviera en caída libre con el eje Z perpendicular al piso, devolvería valores muy cercanos a 0 para la aceleración en Z. Por el contrario, si

se mantuviera el Wiimote en reposo con la misma orientación, entregaría en este eje un valor positivo equivalente a la fuerza de gravedad g (aproximadamente $9.8\ m/s^2$). En la dirección contraria este valor sería negativo. A partir de esta relación se puede conocer la orientación del Wiimote, siempre y cuando esté razonablemente quieto [26].

En este proyecto, el acelerómetro integrado en el Wiimote obtiene la información de los gestos o movimientos físicos que realiza el usuario con el control.

2.4 Adquisición de muestras

2.4.1 LibWiimote (Linux)

Para la obtención de las muestras se utilizó la librería en el lenguaje de programación C LibWiimote [27], que provee una interfaz para la comunicación con el Wiimote en un sistema Linux.

Se utilizan instrucciones como `wiimote_connect`, `wiimote_update` y `wiimote.force` para conectarse y obtener información actualizada del dispositivo.

2.4.2 WiimoteLib for .NET (Windows)

Del mismo modo, en Windows se establece una interfaz con el dispositivo Wiimote a través de la librería WiimoteLib [28]. Dicha librería provee una API para código administrado (.NET) que fue posible enlazar con la librería para comunicación por bluetooth.

Así mismo, mediante los métodos `Connect`, `WiimoteChanged` y la propiedad `WiimoteState` se puede comunicar con el Wiimote y obtener la información deseada. Esta librería define que, a pesar de que las aceleraciones devueltas por el dispositivo son equilibradas por valores de calibración, se debe tomar en consideración que valores de aceleración ± 3 son imprecisos.

2.5 Preprocesamiento

2.5.1 Interpolación

Se define como interpolación a la estimación de puntos intermedios entre valores discretos conocidos.

La interpolación lineal es la más sencilla y más utilizada, la cual es un caso particular de la interpolación general de Newton. Se utilizan dos puntos, (x_a, y_a) y (x_b, y_b), para obtener un tercer punto interpolado (x, y) a partir de la fórmula que se muestra en la ecuación (2.1).

$$y = y_a + (x - x_a)\frac{(y_b - y_a)}{(x_b - x_a)}$$

(2.1)

Generalmente se aplica en la aproximación de una función complicada en una más simple. Sin embargo, no es muy precisa ya que no se obtienen los mismos resultados evaluando en la función obtenida, que evaluando en la función original. Sin embargo, entre más pequeño sea el intervalo entre los puntos, más exacta será la aproximación.

2.5.2 K-medias

El algoritmo K-medias [29] es un método popular de agrupamiento (clusterización) heurístico que separa un conjunto de observaciones en k grupos (clústeres). El algoritmo es fácil de implementar y además es eficiente. Por otro lado, procesa los patrones secuencialmente por lo que requiere un almacenamiento mínimo, aunque su comportamiento depende del parámetro k.

Para poder definir la pertenencia de una observación a un clúster, se necesita agrupar las observaciones de acuerdo a la similitud entre las mismas, para lo cual se utiliza métrica que permita asociar una observación a un determinado clúster. En el caso de K-medias se usa la norma euclidiana que se muestra en la ecuación (2.2), donde se define D, la distancia de los vectores x_i a $x_{i'}$ en un espacio euclidiano multidimensional.

$$D(x, x') = \|x - x'\|^2 = \sqrt{\sum_{i=1}^{n}(x_i - x_{i'})^2}$$

(2.2)

Suponiendo que se tienen n observaciones $x_1, x_2, \dots, x_n$ agrupados en $k \leq n$ clústeres $S = \{S_1, S_2, \dots, S_k\}$ donde x_i, siendo $1 <= i <= n$, está representado en un espacio multidimensional, y sea μ_i como la media estimada del j-ésimo clúster.

Con esto se puede decir que la observación x_i está asociada al j-ésimo clúster si $D(x_i, \mu_i)$ es el mínimo con respecto a los k clústeres.

$$f(x) = \sum_{i=1}^{k} \sum_{x_j \in S_i} \|x_j - \mu_i\|^2$$

(2.3)

Así se resume que el algoritmo está basado en la minimización de la distancia interna (2.3) la cual se define como la suma de las distancias de cada observación al centroide del clúster al cual está asociado.

2.5.3 Transformada rápida de Fourier

La transformada rápida de Fourier (FFT por sus siglas en inglés) es un algoritmo computacionalmente eficiente para calcular la transformada discreta de Fourier (TDF) y su inversa.

La transformada Fourier es la representación de una señal continua de entrada $h(t)$, es decir un proceso físico descrito por una cualidad h en función del tiempo t, como una función de su amplitud H en términos de su frecuencia $-\infty < f < \infty$, como se muestra en la ecuación (2.4)

$$H(f) = \int_{-\infty}^{\infty} h(t) e^{2\pi i f t} dt$$

$$h(f) = \int_{-\infty}^{\infty} H(f) e^{-i2\pi f t} df,$$

(2.4)

La TDF reemplaza la integral infinita y la reduce a la sumatoria en la ecuación (2.5), para N valores consecutivos en $k = 0,1,2, \dots, N-1$.

$$H(N) \equiv \sum_{k=0}^{N-1} h_k e^{2\pi i k/N}$$

(2.5)

Una TRF es cualquier algoritmo que obtenga los mismos resultados de la TDF en $O(N\ log\ N)$ operaciones en vez de $O(n^2)$ que requiere la definición de la TDF.

$$e^{x+y_i} = e^x(\cos y + i\ \mathrm{sen}\ y)$$

(2.6)

De esta forma, interpretando con la fórmula de Euler en la ecuación (2.6), podemos decir que la TDF transforma, como su nombre lo dice, cualquier señal de entrada en una serie de funciones senoidales y cosenoidales a diferentes frecuencias como se muestra en la ecuación (2.7).

$$H(N) = \sum_{k=0}^{N-1} \left(\cos^2 \frac{\pi k}{N} + \mathrm{i}\,\mathrm{sen}^2 \frac{\pi k}{N}\right)$$

(2.7)

Lo que se obtiene de la transformada son las diversas frecuencias respecto a la amplitud en que se puede representar una función en términos de funciones cosenos y senos. Las primeras frecuencias, las más características de la función, son las armónicas de Fourier, que son la esencia de nuestra función. Están serán nuestras entradas para el clasificador.

2.6 Clasificación

2.6.1 Redes Neuronales Artificiales

Un método que está siendo explorado para comparar resultados y definir su campo de aplicación es el de las Redes Neuronales Artificiales (RNA), las cuales tratan de simular las redes neuronales que existen en el cerebro humano. En el cerebro se realizan tareas complejas y no lineales (como reconcomiendo de patrones, control motriz, etc.) mediante la organización de sus neuronas en redes interconectadas capaces de

adaptarse a la tarea a realizar. Estas redes son las que se modelan en máquinas diseñadas de manera similar a como se organizan en el cerebro.

Las RNA son modelos estadísticos adaptativos que, a pesar de que no son inteligentes, pueden estimar los parámetros de alguna población utilizando un pequeño número de ejemplares (uno o pocos) a la vez [30]. Esto quiere decir que pueden reconocer los patrones y crear reglas simples para resolver problemas complejos. Estas simples reglas les permiten a las RNA generalizar y por lo tanto clasificar después de un entrenamiento con varias muestras similares [31]. Ya que sus capacidades de entrenamiento y aprendizaje son muy amplias, son utilizadas como herramientas estadísticas en diversos campos como la psicología, estadística, ingeniería, econométrica, e incluso física. Por lo mismo, también han sido utilizadas como modelos de procesos cognitivos en inteligencia artificial, siguiendo tal vez el camino de los HMM. Unos ejemplos de esto han sido [15] y [32], quienes implementaron modelos para el reconocimiento de gestos con movimientos.

2.6.1.1 Definición

Las redes neuronales se construyen a partir de unidades simples llamadas neuronas, por analogía con las neuronas que se encuentra en el cerebro. Las neuronas fueron conceptualizadas como componentes del cerebro desde [33]. Las neuronas vienen en diferentes formas y tamaños, pero se pueden reconocer como células con dos filamentos característicos, uno axón para transmitir impulsos, y uno o más dentritas para recibir estos impulsos a través de la sinapsis, que media la interacción de neuronas.

En las redes neuronales artificiales, las neuronas están unidas por un conjunto de conexiones ponderadas. El aprendizaje se logra generalmente mediante la modificación de los pesos de las conexiones, amplificando o atenuando las entradas. Cada unidad codifica o corresponde a una función o característica de un patrón que se quiere analizar o se desea utilizar como un factor de predicción.

$$y(x) = g\left(\sum_{i=0}^{n} w_i x_i\right)$$

(2.8)

La definición matemática de una RNA general está dada por la ecuación (2.8), donde x es la neurona con n entradas $(x_0 \ldots x_n)$ y una neurona de salida $y(x)$, y donde $(w_0 \ldots w_n)$ son los pesos. La función de activación g es la función que mide que tan alta debe ser la salida de la neurona. Esto es el resultado de la sumatoria de las entradas por sus pesos. La función de activación puede ser una simple función escalón, que devuelva 1 o 0.

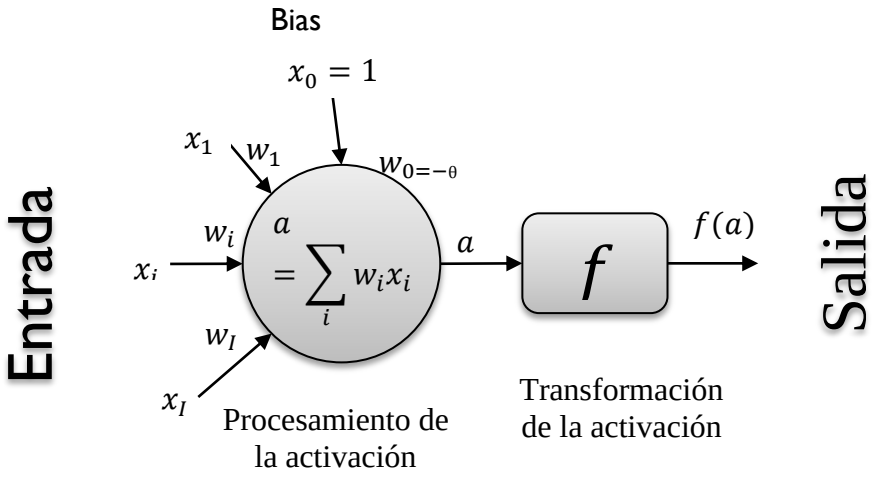


Figura 2.2 - Perceptrón (basada en [30])

2.6.1.2 Topología

Las RNA suelen organizar sus unidades en varias capas. La primera capa se llama la *capa de entrada*, la última la *capa de salida*. Las capas intermedias (si las hay) se llaman las *capas ocultas*. La información que será analizada por la red neuronal se alimenta a las neuronas de la primera capa y después se propagan a las neuronas de la segunda capa para su posterior procesamiento. El resultado de este proceso entonces se propaga a la capa siguiente y así sucesivamente hasta la última capa. Cada unidad recibe alguna información de otras unidades y procesa esta información, que se convertirá en la salida de la unidad. La Figura 2.2 muestra un perceptrón, es decir, una RNA de una sola capa.

La mayoría de las redes multicapa utilizan un algoritmo de aprendizaje por la regla de Widrow-Hoff[I] y la función logística como su función de transferencia de las unidades de la capa oculta (la función de transferencia es en general no-lineal para estas neuronas), que se verán más adelante.

2.6.1.3 Función de activación

Cualquier función cuyo dominio sea los números reales puede ser utilizada como una función de transferencia o activación [30]. Las funciones de activación se pueden dividir en cuatro categorías:

- Lineal ($o \propto a$) – La actividad de salida es proporcional a la salida total ponderado.
- Escalón ($\{-1,0,1\}$) – La salida se determina en uno de los dos niveles, dependiendo de si la entrada total es mayor o menor que cierto valor umbral.
- Sigmoidea - La salida varía continuamente, pero no de forma lineal, a medida que cambia la entrada. La más popular es la función logística, determinada por la fórmula (2.9) donde β es una constante.

$$\varphi(e) = \frac{1}{1 + exp(-\beta e)}$$

(2.9)

- Normal o Gaussiana – Esta función es ideal para unidades radiales, que es representada por la ecuación (2.10), donde σ es la desviación típica y (σ^2 corresponde a la varianza), a corresponde a la diferencia entre x y la media μ.

$$\boldsymbol{o = \left(\sigma\sqrt{2\pi}\right)^{-1} e\left\{-\frac{1}{2}\left(a/\sigma\right)^2\right\}}$$

(2.10)

Las unidades sigmoideas se parecen más a las neuronas reales que las unidades lineales o de umbral, pero los tres deben considerarse aproximaciones.

[I] También conocida como la regla Delta, la regla de Widrow-Hoff utiliza la diferencia entre la entrada actual de la neurona y la salida esperada como una señal de error para las unidades en la capa de salida.

2.6.1.4 Aprendizaje

El objetivo de la red neuronal es analizar, aprender, descubrir alguna relación entre los patrones de entrada y los de salida, o encontrar la estructura de los patrones de entrada. Gracias a que las redes neuronales son dispositivos estadísticos adaptativos, se pueden alterar iterativamente los valores de sus parámetros (pesos) como función de su desempeño.

El proceso de aprendizaje se logra mediante la modificación de los pesos de conexión entre las unidades en cada iteración del entrenamiento para satisfacer los valores de entrada. Los pesos entre las capas ocultas determinan la bondad del desempeño de la RNA. En términos estadísticos, esto es equivalente a interpretar el valor de las conexiones entre las unidades como parámetros (por ejemplo, como los valores de a y b en la ecuación de regresión $\hat{y} = a + bx$) a estimar. El proceso de aprendizaje especifica el "algoritmo" que se utiliza para estimar los parámetros.

