Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias de la Electrónica

Simulación de ubicación de un minisatélite basado en el reconocimiento de patrones.

T E S I S

PARA OBTENER EL GRADO DE:

Licenciatura en Ingeniería Mecatrónica

PRESENTA:

Kevin Arturo Ramirez Zavalza

ASESORES:
Dr. José Eladio Flores Mena
Dr. Gustavo Mendoza Torres

Puebla, Pue, 2016

Agradecimientos

Primero y muy importante, me gustaría agradecer a mis padres, por haberme apoyado hasta ahora para obtener una carrera universitaria y también por siempre animarme a cumplir todo lo que me proponga.

Al Dr. José Eladio Flores Mena por otorgarme su apoyo en la universidad desde mi ingreso, además de motivarme a realizar investigación y por su apoyo para realizar este trabajo de tesis.

Al Dr. Gustavo Mendoza Torres, ya que gracias a uno de sus cursos impartidos además de mis prácticas profesionales realizadas con el empezó mi interés hacia el área de inteligencia artificial, y también por el apoyo brindado para este trabajo de tesis.

A la BUAP al otorgarme una beca de tesis a través del Plan Anual 2015 del Cuerpo Académico Sistemas No Lineales: Modelado, Simulación e Implementación.

Y por último, también agradezco a mis amigos y profesores ya que gracias a su apoyo y amistad aportaron a mis ganas de continuar estudiando y superándome.

Índice general

Resumen

En este trabajo de tesis se propone un método para identificar la ubicación de un minisatélite, cuya órbita es una trayectoria circular. Tomando como referencia una fotografía en un determinado instante y comparando esta con una base de fotografías de la bóveda celeste es posible identificar la ubicación del minisatélite. En base a conocimientos de inteligencia artificial, la teoría de redes neuronales, programación y procesamiento de imágenes, en este trabajo de tesis se hace una propuesta para resolver el problema planteado.

El método propuesto se aplica a una fotografía de la constelación de Andrómeda "tomada" por el minisatélite, al ser una fotografía de gran tamaño solo se toma una cuarta parte de esta y posteriormente de esta cuarta parte la imagen es dividida en nueve diferentes imágenes del mismo tamaño las cuales posteriormente serán transformadas a datos únicos para cada imagen, ademas las imágenes a probar en el sistema poseen el mismo tamaño que cualquiera de las nueve imágenes obtenidas anteriormente.

Las fotografías a utilizar y las fotografías de las cuales se determina su ubicación requieren de un pre-procesamiento de imagen, con el fin de preparar las fotografías para poder extraer información de estas, en este caso se eliminan colores y solo se mantienen los objetos más brillantes (que pueden ser estrellas, cúmulos), para hacer esto se aplican diversos filtros a las fotografías que se encuentran en el toolbox de procesamiento de imágenes de MATLAB.

Después de tener listas las fotografías pre-procesadas, se procede a caracterizar cada una de estas de las cuales se obtiene información única, para generar esta información se partió de cada una de estas imágenes, en estas se detectaron los puntos más brillantes, estos al ser mayormente estrellas tienen forma circular por lo que se les asigno un radio en función a su tamaño el cual depende del brillo, mientras más brillantes poseen un mayor radio y por ende un mayor tamaño, posteriormente se escoge el elemento de mayor tamaño y los tres siguientes en tamaño, entonces se calcula la distancia entre el primer elemento y cada uno de los tres siguientes, lo que al final nos genera una serie de datos con tres distancias únicas para cada imagen. Posteriormente se crean dos archivos en los cuales se almacenan todos los datos generados y su salida correspondiente para cada uno de esos datos, estos archivos se utilizan para entrenar la red neuronal la cual se realiza mediante el toolbox de redes neuronales de MATLAB, y la cual se encarga de otorgar una clasificación (la cual es una ubicación definida) única de cada uno de

los datos de las fotografías en la base de datos para poder generar una salida única y tener la capacidad de generar nuevas salidas ante nuevas fotografías que se encuentren en el rango de la cuarta parte tomada de la constelación. Con el fin de identificar mejor la ubicación se propone realizar un seguimiento en la fotografía y estableciendo un sistema de coordenadas relativo, para hacer esto una vez obtenida la ubicación con la red neuronal, se procede a establecer un sistema de referencia sobre la imagen, después se localiza el objeto más brillante y se calculan sus coordenadas con respecto a un punto de referencia sobre la imagen, con esto se puede observar en qué dirección se encuentra la imagen con respecto a la ubicación detectada de la base de datos mediante el desplazamiento de este objeto.

Para la red neuronal entrenada y de acuerdo a los datos arrojados por el toolbox de MATLAB la red posee un 88.9 % de efectividad para clasificar correctamente fotografías que se encuentren en el rango de la cuarta parte tomada de la constelación. Realizando pruebas reales con diferentes fotografías "tomadas" por el minisatélite se tiene un 80 % de éxito al identificar de manera correcta la ubicación de fotografías desplazadas en diferentes direcciones además de solo requerir en promedio 14 segundos para hacer todo el proceso.

Introducción

Desde la antigüedad para las personas ha sido necesario determinar la ubicación aproximada en la que se encontraban, esto con el fin de poder orientarse y llegar a un destino determinado. Y desde ese entonces hasta la actualidad los métodos para lograr ese objetivo han ido cambiado drásticamente, en la antigüedad si se trataba de navegación existían distintas maneras de poder conocer si se encontraban orientados de manera correcta hacia su destino, uno de los principales métodos y que era además aplicado a el transporte terrestre era mediante la observación del cielo y mediante la dirección del viento [1-2].

En la antigüedad, para poder navegar los marinos consideraban cuatro puntos cardinales adicionales a los mencionados: noreste, noroeste, sureste y suroeste, esto los obtenían subdividiendo los cuatro puntos cardinales principales. Si era de día, se aplicaba el uso de los puntos cardinales, pero si era de noche se hacía uso del cielo nocturno, específicamente de las estrellas, se tomaba como referencia la estrella del norte o estrella polar, ya que a la hora del ocaso esta parece estar situada sobre el polo norte, y mediante la cruz del sur con la cual se lograba localizar el sur y así poder tener puntos de referencia para realizar el viaje. Algunas civilizaciones antiguas usaban como referencia el movimiento del sol, lo que además les permitió definir el primer calendario exacto [3-4].

Otro método utilizado para conocer la trayectoria correcta o para determinar su ubicación era mediante el establecimiento de un punto de referencia, esto se hacía mediante la selección de una estrella que cuya salida o puesta estuviera en dirección al destino, y durante la noche mediante la observación se examinaba la alineación de esa estrella de referencia con otros astros para confirmar el rumbo, para esto memorizaban características de las estrellas como el brillo, el color o el tiempo en que estas se presentaban, estos procesos de abstracción de información son conocidos como reconocimientos de patrones, la capacidad de poder predecir, detectar o identificar algo para llevar a cabo un proceso [5-6].

A partir del siglo XIX la localización de un punto en la superficie terrestre se realizaba mediante el uso de coordenadas geográficas, esto se logra mediante un sistema imaginario constituido de meridianos y paralelos que al ser proyectados sobre un plano o un mapa geográfico se puede identificar la latitud y longitud de cualquier punto situado sobre el planeta Tierra.

Los meridianos son unos círculos imaginarios dibujados sobre el globo terráqueo y que pasan por los polos, o los semicírculos que van de polo a polo y esta medición es conocida como longitud, los paralelos son círculos imaginarios paralelos que cortan a los meridianos formando ángulos rectos y esta medida es conocida como latitud y finalmente para la localización de un punto en el planeta se hace uso de la longitud y la latitud. La longitud es la distancia existente entre el meridiano que pasa sobre la superficie y el meridiano 0 o de Greenwich y la latitud es la distancia entre el punto a localizar y el ecuador, esta puede ser norte o sur en función de la posición con respecto al meridiano cero [7].

En la actualidad se disponen de sistemas como el GPS que es un sistema que permite determinar la localización de un objeto con precisión de hasta centímetros en cualquier parte del mundo. Este sistema funciona mediante el uso de 24 satélites posicionados en órbita distribuidos entre 6 planos orbitales. Para poder conocer la localización mediante el uso de este sistema se requieren de conocer mínimo tres puntos (x, y, z) para poder realizar una triangulación aproximada, para poder conocer las distancias se mide el tiempo de respuesta de una señal entre el emisor (localización a identificar) y un receptor (satélite). El sistema descrito anteriormente funciona de manera correcta para determinar la localización de un objeto en la Tierra mientras los satélites estén colocados de manera correcta en una órbita geoestacionaria [8].

En este trabajo de tesis se estudió el problema de identificar la ubicación de un minisatélite, el cual orbita en una trayectoria circular. Conociendo una fotografía en un determinado instante y comparando esta con una base de fotografías de la bóveda celeste es posible determinar la ubicación del minisatélite. En base a conocimientos de inteligencia artificial, la teoría de redes neuronales, programación y procesamiento de imágenes. Para este trabajo de tesis nos propusimos los siguientes objetivos:

Objetivo General

Simular el sistema de ubicación para un minisatélite en el espacio exterior, esto se realiza mediante la comparación de códigos que se obtienen empleando reconocimiento de patrones y redes neuronales.

Objetivos particulares

1. Investigar y recabar información sobre que es el reconocimiento de patrones y sus aplicaciones.
2. Plantear un método para realizar el procesamiento y tratamiento de las imágenes "tomadas".
3. Proponer el uso de un método o algoritmo para la asignación de valor de peso a las imágenes procesadas.
4. Proponer un método de clasificación para el valor asignado y obtener un resultado único para cada imagen clasificada.

5. Establecer una base de datos con los diferentes valores obtenidos del clasificador para así identificar una ubicación.

6. Realizar diversas simulaciones con el método de clasificación propuesto.

Para una mejor lectura del presente documento se ha dividido el trabajo de tesis de la siguiente manera: el primer capítulo contempla el fundamento en el cual se basa este trabajo, el reconocimiento de patrones. El segundo capítulo contempla la información de los elementos teóricos necesarios para elaborar el trabajo de tesis. El tercer capítulo detalla el procedimiento seguido para elaborar la parte práctica del trabajo. Y en el cuarto capítulo se muestran los resultados obtenidos de este trabajo. Y finalmente, se presenta una sección con las conclusiones y comentarios del presente trabajo de tesis.

Capítulo 1

Fundamentos del reconocimiento de patrones

En este capítulo se presenta una breve revisión de los elementos teóricos que son necesarios para el desarrollo de este trabajo. En la sección 1.1 de este capítulo se revisa que es el reconocimiento de patrones. En la sección 1.2 se conocerán los conceptos básicos para el reconocimiento de patrones. En la sección 1.3 se repasan los distintos métodos de reconocimiento de patrones existentes. En la sección 1.4 se ven los elementos requeridos para el desarrollo del reconocimiento de patrones y por último se enuncian algunas de las principales aplicaciones de este.

1.1. ¿Qué es el reconocimiento de patrones?

De acuerdo a la definición de la Real Academia Española, reconocimiento o reconocer es "examinar con cuidado algo o alguien para enterarse de su identidad, naturaleza y circunstancias", mientras que patrón nos dice que es "modelo que sirve de muestra para sacar otra cosa igual."

De manera más concreta, un patrón describe un objeto en palabras para así poder definirlo y ser capaz de distinguirlo de entre otros objetos más, también es la relación que puede existir entre diversos datos (analógicos o digitales), entre eventos o conceptos. Los patrones están presentes en diferentes elementos a nuestro alrededor, algunos pueden ser notados fácilmente mediante la observación y estos tienen periodos definidos, son consistentes, para poder identificar otros tipos de patrones es necesario hacer uso de otros elementos, que nos permitan analizar señales o grandes cantidades de datos con el fin de encontrar patrones en diferentes intervalos.

Mientras que la palabra reconocer, haciendo una relación con lo que los humanos percibimos a través de nuestros sentidos, reconocer implica otorgar un nombre o etiqueta a todo aquello que escuchamos, sentimos y observamos para posteriormente ser capaces de asociar y clasificar todo lo que identificamos y posteriormente ser capaces

de discriminar entre diferentes objetos.

Por lo tanto, una definición para el reconocimiento de patrones es el campo de estudio el cual tiene como objetivo asignar un objeto identificado a una o varias categorías con el fin de ser capaz de discernir entre ellos, esto mediante el uso de características únicas en los objetos, para esto es necesario saber qué características se requieren conocer y posteriormente buscar un método para poder extraer esas características [9].

En la Figura 1 se puede observar un ejemplo de la clasificación realizada mediante reconocimiento de patrones, en esta clasificación se busca discernir entre perros y gatos mediante las siguientes características: peso y tamaño de hocico.

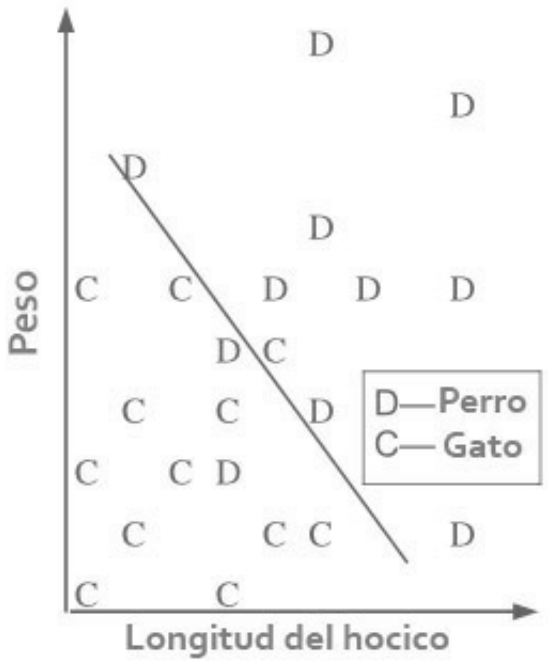


Figura 1: Ejemplo de Clasificación.

Es importante mencionar que el objeto a reconocer también puede verse afectado por su entorno lo que puede llegar a presentarse como ruido, lo que puede ocasionar un aumento de dificultad para discernir los patrones presentes, o en errar al momento de clasificar.

El reconocimiento de patrones también puede ser aplicado para la toma de decisiones mediante el aprendizaje de nuevos patrones además de la generalización y predicción de nuevos, esto mediante una base de entrenamiento en el que se presentan patrones similares.

1.2. Conceptos básicos del reconocimiento de patrones

Para la realización de este trabajo de tesis, es necesario conocer algunos términos básicos que serán utilizados de manera constante y que son fundamentales para el entendimiento del tema y se presentan a continuación:

Reconocer o identificar: existen 2 tipos de reconocimiento, el reconocimiento de objetos concretos que involucra un reconocimiento sensorial y el reconocimiento abstracto. Reconocer es la capacidad de identificar el objeto mediante los sentidos como lo son la vista, el tacto o el olfato. Para un sistema eléctrico y electrónico esto se realiza mediante el uso de sensores.

Patrón o clase: es una o varias propiedades particulares que permiten describir un objeto, es el conjunto de atributos que retenemos para definir un objeto. La clase está determinada por uno o varios atributos comunes.

Sensor: es el dispositivo que se encarga de buscar parámetros específicos en un objeto con el fin de identificarlo.

Proceso de reconocimiento: es la capacidad que tenemos para discernir los datos de entrada y decidir que un objeto pertenece a una y sólo una de las clases o patrones establecidos.

Vector de Características: es el conjunto de características que son utilizadas para distinguir los objetos de las clases, sólo se retienen las propiedades que diferencian los objetos y se dejan de un lado los detalles irrelevantes.

Sistema de reconocimiento de patrones: es un conjunto de reglas que permiten determinar a qué clase pertenece un objeto de una población en estudio.

1.3. Funcionamiento del reconocimiento de patrones

Un sistema de reconocimiento de patrones es un procedimiento llevado a cabo para resolver un problema, normalmente de clasificación, por lo que este sistema consta de ciertos elementos básicos para su funcionamiento. Para poner en funcionamiento un sistema de reconocimiento de patrones, los siguientes son los elementos básicos para su elaboración:

1. Adquisición de Datos
2. Pre-procesamiento
3. Extracción de Características

4. Entrenamiento y aprendizaje del sistema
5. Clasificación
6. Evaluación del funcionamiento

Adquisición de Datos
Este proceso es la entrada del sistema, dependiendo de la aplicación que se desarrolle, esta entrada puede ser de tipo imágen, voz, huellas dactilares, texto, imágenes de retina o iris, etc. Dependiendo del tipo de entrada que se tenga en el sistema, es como se irán constituyendo las demás partes de este.

Pre-procesamiento
En este proceso se transformara nuestra entrada en información que pueda ser manejada por el sistema, además de que en esta parte se podrá llevar a cabo procedimientos como la eliminación de ruido, tratamiento y mejoramiento de imagen, este procedimiento es de vital importancia, ya que mientras nuestra entrada esté lo mejor adaptada posible, será más fácil de manejar en los siguientes procesos, además de que nos permitirá mejorar nuestros resultados.

Extracción de características
Una vez que se preparó la información de entrada de acuerdo a los requerimientos para el manejo de la información del sistema, se procede a realizar la extracción de características, estas características permitirán llevar a cabo la clasificación, dependiendo de la aplicación se puede optar por el uso de diversos sensores como detectores de temperatura, de color, de luminosidad, puede ser utilizado todo aquel sensor que nos ayude a discernir entre los objetos a clasificar, también se pueden aplicar diversos software de procesamiento de imagen para transformar una imagen y posteriormente buscar elementos particulares en cada una de estas, lo que igualmente permitirá discernir entre los distintos objetos.

Entrenamiento y aprendizaje del sistema
Ya que se establecieron las características mediante las cuales funcionará el sistema, se procede a la construcción de la base de entrenamiento del sistema, esta permitirá dependiendo del algoritmo aplicado para la clasificación, ajustar el sistema para generar una salida deseada o en otras palabras aprender en base a los casos que se presenten con el fin de generar una salida aproximada a la deseada.

Para poder establecer el set de entrenamiento, primero es necesario definir la cantidad de clases en las cuales se realizará la clasificación, después se establece la cantidad de elementos de entrenamiento llevará cada una de las clases, mientras mayor sea la cantidad de elementos en cada una de las clases de entrenamiento se mejorará la clasificación, pero a cambio la velocidad de entrenamiento del sistema disminuirá [10].

Es importante mencionar que para los elementos de cada una de las clases, un requisito fundamental es el que cada uno de los elementos sean similares, pero no exac-

tamente iguales, ya que esas pequeñas diferencias son las que permitirán incrementar el rango de detección, un ejemplo sencillo es la detección de letras, para realizar este sistema, es necesario que en la base de entrenamiento se tengan variaciones de las letras a detectar como se puede ver en la Figura 2, ya sean variaciones como inclinación, tamaño o estilo, esto, como se mencionó anteriormente, permitirá incrementar el porcentaje de éxito en la detección y posterior clasificación.