Estos cambios se realizan de acuerdo a las reglas de aprendizaje que pueden ser clasificadas como supervisadas (*ver sección* 2.6.1.4.1) (cuando la salida deseada es conocida y se utiliza para calcular una señal de error) o no supervisadas (*ver sección* 2.6.1.4.2) (cuando no se utiliza señal de error del). [34].

2.6.1.4.1 Supervisado

Este tipo de aprendizaje consiste en que, dado un set de entrenamiento de N muestras de pares de entrada-salida $(x_1, y_1), (x_2, y_2), \dots (x_N, y_N)$ donde cada y_j haya sido generada por una función desconocida $y = f(x)$, se encuentre la función hipótesis h que aproxime la verdadera función f. La hipótesis h entonces relacionará las entradas con sus respectivas salidas mediante iteraciones donde se ajusten los parámetros para disminuir el error, es decir, las diferencias entre los valores de salida real y los valores deseados.

Para medir la exactitud de una hipótesis se debe proporcionar un set de prueba cuyas muestras no estén incluidas en el set de entrenamiento. Se dice que una hipótesis ***generaliza*** cuando predice correctamente el valor de y para ejemplares nuevos.

Si la salida y se encuentra en un set finito de valores se considera al problema como de ***clasificación***, como el de este estudio, donde las salidas solo pueden ser los gestos a evaluar.

2.6.1.4.1.1 Backpropagation

El algoritmo de backpropagation tardó 30 años en popularizarse como una forma de entrenar a las unidades ocultas, que resultó en una nueva ola de investigación de redes neuronales y aplicaciones.

Este algoritmo utiliza la diferencia entre la entrada real de la neurona y la salida deseada como una señal de error para las unidades en la capa de salida. Las unidades en las capas ocultas no pueden calcular directamente la señal de error, pero pueden estimarlo como una función (e.g. un promedio ponderado) del error de las unidades en la capa siguiente.

La corrección de los pesos sinápticos es proporcional a la señal de error multiplicada por el valor de la activación dada por la derivada de la función de transferencia. Utilizando la derivada tiene el efecto de hacer correcciones ajustadas cuando la activación está cerca de sus valores extremos (mínimo o máximo) y correcciones mayores cuando la activación se encuentra en su rango medio. Cada corrección tiene el efecto inmediato de hacer la señal de error más pequeño si una entrada similar se aplica a la unidad.

En general, las reglas de aprendizaje supervisado aplican algoritmos de optimización similares a técnicas de descenso, ya que buscan un conjunto de valores para los parámetros libres (i.e., los pesos sinápticos) del sistema, de tal manera que una función de error calculado para toda la red se minimiza [30]. Entre algunas optimizaciones se encuentran:

a) Incremental, que es el algoritmo de Backpropagation común, donde los pesos se ajustan por cada set de entradas y salidas, es decir, que el ajuste sucede varias veces por iteración [35],
b) Batch, donde los ajustes a los pesos sinápticos y los biases se realizan de epoch en epoch, dando por resultado que un estimado más exacto del vector gradiente es utilizado en los cálculos [36],

c) RPROP, o Resilient Propagation por [37], algoritmo donde el ajuste de cada peso se efectúa solo cuando hay cambios de signo de la derivada parcial respecto a la iteración anterior,
d) Quick Prop [38], método inspirado en el método de Newton que aplica la segunda derivada a la función de error para un mejor aproximación,

2.6.1.4.2 No Supervisado

La regla de Hebb[2] es la más conocida regla de aprendizaje no supervisado, está basada en el trabajo por el neuropsicólogo canadiense Donald Hebb, quien teorizó que el aprendizaje neuronal (es decir, el cambio sináptico) es un fenómeno local expresable en términos de la correlación temporal entre los valores de activación de las neuronas [30].

En concreto, el cambio sináptico depende tanto de actividades presinápticas como postsinápticas y define que el cambio de un peso sináptico es una función de la correlación temporal entre las actividades presináptica y postsináptica.

Específicamente, el valor del peso sináptico entre dos neuronas aumenta siempre que estén en el mismo estado y disminuye cuando se encuentran en diferentes estados.

2.6.1.5 FANN

Para estructurar el algoritmo que manipula la red neuronal, se hace uso de la librería para redes neuronales llamada FANN [39].

Esta librería nos permite configurar la red neuronal desde su creación, de tal manera que se pueda ir adaptando sus características como topología (capas, entradas, y neuronas en las capas ocultas), algoritmo de entrenamiento, función de activación y su pendiente en las capas ocultas y la de salida, interconectividad, aprendizaje, etc.

---

[2] También llamada *aprendizaje de coincidencia* o *aprendizaje Hebbiano*, fue sugerida por Donald Hebb a partir de estudios con neuronas reales. La regla de Hebb dice que "cuando un axón de una célula A está lo suficientemente cerca de una célula B, como para excitarla, y participa repetida o persistentemente en su disparo, ocurre algún proceso de crecimiento o cambio metabólico, en una o en ambas células, de modo tal que aumentan tanto la eficiencia de A como la de una de las distintas células que disparan a B" [46].

Por el tipo y cantidad de información que se obtiene de las muestras, se optó por una topología de tres capas, con seis neuronas en la capa oculta.

También se debe considerar el generar varios conjuntos de muestras, unos de entrenamiento, otros de validación y otros de prueba. De esta forma la red tendrá suficiente información sobre la cual trabajar. De la misma forma que se trabajó con la información de inicio, se manipulará esta información para tener como resultado las entradas a la red neuronal.

3 Metodología

El procedimiento que se siguió en este proyecto muestra en la Figura 3.1, donde básicamente se sigue un proceso de adquisición de muestras, preprocesamiento y clasificación.

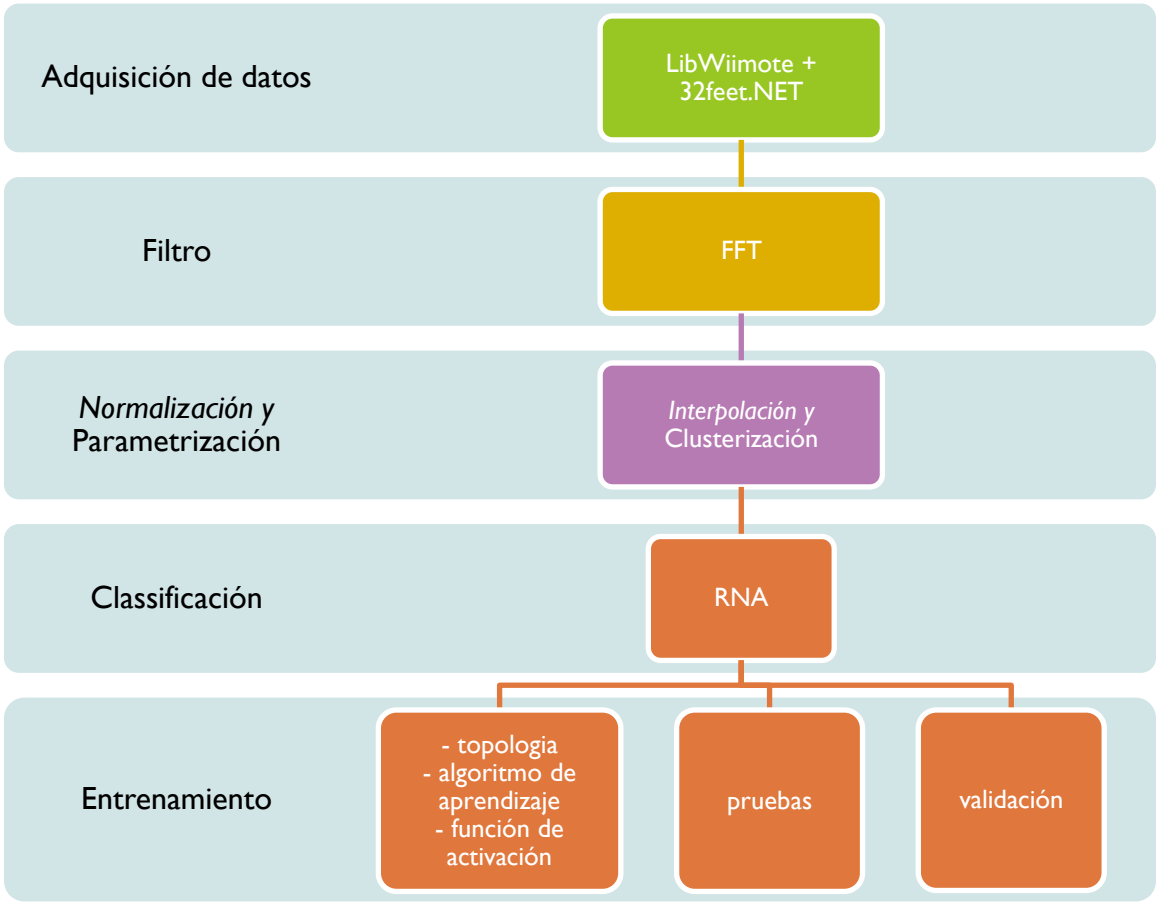


Figura 3.1 - Metodología

A continuación se describe la aplicación de la teoría mostrada en la sección anterior (2 Marco Teórico).

3.1 Adquisición De Datos

Mediante las librerías mencionadas en la sección 2.4, se puede obtener la información de los acelerómetros del Wiimote. Pero para esto, primero se definió la frecuencia de la captura de muestras.

Dado que los movimientos a evaluar duran a lo más 2.6 segundos[3], y por cada movimiento, la frecuencia de muestreo debe ser entonces de aproximadamente 60 por segundo. Esto es posible ya que, como se vio en la sección 2.3.2, el Wiimote contiene un acelerómetro con una capacidad de muestreo de 100Hz. Por lo tanto, se decidió aprovechar esta capacidad al máximo, y tomar muestras cada 10ms, como se muestra en la Tabla 1 (*ver anexo* 9.2), donde se muestran aceleraciones en x, y, z obtenidas del Wiimote.

X	Y	Z	*ms*
0.222	-0.154	0.880	**0**
0.259	-0.077	1.000	**10**
0.333	-0.154	1.040	**20**
0.370	-0.231	1.040	**30**
0.370	-0.308	1.080	**40**

Tabla 1 - Extracto de anexo 9.2

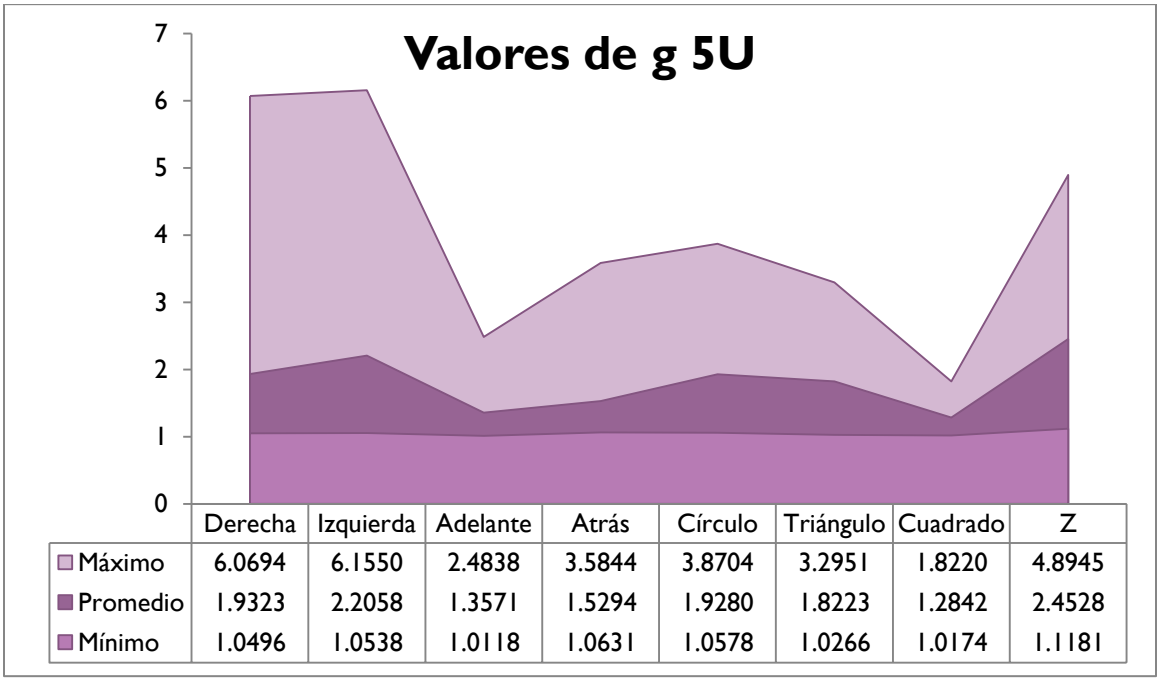
De esta captura se obtuvieron unas líneas de datos como se muestran en la Tabla 1, donde las primeras tres columnas corresponden a los valores de las aceleraciones en x, y y z, y la última columna, son los milisegundos en los que se hizo la captura.

Los primeros datos obtenidos fueron los provenientes de las muestras en crudo. Los datos que se evaluaron fueron los valores ***g*** y ***delta (Δ)*** por figura. Los datos se obtuvieron de ***5*** usuarios en ***8*** figuras que son ***círculo***, ***triángulo***, ***cuadrado***, ***derecha***, ***izquierda***, ***adelante***, ***atrás*** y ***Z*** (para una representación gráfica de los movimientos realizados, *ver anexo* 9.1).

[3] Basado en un promedio para la figura cuadrado (*ver anexo* 9.6)

3.1.1 Valores en g

Lo primero a evaluar son los datos de g (aceleración en x, y y z) de los que se obtuvieron estadísticas de valores en g por figura, apreciando valores máximos y mínimos.