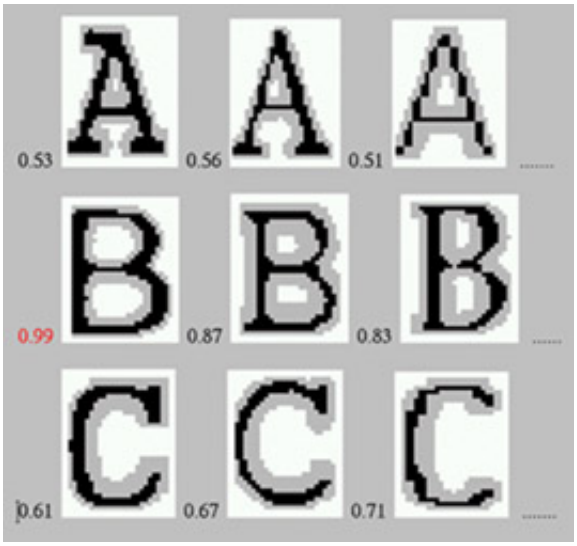


Figura 2: Ejemplo de set de entrenamiento con variaciones.

En el siguiente paso del proceso de un sistema de reconocimiento de patrones se poseen diversas opciones a seguir, ya que aunque este tipo de sistemas son utilizados mayormente para clasificación de las entradas realizando esto en clases separables mediante características, el sistema también tiene la capacidad de realizar asociaciones entre diversos objetos que tienen alguna relación pero no necesariamente por ser elementos del mismo tipo o clase, se le conoce como aprendizaje de asociación. También se puede realizar algo llamado regresión, en el sistema la regresión nos permite predecir una salida en función de la o las entradas que se tengan.

Clasificación
Esta es la salida final del sistema, para poder realizar el sistema de clasificación, es necesario tener un archivo con las entradas que se utilizaran, pero además se debe tener un archivo con las salidas esperadas del sistema. En función a los valores establecidos en los archivos de entrada y salida se podrá ajustar en función a la salida obtenida y la esperada en la clasificación. Para la realización de clasificación se le conoce como método de aprendizaje supervisado [11].

Evaluación de Funcionamiento
Una vez obtenidos los resultados del sistema de clasificación, es necesario llevar un

registro de estos con el fin de saber si se obtuvo la salida esperada con respecto de la entrada definida. Esto permitirá ajustar el sistema en caso de que sea necesario con el fin de mejorar la salida hasta ser o aproximarse a lo esperado.

1.4. Aprendizaje Automático

Con el paso del tiempo y más ahora que nos encontramos en la era digital, diariamente se genera una gran cantidad de información y por eso mismo una gran cantidad de datos, dependiendo del área que se requiera, a esta información se le puede dar una gran cantidad de usos. Por lo tanto tenemos la necesidad de analizar grandes cantidades de datos con el fin de conseguir algún propósito en específico.

En algunos casos esta es una tarea sencilla de realizar ya que es posible crear y hacer uso de algoritmos los cuales pueden ser utilizados para buscar información específica entre una gran cantidad de información recolectada, esto significa en otros términos, establecer una serie de instrucciones las cuales nos permitirán convertir información de entrada (lo que deseamos analizar) en una o varias salidas (saber si se obtuvo un resultado satisfactorio o no).

Pero para realizar algunas tareas, no se posee algún algoritmo que nos permita realizarlas correctamente ya que se puede dar el caso en el que no existan marcadores claros en la información y los cuales puedan ser usados para completar la tarea, también puede ser que la información varíe con respecto a cada individuo por lo que no siempre se pueden seguir las mismas reglas para completar la tarea. Un ejemplo que está presente en la actualidad es la gestión del correo electrónico, uno de los problemas que se presentan es saber detectar y separar los correos basura de los que no lo son, esta no es una tarea sencilla ya que la información contenida en cada uno de los correos electrónicos muchas veces no poseen indicadores específicos para poder detectar rápidamente este tipo de correos.

En base al ejemplo mencionado anteriormente se puede notar que en algunos casos nuestra falta de conocimiento de lo que se tiene que buscar mediante un algoritmo ya establecido entonces es necesario aproximarse al problema de diferente manera.

Al no poseer un algoritmo para esto es necesario buscar una manera para generar uno, para poder realizar esto es necesario recabar una gran cantidad de elementos que permitan ser usados de guía para que el sistema sea capaz de aprender de esos elementos y ser capaz de detectarlos. En otras palabras esto significa que se desea que el sistema sea capaz de generar un algoritmo para una tarea en específico haciendo uso como base diversos ejemplos buscando abstraer toda la información posible de estos ejemplos, esto mediante la búsqueda de similitudes, patrones y regularidades presentes en la información.

En la actualidad para poder realizar esto, existe una rama del área de inteligencia artificial llamada Aprendizaje Automático que tiene como fin permitir a un sistema tener la capacidad de aprender, esta rama se dedica al estudio y a la creación de algoritmos que puedan aprender de casos anteriores y que además sea capaz de realizar

predicciones a futuro. Si la base de datos para el sistema contiene una gran cantidad de información, las predicciones realizadas tendrán mayor probabilidad de ser correctas.

Esta área además de ser aplicada para el análisis de grandes volúmenes de información, como fue mencionado, también tiene su aplicación en el área de inteligencia artificial. Para que un sistema sea inteligente tiene que ser capaz de aprender de su entorno, pero además debe de ser capaz de ajustarse a nuevos cambios, esto significa que no habrá necesidad de intervenir en este para hacer modificaciones ya que el propio sistema se encargará de hacer las modificaciones [12].

1.5. Métodos de Aprendizaje

La forma en que aprenden los humanos siempre ha sido objeto de estudio, esto con el fin de aprender cómo aprendemos, pero además tener la capacidad de generar métodos que puedan ser aplicados a otros sistemas, como son los computacionales y así dotarlos de capacidad de aprendizaje y de saber responder ante nueva información que ingrese al sistema y no solo dependiendo de casos pre establecidos en la programación.

El aprendizaje para este tipo de sistemas está clasificado en tres:

1. Aprendizaje Supervisado
2. Aprendizaje no Supervisado
3. Aprendizaje por Refuerzo

1.5.1. Aprendizaje Supervisado

El método de aprendizaje supervisado es un método el cual hace uso de sets de ejemplo o de entrenamiento para generar una función con el fin de obtener una salida ya establecida. Para poder realizar este procedimiento es necesario establecer cómo será la entrada y posteriormente aplicar un algoritmo con el fin de generar un vector de características, este vector de características estará relacionado con una clase específica a detectar y clasificar, pero además se ha de asignar una salida para cada uno de los diferentes vectores de características para las clases que se deseen identificar y clasificar.

Esto se realiza mediante el análisis de datos de entrada y usando como referencia la salida establecida, esto con el fin de inferir una función que nos proporcione la salida deseada, pero que además esta función puede ser usada para generalizar nuevas salidas ante la presencia de nuevos casos, esto quiere decir que el sistema después de haber sido entrenado, y ante la presencia de entradas similares a el set de entrenamiento, este puede generar una respuesta similar [12].

El aprendizaje supervisado puede ser generalizado de diferentes maneras mediante la presencia de diferentes casos, los cuales son los siguientes:

- Aprendizaje Activo
- Aprendizaje Semi Supervisado
- Predicción Estructurada
- Máquina de aprendizaje de clasificación

Aprendizaje Activo
Este es un subtipo de aprendizaje derivado del aprendizaje automático y aplicado en el área de inteligencia artificial, la hipótesis que nos propone este tipo de aprendizaje es permitir al sistema de aprendizaje escoger la información de la cual aprende, esto con el fin de utilizar solo información que se considere más relevante para el aprendizaje, esto con el fin de aumentar el rendimiento del aprendizaje además de reducir el tiempo requerido para este [13].

Aprendizaje Semi Supervisado
Es un tipo de técnica proveniente del aprendizaje supervisado, esta técnica de aprendizaje se encuentra entre el aprendizaje supervisado y no supervisado, ya que para realizarlo, este solo requiere de un pequeño set de entrenamiento y además de una gran cantidad de entradas sin procesar y sin etiquetar por lo que se desconocen las características del objeto de entrada y no se conoce la salida [14].

Predicción Estructurada
La predicción estructurada es una técnica derivada del aprendizaje supervisado, esta tiene como objetivo predecir objetos estructurados, de segmentarlos y de clasificarlos. Para el entrenamiento de este tipo de sistemas, se hace mediante la observación de los datos, con el fin de identificar cuando es correcta la predicción con el fin de ajustar el sistema. Esta técnica es usada principalmente para aplicaciones que requieran el uso de lenguaje humano como el reconocimiento de voz o traducción de texto de un idioma a otro.

Máquina de aprendizaje de clasificación
Esta técnica se enfoca en la generación y uso de modelos de clasificación de información para sistemas de búsqueda de información, esto con el fin de generar resultados más aproximados. Para realizar el entrenamiento mediante el uso de esta técnica, se parte de un set de entrenamiento que contiene una lista de objetos, y entre cada uno de estos objetos están parcialmente ordenados, esto significa que existe un orden, secuencia o disposición en el set, esto con el fin de dar mayor o menor importancia a cada uno de los elementos contenidos.

En el aprendizaje supervisado, existen diferentes tipos de clasificadores, entre los cuales los más importantes son:

- Clasificadores basados en distancias

- Clasificadores Bayesianos
- Redes Neuronales
- Máquinas de Vectores de Soporte (SVM)
- Algoritmo de agrupamiento (clustering)

Clasificadores basados en distancias
Existen diferentes tipos de clasificadores basados en distancias, su funcionamiento está basado, como su nombre lo menciona, en calcular las distancias que existen en cada uno de los elementos del conjunto de entrada con respecto a un patrón de prueba y dependiendo del tipo de clasificador que se aplique los elementos pueden ser ordenados de acuerdo de mayor a menor distancia del patrón prueba, esto significa que mientras la distancia sea menor, el elemento pertenece más a la clase.

Clasificadores Bayesianos
También conocido como clasificador probabilístico de Naive Bayes, este es un tipo de clasificación y predicción supervisada ya que requiere de ejemplos previos para realizar su entrenamiento y así ser capaz de realizar la clasificación. Este tipo de clasificador tiene sus fundamentos en el Teorema de Bayes, este teorema expresa la probabilidad condicional de un evento A dado B con la probabilidad de B dado A.

Redes Neuronales
Las redes neuronales artificiales buscan imitar el funcionamiento de las redes neuronales de organismos vivos, estas redes artificiales están conformadas por diversas capas, se tiene una capa de entrada, una capa oculta y una capa de salida. Estas redes hacen uso del aprendizaje supervisado por lo que para el entrenamiento de la red se requiere de diversos conjuntos de ejemplo para la entrada y se requiere conocer la salida, esto es con el fin de ajustar las conexiones entre las neuronas ya que estas conexiones son las que almacenan el "conocimiento" adquirido.

Máquina de Vectores de Soporte
También conocidas como SVM (Support Vector Machine), es un modelo de clasificación el cual requiere que los datos de entrada y las clases establecidas sean linealmente separables, esto significa que debe haber una clara separación entre las clases existentes, en caso de que existan elementos que no sean linealmente separables los SVM tienen la capacidad de mapear la entrada original en un nuevo espacio. Algunas de sus aplicaciones son: reconocimiento de firmas, reconocimiento de imágenes, categorización de textos y como clasificador.

Algoritmo de agrupamiento o Clustering
El algoritmo de agrupamiento permite agrupar un conjunto de objetos de tal manera que estos objetos sean del mismo grupo o tipo, esto significa que los objetos pertene-

ciente a un grupo deben de ser similares entre sí, este es aplicado en áreas como el aprendizaje automático, la minería de datos y en el reconocimiento de patrones.

1.5.2. Aprendizaje no Supervisado

El aprendizaje no supervisado es un método de aprendizaje en el cual las entradas son conocidas y las salidas son desconocidas. El objetivo de este método es el encontrar regularidades o patrones en la entrada, esto con el fin de identificar una periodicidad o uniformidad y esto suceda de manera constante. El aprendizaje no supervisado está estrechamente relacionado con el problema de estimación de densidad en el área de estadística.

Para realizar el aprendizaje supervisado es necesario aplicar una estimación de densidad, esta es una técnica en el Análisis Exploratorio de Datos (DEA por sus siglas en inglés). Esta técnica busca generar un valor de salida mediante el análisis de las entradas, esto buscando patrones o regularidades presentes en la entrada y se pueda obtener una salida consistente [12].

Para este método de aprendizaje se puede realizar desde diferentes aproximaciones, entre las principales se encuentran:

- Clustering o Algoritmo de Agrupamiento
- Algoritmo E-M
- Separación Ciega de Fuentes

Algoritmo de Agrupamiento o Clustering
Este algoritmo tiene como objetivo agrupar un conjunto de objetos, de manera que los objetos parecidos pertenezcan a un grupo propio. Esto se realiza mediante usualmente por la distancia o similitud que existe entre cada uno de los objetos del conjunto. Este algoritmo es aplicado en el aprendizaje automático, reconocimiento de patrones, análisis de imágenes y bioinformática.

Algoritmo E-M
Este algoritmo es usado también en estadística, este es realizado de manera iterativa y permite encontrar una estimación de máxima verosimilitud y la máxima a posteriori que son conceptos de probabilidad y estadística, estos permiten encontrar un parámetro determinado de una distribución. Es aplicado comúnmente para algoritmos de agrupamiento.

Separación Ciega de Fuentes
Es un proceso en el que se realiza la separación de un conjunto de señales de entrada y en el cual no se posee información sobre los tipos de señales que contiene o de cómo fueron mezcladas.

1.5.3. Aprendizaje Reforzado

Este es un método aprendizaje perteneciente al área de aprendizaje autónomo, este método está inspirado en el área de psicología, específicamente en el conductismo, este método busca cómo se realiza la toma de decisiones en un entorno con el fin de resolver un problema de manera satisfactoria.

Algunas de las áreas en las que es aplicado este método son la teoría de juegos, teoría de control, optimización basada en simulación y algoritmos genéticos.

Para algunas aplicaciones, se requiere de una secuencia de pasos con el fin de llegar a un resultado satisfactorio, por lo que para este método no es importante que cada uno de estos pasos sea correcto, lo que busca este método de aprendizaje es una norma la cual permita generar una serie de pasos correctos para alcanzar un resultado positivo, pero además con este método, el sistema es capaz de generar diversas normas para llegar a un mismo resultado y también es capaz de distinguir cuál de estas normas generadas es más efectiva. Este método es iterativo, ya que requiere de varios intentos con el fin de llegar a resolver un problema o cumplir su objetivo [12].

1.6. Diseño y características del reconocimiento de patrones

1. Diseño algoritmo localizador del objeto: Seleccionar un algoritmo de segmentación que aislará los objetos individuales de la imagen.
2. Selección de características: Decidir qué propiedades de los objetos discriminan mejor y cómo medirlas.
3. Diseño clasificador: Establecer las bases matemáticas del algoritmo de clasificación y escoger la estructura del clasificador que se usará.
4. Aprendizaje del clasificador: Fijar parámetros ajustables, por ejemplo, fronteras de decisión en la clasificación.
5. Evaluación del funcionamiento: Estimar las tasas de los posibles errores al clasificar.

Con el fin de desarrollar un sistema de reconocimiento de patrones basado en imágenes se requieren de un mínimo de tres pasos para su desarrollo, estos tres pasos son:

Segmentación de imagen: El objetivo de la segmentación es realizar el procesamiento de una imagen con el fin de mejorarla y/o simplificarla para que esta sea de más fácil uso y manejo para el sistema.

Extracción de características: con el fin de ser capaz de discriminar entre las imágenes se requiere de la extracción de las características de estas, dependiendo de lo que se desee detectar es como se establecerán las características a buscar.

Clasificación: Una vez que las características han sido extraídas se procede a realizar el sistema de clasificación de imágenes, el cual separara las diferentes entradas en diferentes clases.

A continuación se muestra un esquema general de un sistema de reconocimiento de patrones para imágenes:

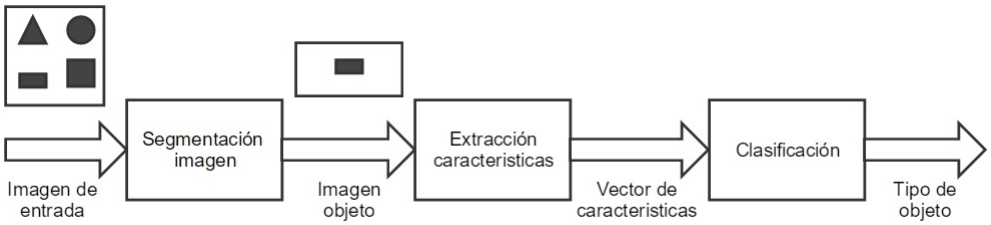


Figura 3: Esquema de un sistema de reconocimiento de patrones para imágenes [15].

1.7. Aplicaciones

En este apartado se muestran algunas de las aplicaciones del reconocimiento de patrones, como se puede notar el reconocimiento de patrones puede ser aplicado en diferentes áreas.

En el área de medicina, algunas de las aplicaciones del reconocimiento de patrones son: el diagnóstico de enfermedades y como sistema de apoyo para toma de decisiones sobre un diagnóstico médico, análisis de cromosoma para la detección de irregularidades, el análisis de electrocardiogramas y encefalogramas con el fin de detectar anormalidades.

También se ha desarrollado para realizar el reconocimiento de voz, para la conversión de voz a texto y de texto a voz, para sistemas OCR (Optical Character Recognition) con el fin de poder convertir la escritura de un documento a texto que pueda ser editado desde la computadora.

En el área de biometría, el reconocimiento de patrones ha sido utilizado en el reconocimiento de huellas dactilares, para el reconocimiento de iris, para la detección de rostros que también es de uso comercial como se puede ver en las cámaras que poseen la capacidad de detectar la sonrisa de las personas.

En el área de visión artificial el reconocimiento de patrones es utilizado para el reconocimiento y detección de objetos, para la búsqueda de imágenes similares y para la generación de modelos 3D partiendo de imágenes.

Capítulo 2

Elementos disponibles para la aplicación al sistema de ubicación

En este capítulo se presenta toda la teoría necesaria para el desarrollo de la parte práctica del trabajo de Tesis. En la sección 2.1 se presenta la información relacionada con las imágenes digitales, algunos métodos y técnicas que serán de utilidad para realizar el pre-procesamiento de estas. En la sección 2.2 se presenta la teoría necesaria del apartado de redes neuronales las cuales serán utilizadas para asignar una primera ubicación a los datos de las imágenes utilizadas como base de entrenamiento y que además asignan un valor a las nuevas imágenes ingresadas, que posteriormente serán utilizadas para hacer una primera identificación de la ubicación. En la sección 2.3 se definirán los elementos que serán utilizados para el desarrollo de la interfaz gráfica que será realizada en MATLAB y la cual mostrará la información relevante del sistema.

2.1. Procesamiento de Imágenes

Los humanos perciben el mundo a partir de los sentidos, gracias a estos es posible identificar y dar un nombre a cada una de las cosas que nos rodean. Esto puede ser realizado mediante el uso de uno o varios de los cinco sentidos: el olfato, el oído, el gusto, el tacto y la vista.

La vista es el sentido que nos permite ver las cosas. Con este sentido podemos ver imágenes, o dicho de otra forma el percibir la forma, el tamaño o la distancia que se encuentran los objetos que observamos. Las imágenes no están limitadas solo a lo que vemos a través de la vista, ya que estas pueden ser una representación mental de lo que imaginamos, de conceptos que visualizamos internamente y a los cuales buscamos otorgarles una representación que pueda ser mostrada a otras personas.

Una vez captado un objeto mediante la vista se poseen diversos métodos de capturar esa imagen con el fin de representarla, almacenarla y ser capaces de transmitirla a otras personas. Entre los principales métodos que existen para capturar una imagen se

encuentran: el dibujo, la pintura, la fotografía o el vídeo [16].

2.1.1. ¿Qué es una imagen digital?

En la actualidad nos encontramos en un momento en el que la información es almacenada de manera digital, esto con el fin de hacer más fácil su manejo además de hacer esta información accesible desde cualquier punto del planeta y accesible para cualquier persona. Actualmente existe gran variedad de información que puede ser transformada y almacenada de manera digital.

Un tipo de información digital son las imágenes digitales, estas pueden ser dibujos, fotografías, pinturas artísticas o cualquier otra imagen que puede ser transformada y almacenada de manera digital. De acuerdo al origen de las imágenes estas pueden ser clasificadas en imágenes reales o imágenes sintéticas. Las imágenes reales son aquellas que han sido digitalizadas mediante elementos externos como con el uso de escáner o de tabletas de dibujo o tomadas mediante una cámara digital y las imágenes sintéticas son aquellas que han sido generadas mediante un software de edición de imagen.

En las imágenes digitales podemos distinguir entre dos tipos principales de imágenes digitales: las imágenes vectoriales y las imágenes bitmap o mapa de bits.

Las imágenes vectoriales son aquellas que están compuestas por elementos geométricos sencillos, los cuales son controlados mediante cálculos y fórmulas matemáticas y de los cuales algunos puntos son seleccionados como puntos de referencia con el fin de construir la imagen completa. También cada uno de estos elementos poseen diversos parámetros como coordenadas de inicio y fin, grosor, color de relleno, brillo, intensidad, etc. Una de las principales ventajas de una imagen vectorial, es que al ser conformada por elementos geométricos, este tipo de imágenes pueden ser ampliadas o reducidas sin perder calidad, además de su reducido uso de espacio. Una de las principales desventajas de este tipo de imagen es que no son eficaces para representar imágenes de tipo fotográfico.

En cuanto a las imágenes bitmap o de mapa de bits, estas están conformadas por valores también llamados píxeles que están dispuestos de manera ordenada y en una cuadrícula, cada uno de los valores que conforma la imagen poseen un color uniforme, los cuales al integrarse con los demás valores existentes proveen de la imagen completa además de ser capaces de mostrar elementos como iluminación, sombra, etc. Mientras más píxeles contenga la imagen se tendrá una mayor resolución y por ende una mejor calidad. Una de las desventajas que posee este tipo de imagen es que si se desea ampliar la imagen a un tamaño mayor del original, este proceso puede llegar a generar pérdidas de calidad [17].

En la Figura 4 se muestran los dos tipos de imagen existentes: una imagen vectorial que está conformada por formas geométricas y una imagen de mapa de bits que está conformada por píxeles.

Figura 4: Imagen vectorial e Imagen de mapa de bits.

2.1.2. Conceptos básicos de imágenes digitales

Con el fin de entender un poco mejor la información que será presentada posteriormente es necesario familiarizarse con algunos conceptos que son usados en el área de imágenes digitales.

Píxeles: El píxel que es una abreviatura de Picture Element, es la unidad más pequeña existente en una imagen digital. Estas unidades están contenidas en cada uno de los elementos de la cuadrícula que conforman una imagen digital, cada uno de estos píxeles almacena un color, esto lo logra combinando los tres colores básicos: rojo, verde y azul (también conocido como RGB). Entre las principales características de los píxeles se encuentran: poseen forma cuadrada, poseen una posición relativa con respecto al resto de píxeles de una imagen y la profundidad de color.

Resolución de la imagen: La resolución de una imagen es la cantidad de píxeles que componen una imagen digital, estos suelen medirse en píxeles por pulgada (ppi) o píxeles por centímetro (ppc), esto significa que mientras mayor cantidad de píxeles se tengan, se tendrá una imagen de mayor tamaño y además se tendrá mayor resolución lo que puede significar una mejora de calidad pero también un incremento de peso en el tamaño final del archivo.

Profundidad de color: esta es una característica de los píxeles la cual posee la capacidad de almacenar el color, mientras más grande sea este valor se podrán almacenar más variantes de color, esta característica se mide en bits, esto quiere decir, por ejemplo, si solo usáramos 1 bit, esto implicaría que solo podremos usar el 0 o 1 como valores de color. Los valores más comunes son: 8 bits (256 colores) y 24 bits (16,777,216 colores).

Niveles o escalas de gris: esta es una característica de los píxeles la cual posee la capacidad de almacenar su nivel de luminiscencia, este se maneja con una escala entre los colores blanco, gris y negro. Dependiendo de la aplicación que se le dé a la escala de grises, es la cantidad de bits que maneja y el rango de valores que poseerán cada uno de los colores. En la Figura 5 podemos ver la imagen original y a su derecha se puede ver una imagen con un filtro que representa la imagen en escala de grises.

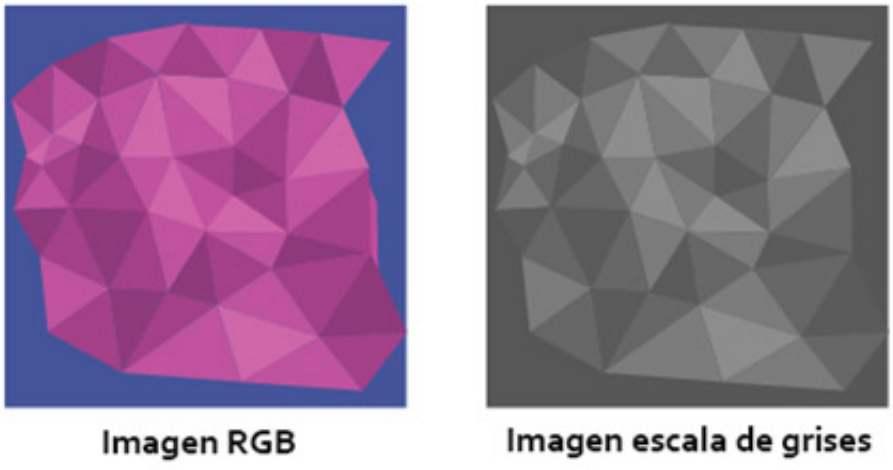


Figura 5: Imagen original e imagen con escala de grises.

Tamaño de archivo: Este dato indica la cantidad de información que posee la imagen, además de que con este se puede saber qué cantidad de espacio será requerido para su almacenamiento. El tamaño del archivo puede variar por diversos factores, entre los principales están:

- Tipo de archivo
- Modo de color
- Tipo de imagen digital
- Resolución de imagen

Tipo de archivo: el tipo de archivo, también conocido como formato de archivo tiene un gran impacto en el tamaño final del archivo. Esto se debe a que dependiendo del tipo de archivo a utilizar será la forma en que si se realizara o no una compresión de los datos. Entre los formatos de imágenes más comunes se tiene: PNG, BMP, GIF y JPG.

El formato JPG es un formato de compresión con pérdidas por lo que este formato elimina toda información que no es visible, este formato divide la imagen en dos partes, en color y en luminosidad. Para este formato la profundidad de color puede variar entre

los 8 bits y los 24. Este formato es muy utilizado para los sitios web debido a su bajo tamaño de archivo y su buena calidad.

El formato GIF es un formato que genera imágenes de tamaño reducido, esto se debe a que su profundidad solo hace uso de 8 bits (256 colores). Algunas de las ventajas que posee este formato es la capacidad de manejar transparencias además de generar secuencias animadas.

El formato PNG es un formato que surgió con el fin de tomar las mejores características del formato JPG y GIF. Este es un formato de compresión sin pérdidas ya que este maneja una profundidad de color de 24 bits además de que tiene la capacidad de manejar transparencias como GIF.

Dependiendo del formato que sea utilizado y el tipo de imagen también es posible variar el tamaño final del archivo si se modifica la resolución, generalmente a imágenes con resoluciones mayores generarán un archivo de un tamaño mayor que si se usara una imagen con una resolución menor.

Modo de color: Para poder percibir los colores, el ojo humano lo realiza mediante la longitud de onda que estos reciben, esta longitud de onda que podemos percibir también es conocida como espectro visible, esto es desde los 400 a los 700nm. La luz blanca contiene todo el espectro de color mientras que la ausencia de luz es percibida como el color negro. Para los software de edición de imágenes es común manejar tres modos de color: HSB, RGB y CMYK. El modo HSB clasifica los colores de acuerdo a tres valores básicos: el tono, la saturación y la luminosidad. El modo RGB obtiene los colores mediante la mezcla de los tres colores primarios: rojo, verde y azul. Y el modo CMYK, este describe el color que se obtendría si se tiñese un papel con tinta de color, este modo se basa en los pigmentos de los siguientes colores: cían, magenta, amarillo y negro. Este modo es utilizado principalmente cuando se realizara alguna impresión.

Tipo de imagen: como se mencionó anteriormente, tenemos las imágenes de mapas de bits y las imágenes vectoriales, las imágenes vectoriales hacen uso de un menor tamaño de imagen ya que estas están conformadas por elementos geométricos y hace uso de fórmulas matemáticas lo que además permite modificar su resolución sin presentar grandes aumentos de tamaño.

Relación de aspecto: esta es la proporción existente entre el ancho y el alto de una imagen, para poder calcular esta relación de aspecto solo es necesario dividir el ancho entre el alto de una imagen y es expresado normalmente de la forma X:Y. Entre las relaciones de aspecto más comunes y estandarizadas en cámaras digitales son las relaciones de 4:3 y la relación 16:9, la primera es más usada en fotografías digitales mientras que la segunda, también conocida como pantalla ancha y la cual es usada en televisores de alta definición [17].

2.1.3. ¿Qué es el pre-procesamiento de imagen?

El pre-procesamiento de imagen es el proceso de transformación que se aplica a una imagen con el fin de mejorarla, cambiarla con el objetivo de resaltar características, de eliminar elementos que no sean necesarios o de prepararla para su uso en un proceso posterior.

Este proceso es fundamental para las fases posteriores como lo es el reconocimiento y la interpretación de los datos. Estas operaciones se realizan a nivel píxel y estas operaciones pueden ser de tipo lógicas para imágenes binarias (and, or, not, xor), de tipo geométricas (modifican las relaciones espaciales entre píxeles).

La técnica en la que nos enfocaremos será la técnica de umbralización o thresholding, la cual nos permitirá crear una imagen binaria a partir de una imagen en escala de grises, esto significa que nos enfocaremos en la luminosidad que poseen los píxeles de la imagen a pre-procesar. En el siguiente apartado el cual trata sobre el toolbox de MATLAB se incluye la información sobre la realización de esta técnica de pre-procesamiento [18].

2.1.4. Toolbox procesamiento de imagen de MATLAB

El toolbox de procesamiento de imagen o image processing toolbox por su nombre en inglés es un conjunto de funciones que son utilizadas para aumentar las capacidades del software MATLAB. Este toolbox puede realizar diversas operaciones de procesamiento de imagen ya que este otorga un conjunto de herramientas para visualizarlas y manipularlas.

Este toolbox contiene diversos algoritmos, funciones y aplicaciones para el análisis, procesamiento y visualización de imágenes, con todas estas herramientas se pueden realizar diversas operaciones como: análisis y mejora de imágenes, reducción de ruido, registro de imágenes y transformaciones geométricas. Una de las ventajas del uso de este toolbox es que tiene la capacidad de manejar varios tipos de archivos de imagen además de que en caso de ser necesario para un procesamiento más intensivo posee la capacidad de hacer el uso de la unidad GPU de una computadora para acelerar el trabajo y reducir el tiempo de ejecución.

Entre los tipos de imágenes que soporta el toolbox se encuentra las imágenes indexadas, imágenes binarias e imágenes RGB, para cumplir nuestro objetivo estaremos manejando imágenes de tipo RGB e imágenes binarias las cuales han sido mencionadas anteriormente. Por defecto MATLAB almacena normalmente las imágenes en matrices, en las cuales cada uno de los elementos de la matriz corresponde a un píxel de la imagen original. Un ejemplo sencillo para comprender esto es: si se usara una imagen en MATLAB con una resolución de 1024 x 768 píxeles darían como resultado una matriz de tamaño 1024 x 768. Esto nos da como resultado que si se posee conocimiento de MATLAB se pueden manejar fácilmente los datos de la imagen.

Si se desea acceder a un píxel en específico en la matriz de una imagen es tan simple como conocer la posición en la que se encuentra, para esto se realiza de la siguiente manera, suponiendo que tenemos una matriz nombrada I de tamaño m x n si deseamos

acceder a alguno de los elementos que están contenidos en esta solo se requiere conocer el número de fila y columna en el que se encuentra y para obtener el valor se hace uso de la instrucción I(f,c) en donde f es el número de fila y c es el número de columna. En MATLAB a diferencia de otros programas y otros lenguajes de programación, el número de filas y columnas inician en el número 1.

En el toolbox existen diversas instrucciones para el manejo de las imágenes, pero para la realización de este trabajo de tesis solo se hace uso de algunas de estas funciones. El toolbox divide las herramientas en diversas categorías, las categorías en las cuales estará más enfocado este trabajo para el manejo de las imágenes es:

Importar, exportar y conversión, de las cuales se utilizaran las funciones del apartado: básicos de importación y exportación y las funciones del apartado conversión de tipo de imagen.

En las funciones del apartado básicos de importación y exportación se encuentran las funciones necesarias para leer y escribir la información de las imágenes además de obtener información sobre las imágenes.

En el apartado de conversión de tipo de imagen se encontrarán las funciones necesarias para convertir entre tipos de imágenes como son de tipo RGB, binarias, escala de grises [19].

Las funciones de este toolbox que son utilizadas en el desarrollo del sistema serán descritas en el capítulo 3 en donde se muestra el desarrollo completo del sistema.

2.2. Redes neuronales

Las redes neuronales artificiales están inspiradas en el funcionamiento y estructura de una red neuronal biológica. Una red neuronal biológica es un conjunto de neuronas que se encuentran conectadas y se comunican entre sí. Para poder comprender lo que es una red neuronal primero tenemos que enfocarnos en las unidades fundamentales que conforman a las redes neuronales biológicas, la neurona.

La neurona es una célula viva que tiene algunos elementos que la diferencian de las otras células vivas, esta tiene una forma más o menos esférica y esta posee un tamaño entre 5 y 10 micras, en contraste con una compuerta lógica de silicio las neuronas son más lentas ya que estas funcionan en orden de mili-segundos mientras que una compuerta está en el orden de nanosegundos. Una neurona está compuesta por una parte llamada axón que está conectada al cuerpo de la célula y que en su extremo contiene ramificaciones más pequeñas con las cuales emiten la salida de la neurona y realizan la conexión con otra u otras neuronas. También del cuerpo de la célula salen otras ramificaciones llamadas dendritas las cuales son utilizadas para recibir la conexión de otra neurona y las cuales llevan la señal al interior de la célula. Por lo general una neurona recibe miles de conexiones e información de otras neuronas y a su vez puede enviar información a miles de neuronas. En la Figura 6 se puede observar la forma general de la neurona en donde se puede ver el cuerpo de la célula, el axón, las dendritas y los puntos de conexión de entrada y de salida de la neurona.

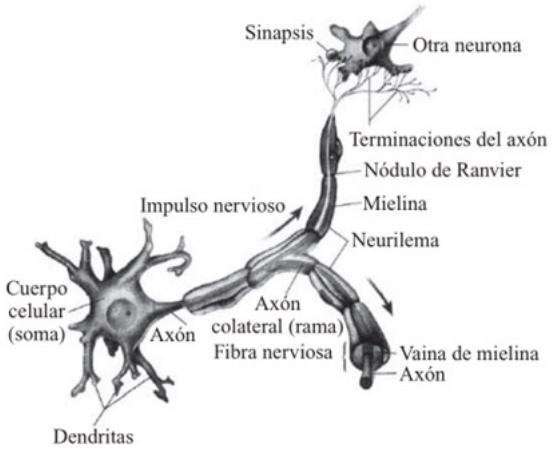


Figura 6: Forma general de la neurona.

Las señales que utiliza una neurona son de dos naturalezas diferentes, de tipo eléctrica y de tipo química. La señal que se transmite desde el axón es de tipo eléctrica, mientras que la señal que se transmite desde las ramificaciones del axón hasta las dendritas de la siguiente neurona son de tipo químicas, a esta conexión se le llama sinapsis. En la Figura 7 se puede ver cómo se lleva a cabo el proceso de la sinapsis, la distancia que existe entre la terminal del axón y de la dendrita de la siguiente neurona es de apenas 50 nm.

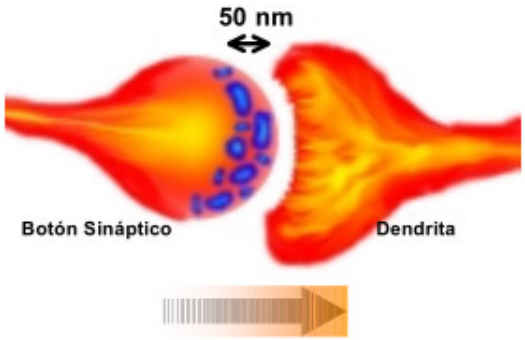


Figura 7: Sinapsis entre dos neuronas.

La sinapsis química es el tipo más común, como se mencionó anteriormente una señal eléctrica viaja a través del axón hasta llegar al botón sináptico en la Figura 7, ahí se rompen las vesículas sinápticas representadas de color azul y así liberándose una sustancia llamada neurotransmisor, esta sustancia es captada por la dendrita de la siguiente neurona y esta estimula la emisión de una nueva señal eléctrica en la nueva neurona y que va en dirección derecha para poder pasar a la siguiente neurona. En la neurona existen dos comportamientos que son importantes:

El primero nos dice que el pulso que llega a la sinapsis y el que sale no es necesariamente igual, el pulso de salida dependerá de la cantidad de neurotransmisor liberada, la cantidad del neurotransmisor varía durante el proceso de aprendizaje de la neurona haciendo que el pulso sea reforzado o debilitado.