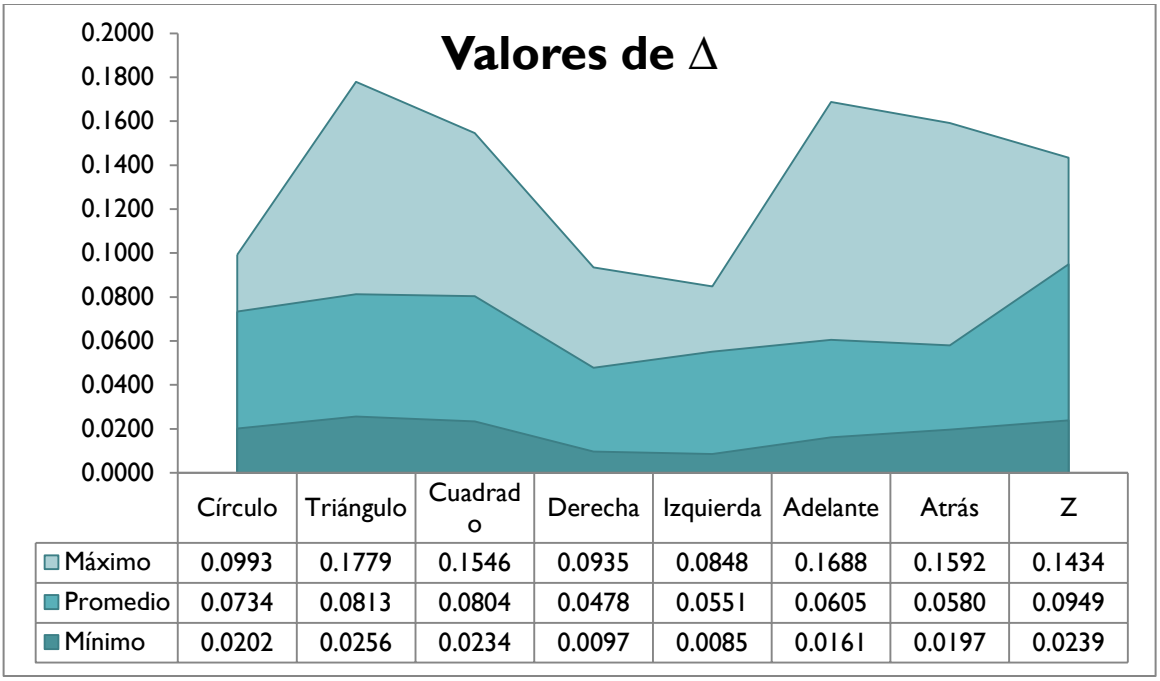
	Derecha	Izquierda	Adelante	Atrás	Círculo	Triángulo	Cuadrado	Z
Máximo	6.0694	6.1550	2.4838	3.5844	3.8704	3.2951	1.8220	4.8945
Promedio	1.9323	2.2058	1.3571	1.5294	1.9280	1.8223	1.2842	2.4528
Mínimo	1.0496	1.0538	1.0118	1.0631	1.0578	1.0266	1.0174	1.1181

Figura 3.2 - Valores en g de figuras muestreadas

En la Figura 3.2 podemos apreciar el gran distanciamiento entre los valores máximos y mínimos de las aceleraciones, siendo que los valores promedio nos dan datos más reales. Es por eso que esto nos indica que es necesario hacer un filtrado inicial para eliminar de los valores extremos (*ver sección* 3.2.1).

3.1.2 Valores en delta (Δ)

Por valores delta (Δ) denominamos a la diferencia en aceleración de un tiempo a otro en el momento de muestreo, como se define en la sección 3.2.1. Esto para detectar y descartar los valores que no aporten a la esencia de la función.



	Círculo	Triángulo	Cuadrado	Derecha	Izquierda	Adelante	Atrás	Z
Máximo	0.0993	0.1779	0.1546	0.0935	0.0848	0.1688	0.1592	0.1434
Promedio	0.0734	0.0813	0.0804	0.0478	0.0551	0.0605	0.0580	0.0949
Mínimo	0.0202	0.0256	0.0234	0.0097	0.0085	0.0161	0.0197	0.0239

Figura 3.3 - Valores de Δ de figuras muestreadas

Con la Figura 3.3 se puede apreciar el ruido en las muestras, determinado por los valores cuyo Δ sale muy fuera del promedio. Además con estos resultados se puede definir el Δ mínimo requerido por los movimientos, que resultó en un valor definido empíricamente entre el mínimo y le promedio, y el máximo, que por mucho pudiera ser el máximo ya obtenido en las muestras de entrenamiento.

3.1.3 Cambios de *g* por figura

También se obtuvo el promedio de datos por muestreo, que se requirió para definir los tamaños de las funciones a normalizar.

Estos resultados se pueden traducir en el tiempo requerido para realizar un gesto, ya que los muestreos fueron realizados a intervalos periódicos. Es decir, que el número de valores determina el tiempo para realizar cada uno. Por ello, los movimientos sencillos como *derecha* o *adelante* tienen la mitad de valores que los más complejos como *cuadrado* o *Z*.

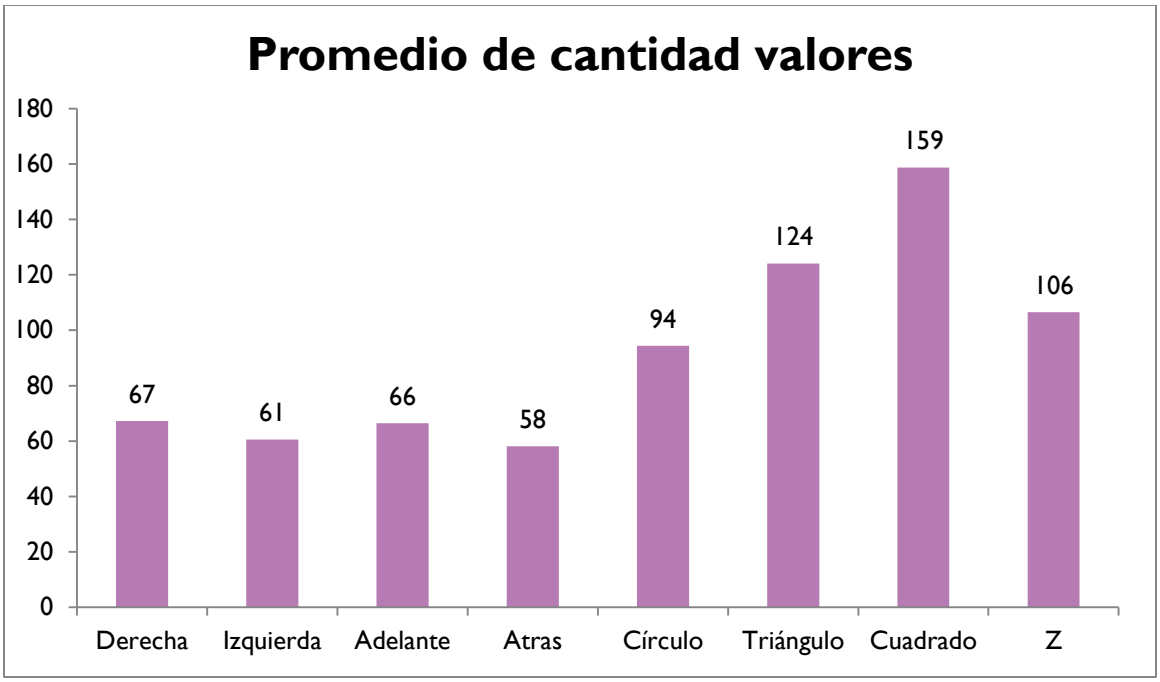


Tabla 2 - Promedio de cantidad de valores (equivalente a tiempo en *s*) por figura de un promedio de un promedio de 5 usuarios diferentes

De esta forma, al ver que los movimientos más simples (derecha, izquierda, adelante y atrás) se realizaban en un tiempo muy corto, se decidió tomar en cuenta solo las figuras más complejas para las siguientes pruebas.

3.2 Preprocesamiento

Para analizar los vectores de información obtenidos del Wiimote y poder reconocer los patrones, se debió preprocesar la información para obtener parámetros definidos como número de muestras y tamaño de las mismas.

De entrada, las muestras, como cualquiera otra evaluación, contienen ruido y datos irrelevantes a este estudio, por lo que el primer proceso llevado a cabo fue el desarrollo de un filtro que obtuviera solo los datos que definen el movimiento realizado.

Así mismo, debido a que el acelerómetro devuelve múltiples valores en un periodo de tiempo muy corto y que no todas las muestras tienen la misma duración (y por tanto, la misma cantidad de datos) se decidió aplicar un *parametrizador* que sin importar el número de entradas, devuelva un número fijo de salidas. Estas salidas fueron entrada

para nuestra red neuronal, que previamente se estructuró para este número fijo de entradas.

3.2.1 Filtrado

El primer proceso que afecta las muestras es un filtro que elimina valores redundantes o extremos, es decir, que no contribuyen de manera significativa a la esencia de la función.

Dada una matriz de puntos de puntos $A = \begin{bmatrix} x_1 & y_1 & z_1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{bmatrix}$ que se tiene por la trayectoria de cada movimiento realizado, se busca eliminar los valores que cumplan con dos condiciones,

- Que la magnitud del vector de aceleración no esté fuera del límite g, esto es

$$\|A_i\| > g$$

- Que la diferencia respecto al punto más cercano sea mayor a Δ, es decir, que la distancia euclidiana entre estos dos puntos fuera mayor a Δ, esto es

$$|\|A_i\| - \|A_{i-1}\|| > \Delta$$

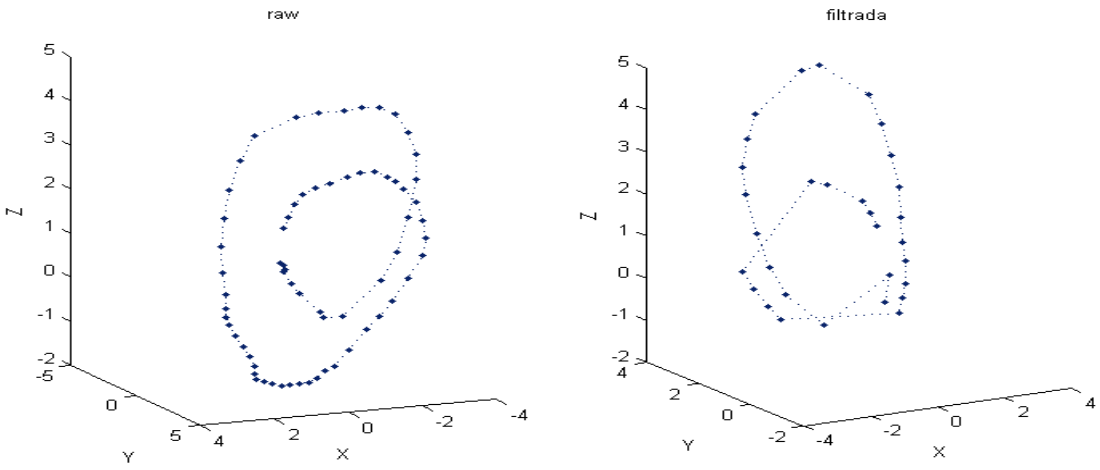


Figura 3.4 - Conjunto de puntos de un círculo filtrado por límites y similitudes

Los valores g y Δ fueron definidos empíricamente basándose en los resultados de los datos Figura 3.2 y Figura 3.3: $\boldsymbol{g = 1.0}$ y $\Delta = \mathbf{0.5}$. Con esto se obtuvo una matriz reducida como se muestra en la Figura 3.4, donde se eliminaron los puntos que

sobrepasan el límite determinado, o que son relativamente similares a los puntos cercanos, reduciendo la muestra de 74 a 31 para este ejemplo, esto es, el 41%.

3.2.1.1 Transformada Rápida de Fourier (FFT)

Una forma muy sencilla de definir un movimiento (determinado por valores en x, y, z) sería obtener el máximo, mínimo, y media de cada eje, obteniendo lo siguiente [40]:

$$x_{min},\ x_{max},\ x_{med},\ y_{min},\ y_{max},\ y_{med},\ z_{min},\ z_{max},\ z_{med}$$

Sin embargo este filtro es insuficiente pues tiene un rango de error muy alto al momento de hacer a clasificación.

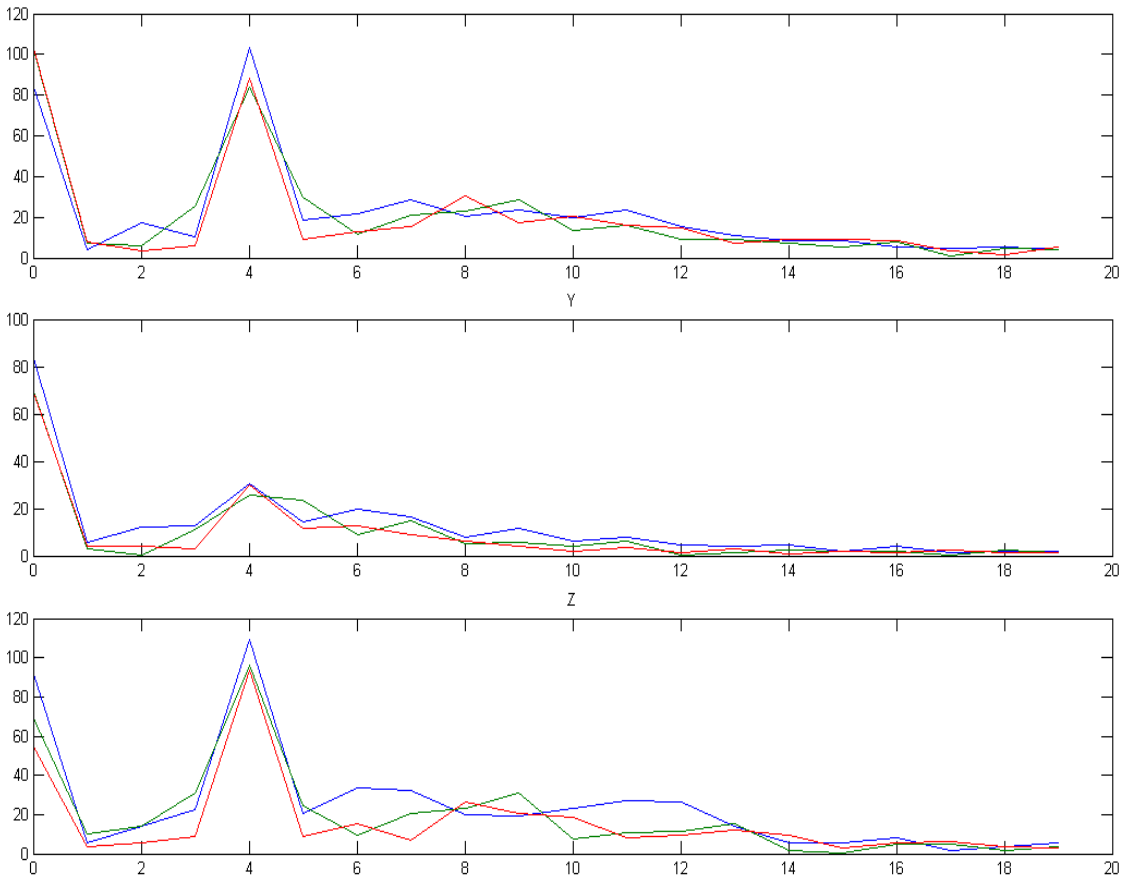


Figura 3.5 - Espectro de Fourier de una muestra de una trayectoria de un círculo

En un intento por evaluar las funciones y encontrar patrones entre las diversas figuras, se aplicó la FFT a algunas de las muestras tomadas al azar.