El segundo comportamiento nos dice que se suman todas las entradas de las dendritas de una neurona y si la suma de esas entradas sobrepasa un umbral entonces se mandará un pulso a través del axón, en caso de no superar el umbral no se mandara un pulso. Las neuronas poseen un tiempo refractario, esto significa que durante la transmisión entre mensaje y mensaje requieren un tiempo de entre 0.5 y 2 mili-segundos para poder transmitir un nuevo mensaje [20-21].

2.2.1. Redes neuronales artificiales

La motivación por la cual se buscan desarrollar redes neuronales artificiales es la búsqueda de modelar el funcionamiento de una red neuronal biológica, esto específicamente por la forma de funcionar del cerebro humano ya que este es un sistema complejo, no lineal y funciona de manera paralela, esto significa que puede realizar varias operaciones de manera simultánea a diferencia de las operaciones realizadas por una computadora que son realizadas de manera secuencial, lo que significa que para poder realizar operaciones tienes que realizarlas una a la vez, con esto podemos decir que una red neuronal artificial puede funcionar como un procesador, esta red está compuesta por neuronas que serían las unidades encargadas del procesamiento. Algunas de las principales características de una red neuronal artificial son:

- Estas poseen la capacidad de adquirir conocimiento a través de la experiencia mediante el uso de datos de ejemplo, con los cuales la red neuronal ajusta las conexiones entre las neuronas para responder a determinados estímulos.
- Poseen capacidad de plasticidad y adaptabilidad, esto significa que una red neuronal puede ser moldeada para aprender a ejecutar nuevos procesos como el reconocimiento de patrones, clasificaciones y asociaciones de la información de entrada además de poder responder de forma correcta a nueva información por su capacidad de generalización.
- Tienen un alto nivel de tolerancia a fallas, esto significa que la red puede llegar a presentar fallos por causas externas y la red podrá seguir teniendo un buen funcionamiento, esto imitando la red neuronal biológica.

2.2.1.1. Modelo de la red neuronal

A continuación se presenta un modelo sencillo para la construcción de redes neuronales, este modelo busca modelar las características más importantes del comportamiento biológico.

Para el modelo de la red neuronal artificial, como se puede observar en la Figura 8 contamos con n neuronas de entrada denotadas como $x_1 ... x_n$ y estas estarán mandando los valores de entrada a la neurona de salida y_j. Los elementos denotados w_{j0}, w_{j1}, w_{ji} son los pesos sinápticos de las conexiones con la neurona y_j, en la notación de los pesos el primer índice indica a qué neurona se dirige y el segundo índice indica de qué neurona proviene la información.

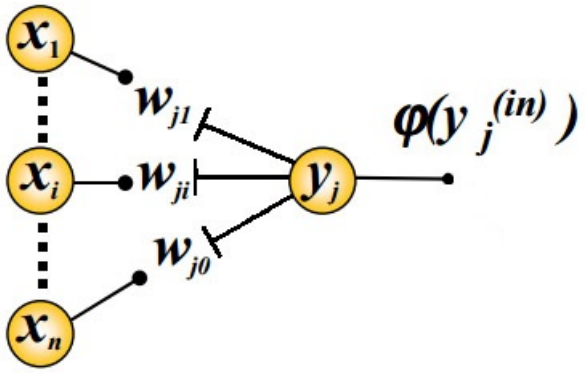


Figura 8: Esquema de la neurona artificial.

La función de cada uno de los pesos sinápticos w_{ji} es realizar una multiplicación con su correspondiente entrada y este peso define la importancia de cada entrada. Si se recuerda de lo mencionado anteriormente, en una neurona biológica se suman todas las entradas, para el caso del modelado de la neurona artificial se realiza la misma operación, por lo que para obtener la entrada total y_j se tiene:

$$y_j^{(in)} = \sum_{i=1}^{n} w_{ji} x_i \tag{1}$$

Se incluye el índice (in) para denotar que es la entrada. Como se mencionó en el funcionamiento de la neurona biológica, para poder generar una salida es necesario que la suma de las entradas supere un umbral. Para poder realizar esto es necesario aplicar una función de activación φ la cual se aplica sobre $y_j^{(in)}$, para realizar esto existen diversas funciones de activación las cuales se verán a detalle en el siguiente apartado. Por lo tanto nuestra señal de salida de la neurona y_j es:

$$y_j = \varphi(y_j^{(in)}) \tag{2}$$

Entre los componentes más importantes que podemos encontrar en una red neuronal son:

- Función de activación
- Tipo de red neuronal artificial

- Regla de aprendizaje
- Algoritmo de entrenamiento

La red neuronal puede estar organizada en función de:

- Número de capas
- Número de neuronas por capa
- Patrones de conexión
- Flujo de información

Para poder realizar la distribución de las neuronas en una red neuronal, esta se realiza mediante capas o niveles y cada una con un determinado número de neuronas. En una red neuronal existen tres tipos de capas: de entrada, ocultas y de salida.

La capa de entrada es donde se recibe la información de la entrada original proveniente de otro medio y que ha sido pre-procesada para su uso en la red. Las capas ocultas son internas y no poseen alguna conexión directa con el exterior de la red, con la posibilidad de que pueden existir varias capas ocultas, estas pueden estar conectadas entre sí. Y la capa de salida la cual transfiere la información interna de la red al exterior.

Para el flujo de información, dependiendo del tipo de red existen las redes con conexiones hacia adelante o feedforward en las cuales se propaga la información hacia adelante, desde la capa de entrada a la capa de salida. También existen las redes con conexiones hacia adelante y hacia atrás, o que significa que la información puede propagarse en ambos sentidos de las conexiones de la red.

Con esto se da la topología de las redes neuronales, las cuales serán de tipo monocapa y de tipo multicapa. Las redes de tipo monocapa solo poseen una capa y esta capa realiza conexiones laterales a las neuronas a las que pertenece esta capa. Las redes de tipo multicapa poseen varias neuronas divididas en diversos niveles o capas [20-22].

2.2.1.2. Funciones de activación

Como se mencionó, para que una neurona artificial pueda generar una salida se requiere que la entrada total supere un umbral. Por lo que para hacer esto es necesario aplicar una función de activación sobre $y_j^{(in)}$ o en algunos casos donde recibe el nombre de x . Entre las funciones de activación más comunes se encuentran:

- Función escalón
- Función lineal y mixta
- Función sigmoidal

Función escalón

Este tipo de función es utilizada para neuronas de tipo binarias, en las cuales cuando la suma de las entradas es mayor o igual que el umbral de la neurona la activación es 1 y si es menor la activación es 0 aunque también puede tomar el valor de -1 como se puede ver a continuación para el caso de valores de 0 y 1:

$$y_j = \begin{cases} 1 & si \quad x \geq 0 \\ 0 & si \quad x < 0 \end{cases} \tag{3}$$

Y a continuación el caso para valores de 1 y -1:

$$y_j = \begin{cases} 1 & si \quad x \geq 0 \\ -1 & si \quad x < 0 \end{cases} \tag{4}$$

En la Figura 9 podemos ver la gráfica de la función escalón tomando valores de 0 y 1, y en la Figura 10 podemos ver la gráfica de la función escalón con valores de 1 y -1.

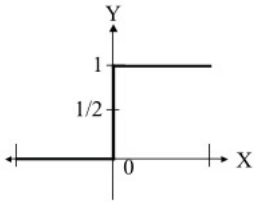


Figura 9: Gráfica de función escalón con valores de 0 y 1.

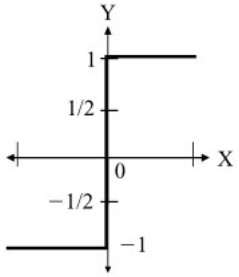


Figura 10: Gráfica de función escalón con valores de -1 y 1.

La función de activación de tipo escalón es usualmente utilizada para redes neuronales de tipo perceptrón monocapa y multicapa.

Función lineal y mixta

La función lineal o también conocida como identidad está representada por la función de activación $y_j = x$ y la cual podemos ver representada en la Figura 11. Este tipo de función producirá usualmente sólo valores positivos dentro del rango de los números

reales y es usada para redes de tipo predictivas. Esta función es empleada generalmente para redes neuronales de tipo Adaline.

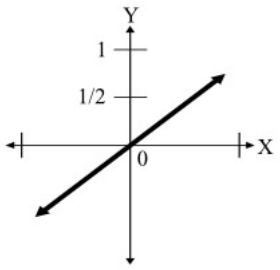


Figura 11: Gráfica de la función lineal.

En la función de activación mixta en la cual si la suma de las entradas es menor que el límite inferior la función se define como 0 o -1, y si la suma es mayor o igual al límite superior entonces se da la activación y da como salida un 1. Si la suma de la entrada está comprendida entre los dos límites, la activación también es 1 y si la suma de las entradas está entre ambos límites, la activación se define como una función lineal como se muestra a continuación:

$$y_j = \begin{cases} 0 & si \quad x < -c \\ 1 & si \quad x > c \\ \frac{x}{2c} + \frac{1}{2} & en \quad otro \quad caso \end{cases} \tag{5}$$

En la Figura 12 podemos ver la representación gráfica de la función de activación mixta:

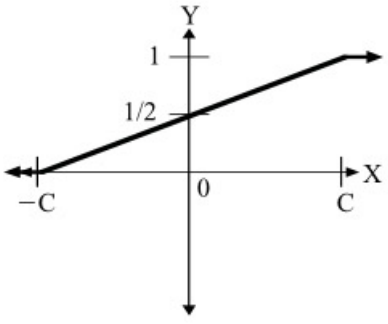


Figura 12: Gráfica de una función mixta.

Función sigmoidal

En la función de activación sigmoidal el valor de salida de la función es cercano a uno de los valores asintóticos por lo que la salida está comprendida en la zona alta o baja de la sigmoidal, esta función toma valores comprendidos entre 0 y 1. Si la pendiente

de la sigmoidal es elevada, a esta le pueden ser aplicadas reglas de aprendizaje para una función de tipo escalón. Esta función es muy utilizada para cuándo se realizará el entrenamiento de una red para reconocimiento de patrones y clasificación y es la cual será utilizada para el desarrollo de la parte práctica. La función sigmoidal es de la siguiente manera:

$$y_j = \frac{1}{1 + e^{-x}} \tag{6}$$

Y en la Figura 13 se puede ver la representación gráfica de la función de activación sigmoidal [20-22].

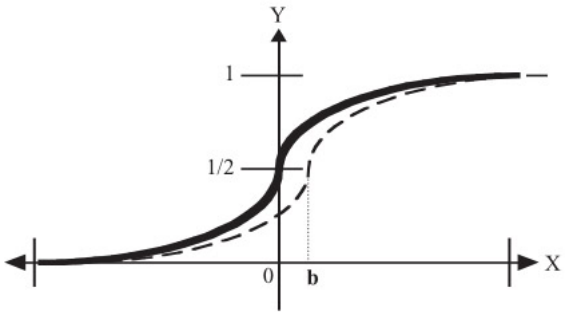


Figura 13: Gráfica de la función de activación sigmoidal.

2.2.1.3. Red neuronal backpropagation

Existen una gran cantidad de tipos de redes neuronales artificiales, estas varían en función de la aplicación que se les dará, entre las principales encontramos:

- Perceptrón
- Adaline
- Backpropagation

Para la realización de este trabajo de tesis se hará uso de la red de tipo backpropagation. La red backpropagation es un tipo de red que puede hacer uso de más de una capa oculta. Esta red hace uso del algoritmo de propagación hacia atrás o backpropagation, este es una regla de aprendizaje que es aplicable para redes de dos o más capas.

El funcionamiento de la red backpropagation consiste en realizar el aprendizaje mediante el uso de set de entrenamientos los cuales contienen todas las entradas con sus salidas deseadas, las cuales serán utilizadas para ajustar los pesos de las conexiones de las neuronas, el tipo de aprendizaje que maneja la red backpropagation es de tipo supervisada, para el cálculo de estos pesos se hace uso del algoritmo backpropagation o de propagación hacia atrás y consiste en:

- Empezar con los pesos de manera aleatoria.
- Introducir los datos de entrada del set o conjunto de entrenamiento.
- Dejar que la salida genere un vector de salida lo que también es conocido como propagación hacia adelante, ya que se va desde la capa de entrada, posteriormente la información pasa por la capa oculta para finalmente llegar a la capa de salida.
- Comparar la salida generada por la red con la salida deseada.
- La diferencia de la salida deseada y la salida generada denominada error se usa para ajustar los pesos de las neuronas de la capa de salida.
- El error se propaga hacia atrás hacia las capas anteriores y se usa para ajustar los pesos sinápticos en esa capa.
- El proceso anterior se aplica y se continúa propagando el error hacia atrás, ajustando los pesos hasta alcanzar la capa de entrada.
- Este proceso se repetirá con cada uno de los datos de entrada de cada uno de los sets de entrenamiento.

Para hacer esto se hace uso de la regla delta generalizada la cual fue utilizada para la creación de la red y el algoritmo de propagación hacia atrás o backpropagation. La regla delta se basa en ajustar los pesos de manera proporcional a la delta o diferencia entre la salida deseada y la obtenida:

$$\delta = (salida \quad deseada - salida \quad obtenida) \tag{7}$$

Una vez obtenida la delta en la salida o salidas de la red neuronal, esta comienza a propagarse, posteriormente se calcula la delta en cada una de las neuronas, se tendrán tantas deltas como número de neuronas presentes en la red, por cada capa que se propaga hacia atrás se va sumando el delta anterior.

Para actualizar los pesos se realiza de la siguiente manera:

$$w_{ij}^{'} = w_{ij} + \eta\delta_j \frac{df_j(e)}{de} y_i \tag{8}$$

En donde:

1. $w_{ij}^{'}$ es el nuevo peso de la neurona i que realiza una conexión con la neurona j, al ser un nuevo peso se denota con un apostrofe.
2. w_{ij} es el peso actual de la neurona i conectada con la neurona j.
3. La tasa de aprendizaje η, por lo general se busca que este no sea un valor muy grande o muy pequeño, ya que al ser muy pequeño al algoritmo de propagación hacia atrás realizara muchas iteraciones hasta encontrar un valor óptimo para los pesos y si el valor es muy grande, realizará una cantidad menor de iteraciones

y se puede dar el caso en el cual no se llegue a valores óptimos para el peso, se recomienda que este valor se encuentre en el orden de 0.05 a 0.25.

4. δ_j es la delta de la neurona conectada j.

5. $f_j(e)$ es la función de activación de la neurona j evaluada en el valor de la suma de las entradas ij.

6. y_i es el valor de salida de la neurona i.

Para este método se aplica un gradiente descendiente, éste se basa en buscar la dirección en la que una pequeña variación de los pesos hace que el error disminuya más rápidamente y localizar el punto mínimo [20-22].

Para el caso del desarrollo de este trabajo de tesis la red al ser aplicada para reconocimiento de patrones y clasificación es necesario agregar una función extra de activación en la capa de salida para así poder normalizarla y asegurar tener valores de salida entre 0 y 1 ya que para el caso de clasificación ese es el único posible rango y el cual debe ser distribuido entre todas las neuronas por lo que al final sumamos la salida de cada una de nuestras neuronas esta nos debe dar 1. Esta función se conoce como softmax o exponencial normalizada.

2.2.2. Toolbox red neuronal MATLAB

Para la elaboración de la red neuronal, el software MATLAB cuenta con un toolbox para la creación de redes neuronales. El toolbox de redes neuronales puede manejar redes neuronales supervisadas y no supervisadas además de manejar redes con propagación hacia adelante.

También maneja diversos algoritmos de entrenamiento para el ajuste de pesos de las redes neuronales, en el cual nuestro interés está en el gradiente descendiente para la realización de una red de propagación hacia atrás.

El toolbox cuenta con una gran cantidad de funciones además de poder hacer uso de otras funciones de MATLAB. También es posible generar directamente la función de la red neuronal en un archivo m en la cual solo es necesario introducir nuestro vector de entrada y automáticamente obtener el resultado de la red, las funciones a utilizar serán detalladas en el capítulo tres en donde se mostrará cómo se realizó la red neuronal [24].

2.3. Interfaz Gráfica

Con el fin de hacer más amigable el sistema y más sencillo de utilizar, se hará uso del entorno de diseño de interfaz gráfica de usuario de MATLAB también conocida como GUIDE por sus siglas en ingles. Para abrir esta herramienta solo es necesario escribir la palabra “guide” en la ventana de comandos de MATLAB y la interfaz de GUIDE

nos ofrecerá diversas opciones como crear una aplicación en blanco o con controles predefinidos.

Esta herramienta de MATLAB permite crear aplicaciones, las cuales usualmente pueden contener menús, barras de herramientas, botones, áreas de texto, etc. Para desarrollar una aplicación visual o con interfaz gráfica de usuario también conocido como GUI por sus siglas en inglés se puede realizar desde dos aproximaciones diferentes, se puede desarrollar de manera visual haciendo uso del editor de diseño de GUIDE con el cual se puede diseñar la interfaz de usuario de manera gráfica y GUIDE generará automáticamente el código de MATLAB para la interfaz. En la Figura 14 se puede ver como es el editor de diseño junto con las diversas herramientas y elementos que pueden ser agregados.

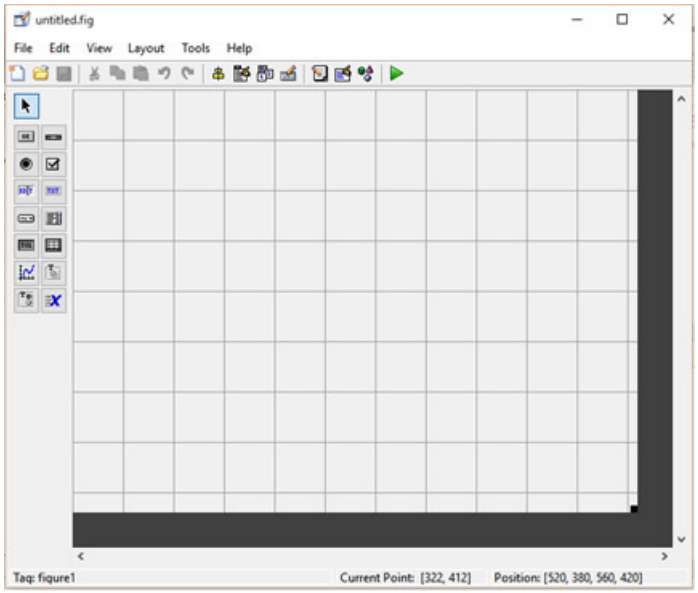


Figura 14: Editor de diseño GUIDE de MATLAB.

Para agregar elementos en el editor de elementos es tan sencillo como seleccionar el elemento en la barra de herramientas de la izquierda y arrastrar el elemento al área de trabajo, y si se desean editar las propiedades del elemento, solo es necesario dar doble clic sobre el elemento una vez esté en el área de trabajo.

Si se desea tener más control sobre el diseño y control de la aplicación desarrollada en MATLAB se puede complementar con el uso de programación mediante el lenguaje de programación M. En el capítulo tres se explica a detalle los elementos que son utilizados para el desarrollo de la parte de la interfaz gráfica [25].

Capítulo 3

Elaboración del sistema de ubicación

En este capítulo se describe a detalle el procedimiento llevado a cabo para la realización de cada uno de los elementos que conforman el sistema de ubicación. En la sección 3.1 se presenta el desarrollo llevado a cabo para realizar el procesamiento de las imágenes las cuales serán utilizadas como base de entrenamiento, esto mediante el toolbox de imágenes de MATLAB. En la sección 3.2 se presenta el método propuesto para la creación de patrones únicos para cada una de las imágenes a utilizar como base de entrenamiento para la red neuronal, esto mediante el uso del lenguaje de programación de MATLAB y sus funciones. En la sección 3.3 se presenta el desarrollo de la red neuronal a utilizar haciendo uso de los patrones generados en el segundo inciso y el cual nos dará como resultado un código único de una aproximación para cada ubicación. En la sección 3.4 se muestra a detalle cómo se determinó de manera más precisa la ubicación haciendo uso de los datos generados por la red neuronal y los patrones generados. En la sección 3.5 se presenta el desarrollo de la interfaz gráfica para realizar las pruebas del sistema y poder observar de manera más sencilla los resultados de cada una de las imágenes probadas. En la Figura 15 se muestra un diagrama del procedimiento a seguir por el sistema para determinar la ubicación.

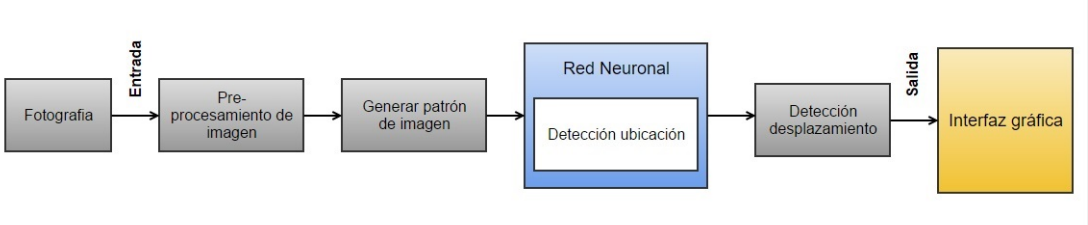


Figura 15: Diagrama de flujo del sistema de ubicación

3.1. Pre-procesamiento de imágenes

Para poder entender este procedimiento primero es necesario definir las características con las que debe contar cada una de las imágenes a utilizar, esto con el fin de poder definir todas las operaciones a aplicar a cada una de las imágenes y así poder hacer un procesamiento de manera óptima.

Las características que deben de poseer las imágenes para poder ser utilizadas en el sistema son las siguientes:

- Relación de aspecto de 4:3 el cual es común en las cámaras digitales.
- Profundidad de color preferentemente de 24 bits, esto evitará que haya pérdidas de información de la imagen en todo el proceso.
- Tipo de archivo jpg por su buena compresibilidad y buena calidad además de ser utilizado por cámaras digitales.
- El modo de color de la imagen debe ser RGB ya que este es utilizado comúnmente en imágenes digitales además de ser utilizado por cámaras digitales.

Una vez establecidas las características de las imágenes es necesario definir qué fotografías serán utilizadas y el porqué de estas, para poder entender esto es necesario remontarse a lo que es el área de astronomía.

Cuando uno ve al cielo puede ver distintas estrellas agrupadas y si uno mira detalladamente puede notar que algunas estrellas brillan más que otras además de que esos patrones de estrellas son únicos y usando un poco de imaginación se pueden trazar figuras imaginarias con formas únicas haciendo uso de las estrellas que se ven. Haciendo uso de ese conjunto de estrellas visiblemente agrupadas es posible delimitar diversas áreas en el cielo las cuales ocupan cierto porcentaje del cielo visto desde la Tierra y en órbitas de esta. Estas son conocidas como constelaciones las cuales son áreas perfectamente delimitadas. En el año de 1928 la Unión Astronómica Nacional estableció las 88 constelaciones que actualmente son usadas en el área de astronomía, esta división se da en base a lo que es observado desde la Tierra, en la Figura 16 se ve un ejemplo de cómo son observadas algunas de las 88 constelaciones desde la Tierra.

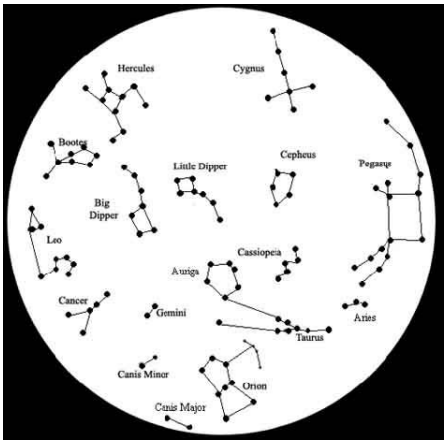


Figura 16: Constelaciones vistas desde la Tierra.

Por lo tanto con el fin de poder identificar la ubicación aproximada del minisatélite en órbita se hace uso de las constelaciones vistas desde órbita. Para la realización de este trabajo se hace uso sólo de una cuarta parte de una constelación con la finalidad de mostrar la funcionalidad del sistema y la capacidad de identificar la ubicación aproximada y un sistema de referencia para conocer la dirección de desplazamiento. Para realizar esto se hace uso de fotografías más pequeñas de diversas regiones de la fotografía de la cuarta parte de la constelación a utilizar para así poder generar la base de entrenamiento de la red neuronal, en la Figura 17 se muestra las áreas vistas desde un mapa estelar y las constelaciones contenidas, mientras que la Figura 18 muestra la imagen pre-procesada.

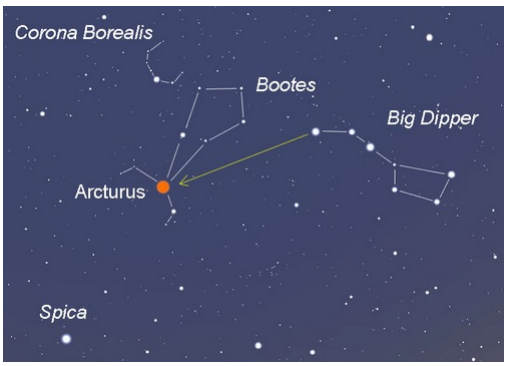


Figura 17: Ejemplo de área del espacio y las constelaciones contenidas en esta área.

Figura 18: Imagen de ejemplo pre-procesada con MATLAB.

El pre-procesamiento que se lleva a cabo es aplicado a las imágenes usadas para generar los patrones únicos y que serán utilizados para el entrenamiento, además este pre-procesamiento es utilizado para las fotos que se deseen probar en el sistema y del cual se desee conocer la ubicación aproximada.

Como se mencionó anteriormente, para la realización de este trabajo se hará uso del software MATLAB en su versión 2013b el cual contiene diversas herramientas para procesamiento, para redes neuronales y para la interfaz gráfica además de poseer su propio lenguaje de programación por lo que los código citados en el trabajo se garantiza su funcionamiento correcto para la versión 2013b.

Para iniciar el pre-procesamiento es necesario cargar la imagen en MATLAB además de almacenarla en una variable de tipo array, el toolbox de procesamiento de imagen nos proporciona la instrucción imread la cual permite hacer lo descrito anteriormente, posteriormente se requiere obtener la información de la imagen cargada, específicamente el ancho y el alto, para realizar esto se hace uso de la función imfinfo la cual almacena todos los datos de la imagen en una variable como son: resolución, formato, profundidad de color, esta información es requerida para conocer si la imagen posee la resolución de 512x384 px, en caso de ser mayor o menor se hace uso de la función imresize la cual permite ajustar el tamaño de la imagen por un tamaño especificado

Una vez realizado el ajuste al tamaño correcto se procede a realizar la primer transformación a la imagen, para hacer esto se convierte la imagen que se encuentra en RGB al ser tomada por una cámara digital a escala de grises, esta transformación hace que los colores de la imagen se transformen en diferentes tonalidades de grises, con esto se reduce la información necesaria para cada píxel de la imagen pero manteniendo la luminosidad de cada una de los objetos que se encuentran en la imagen, esta transformación se realiza mediante la función rgb2gray y que se puede observar el antes en la

Figura 19a y el después en la Figura 19b.

(a) **(b)**

Figura 19: Constelación de Andrómeda. En (a) se muestra esta constelación sin filtro. Mientras que en (b) aplicando filtro con función MATLAB.

Posteriormente a la imagen en escala de grises se le aplica otra transformación llamada complemento la cual es usada mediante la función imcomplement, en una imagen binaria el complemento invierte los valores de la imagen, mientras que en una imagen en escala de grises los píxeles más oscuros se vuelven más claros mientras que los más claros se vuelven más oscuros como se puede ver en la Figura 20.

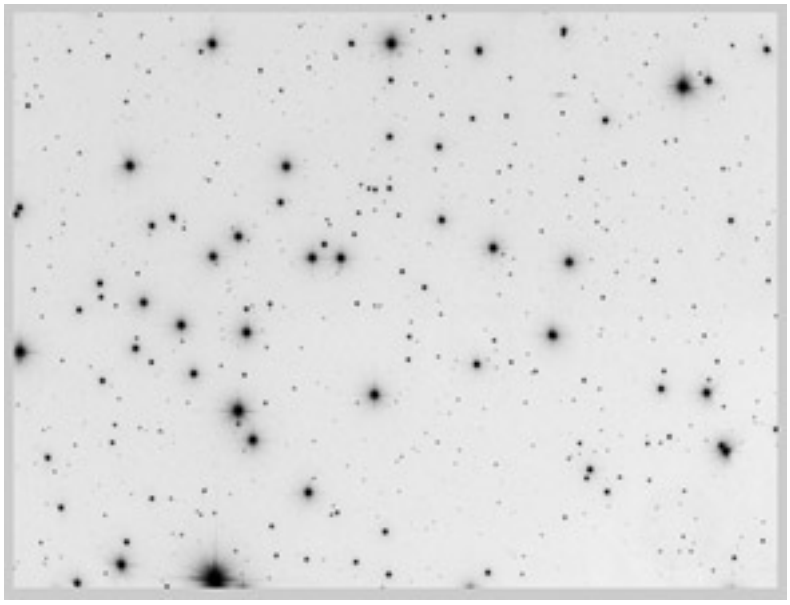


Figura 20: El complemento de la imagen en escala de grises.

La siguiente parte con la que se sigue es la transformación del complemento de la imagen a blanco y negro, el complemento es necesario ya que nos permite identificar los objetos brillantes en un rango de negro a gris en lugar de la escala blanco a gris que nos da la escala de grises, el cambio no se hace de manera directa de la imagen en RGB a blanco y negro ya que esto causa que algunos de los objetos de la imagen desaparezcan además de que de esta manera propuesta son más visibles los resultados de los objetos más visibles en la fotografía. Para realizar la transformación a blanco y negro se hace uso de la función im2bw la cual requiere de dos argumentos, el primero es la imagen a utilizar, en este caso el complemento de la imagen, y como segundo el nivel el rango de detección de grises y el cual varía entre 0 y 1, mientras más grande sea el número mayores tonalidades de grises detectara y transformara a color negro, para poder elegir este valor se hizo de manera experimental buscando disminuir la cantidad de objetos visibles en la fotografía pero sin desaparecerlos en su totalidad usando el nivel de detección en 0.2 y que da como resultado lo que se puede ver en la Figura 21.

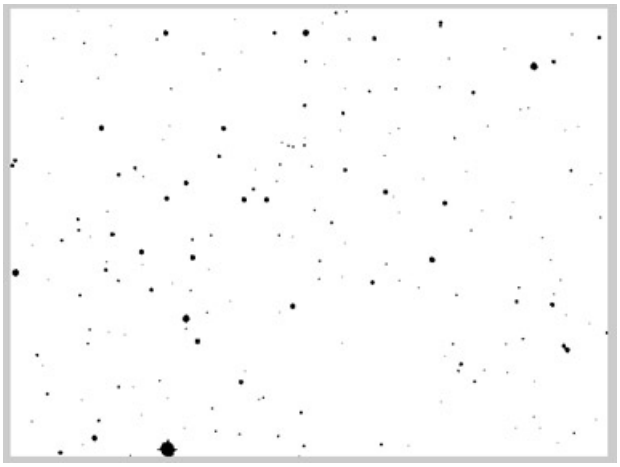


Figura 21: Transformación del complemento de la imagen a blanco y negro.

Para finalizar el pre-procesamiento se vuelve a aplicar la función imcomplement para volver a invertir los colores, esto hará que los objetos en negro pasen a blanco y lo blanco pase a negro, esto se hace con el fin de la siguiente parte del pre-procesamiento la cual consiste en eliminar los puntos más pequeños de la imagen para intentar disminuir más la cantidad de objetos en función a su tamaño, para realizar esto se hace mediante la función bwareaopen la cual posee dos argumentos, el primero es la imagen en la que se desean eliminar los objetos, y el segundo argumento es el máximo número de píxeles a eliminar en la imagen, en este caso se usó el valor de 20 y se puede ver en la Figura 22.

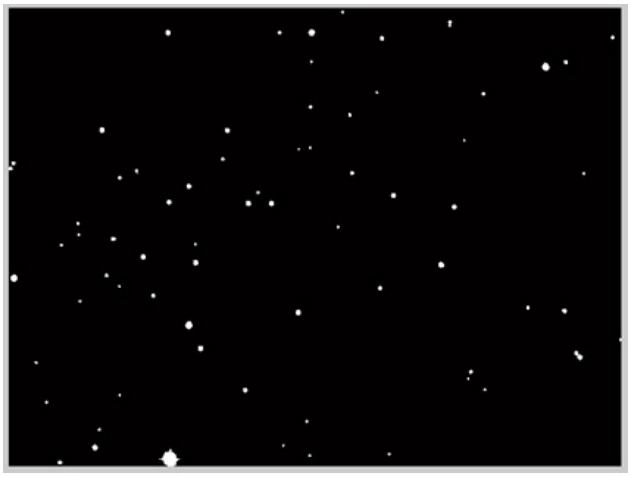


Figura 22: Complemento de la imagen y eliminación de objetos menor o igual a 20 píxeles.

3.2. Generar patrones para imágenes

Una vez terminado el pre-procesamiento de las imágenes se procede a desarrollar la parte del sistema la cual se encarga de generar una serie de patrones únicos para cada una de las imágenes que es utilizada para generar los sets de entrenamiento además de que con esta misma parte del sistema poder generar un patrón de entrada en función a la fotografía que se desee detectar.

Para poder generar los patrones se procede con la extracción de características la cual consiste en detectar todos los objetos que existen en la imagen pre-procesada los cuales tienen diferentes grados de brillo, al ser objetos que poseen un contorno cerrados se procede a usar la función bwboundaries de MATLAB la cual permite trazar los límites exteriores de los objetos en blanco que se encuentran en la imagen binaria y posteriormente son almacenados en un arreglo de celdas de tamaño P x 1 en donde P es el número de objetos y/o huecos encontrados en la imagen y en cada una de las celdas se almacena otro arreglo de celdas de tamaño Q x 2 el cual contiene las coordenadas de cada uno de los píxeles que conforman el límite o contorno de la imagen, en la Figura 23a podemos ver un ejemplo del arreglo de celdas generado de la Figura 22 y en la Figura 23b se puede ver las coordenadas almacenadas de cada uno los píxeles que conforman el contorno de un objeto detectado.

B

143x1 cell

	1
1	14x2 double
2	15x2 double
3	17x2 double
4	26x2 double
5	18x2 double
6	28x2 double
7	20x2 double
8	20x2 double
9	39x2 double
10	14x2 double
11	19x2 double
12	28x2 double
13	36x2 double
14	32x2 double

(a)

B B{1, 1}

14x2 double

	1	2
1	718	27
2	718	28
3	718	29
4	719	30
5	720	31
6	721	30
7	722	30
8	722	29
9	722	28
10	722	27
11	721	27
12	720	27
13	719	27
14	718	27

(b)

Figura 23: Elementos detectados. En (a) se observan en las celdas cada uno de los objetos detectados. Mientras que en (b) se observa una de las celdas de la figura 22a en donde se encuentran las coordenadas en píxeles del contorno del objeto detectado.

Una vez obtenidos todos los objetos existentes en la imagen y las coordenadas de sus contornos se procede a estimar las coordenadas aproximadas de los centros de cada uno de los objetos, para hacer eso se buscan los puntos mínimos y máximos de las coordenadas obtenidas para cada objeto y se estima haciendo un promedio.

Una vez obtenidos los centros aproximados, usando las mismas coordenadas obtenidas de cada punto podemos estimar el radio en píxeles de cada uno de los objetos, por lo que entonces se procede a crear una nueva matriz con tres columnas, en donde se almacena la coordenada en x y en y del centro además el radio de cada uno de los elementos detectados como se puede ver en la Figura 24. La cantidad de filas varía de acuerdo a la cantidad de objetos detectados en la imagen.

10x3 double

	1	2	3
1	328.5000	53	1.5700
2	108	352	2.0300
3	256	78.5000	2.0600
4	379	251	2.0800
5	509.5000	150	2.5900
6	14.5000	25	3.5000
7	19	148.5000	4.0100
8	235	220.5000	5.0500
9	106.5000	196	5.5200
10	207	343	6.0400

Figura 24: Matriz la cual contiene los datos de los objetos detectados, estos datos son: coordenadas del centro y tamaño del objeto.

Posteriormente se procede a identificar cual es el objeto más grande en función al radio y los siguientes tres objetos más grandes, viendo esto en contraste con la imagen pre-procesada significa que mientras mayor sea el radio del objeto, el brillo de la estrella es mucho mayor. Estos 4 elementos más grandes son almacenados en una nueva matriz de tamaño 4x3 y re ordenados en función al tamaño de radio con lo cual el elemento más grande queda en la primera fila.

Con los datos almacenados en el anterior paso se procede a generar lo que serán los patrones para la imagen de entrada, los patrones de cada imagen constaran de distancias en píxeles entre el objeto más grande y cada uno de los siguientes tres objetos almacenados como se puede observar en la Figura 25 en donde se ven conexiones con líneas amarillas desde el objeto más grande a los siguientes tres objetos con mayor tamaño.