La FFT se utilizó para evaluar la función resultante de cada movimiento, y obtener las armónicas características.

La FFT nos ayuda a extraer las características esenciales de la función. La parametrización con Fourier se hace por aparte para cada uno de las aceleraciones de los ejes X, Y, Z.

Nótese que cada movimiento tiene en el espectro de Fourier un valor medio en la intersección de la curva con el eje Y, valles y armónicas sucesivas, lo cual sirve para caracterizar el tipo de movimiento.

En particular se puede caracterizar el movimiento de un círculo por la posición y la amplitud de las armónicas dominantes para cada una de las aceleraciones tomadas en los ejes X, Y, Z.

En un principio esta información sería tomada para el procesamiento y entrenamiento con una RNA. Sin embargo solo sirvió para evaluar y corroborar que la información que se estaba analizando era la correcta.

3.2.2 Normalización

Una vez que se descartaron los valores de las muestras que no aportan algo significativo a la función, se normalizó el tamaño de las matrices a un valor predeterminado para todos los gestos. Esto se hizo para facilitar el procesamiento y clasificación de todos los datos.

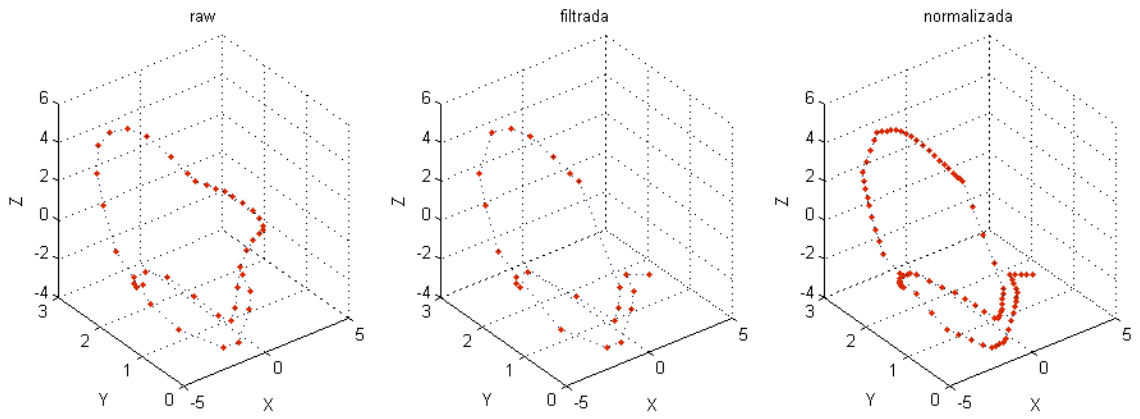


Figura 3.6 - Progresión de la trayectoria a través de un filtro y normalizacion

Existen dos casos a considerar al obtener las muestras, uno en el que se requiere reducir la matriz, y otro en el que se necesita interpolar los valores faltantes. Para este segundo caso se aplicó una interpolación lineal que obtiene los valores intermedios faltantes para completar el número. Para esto se definió una longitud de valores de $n = 40$ por deducción de los valores obtenidos y mostrados en el anexo 9.6. En la Figura 3.6 se muestra la progresión de los valores de entrada, pasando por el filtro, y por último, normalizando a la longitud preestablecida.

3.2.3 Clusterización

Como en [14], se optó por utilizar el algoritmo de K-medias para reducir la cantidad de valores que se ingresarán a la red neuronal y simplificar el trabajo de la misma.

La Figura 3.7 muestra la trayectoria de un movimiento circular representada por los centroides de los 15 clústeres.

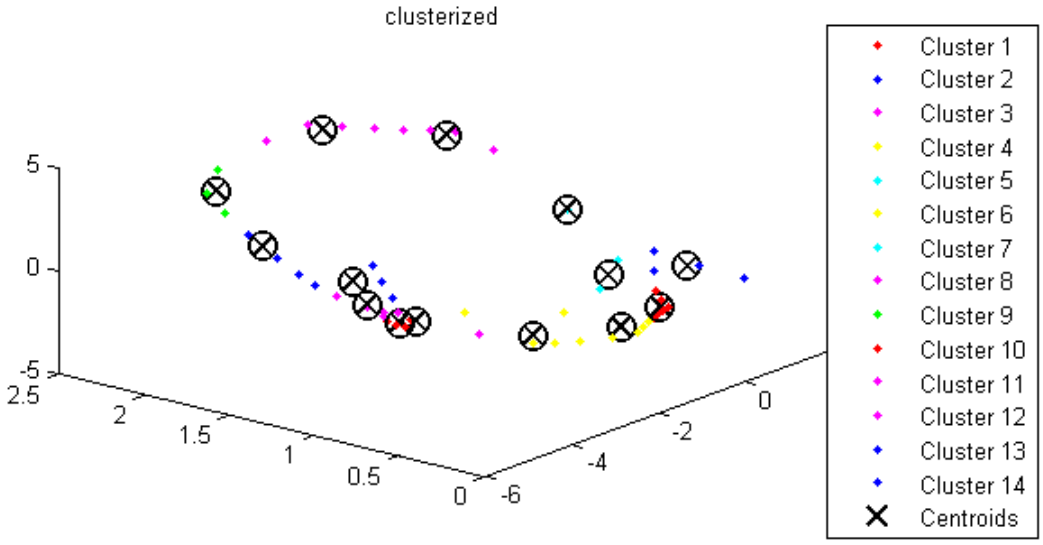


Figura 3.7 - Centroides de un clústeres de un movimiento circular

Para este proyecto, y después de varias pruebas, se determinó que el valor ideal es $\mathbf{k = 5}$, ya que este valor, multiplicado por cada eje (3) resulta en ***15*** entradas para la red neuronal.

4 Resultados

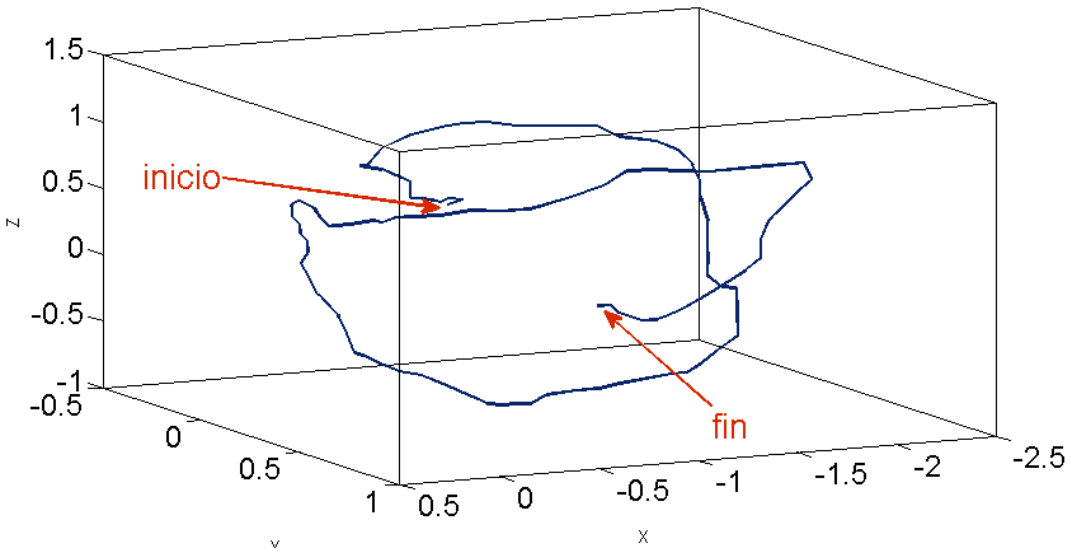


Figura 4.1 - Trayectoria de movimiento circular tomada con Wiimote

En la Figura 4.1 se puede apreciar el punto del inicio y del final de un movimiento circular, al igual que su dirección y trayectoria. Sin embargo se observa que la imagen a primera vista es poco comprensible, pues incluye cambios bruscos de concavidad, ruido y trazos no uniformes.

4.1 Gráficas de figuras

La Figura 4.2 muestra la representación gráfica en 3D de una muestra de cada uno de los movimientos. Se puede apreciar cómo cada gesto se distingue de los demás por una trayectoria característica, definida por su magnitud y complejidad, entre otros rasgos. Mientras que trazos simples se muestran en las primeras dos filas, i.e. derecha, izquierda, adelante, atrás, movimientos más complejos (círculo, triángulo, cuadrado, Z) se muestran en la mitad inferior de la tabla. Así se muestra el gesto hacia la izquierda como una gráfica circular parcial, en tanto que el cuadrado no asemeja a simple vista ninguna función conocida.

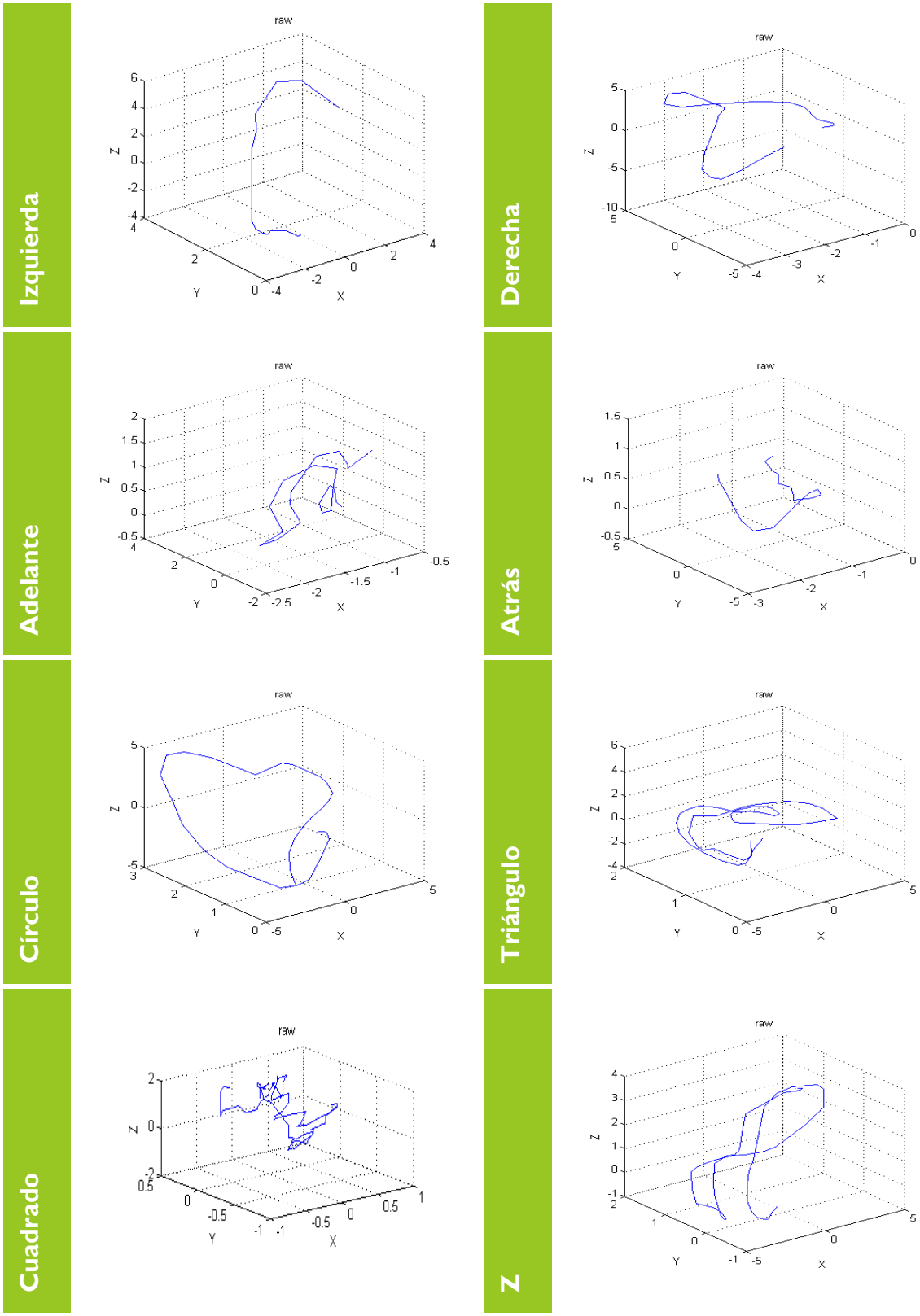


Figura 4.2 - Gráficas de movimientos por gesto

4.2 Comparación de resultados por algoritmo de entrenamiento

Una vez definidas todas las precondiciones y parámetros necesarios, se procedió a evaluar el desempeño de la red neuronal en diversas configuraciones, combinando algoritmos de entrenamiento, funciones de activación y diversas configuraciones de topología (todas feedforward).

Todas las topologías que se compararon cuentan con tres capas, una de entrada, una de salida y una oculta. El número de neuronas en las capas de entrada y de salida no varía, solo las de la capa oculta.

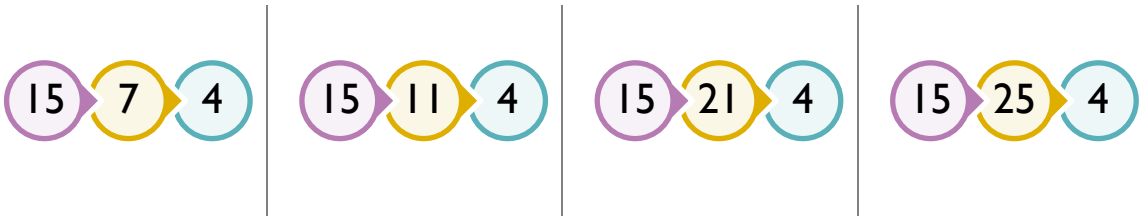


Figura 4.3 - Representación de las topologías evaluadas

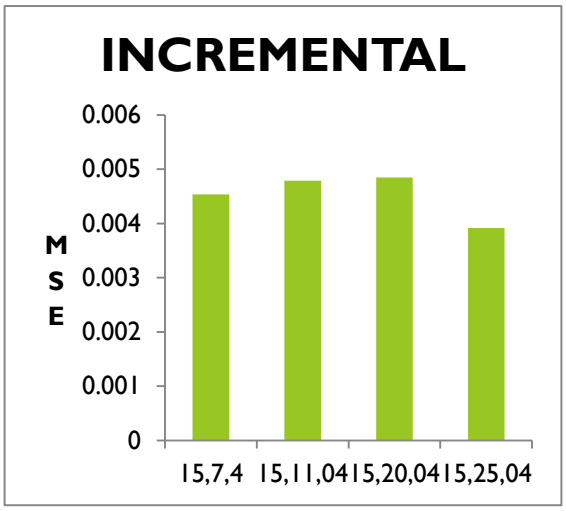
En la Figura 4.3 se muestra una representación simbólica de las topologías que se evaluaron, donde cada forma representa una capa y el número en su interior el número de neuronas en dicha capa. La capa de entrada se muestra con el color ***lila***, la capa de salida con el ***azul*** y la oculta con el ***naranja***.

Al elegir el algoritmo de entrenamiento, los parámetros determinantes fueron el error cuadrático medio (MSE, por sus siglas en inglés *Mean Squared Error*) y el Bit Fail, es decir, el número de neuronas, o bits, de salida con valores diferentes a la salida deseada.

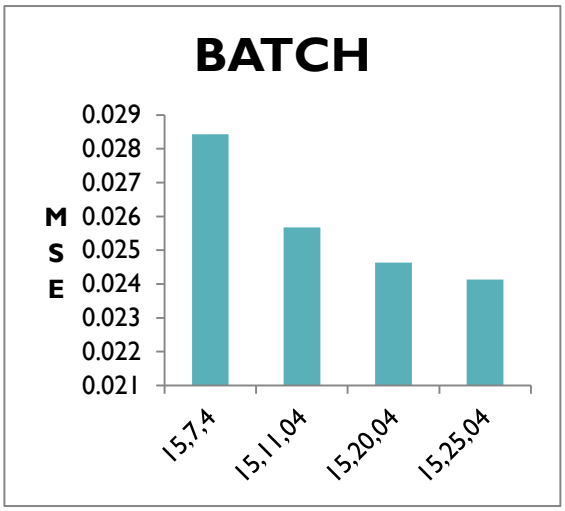
Los algoritmos a evaluar fueron todos del tipo backpropagation, por su eficiencia y buenos resultados en la mayoría de los casos [31].

La Figura 4.4 muestra cuatro gráficas mostrando el desempeño de cada algoritmo de entrenamiento a evaluar aplicado a las topologías de la Figura 4.3. La subfigura 4.4 a) muestra un desempeño relativamente uniforme del algoritmo incremental a través de las diversas topologías, siendo el máximo MSE de la topología ***15-20-4*** con ***0.004848*** y el

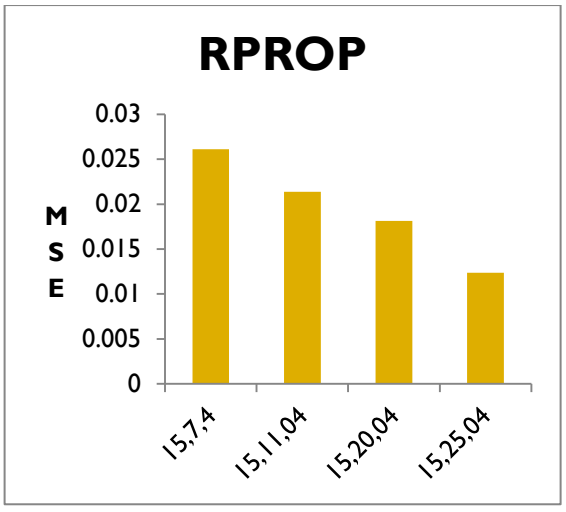
mínimo de ***0.003913*** de la topología ***15-25-4***. En tanto, los demás algoritmos muestran una mejora significativa conforme disminuye el número de neuronas de la capa oculta. El algortimo Batch, en la subfigura 4.4 b) decrementa el MSE en un ***9.68%*** al aumentar ***4*** neuronas a la capa oculta de la primer topología, disminuye otro ***4.04%*** al aumentar otras ***9*** neuronas a la misma capa y por ultimo ***2.07%*** al aumentar ***5*** neuronas respecto de la última topología.



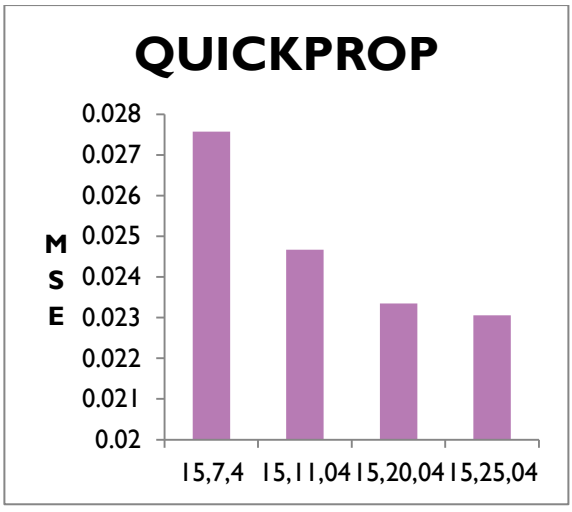
a) MSE del algoritmo Incremental



b) MSE del algoritmo Batch



c) MSE del algoritmo Rprop



d) MSE del algoritmo Quickprop

Figura 4.4 - MSE de diferentes algoritmos aplicados a diferentes topologías

La Figura 4.5 muestra el resultado del cálculo del error con las topologías en la Figura 4.3 y los algoritmos incremental, batch, rprop, y quickprop.

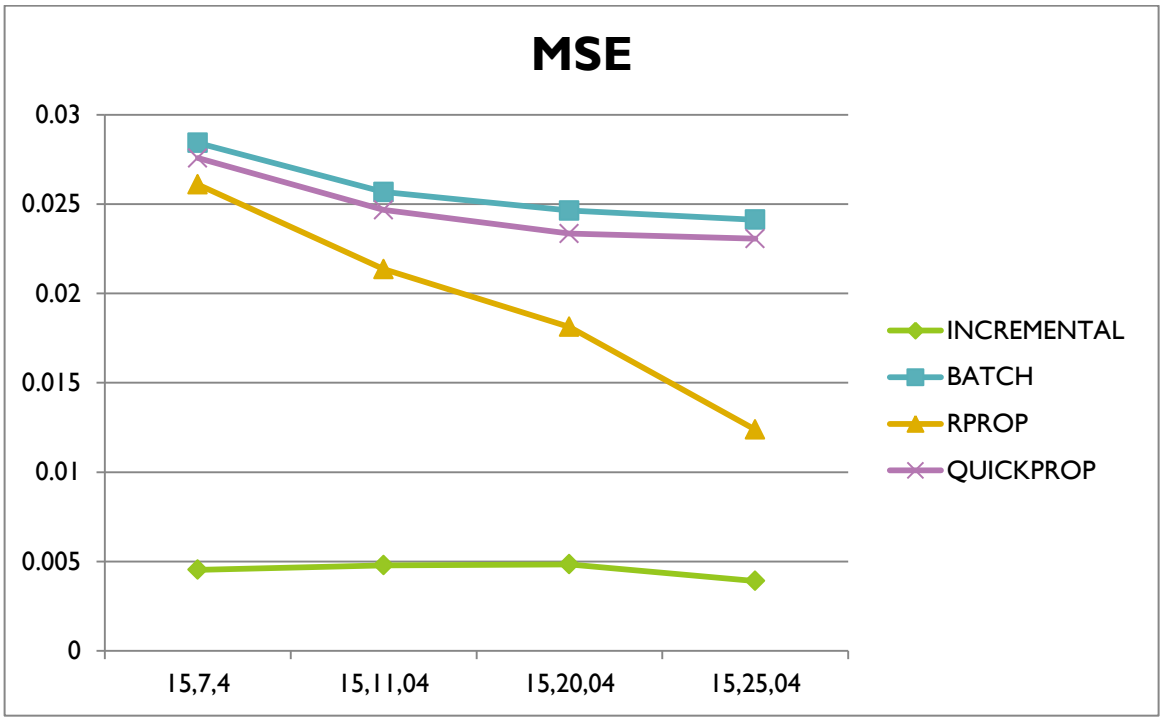


Figura 4.5 - Comparación de error cuadrático medio por topología por algoritmo de entrenamiento

Así mismo, se evaluó el conteo de neuronas, o bits, erróneos en la capa de salida. La Figura 4.6 muestra una gráfica comparativa de los algoritmos aplicados a las topologías de la Figura 4.3, donde se muestra como al igual que el MSE, el bitfail desciende conforme aumenta el número de neuronas en la capa oculta. En esta comparativa, el mejor desempeño lo tuvo el algoritmo Batch en las topologías ***15-11-04*** y ***15-20-4***.

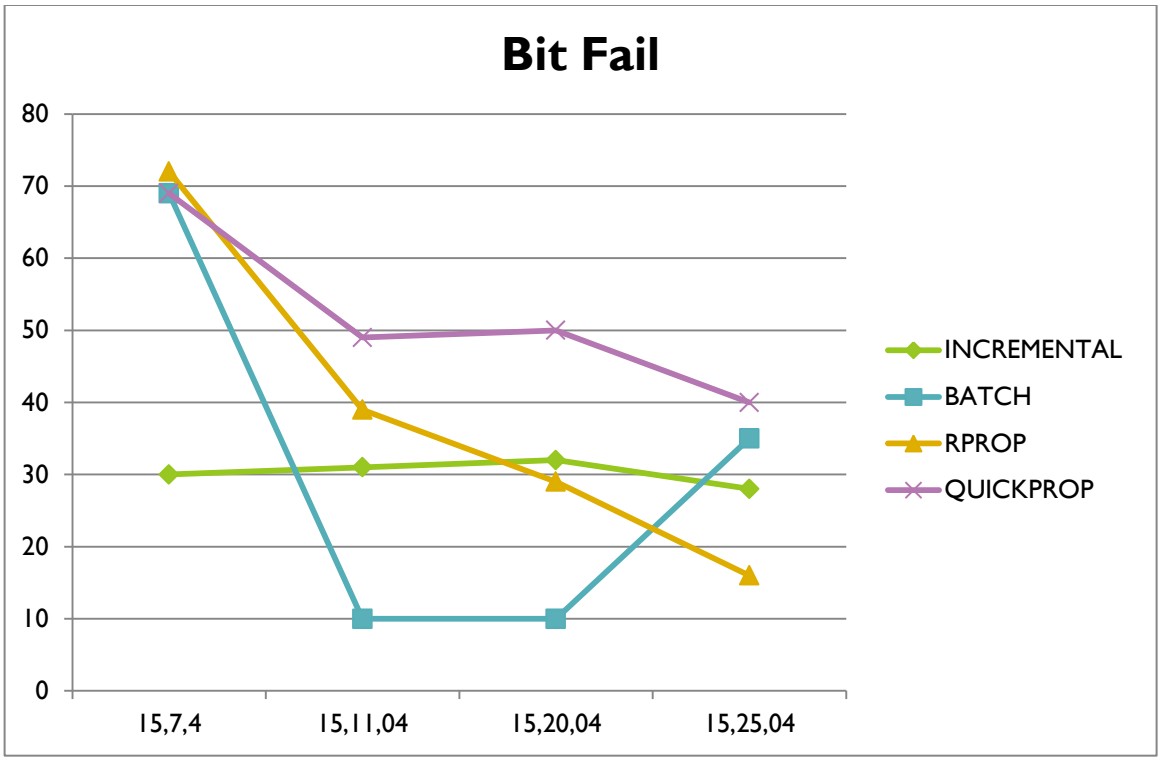


Figura 4.6 - Comparación de neuronas con error por topología por algoritmo de entrenamiento

4.3 Resultados con valores de prueba

Después los datos se entrenaron con la mejor configuración, que fue ***15-25-4***, que se muestra en la Figura 4.7, con el algoritmo ***RPROP***, y se evaluó el desempeño por figura específicamente.

El desempeño total fue de un error medio de tan solo ***0.048254862***. En la Figura 4.8 se muestran los resultados por figura.

Cada gráfica representa los valores de salida de cada una de las cuatro neuronas de salida después de correr la red neuronal con muestras nuevas, diferentes a las de entrenamiento.