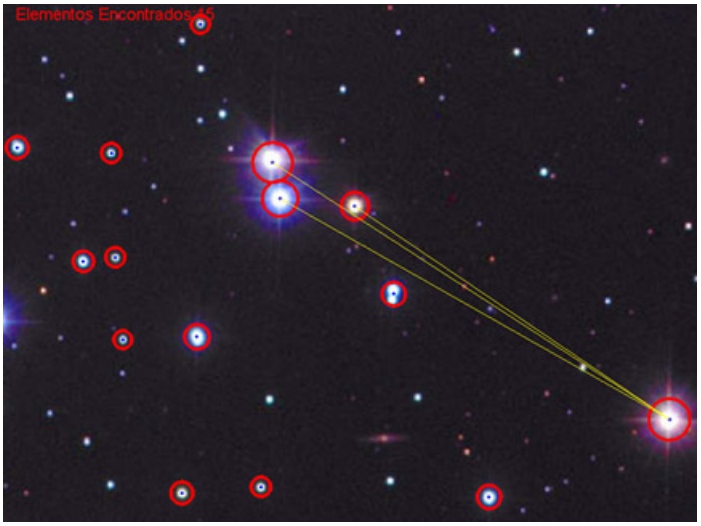


Figura 25: Conexiones entre el elemento más brillante y los siguientes tres más brillantes.

Con esto se generan 3 datos de distancias en píxeles para crear los patrones para cada imagen, pero con el fin de generar más conexiones neuronales y así tener más posibilidad de ajuste en los pesos y mejorar los resultados se genera un vector de 16 elementos lleno de ceros y del cual sólo tres elementos se llenan con los patrones generados como se puede ver en la Figura 26, esto dicho en otras palabras significa que a los 3 valores generados se agregan ceros en los demás elementos del vector hasta tener 16 elementos. Esto también permite en un futuro agregar más valores al patrón generado sustituyendo los valores de 0 por más distancias medidas entre objetos detectados en el vector de 16 elementos.

DistA1

1x16 double

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0	178.0709	125.6593	270.5074	0	0	0	0	0	0	0	0	0	0	0	0

Figura 26: Ejemplo de patrón generado para una imagen.

Una vez generados todos los patrones que serán utilizados para la red neuronal estos han de almacenarse en un archivo de texto en donde cada una de las líneas debe ser un vector patrón generado y en un segundo archivo es donde se almacenan los objetivos o salidas definidas para cada uno de los patrones siguiendo el mismo formato que para los vectores patrón, una línea para cada objetivo o salida establecido, la primer línea de texto del vector patrón está ligada a la primer línea de texto del objetivo definido por lo que es necesario cerciorarse que estén de manera correcta. Estos archivos serán utilizados en la siguiente fase la cual consiste en generar la red neuronal.

3.3. Red neuronal para reconocimiento de patrones

Para poder entrenar una red neuronal de tipo supervisada es necesario contar con una serie de patrones de entrenamiento y salidas definidas las cuales son generadas en el apartado anterior. Una vez creada la parte encargada de generar los patrones de entrenamiento y crear los archivos que los contienen los patrones de entrenamiento y además de tener la capacidad de generar los patrones de las imágenes de prueba se procede a crear la red neuronal usando el toolbox de MATLAB de redes neuronales. Para iniciar el toolbox solo es necesario ejecutar la función nnstart en la línea de comandos, lo cual nos da las opciones mostradas en la Figura 27.

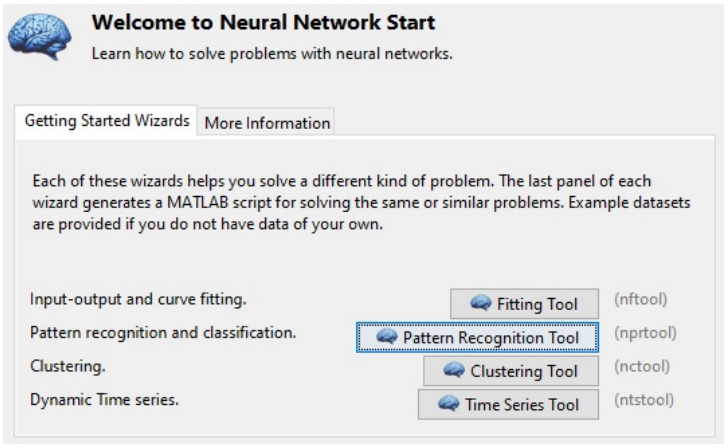


Figura 27: Opciones del toolbox de redes neuronales.

Posteriormente se elige la opción Pattern Recognition Tool la cual nos configura automáticamente la red para ese uso en específico, en esta opción se pide que se seleccionen los archivos que serán utilizados como patrones de entrenamiento (Inputs) y sus salidas definidas (Targets) como se puede ver en la Figura 28, estos archivos fueron generados en el anterior paso.

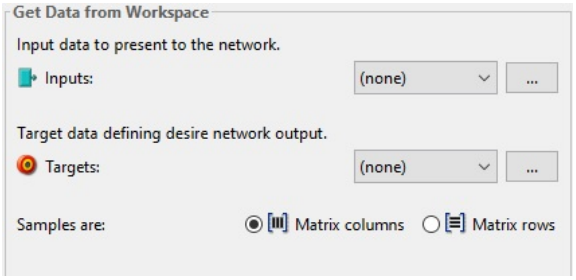


Figura 28: Selección de patrones de entrenamiento y sus salidas deseadas.

En el siguiente paso se procede a seleccionar de todos los patrones de entrenamiento, una cantidad de muestras de los patrones ingresados, los cuales serán asignados a cada una de las siguientes opciones: Entrenamiento, Validación y Prueba como se ven en la Figura 29. Las muestras asignadas a la parte de entrenamiento son utilizados para ajustar los pesos de la red neuronal hasta disminuir el error de las salidas aproximándose a 0. Las muestras asignadas a validación son utilizadas para medir la capacidad de generalización de la red neuronal y para detener el entrenamiento cuando ya no exista un decremento en el desempeño de la red. Y por último las muestras asignadas a la opción de prueba sirven para poder medir el desempeño de la red neuronal durante y después del entrenamiento.

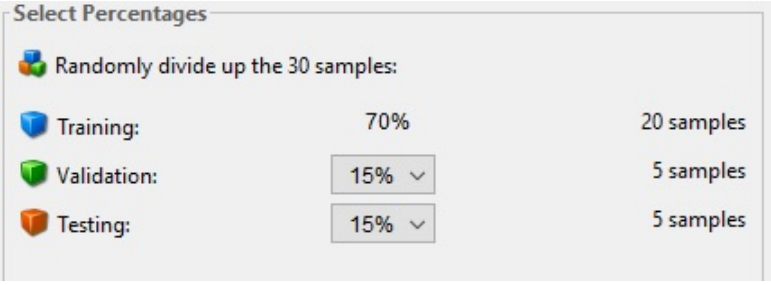


Figura 29: Asignación de muestras a cada una de las opciones para entrenamiento, validación y prueba de la red neuronal.

En la siguiente opción del toolbox, se requiere ingresar el número de neuronas que serán utilizadas en la capa oculta como se puede ver en la Figura 30.

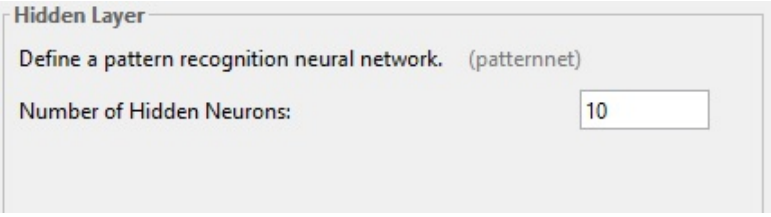


Figura 30: Asignación de neuronas a la capa oculta.

La red neuronal sigue una forma piramidal, lo que significa que se presenta un número decreciente de neuronas de la entrada a la salida. Para poder calcular el número de neuronas se hace uso de la regla de la pirámide geométrica que se utiliza cuando solo se tiene una capa oculta y es de la siguiente manera:

$$h = \sqrt{m \times n} \tag{9}$$

En la ecuación 9 se tiene que h es el número inicial de neuronas de la capa oculta, m es el número de neuronas de entrada y n es el número de neuronas de salida. Para

obtener el número óptimo de neuronas de la capa oculta se debe ir incrementando de uno en uno el número de neuronas ocultas, entrenar y después probar. Se repite lo anterior hasta que el error sea lo más pequeño posible o que no haya una mejora significativa en el error. En el siguiente paso se indica que se puede proceder a hacer el entrenamiento dando clic en el botón Train, lo cual nos lleva a la ventana en donde se puede ver el proceso del entrenamiento, en caso de ser necesario el botón de Train puede ser apretado varias veces hasta alcanzar el valor deseado en el entrenamiento usando como base los datos que aparecen en la Figura 31.

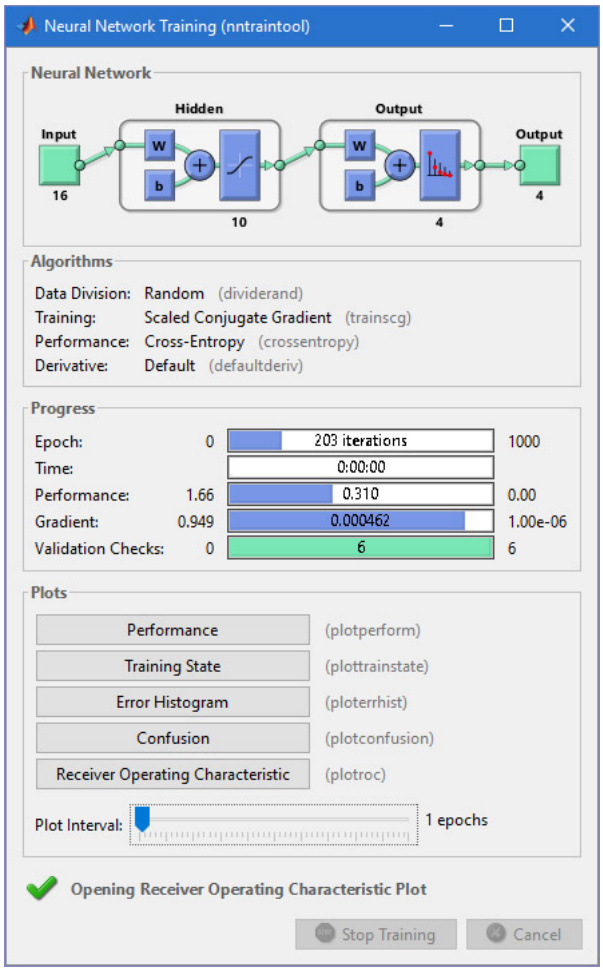


Figura 31: Entrenamiento de la red neuronal.

El toolbox de redes neuronales configura automáticamente las opciones necesarias para el entrenamiento de redes neuronales, las cuales se pueden observar en la Figura 31 y son Data Division, Training, Performance y Derivative. La opción Data Division es definida cuando se establecen los porcentajes de muestras a utilizar para el entrenamiento, validación y prueba de la red en la Figura 29, esta opción está definida con la

función dividerand() la cual establece los porcentajes, el número total de muestras de los objetivos definidos, además de escoger las muestras para cada porcentaje de manera aleatoria.

La opción Training es definida automáticamente por el toolbox para reconocimiento de patrones con la función trainscg, la cual es la función que es utilizada para el entrenamiento para hacer el ajuste de los pesos de la red neuronal, el algoritmo usado en esta función es conocido como scaled conjugate gradient backpropagation o gradiente conjugado escalado de propagación hacia atrás, este método particular de backpropagation hace uso de gradientes para conocer la dirección en la que se da una variación más pronunciada para intentar converger al mínimo error posible. La función trainscg además establece ciertos parámetros, que al cumplirse, se detiene el entrenamiento automáticamente, entre los principales está cuando se alcanza el gradiente mínimo de 1e-6, cuando el error se hace 0 que en MATLAB es conocido como objetivo de rendimiento o cuando se alcanza el límite de iteraciones el cual es de 1000. Todos estos valores están establecidos por defecto con el fin de intentar converger lo más rápido posible con la menor cantidad de iteraciones posibles, en caso de querer mejorar más los resultados es posible ajustar los anteriores parámetros de la función [26].

La opción Performance por defecto para reconocimientos de patrones está definido por la función crossentropy(), esta es conocida como función de error, la cual es usada para calcular el error que será utilizado para los valores asignados en validación por la función de backpropagation y para evaluar el desempeño buscando que durante 6 pruebas continuas no exista una disminución en el error y en caso de ser así terminar el entrenamiento. Esta función es elegida por defecto por su buen desempeño para redes neuronales enfocadas a clasificación y reconocimiento de patrones que usan como objetivos o valores de salida definidos [0,1] y se presenta de la siguiente manera, la cual puede ser simplificada para el algoritmo backpropagation:

$$\frac{\partial E}{\partial net} = y - t \tag{10}$$

En donde y es el objetivo o salida obtenida y t es el objetivo o salida definida para el patrón que se está entrenando [27].

La opción derivative está establecida por defecto con la función defaultderiv() la cual está encargada de elegir de manera automática el algoritmo utilizado para computar las derivadas usadas en las funciones establecidas en las anteriores funciones de la red neuronal, para este caso, la función defaultderiv() hace uso de la función staticderiv(), esta función calcula las derivadas haciendo uso de la regla de la cadena.

La red neuronal puede requerir ser entrenada varias veces hasta que se logre llegar a un valor muy cercano 0 o se cumpla alguna de las otras condiciones antes mencionadas. En el caso de que la red después de varios entrenamientos no otorgue los resultados favorables es necesario ir incrementando el número de neuronas haciendo uso del método de prueba y error.

Una vez que la red neuronal fue entrenada correctamente MATLAB ofrece la oportunidad de generar un archivo .m con una función para la red neuronal, esto con el fin

de para poder probarla más fácilmente con otros valores para poder observar el desempeño para la detección de las fotografías y su capacidad para determinar de manera inicial la ubicación. Este archivo .m con la función es colocado en el mismo directorio que la interfaz gráfica, ya que esta función es utilizada para mostrar las salidas de la red neuronal en la interfaz.

3.4. Sistema de seguimiento

Una vez identificada una ubicación aproximada mediante la red neuronal se procede a establecer un sistema de referencia para determinar una ubicación más certera usando como base el objeto más brillante detectado en la fotografía de entrada del sistema. Para realizar esto se establece como 0 de referencia en la imagen el punto 256x384 en píxeles, como se puede ver en la Figura 32 marcado con un círculo de color verde.

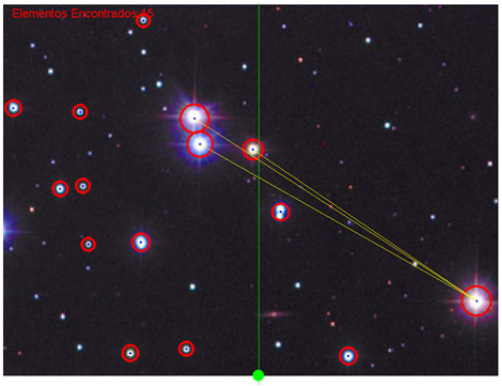


Figura 32: Punto de referencia en la imagen.

Posteriormente se calculan las coordenadas en la imagen del punto más brillante con respecto al punto verde de referencia, esto significa que para determinar de mejor manera la ubicación una vez detectada de manera general el área con la red neuronal, se seguirá al objeto más brillante con lo cual se puede determinar un desplazamiento usando los valores en X y en Y con respecto a la referencia como se puede observar en la Figura 33.

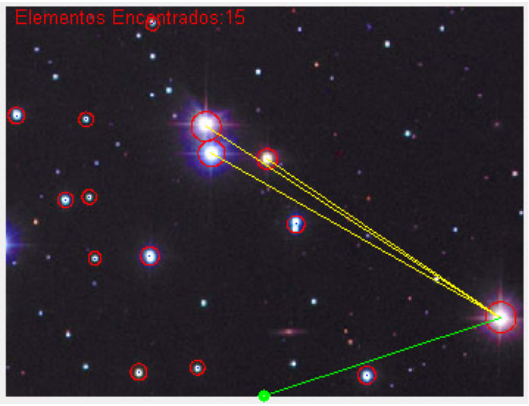


Figura 33: Seguimiento del objeto más brillante.

3.5. Interfaz gráfica

Por último se procede a hacer la interfaz gráfica haciendo uso de GUIDE de MATLAB, con la interfaz gráfica creada se pueden hacer pruebas directas de la red neuronal sin necesidad de hacer uso de la línea de comandos de MATLAB como se puede ver en la Figura 34.

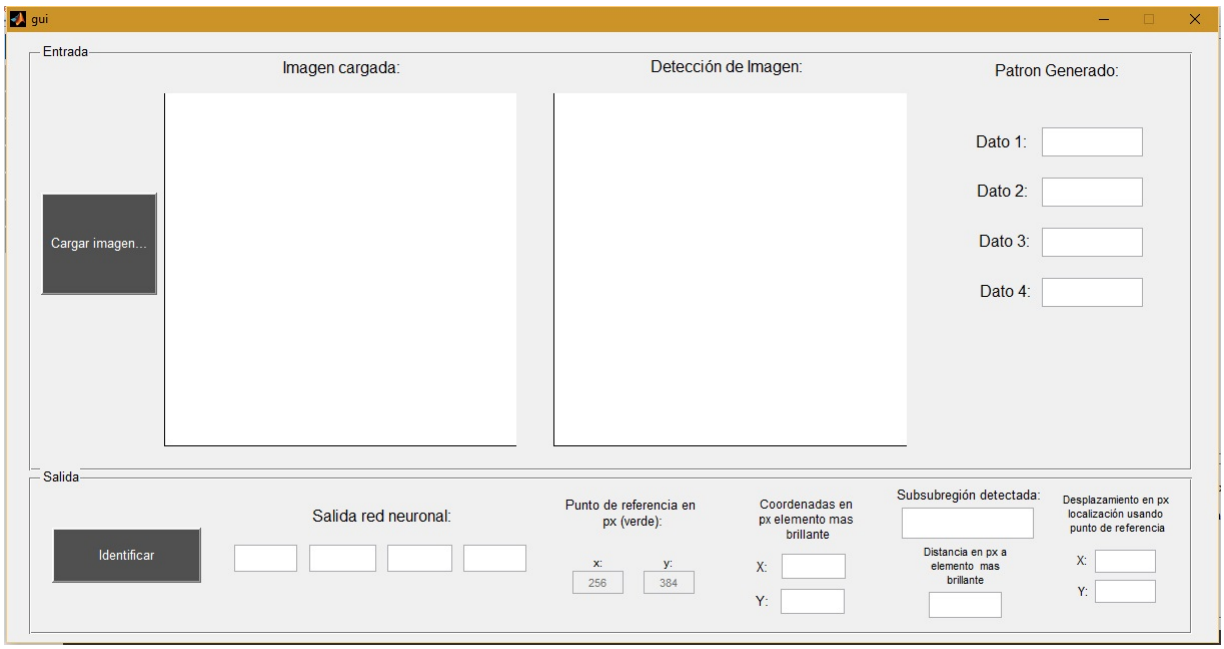


Figura 34: Interfaz Gráfica para la prueba de la red neuronal.

Para hacer uso de la interfaz gráfica solo se requiere cargar una fotografía, posteriormente hacer clic en el botón identificar, esperar a que termine el proceso y con esto se puede observar la salida generada por la red neuronal, a partir de la salida de la red se determina el área detectada de la constelación, después muestra la distancia al elemento más brillante y el desplazamiento usando el punto de referencia. Además la interfaz gráfica muestra algunos de los procesos realizados como: la imagen cargada, la imagen con los elementos detectados, la cantidad de objetos detectados, el patrón generado para esa imagen, coordenadas del objeto más brillante como se puede observar en la Figura 35 en la cual se puede observar la estimación realizada para la ubicación aproximada.

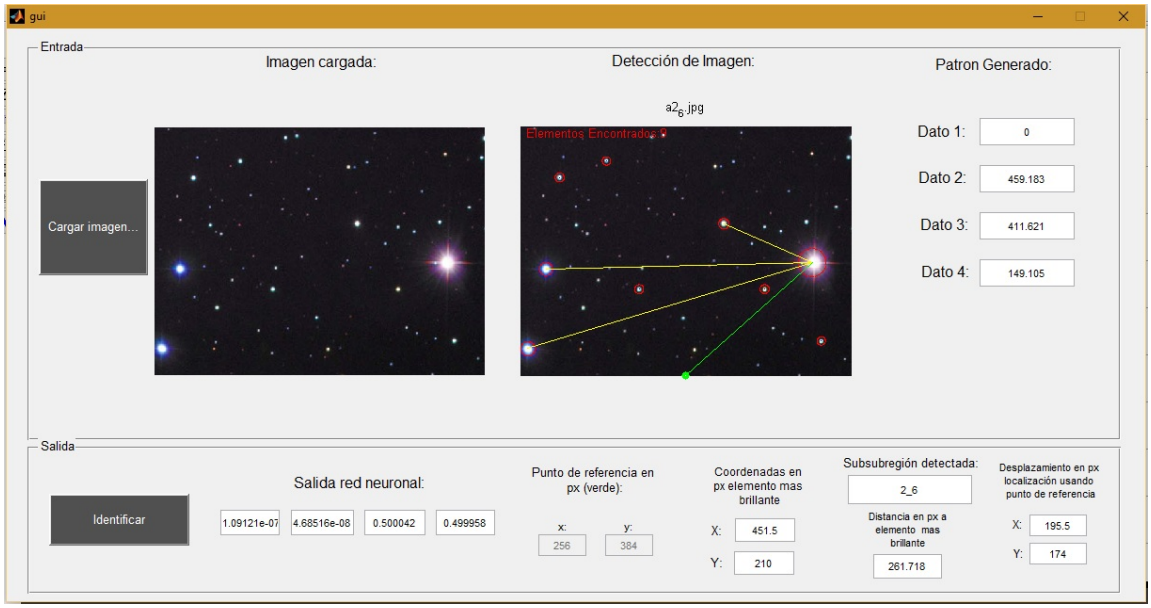


Figura 35: Interfaz Gráfica en funcionamiento.

Capítulo 4

Método para ubicar un minisatélite

En este capítulo se presentan los resultados que se obtienen al implementar los elementos propuestos en el capítulo 3. En la sección 4.1 se implementa el procedimiento para generar las imágenes las cuales se van a utilizar como referencia para crear una base de datos. En la sección 4.2 se muestran los patrones que se obtuvieron de las imágenes de la sección 4.1, los cuales forman la base de datos para la red neuronal. En la sección 4.3 se muestran todos los resultados obtenidos del entrenamiento de la red neuronal utilizando la base de datos. Y por último en la sección 4.4 se muestran los resultados obtenidos para diversas imágenes de pruebas.

4.1. Preparación de imágenes para la base de datos

En esta subsección se utiliza una fotografía de la constelación de Andrómeda para generar una serie de imágenes las cuales sirven de ejemplo para entrenar posteriormente la red neuronal. La fotografía posee una resolución de 4000x2658 píxeles o en su equivalente a 10.6 megapíxeles, la cual después fue reducida a un tamaño de 2000x1329 píxeles para hacerla más manejable para los procesos posteriores, la imagen se puede observar en la Figura 36 [28].

Figura 36: Fotografía de la constelación de Andrómeda, en resolución 4000x2658.

Como se mencionó anteriormente, se genera la base de datos de sólo una cuarta parte de la constelación que aparece en la fotografía, esto con el fin de demostrar la funcionalidad del sistema y su capacidad para identificar la ubicación, esto se hizo principalmente porque el tiempo requerido para generar la base de imágenes se incrementa ya que esto se hace de manera manual, además el tiempo de funcionamiento del sistema se irá incrementando conforme se incrementa la cantidad de imágenes a usar para el entrenamiento ya que esto implica generar más patrones, además del tiempo requerido para el entrenamiento de la red neuronal con el fin de obtener los mejores resultados posibles y para esto se realiza por el método de prueba y error hasta alcanzar los mejores valores posibles para la red. En la Figura 37 se puede observar el área de la constelación que es utilizada para generar los datos que serán utilizados, se tomó una subregión ligeramente más grande de tamaño 1024x768 píxeles para mantener la relación de aspecto 4:3 y mantener los requerimientos establecidos para las fotografías ya que la fotografía original posee una relación de aspecto 16:9.

Figura 37: Subregión de la constelación a detectar y ubicar.

Una vez recortada la imagen de la subregión a procesar, se procede a crear subsubregiones las cuales son imágenes que parten de la subregión de la Figura 37 y que son utilizadas como base para generar diversos patrones para caracterizar la subregión marcada en rojo, para realizar esto la imagen fue dividida en 9 distintas subsubregiones de resolución 512x384 píxeles, estas se escogieron de diferentes partes de la imagen sin dejar ningún espacio de la imagen sin usar, esto con el fin de generar más patrones diferentes pero pertenecientes a esa subregión y así de aumentar el rango de detección, las subsubregiones elegidas se pueden ver en la Tabla 1 de la sección 4.2.

4.2. Generar patrones de imágenes

A continuación en la Tabla 1 se muestra cada una de las subsubregiones creadas, las cuales son utilizadas para el entrenamiento de la red neuronal, además se muestra el patrón generado para cada imagen, la salida deseada y una etiqueta la cual identifica cada una de las subsubregiones las cuales son mostradas en la interfaz gráfica.

En la Tabla 1 los términos de las columnas tienen el siguiente significado: el patrón generado representa los valores únicos generados para esa imagen por MATLAB en función a las distancias medidas entre los objetos más brillantes, la salida deseada representa los valores que definimos para nuestra salida en función de los patrones de entrada generados, la etiqueta se establece con el fin de mostrar en la aplicación gráfica la subsubregión a la que pertenece una fotografía de acuerdo a cada una de las 9 imágenes de la tabla, la etiqueta representa la ubicación aproximada con respecto a la constelación de Andrómeda y comparando con las 9 imágenes originales, por ultimo las coordenadas de referencia del objeto más brillantes son utilizadas posteriormente para calcular el desplazamiento. Con los valores de la Tabla 1 anterior se genera el archivo con patrones generados y el archivo con las salidas deseadas de los patrones generados los cuales son utilizados con el toolbox de MATLAB, para esto los patrones de entrada se repitieron once veces cada uno y lo mismo para los patrones de salida dando un total de 99 patrones de entrada y 99 patrones de salida esto con el fin de que al realizar las iteraciones para los ajustes se compruebe de manera constante que la red funciona bien con todos los patrones de entrada. Posteriormente como se mencionó en el apartado 3.3 se procede a asignar una cantidad de patrones a la parte de entrenamiento, validación y prueba, en el primero se asignaron 69 muestras, en el segundo 15 y en el tercero 15 muestras.

Tabla 1: Patrones de subsubregiones.

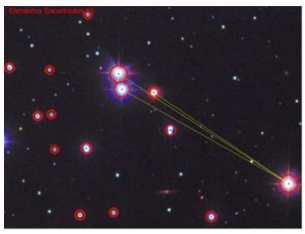
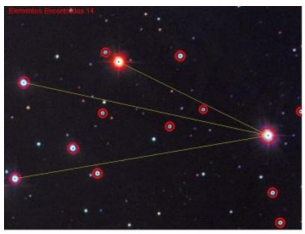
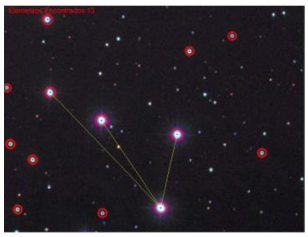
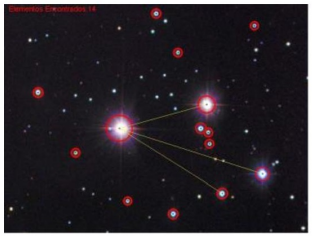
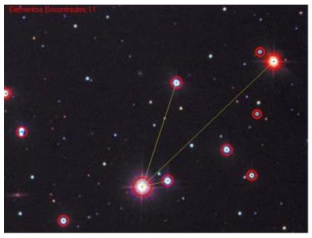
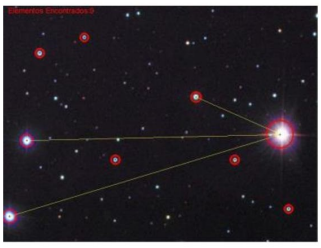
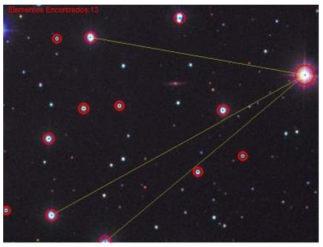
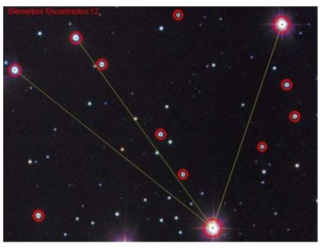
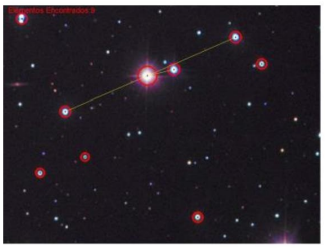
	Imagen	Patrón generado por MATLAB	Salida deseada red neuronal	Etiqueta única de identificación	Coordenadas de referencia de objeto más brillante
1		0;347.14;328.69 ;279.02;0;0;0;0; 0;0;0;0;0;0;0;0	1;0;0;0	2_1	(490,307)
2		0;285.33;428.66 ;440.26;0;0;0;0; 0;0;0;0;0;0;0;0	0;1;0;0	2_2	(454,223)
3		0;178.07;126.15 ;270.51;0;0;0;0; 0;0;0;0;0;0;0;0	0;0;1;0	2_3	(268,343)
4		0;151.86;252.37 ;204.21;0;0;0;0; 0;0;0;0;0;0;0;0	0;0;0;1	2_4	(195,210)
5		0;304.5;183.73; 43.661;0;0;0;0;0 ;0;0;0;0;0;0;0	1;1;0;0	2_5	(234,307)

	Imagen	Patrón generado por MATLAB	Salida deseada red neuronal	Etiqueta única de identificación	Coordenadas de referencia de objeto más brillante
6		0;459.18;411.62;149.1;0;0;0;0;0;0;0;0;0;0;0;0	0;0;1;1	2_6	(451,210)
7		0;419.71;352.23;467.92;0;0;0;0;0;0;0;0;0;0;0;0	1;0;1;0	2_7	(490,115)
8		0;349.6;411.6;380.48;0;0;0;0;0;0;0;0;0;0;0;0	0;1;0;1	2_8	(342,362)
9		0;43.661;154.95;144.24;0;0;0;0;0;0;0;0;0;0;0;0	1;1;1;1	2_9	(234,115)

4.3. Red neuronal artificial

A continuación vemos la Figura 38, esta se obtiene del toolbox de MATLAB y muestra la cantidad de neuronas que se emplean para cada capa de la red definidas en la sección 2.2.1.1., la capa de entrada consta de 16 neuronas en las cuales se usaran los patrones de cada imagen, para la capa oculta se aplicó la regla de la pirámide geométrica que se vio en el apartado 3.3 del capítulo 3 y esto dio como resultado 8 neuronas, pero después de realizar el entrenamiento varias veces se llegó a la conclusión que el mejor resultado se encuentra con 10 neuronas, este valor se obtiene mediante prueba y error cambiando el número de neuronas de la capa oculta, y por último se tiene la capa de salida en la cual se usaron 4 neuronas ya que como se muestra en la Tabla 1 la salida es un vector que está conformado por 4 elementos. En esta capa se normalizan los valores para obtener en la salida valores entre 0 y 1. Entre la capa de entrada y la capa oculta existen 160 conexiones y de la capa oculta a la capa de salida existen 40 conexiones.

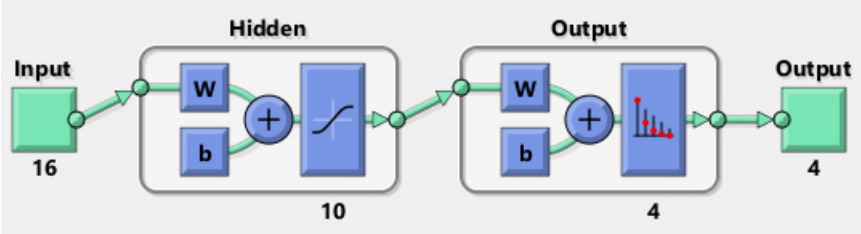


Figura 38: Cantidad de neuronas en cada capa de la red.

Se procedió a realizar el entrenamiento de la red neuronal, en la Figura 39 se pueden observar los valores obtenidos una vez finalizado el entrenamiento. El entrenamiento finalizó cuando se alcanzó el valor de 6.85×10^{-7} en el gradiente, el cual es menor al requerido para terminar el entrenamiento. Se realizaron 625 iteraciones en la red para realizar el ajuste de los pesos de las interconexiones. La opción performance nos muestra como fue el desempeño de la red para patrones de validación ingresados de manera consecutiva. Las condiciones mínimas a alcanzar para terminar el entrenamiento se muestran en el lado derecho de cada una de las barras de progreso de la Figura 39.

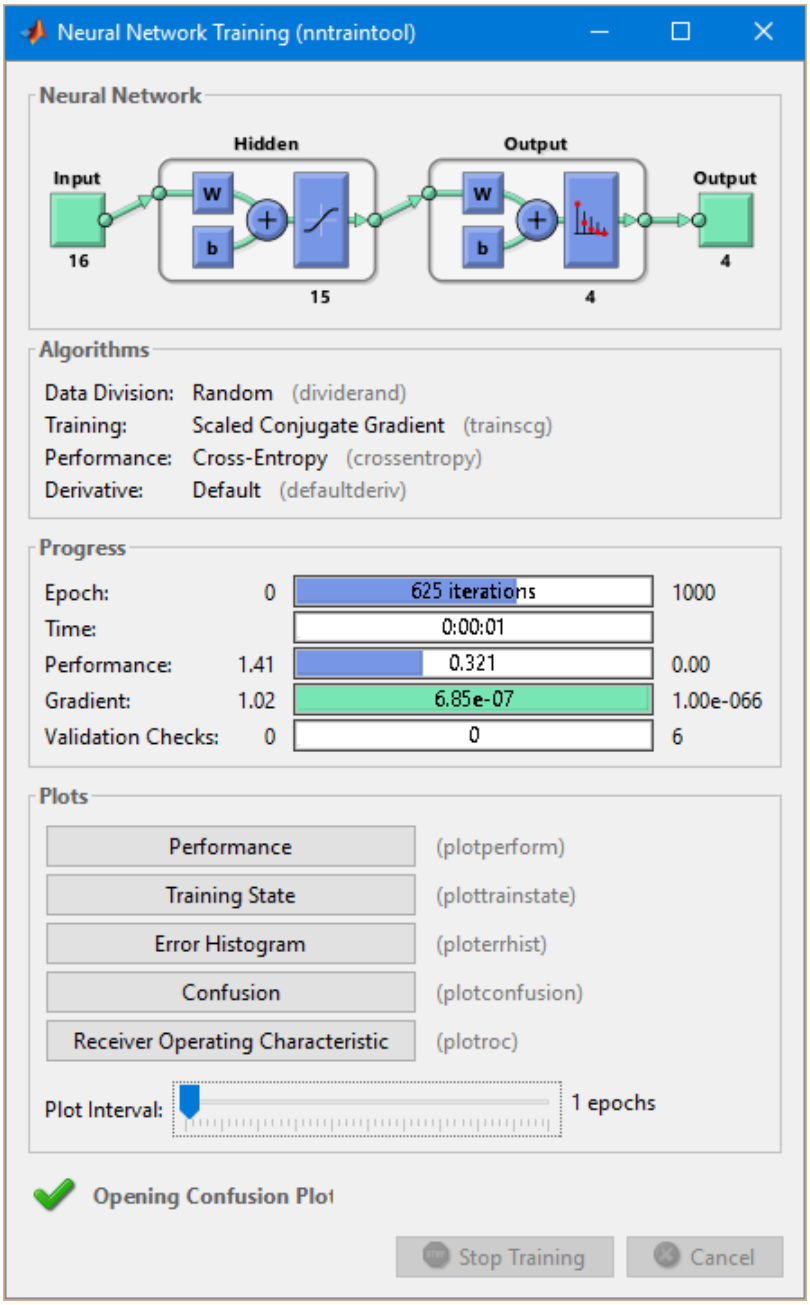


Figura 39: Entrenamiento de la red neuronal.

La ventana de entrenamiento nos da la posibilidad de graficar los resultados de la red, en la Figura 40 se puede ver la gráfica del desempeño de la red para cada iteración del entrenamiento para las muestras de entrenamiento, prueba y validación hasta la iteración 625 en la que cual detuvo el entrenamiento de la red, también se puede de ver

marcado el punto en el que se obtuvo el mejor valor posible para la validación de la red también conocido como BVP (Best Validation Performance) por sus siglas en inglés.