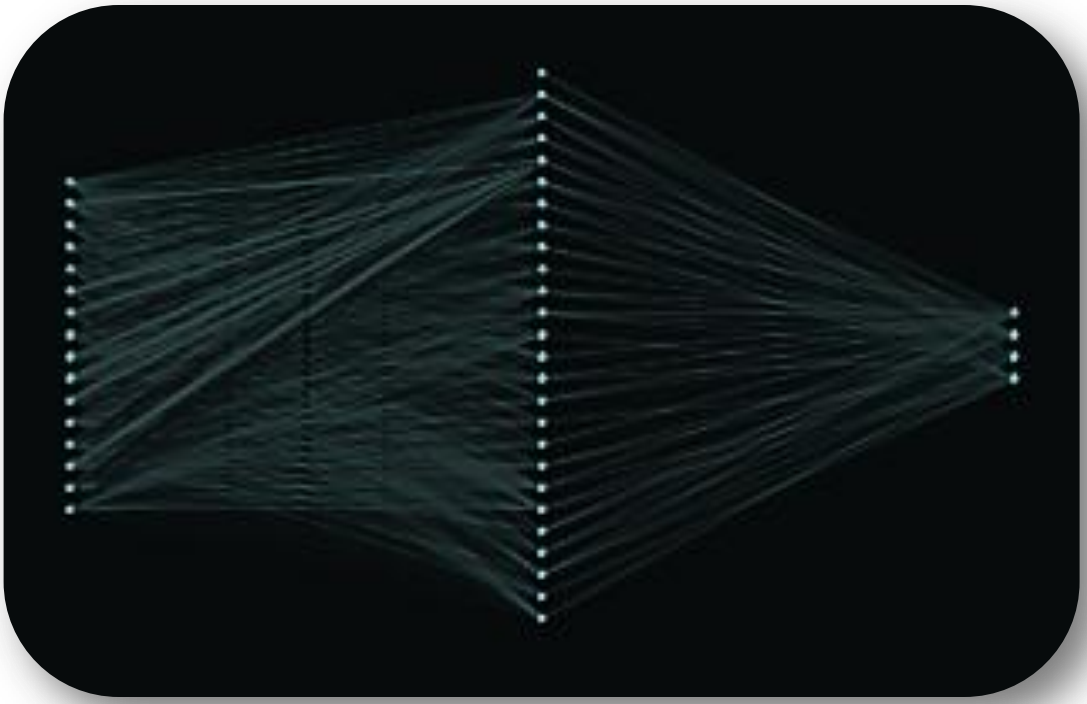


Figura 4.7 - Red neuronal resultado del entrenamiento por RPROP

Las muestras se entregaron en el mismo orden que las neuronas de salida a la red neuronal, es decir que las primeras muestras corresponden a círculos, las segundas a triángulos, las terceras a cuadrados y las ultimas a Z´s.

Entonces, se puede explicar que para los primeros valores en el eje de las x, los círculos, las únicas neuronas que deberían de tener valor cercano a 1 porque deberían ser las primeras de la gráfica círculo, por lo que cualquier otro valor podría significar un error, como en la gráfica Z, donde un valor de los círculos se dispara cerca del 1. Sin embargo, esto no afecta, ya que los valores en la neurona de salida del cuadrado son más altos.

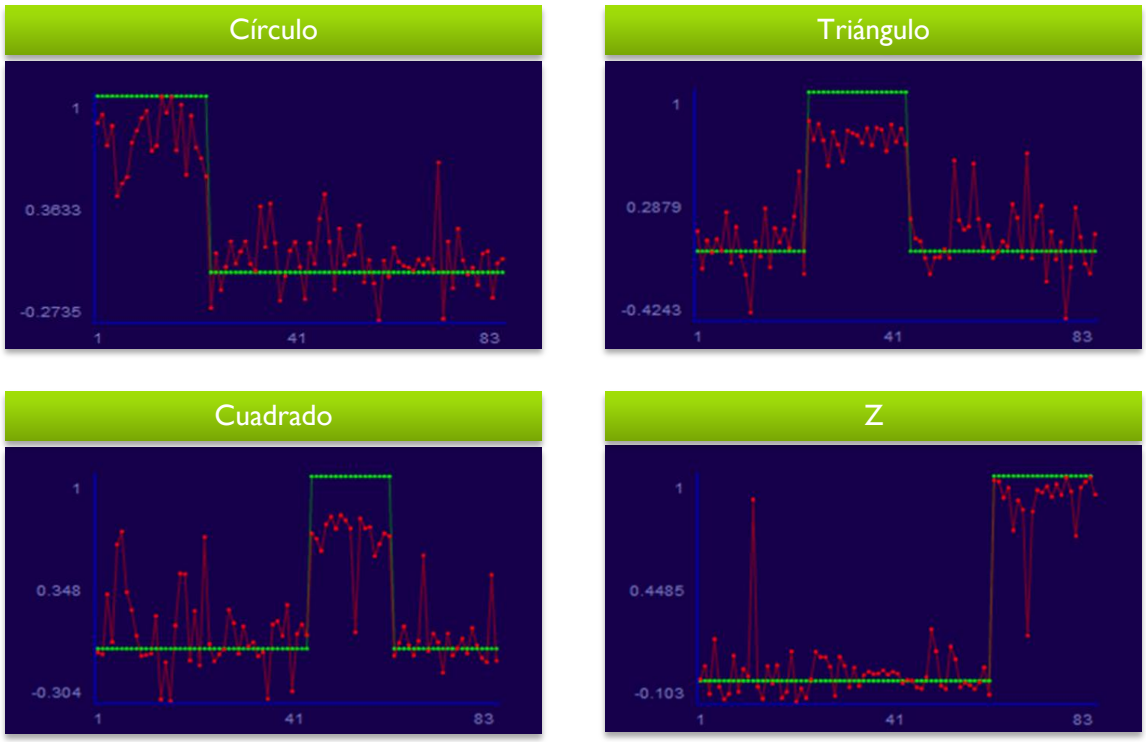


Figura 4.8 - Gráficas representativas de la respuesta de los valores de prueba

4.3.1 Matriz de confusión

Una matriz de confusión nos permite identificar un despeño un tanto vago y donde podría estar desviándose la clasificación con cierto patrón. Esto quiere decir que nos una matriz de este tipo nos puede mostrar los gestos, que por su misma naturaleza, puedan llegar a interpreta como otro gesto parecido, como puede ser el caso del círculo y el cuadrado.

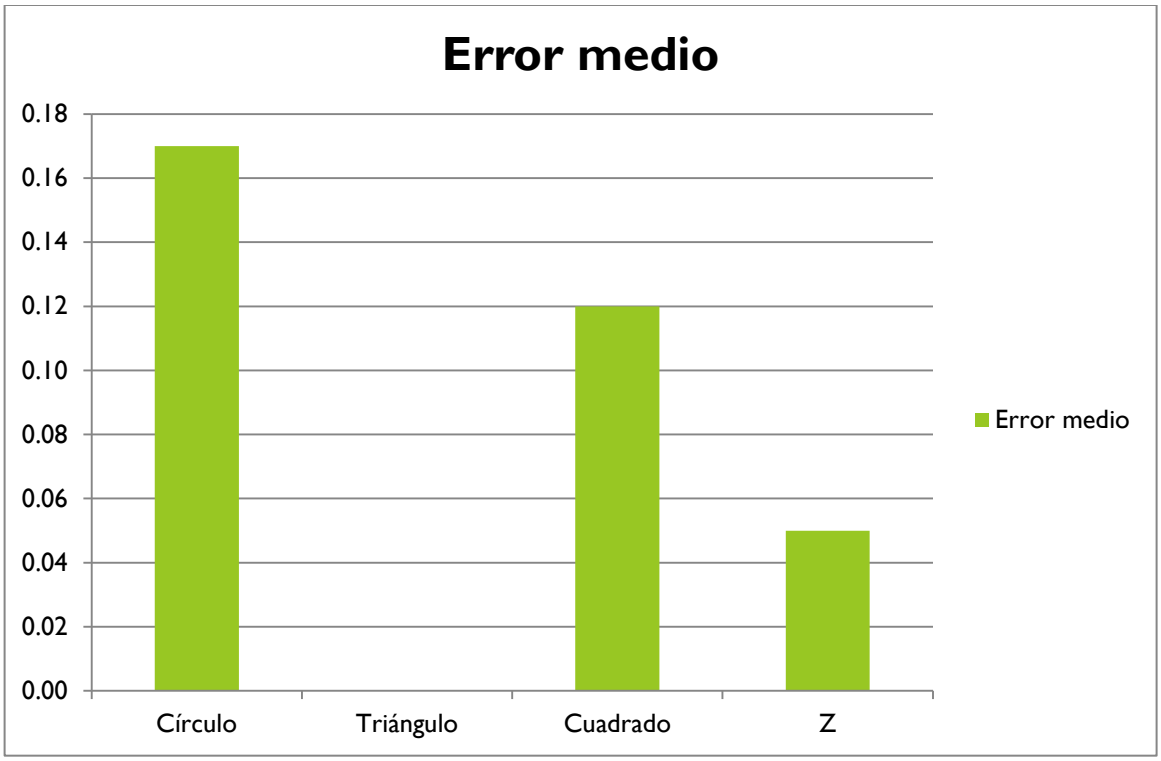


Figura 4.9 - Gráfica de Error medio por figura con valores de prueba

En la Figura 4.9 se muestra el error por figura debido a malinterpretación de red neuronal. En el caso del cuadrado por ejemplo, ***0.12%*** de las figuras no devolvieron el valor esperado.

	Círculo	Triángulo	Cuadrado	Z
Círculo	19	0	3	1
Triángulo	0	21	0	0
Cuadrado	0	2	15	0
Z	0	1	0	21

Tabla 3 - Matriz de confusión

En la Tabla 3 se muestra desglosado los resultados que regresan los valores de prueba, donde se pueden ver cuales gestos se confunden con cuales como el cuadrado y el círculo, llegando a marcar falsos positivos en tres ocasiones. En la Tabla 4 se muestran los mismos valores pero proporcionales al total de muestras.

	Círculo	Triángulo	Cuadrado	Z
Círculo	0.83	0	0.13	0.04
Triángulo	0	1	0	0
Cuadrado	0	0.12	0.88	0
Z	0	0.05	0	0.95

Tabla 4 - Matriz de confusión normalizada

En base a los resultados totales, el desempeño se traduce en un ***0.915746***, siendo ***1*** un desempeño perfecto, más no ideal. Se puede ver el anexo 9.7 para los resultados por muestra al correr la red neuronal.

4.4 Aplicaciones Similares

Existen aplicaciones similares al proyecto presentado en este documento, sin embargo la mayoría son comerciales y costosas. A continuación se describen algunas.

4.4.1 LiveMove AiLive

LiveMove AiLive es una serie de productos cuyo sistema fue introducido en el 2006, utilizando la tecnología de máquinas de aprendizaje para facilitar el desarrollo de paquetes de reconocimiento de movimiento para juegos individuales. Los desarrolladores pueden utilizar este software para construir clasificadores de ejemplos proporcionados de propuestas específicas que se incluirán en cada juego. El sistema LiveMove se utiliza para el desarrollo de juegos para la consola Wii. [41] Aunque inicialmente se lanzó para los desarrolladores profesionales de los juegos de Wii, el software fue diseñado para su uso por desarrolladores independientes, así, con el objetivo de ampliar las posibilidades de la nueva consola y el Wiimote.

5 Conclusiones

En los últimos años se ha investigado extensamente el reconocimiento de patrones, en particular de gestos. Sin embargo, aún no se establecen reglas ni métodos óptimos para cada una de las posibles aplicaciones. La tecnología actual se aplica en diversos dispositivos con técnicas diferentes para alcanzar los objetivos de reconocimiento de patrones o gestos utilizando plataformas multitouch. Entre dichos dispositivos que actualmente están alcance de todo público se encuentran el iPhone, el iPad, las tabletas Android y el Kinect. Todas estas plataformas implementaron una interfaz intuitiva y natural a través de gestos.

Dada la evidente tendencia tecnológica, se decidió explorar este campo mediante la aplicación de algoritmos de IA para desarrollar una herramienta de bajo costo y fácil acceso que pudiera aplicarse después en otros proyectos.

Es por eso que en este trabajo se estudiaron y analizaron diversos métodos para la adquisición, filtrado, parametrización, caracterización, clasificación y reconocimiento de patrones aplicado a gestos con un control Wiimote.

El fundamento de esta investigación ha sido la de la teoría actual en reconocimiento de patrones, en conjunto con métodos de filtrado y clasificación. Para efectos de este trabajo se enfocó a encontrar características específicas para los gestos evaluados y poder así crear una modelo general de clasificación.

Se comprobó que, a pesar de no proveer de mucha precisión, los resultados entregados por los acelerómetros contenían ruido de los mismos movimientos. Para lo cual el filtrado más simple y efectivo fue utilizado, ya que más adelante se procesaría la información. Es por eso que un filtro basado en límites y distancias euclídeas fue utilizado, para agilizar los procesos subsecuentes con datos libres de ruido.

En cuanto a la simplificación y caracterización, la transformada de Fourier proporcionó información útil acerca de las funciones de los diversos gestos. Sin embargo, dicha

información no fue utilizada para el modelado de la red neuronal. Por su parte, la clusterización por k-medias entregó resultados efectivos que sirvieron de entrada para la red neuronal.

Con los datos muestreados, se comprobó la eficiencia del algoritmo RPROP respecto a otras versiones de algoritmos de backpropagation, entregando mejores resultados en combinación de MSE y bitfail en las topologías evaluadas.

Finalmente, se podría aplicar está misma metodología a más gestos y usuarios para probar su desempeño, ya que, como se vio con las figuras de este trabajo, los resultados varían por gesto al punto de existir cierta confusión en los resultados de la red neuronal con valores de prueba que no estaban incluidos en el set de valores de entrenamiento.

Para aplicaciones similares, es posible que haya otros enfoques con igual o mejor desempeño. No obstante, en este proyecto de logró encontrar una combinación de herramientas que aplicadas dieron buenos resultados a los datos adquiridos.

5.1 Trabajo a futuro

El enfoque de este proyecto fue aplicar y comparar diferentes técnicas para analizar, probar y reconocer gestos de movimientos realizados con el Wiimote. La aplicación se basó en la teoría existente de reconocimiento de patrones.

Por el momento, se buscaron características específicas de cada gesto para poder modelar el sistema de clasificación. Aunque en este caso se obtuvieron buenos resultados, no es aplicable a todo tipo de muestras.

Se debe hacer una investigación a profundidad para encontrar el mejor modelado de la red neuronal la cual considere más factores como lo son características del usuario o ambiente, o propiedades de la misma función. Aunado a eso, se integrarán más muestras de otros usuarios y una gama más amplia de movimientos.

6 Referencias Bibliográficas

[1] Kurzweil, R.: The Age of Intelligent Machines. (1990)

[2] Howe, D.: pattern recognition. En: FOLDOC Free On-Line Dictionary of Computing. (Accessado el Septiembre 22, 1995) Disponible en: http://foldoc.org/pattern+recognition

[3] Hofmann, F., Heyer, P., Hommel, G.: Velocity Profile Based Recognition of Dynamic Gestures with Discrete Hidden Markov Models. En : Proc. of the International Gesture Workshop on Gesture and Sign Language in Human-Computer Interaction, London, p.81–95 (2004)

[4] Mäntyjärvi, J., Kela, J., Korpipää, P., Kallio, S.: Enabling fast and effortless customisation in accelerometer based gesture interaction. En : Proc. of the MUM '04, p.25–31 (2004)

[5] AiLive Inc. Disponible en: http://www.ailive.net

[6] Rudnicky, A., Hauptmann, A., Lee., K.-F.: Survey of current speech technology. Communications of the ACM 37(3), 52-57 (1994)

[7] Lecun, Y., Jackel, L., Bottou, L., Brunot, A., Cortes, C., Denker, J., Drucker, H., Guyon, I., Müller, U., Säckinger, E., Simard, P., Vapnik, V.: Comparison of Learning algortihms for Handwritten Digit Recognition. En : International Conference on Artificial Neural Networks (ICANN), Paris, France, pp.53-60 (1995)

[8] Nathan, K., Beigi, H., Subrahmonia, J., Clary, G., Maruyama, H.: Real-Time On-Line Unconstrained Handwriting Recognition Using Statistical Methods. En : International Conference on Acoustics, Speech, and Signal Processing (ICASP), Detroit, MI, USA, vol. 4, pp.2619-2622 (1995)

[9] Kendon, A.: Conducting Interaction: Patterns of behavior in focused encounters. Cambridge University Press, Cambridge (1990)

[10] McNeill, D., Levy, E.: Conceptual Representations in Language Activity and Gesture. John Wiley and Sons Ltd (1982)

[11] Nespoulous, J.-L., Perron, P., Roch Lecours, A.: The Biological Foundations of Gestures: Motor and Semiotic Aspects. Lawrence Erlbaum Associates, Hillsdale, MJ (1986)

[12] Card, S., Mackinlay, J., Robinson, G.: The design space of input devices. En : Proceedings of the CHI '90 Conference on Human Factors in Computing Systems, pp.117-124 (1990)

[13] Wolf, C., Morrel-Samuals., P.: The use of hand-drawn gestures for text editing. En : International Journal of Man-Machine Studies, vol. 27, pp.91-102 (1987)

[14] Poppinga, B., Schlömer, T., Henze, N., Boll, S.: Gesture Recognition with a Wii Controller. En : 2nd International Conference on Tangible and Embedded Interaction (TEI), Bonn, Germany (February 2008)

[15] Borza, P.-V.: Motion-based Gesture Recognition. Bachelor's Thesis, Babeș-Bolyai University of Cluj-Napoca, Romania (2008) http://code.google.com/p/accelges/.

[16] Prekopcsák, Z.: Accelerometer Based Real-Time Gesture Recognition. En : Proceedings of the 12th International Student Conference on Electrical Engineering (POSTER), Prague, Czech Republic (2008)

[17] Mardia, K., Ghali, N., Hainsworth, T., Howes, M., Sheehy, N.: Techniques for online gesture recognition on workstations. Image and Vision Computing 11(5), 283-294 (1993)

[18] Rubine, D.: Specifying gestures by example. Computer Graphics 25(4), 329-337

(1991)

[19] Murakami, K., Taguchi, H.: Gesture Recognition using Recurrent Neural Networks. En : Proceedings of the Conference on Human Factors in Computing Systems (SIGCHI): Reaching through Technology, New Orleans, Louisiana, United States, pp.237 - 242 (1991)

[20] Paradiso, J., Benbasat, A., Hsiao, K., Teegarde, Z.: Design and implementation of expressive footwear. IBM Systems Journal 39(3&4), 511–529 (2000)

[21] Starner, T., Pentland., A.: Visual recognition of American Sigh Language using Hidden Markov Models. En : IEEE International Symposium on Computer Vision (1995)

[22] Wiimote. En: WiiBrew. Disponible en: http://wiibrew.org/wiki/Wiimote

[23] Bluetooth SIG, Inc. En: bluetooth.com. (Accessado el Septiembre 2008) Disponible en: http://bluetooth.com

[24] 32feet.NET - In The Hand. (Accessado en 2009) Disponible en: http://32feet.net/

[25] Beedkar, K., Shah, D.: Accelerometer Based Gesture Recognition for Real Time Applications.

[26] Motion analysis. En: WiiLi.org Wii Linux. Disponible en: http://www.wiili.org/index.php/Motion_analysis

[27] Andersson, J. En: Simple Nintendo Wii Remote Framework for Linux. (Accessado el 2007) Disponible en: http://libwiimote.sourceforge.net/

[28] Peek, B.: WiimoteLib -.NET Managed Library for the Nintendo Wii Remote. (Accessado en Septiembre 4, 2009) Disponible en: http://www.wiimotelib.org/

[29] MacQueen, J.: Some Methods for Classification and Analysis of Multivariate Observations. En Cam, L., Neyman, J., eds. : Proceedings of Fifth Berkeley

Symposium on Mathematical Statistics and Probability, Berkeley, California, vol. 1, pp.281-297 (1967)

[30] Abdi, H.: Neural networks. En M., L.-B., Bryman, A., T., F., eds. : Encyclopedia of Social Sciences. Sage, Thousand Oaks (2003) 725-728

[31] Nissen, S.: Implementation of a Fast Artificial Neural Network Library (FANN). Thesis, University of Copenhagen, Copenhagen (Octubre 2003)

[32] Mlích, J.: Wiimote Gesture Recognition. En : Proceedings of the 15th Conference and Competition (Student EEICT), Brno, CZ, vol. 4, pp.344-349 (2009)

[33] Ramón y Cajál, S.: Histologie du Sytéms Nerveux de l´homme et des vertébrés. Maloine, Paris (1911)

[34] Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice Hall, Upper Sadle River (2010)

[35] Nissen, S.: Neural Networks Made Simple. Software Developer's Journal(02/2005), 14-19 (Febrero 2005)

[36] Haykin, S.: Neural Networks and Learning Machines 3rd edn. Prentice Hall, Ontario, Canada (2009) McMaster University.

[37] Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: The RPROP algorithm. En : Proc. of the IEEE Intl. Conf. on Neural Networks, San Francisco, pp.586-591 (1993) http://citeseer.nj.nec.com/riedmiller93direct.html.

[38] Fahlman, S.: Faster-Learning Variations on Back-Propagation: An Empirical Study. En : Proceedings of the 1988 Connectionist Models Summer School, Los Altos, pp.38-51 (1988) http://www-2.cs.cmu.edu/afs/cs.cmu.edu/user/sef/www/publications/qp-tr.ps.

[39] Nissen, S.: Fast Artificial Neural Network Library (FANN). (Accessado en 2008) Disponible en: http://leenissen.dk/fann/index.php

[40] Wassner, H.: Sujet du TP "neural computing". (Accessado el Febrero 1, 2008) Disponible en: http://professeurs.esiea.fr/wassner/?Ico4168-neural-computing

[41] Nintendo and AiLive Announce New Tool For Creative and Easy Wii Development. En: AiLive. (Accessado el Octubre 12, 2006) Disponible en: http://ailive.net/pressRelease-2006-10-12.html

[42] AiLive Inc.: LiveMove White Paper. (Accessado el 2006) Disponible en: http://www.ailive.net/papers/LiveMoveWhitePaper_en.pdf

[43] Delgado Mata, C., Ruvalcaba Manzano, R., Quezada Patiño, O., Gomez Pimentel, D., Ibanez Martinez, J.: Low cost video game technology to measure and improve motor skills in children. En : AFRICON, 2009. AFRICON '09. , Nairobi, Kenia, pp.1 - 6 (September 2009)

[44] En: Wii at Nintendo. Disponible en: http://www.nintendo.com/wii

[45] Press, W., Flannery, B., Teukolsky, S., Vetterling, W.: Numerical Recipes in C: The Art of Scientific Computing 2nd edn. Cambridge University Press (1992)

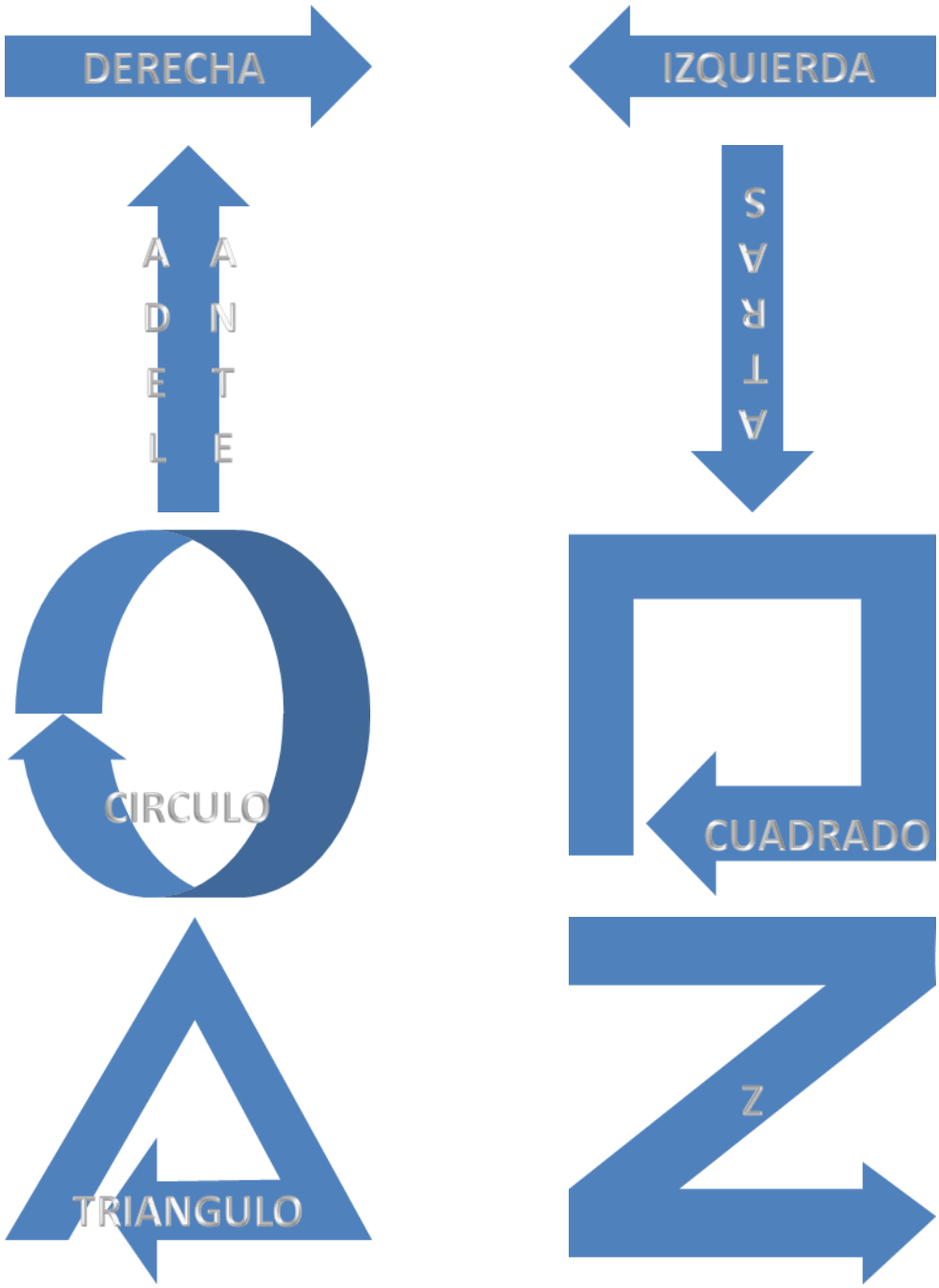
[46] Hebb, D.: The Organization of Behaviour: A Neuropsychological Theory. John Wiley & Sons, New York (1949)

7 Índice de Figuras

8 Índice de Tablas

9 Anexos

9.1 Representación visual de los movimientos realizados con el Wiimote



9.2 Puntos Círculo

0.037037	-0.153846	0.68	0.962963	3.730769	4.6
0.037037	-0.153846	0.68	2.148148	3.884615	4.28
0	-0.115385	0.64	2.148148	3.884615	4.28
0	-0.115385	0.64	2.481482	3.769231	3.72
-0.037037	-0.076923	0.56	2.703704	3.538461	3.04
0	0	0.52	2.740741	3.230769	2.36
-0.185185	0.076923	0.24	2.740741	2.961539	1.68
-0.185185	0.076923	0.24	2.592592	2.692308	1.04
-0.407407	0.038462	0	2.444444	2.461539	0.52
-0.925926	0.153846	-0.44	2.37037	2.230769	0.16
-0.925926	0.153846	-0.44	2.296296	2.076923	-0.08
-0.962963	0.307692	-0.56	2.148148	1.923077	-0.28
-1.407407	0.538462	-0.52	1.925926	1.769231	-0.56
-1.407407	0.538462	-0.52	1.666667	1.576923	-0.84
-2.333333	0.769231	0.24	1.407407	1.423077	-1.12
-2.333333	0.769231	0.24	1.259259	1.269231	-1.36
-2.703704	0.961538	0.88	1.222222	1.192308	-1.56
-2.962963	1.115385	1.68	1.148148	1.153846	-1.68
-3.111111	1.307692	2.56	0.925926	1.076923	-1.76
-3.037037	1.538462	3.16	0.666667	1.038462	-1.84
-2.703704	1.807692	3.72	0.407407	0.961538	-1.92
-2.259259	2.076923	4.2	0.148148	0.884615	-1.92
-1.703704	2.423077	4.44	-0.111111	0.846154	-1.92
-1.148148	2.769231	4.52	-0.407407	0.769231	-1.92
-0.518519	3.115385	4.56	-0.62963	0.769231	-1.84
0.259259	3.461539	4.6	-0.814815	0.846154	-1.68

-1	1	-1.56
-1.333333	1.153846	-1.2
-1.777778	1.230769	-0.76
-2.148148	1.230769	-0.48
-2.481482	1.269231	-0.16
-2.925926	1.230769	0.32
-3.333333	1.153846	0.8
-3.444444	1.076923	1.16
-3.37037	1.038462	1.56
-3.185185	1.038462	2
-2.851852	1.076923	2.32
-2.62963	1.076923	2.52
-2.407408	1.115385	2.64
-2.037037	1.153846	2.8
-1.666667	1.153846	2.8
-1.296296	1.115385	2.72
-0.851852	1.115385	2.6
-0.444444	1.115385	2.52
-0.148148	1.038462	2.4
0.037037	0.923077	2.16
0.185185	0.807692	1.88
0.259259	0.653846	1.6

9.3 Puntos Círculo Filtrado

0	0	0
0.03704	-0.1538	0.68
-1.4074	0.53846	-0.52
-2.3333	0.76923	0.24
-2.7037	0.96154	0.88

-2.963	1.11539	1.68
-3.1111	1.30769	2.56
-3.037	1.53846	3.16
-2.7037	1.80769	3.72
-2.2593	2.07692	4.2
0.25926	3.46154	4.6
0.96296	3.73077	4.6
2.48148	3.76923	3.72
2.7037	3.53846	3.04
2.74074	3.23077	2.36
2.74074	2.96154	1.68
2.59259	2.69231	1.04
2.44444	2.46154	0.52
2.37037	2.23077	0.16
2.14815	1.92308	-0.28
1.92593	1.76923	-0.56
1.66667	1.57692	-0.84
-2.1481	1.23077	-0.48
-2.4815	1.26923	-0.16
-2.9259	1.23077	0.32
-3.3333	1.15385	0.8
-1.2963	1.11539	2.72
-0.8519	1.11539	2.6
0.03704	0.92308	2.16
0.18519	0.80769	1.88
0.25926	0.65385	1.6

9.4 Puntos Círculo Normalizado (90)

0	0	0
0.01248	-0.0519	0.22921
0.02497	-0.1037	0.45843
0.02081	-0.1461	0.66652
-0.4661	0.08729	0.26202
-0.953	0.32066	-0.1425
-1.4282	0.54365	-0.5029
-1.7403	0.62144	-0.2467
-2.0524	0.69922	0.00944
-2.3458	0.77571	0.26157
-2.4707	0.84054	0.4773
-2.5955	0.90536	0.69303
-2.7154	0.96845	0.91596
-2.8027	1.02031	1.18562
-2.8901	1.07217	1.45528
-2.9713	1.12619	1.72944
-3.0212	1.19101	2.02607
-3.0712	1.25583	2.3227
-3.1061	1.32325	2.60045
-3.0811	1.40104	2.8027
-3.0562	1.47882	3.00494
-3.0108	1.55964	3.20404
-2.8985	1.65039	3.39281
-2.7861	1.74114	3.58157
-2.6638	1.83189	3.76315
-2.5139	1.92264	3.92494
-2.3641	2.0134	4.08674
-2.0046	2.21694	4.24045
-1.1556	2.68366	4.37528
-0.3067	3.15039	4.51011
0.33833	3.49179	4.6
0.57553	3.58254	4.6
0.81273	3.67329	4.6
1.15065	3.73552	4.49124
1.66251	3.74849	4.19461
2.17437	3.76145	3.89798
2.51144	3.73812	3.62831
2.58635	3.66033	3.3991
2.66126	3.58254	3.16989
2.70911	3.49352	2.94067
2.7216	3.3898	2.71146
2.73408	3.28608	2.48225
2.74074	3.18842	2.25303
2.74074	3.09767	2.02382
2.74074	3.00691	1.79461
2.71577	2.91616	1.57213
2.66583	2.82541	1.3564
2.6159	2.73466	1.14067
2.56596	2.65082	0.94652
2.51602	2.57303	0.77124
2.46608	2.49525	0.59596
2.4303	2.41746	0.45124
2.40533	2.33967	0.32989
2.38036	2.26188	0.20854
2.32543	2.16854	0.07101
2.25052	2.06482	-0.0773
2.17561	1.96111	-0.2256
2.10071	1.89023	-0.3398

2.0258	1.83838	-0.4342
1.95089	1.78652	-0.5285
1.86767	1.72602	-0.6229
1.78027	1.66119	-0.7173
1.69288	1.59637	-0.8117
0.76654	1.49525	-0.7551
-0.5194	1.37857	-0.6337
-1.8052	1.26188	-0.5124
-2.2305	1.24028	-0.4009
-2.3429	1.25324	-0.293
-2.4553	1.26621	-0.1852
-2.5963	1.25929	-0.036
-2.7462	1.24633	0.12584
-2.896	1.23336	0.28764
-3.0358	1.21003	0.44944
-3.1731	1.1841	0.61124
-3.3104	1.15817	0.77303
-2.7611	1.14304	1.33933
-2.0745	1.13008	1.98652
-1.3878	1.11711	2.63371
-1.1665	1.11539	2.68494
-1.0166	1.11539	2.64449
-0.8668	1.11539	2.60404
-0.5822	1.05704	2.46652
-0.2826	0.99222	2.3182
0.01706	0.9274	2.16989
0.08365	0.88678	2.07191
0.13358	0.84788	1.97753
0.18352	0.80899	1.88315
0.20932	0.75756	1.78876
0.23429	0.7057	1.69438
0.25926	0.65385	1.6

9.5 Centroides Círculo

1.26373	1.45932	0.50088
-2.3237	1.20922	1.56789
1.78409	3.39926	3.40247

9.6 Estadísticas de datos crudos

Tesis_data5U_8G_fc.mat											
		Derecha	*Izquierda*	*Adelante*	*Atras*	Circulo	Triangulo	Cuadrado	Z	Average ALL	Average shapes
g	ave	*1.9323*	*2.2058*	*1.3571*	*1.5294*	1.928	1.8223	1.2842	2.4528	**1.8139875**	**1.871825**
	min	*1.0496*	*1.0538*	*1.0118*	*1.0631*	1.0578	1.0266	1.0174	1.1181	**0**	
	max	*6.0694*	*6.155*	*2.4838*	*3.5844*	3.8704	3.2951	1.822	4.8945	**0**	
Δ	ave	*0.093061*	*0.10179*	*0.072732*	*0.098318*	0.091137	0.11672	0.090176	0.13886	**0.10034925**	**0.10922325**
	min	*0.017698*	*0.02714*	*0.023693*	*0.023233*	0.029323	0.028025	0.023395	0.03388	**0**	
	max	*0.20761*	*0.12661*	*0.17437*	*0.11631*	0.17663	0.16384	0.18245	0.13683	**0**	
count of files (dim)		***161***	***158***	***156***	***122***	**166**	**152**	**120**	**154**	**0**	
countf(mfcountges/dim)		17.2236	17.8101	12.4038	13.5246	31.5723	43.1645	34.4333	44.7403	**26.8590625**	**38.4776**
count (mcountges/dim)		*67.205*	*60.519*	*66.4679*	*58.1475*	94.3313	124.0855	158.7417	106.461	**91.9948625**	**120.904875**

9.7 Resultados de la red neuronal

Salida Real				Salida Esperada			
0.847668	0.126588	-0.021927	0.006886	1	0	0	0
0.896737	-0.109027	-0.031992	0.072183	1	0	0	0
0.719913	0.068963	0.315733	-0.064838	1	0	0	0
0.832098	-0.009273	0.039146	0.203442	1	0	0	0
0.433136	0.076592	0.605166	-0.028936	1	0	0	0
0.504563	0.004563	0.681430	-0.090943	1	0	0	0
0.539843	0.245866	0.327913	-0.063810	1	0	0	0
0.737184	-0.072865	0.224880	0.122942	1	0	0	0
0.804203	0.153859	0.075359	-0.053907	1	0	0	0
0.876499	-0.033340	-0.041533	0.056782	1	0	0	0
0.917884	-0.148325	-0.034868	0.021564	1	0	0	0
0.689818	-0.384626	-0.029605	0.884556	1	0	0	0
0.718990	0.058234	0.190109	-0.029072	1	0	0	0
0.996868	-0.033537	-0.294465	-0.088278	1	0	0	0
0.906783	0.269603	-0.077344	0.072565	1	0	0	0
0.996314	-0.100501	-0.303951	-0.014141	1	0	0	0
0.692467	0.146087	0.134575	0.075982	1	0	0	0

0.951537	0.056774	0.436888	-0.083752	1	0	0	0
0.553713	0.137987	0.433852	-0.054255	1	0	0	0
0.890324	0.021780	-0.067932	0.143821	1	0	0	0
0.708897	0.218868	0.219876	-0.103013	1	0	0	0
0.646634	0.502392	-0.096679	-0.037917	1	0	0	0
0.544824	-0.142573	0.647994	-0.084087	1	0	0	0
-0.202022	0.819953	0.027182	0.007265	0	1	0	0
0.109127	0.701989	-0.073797	0.141832	0	1	0	0
-0.100036	0.801622	-0.032516	0.115961	0	1	0	0
0.031207	0.697084	-0.003475	0.115659	0	1	0	0
0.176509	0.537543	0.226866	0.069067	0	1	0	0
0.049531	0.751430	0.149787	-0.074511	0	1	0	0
0.117599	0.670144	-0.029509	0.116697	0	1	0	0
0.176448	0.565002	0.130998	0.074083	0	1	0	0
0.047537	0.759610	0.013075	-0.031281	0	1	0	0
0.007887	0.741706	0.038867	0.065077	0	1	0	0
0.374981	0.728885	-0.043503	-0.026175	0	1	0	0
0.142921	0.680950	-0.019497	0.027272	0	1	0	0
0.390731	0.774603	-0.291901	0.044333	0	1	0	0
0.167602	0.666921	0.141979	0.032809	0	1	0	0
-0.159628	0.777085	0.160505	0.035833	0	1	0	0
-0.021406	0.764757	0.074236	0.050293	0	1	0	0
0.122604	0.629288	0.254209	0.030130	0	1	0	0
0.171189	0.796745	-0.246804	0.043254	0	1	0	0
0.031083	0.687294	0.086011	0.034828	0	1	0	0
-0.151469	0.771283	0.141622	-0.010523	0	1	0	0
0.166985	0.672224	0.079208	0.004593	0	1	0	0
0.048625	0.202692	0.670397	0.001280	0	0	1	0
0.305073	0.080228	0.638182	-0.032255	0	0	1	0

0.443630	0.063290	0.568545	-0.038131	0	0	1	0
0.173271	-0.042418	0.722094	0.018852	0	0	1	0
-0.098032	-0.144359	0.766461	0.251798	0	0	1	0
0.249030	-0.039315	0.697087	0.145066	0	0	1	0
0.042663	-0.037105	0.774754	-0.024355	0	0	1	0
0.094344	0.015034	0.745519	-0.039609	0	0	1	0
0.101782	-0.044062	0.698643	0.167599	0	0	1	0
0.268007	0.570948	0.095210	0.105192	0	0	1	0
-0.056248	0.195184	0.755907	-0.031888	0	0	1	0
0.071294	0.136082	0.699715	-0.010091	0	0	1	0
-0.062512	0.154734	0.706415	-0.021476	0	0	1	0
-0.273516	0.551614	0.537417	-0.039900	0	0	1	0
0.066674	0.203598	0.608678	-0.007542	0	0	1	0
-0.025050	0.026363	0.669077	0.065868	0	0	1	0
0.139057	0.163473	0.654242	-0.067994	0	0	1	0
0.060308	-0.041862	-0.041152	0.979307	0	0	0	1
0.036965	-0.005646	0.033778	0.973962	0	0	0	1
0.028587	0.061890	0.130105	0.893296	0	0	0	1
0.009285	0.029536	0.020232	0.943340	0	0	0	1
0.071822	0.296950	-0.038740	0.734694	0	0	0	1
0.040995	0.210064	0.045556	0.882345	0	0	0	1
0.078139	-0.037689	0.541370	0.836298	0	0	0	1
0.014744	0.615786	-0.013818	0.220852	0	0	0	1
0.624700	-0.045242	0.087853	0.827151	0	0	0	1
-0.264076	0.216535	0.038109	0.931922	0	0	0	1
0.175348	0.286130	-0.138954	0.919914	0	0	0	1
-0.090024	-0.191248	0.089582	0.949336	0	0	0	1
0.247750	0.123850	-0.039992	0.898253	0	0	0	1
0.068566	-0.051989	0.003844	0.960323	0	0	0	1

-0.013068	0.059116	0.060349	0.907553	0	0	0	1
0.027238	-0.424307	-0.027152	0.993154	0	0	0	1
-0.072270	-0.100398	0.120545	0.925215	0	0	0	1
0.106219	0.274041	0.019000	0.706526	0	0	0	1
0.119748	0.089082	-0.053994	0.943881	0	0	0	1
-0.144287	-0.077512	-0.077551	0.972895	0	0	0	1
0.051269	-0.140295	0.429813	0.992775	0	0	0	1
0.076022	0.108005	-0.068622	0.909621	0	0	0	1