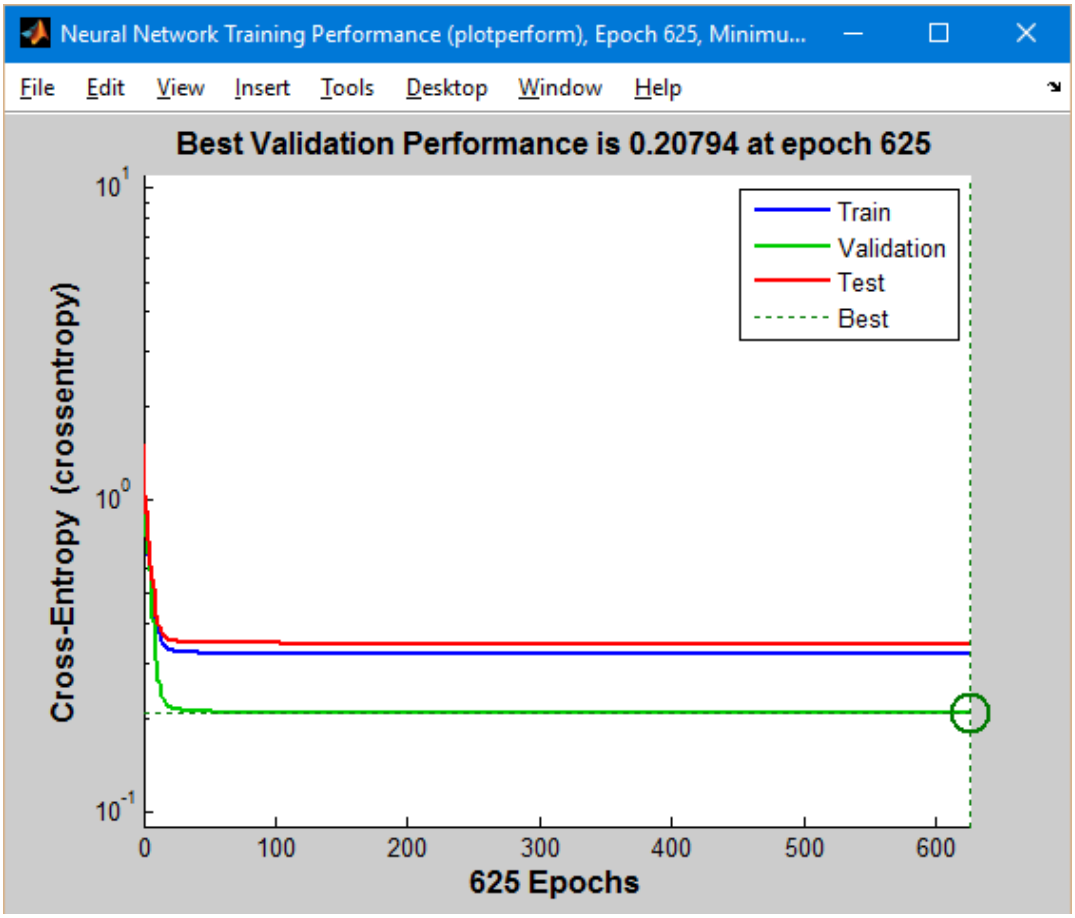


Figura 40: Desempeño de la red para cada una de las categorías de muestras del entrenamiento.

En la Figura 41 se pueden ver la gráficas de la opción training state, en la primera gráfica se observa cómo va variando el gradiente en cada una de las iteraciones hasta llegar a la iteración 625, el gradiente tuvo un valor de 6.853×10^{-7}. En la segunda gráfica de la figura se puede observar cómo se dieron cada una de las pruebas de validación, la prueba de validación busca que de manera consecutiva con 6 muestras de validación (marcadas con punto rosas en la gráfica) no exista un fallo en disminuir el BVP de la Figura 40.

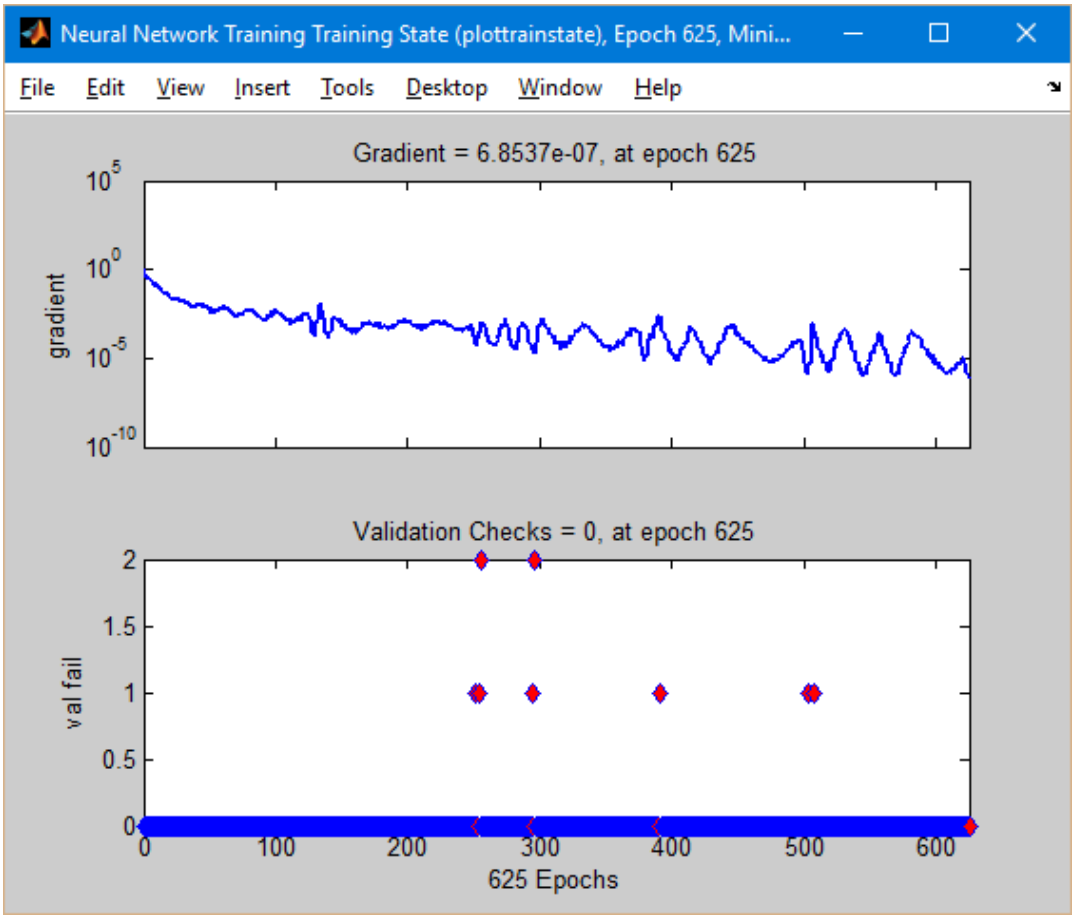


Figura 41: Variación en el gradiente en backpropagation y pruebas de validación.

En la Figura 42 se puede ver la gráfica de la opción error histogram, esta es utilizada para detectar si existen valores anómalos en el error de salida usando los datos de entrenamiento, prueba y validación. Esto significa que si no hay un buen ajuste en la red existirán errores más grandes, para este caso se ve que la mayoría se encuentra con el menor error que es de 0.01875, pero también se puede ver que existen algunos datos anómalos que hacen que el error sea de 0.5063 y 0.7313, esto significa que la mayoría de las entradas y sus valores de salida estarán muy cercanos a los definidos pero además que pueden existir casos en los que nuestra salida varía más de lo esperado o se obtenga una salida incorrecta.

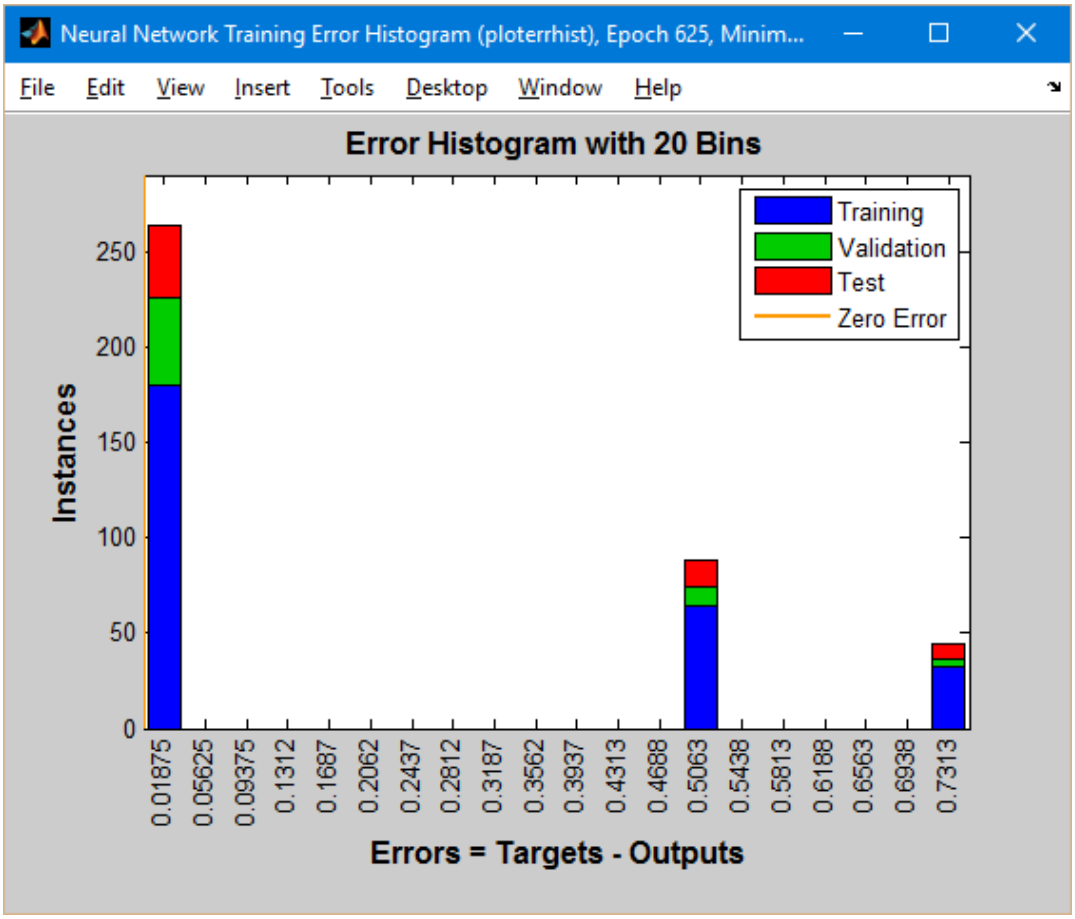


Figura 42: Histograma de error de las salidas.

En la Figura 43 se pueden ver varias gráficas las cuales se obtienen de la opción confusion, estas gráficas representan diferentes matrices de confusión, una matriz de confusión es una herramienta del área de inteligencia artificial que permite ver el desempeño de un algoritmo que hace uso del aprendizaje supervisado. Esta es utilizada para problemas de clasificación viendo el porcentaje de pertenencia a cada una de las clases o valores de salida. En la Figura 42 se pueden ver cuatro diferentes gráficas, cada una con una matriz de confusión, cada gráfica representa el porcentaje de pertenencia a la clase de cada uno de los valores asignados al entrenamiento, prueba y validación. En el eje x se representan las clases de salida establecidas o salida predicha, el número depende de la cantidad de elementos del vector de salida, y en el eje y se representa las clases de salida reales que se obtuvieron al evaluar en la red neuronal. Los elementos de color verde representan la cantidad de aciertos al realizar la clasificación correcta, estos también recibe el nombre de verdaderos positivos. Los elementos en color rojo llamados falsos positivos representan las clasificaciones incorrectas junto con su porcentaje en función a la cantidad de elementos en esa categoría, la columna gris representa los

porcentajes en verde de aciertos totales de cada una de las filas y en rojo el porcentaje de fallos totales en esa fila, la fila de color gris representa los porcentajes en verde de aciertos totales de cada una de las columnas y en rojo el porcentaje de fallos totales en esa columna. El elemento de color azul representa el total del porcentaje de fallos y aciertos que existen en esa matriz, este porcentaje se obtiene mediante una regla de tres tomando la cantidad de elementos totales, la cantidad de elementos totales en verde y la cantidad de elementos totales en rojo, el porcentaje en verde representa la precisión al realizar la clasificación, y el porcentaje en rojo representa la tasa de errores de clasificación para esa matriz de confusión.

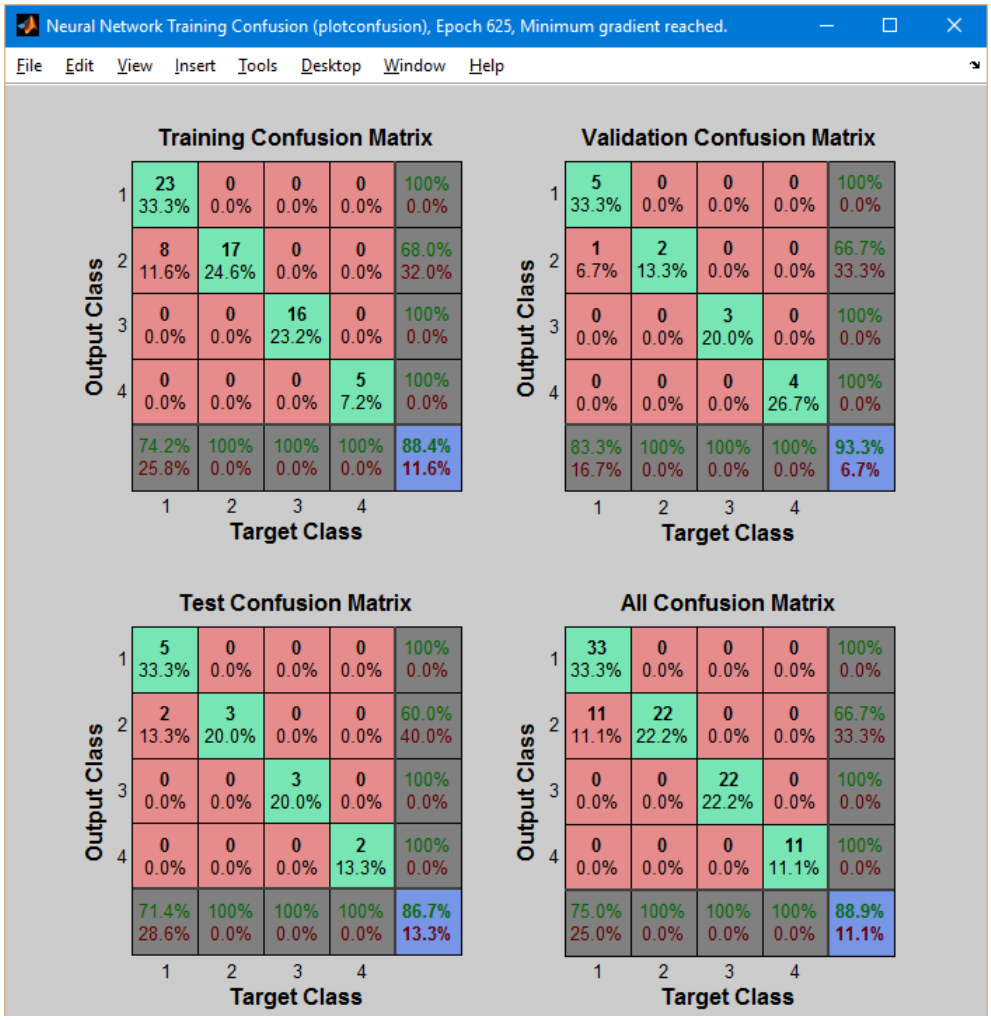


Figura 43: Matriz de confusión de clasificación.

En la primera matriz de confusión se hace uso de los patrones asignados a la parte de entrenamiento, en este caso un total de 69 elementos, se clasificaron correctamente 23 patrones en la clase 1 y fallaron 8 patrones al ser clasificados en la clase 2, se clasificaron

17 patrones correctamente en la clase 2, 16 patrones correctos en la clase 3 y 5 correctos en la clase 4. Con esos valores que se representan en el recuadro azul, se tiene un 88.4 % de éxito y 11.6 % de clasificaciones erróneas.

En la segunda matriz de confusión se hace uso de los patrones asignados a la parte de validación, en este caso de un total de 15 elementos, solo fallo un patrón al ser clasificado en la clase 2, la segunda clase, la tercer clase y cuarta clase tienen todos los aciertos y ningún fallo. Con esos valores para validación como se observa en el elemento azul de la matriz, se tiene un 93.3 % de éxito y 6.7 % de clasificaciones erróneas.

En la tercera matriz de confusión se hace uso de los patrones asignados a la parte de prueba, en este caso 15 elementos, en la primera clase se clasificaron correctamente 5 patrones y dos fallaron al ser clasificados en la clase 2, en la segunda, tercer y cuarta clase se clasificaron todos de manera correcta, y como se observa en el elemento azul esto nos da en total un 86.7 % de éxito y 13.3 % de clasificaciones erróneas.

La última matriz de confusión representa un promedio de todas las matrices de confusión anteriores, con un total de 99 elementos, como se puede ver existió un fallo al clasificar algunos patrones en la primera clase ya que existen 11 patrones clasificados erróneamente en la clase 2, y con esto, como se observa en el elemento azul se obtiene un total un 88.9 % de éxito y 11.1 % de clasificaciones erróneas.

En la Figura 44 se pueden ver varias gráficas las cuales se obtienen de la opción Receiver operating characteristic o también conocido como curva ROC, esta muestra cuatro gráficas para el entrenamiento, pruebas, validación y una para el promedio de todas las gráficas anteriores, en el eje x se tiene la tasa de falso positivo para las clasificaciones. Un falso positivo es aquel que al momento de clasificar un patrón indica que se tiene una clasificación correcta cuando no lo es, en el eje y se tiene la tasa de verdadero positivo para la clasificación, un verdadero positivo es aquel que tiene como resultado una clasificación correcta y la clasificación esperada también como correcta. Estos valores se obtienen a partir de la matriz de confusión. En la Figura 44 podemos ver las gráficas generadas por ROC, como se puede notar todas las líneas de cada una de las clases de cada una de las gráficas se encuentran sobrepuestas por lo que solo se ve un color, en una gráfica ROC el mejor método de predicción se sitúa en la coordenada (0,1) lo que representa un 100 % de sensibilidad o ningún falso negativo y un 100 % de especificidad que significa ningún falso positivo en ese punto, este punto en la gráfica también es conocido como la clasificación perfecta. La línea gris que divide cada una de las gráficas es conocida como línea de no discriminación y es en donde se representan puntos con una clasificación aleatoria, esta línea es la que ayuda a distinguir los buenos de los malos resultados, los puntos arriba de la línea son buenos resultados de clasificación mientras que los malos se encuentran debajo de esta. Para el caso de nuestras gráficas de la Figura 44 se puede ver que todos nuestros puntos se encuentran arriba de la línea gris, lo que implica que tenemos buena clasificación además de que se poseen puntos en la coordenada (0,1) y puntos muy cercanos lo que significa que se tienen clasificaciones perfectas para los patrones de entrenamiento usados para determinar la ubicación.

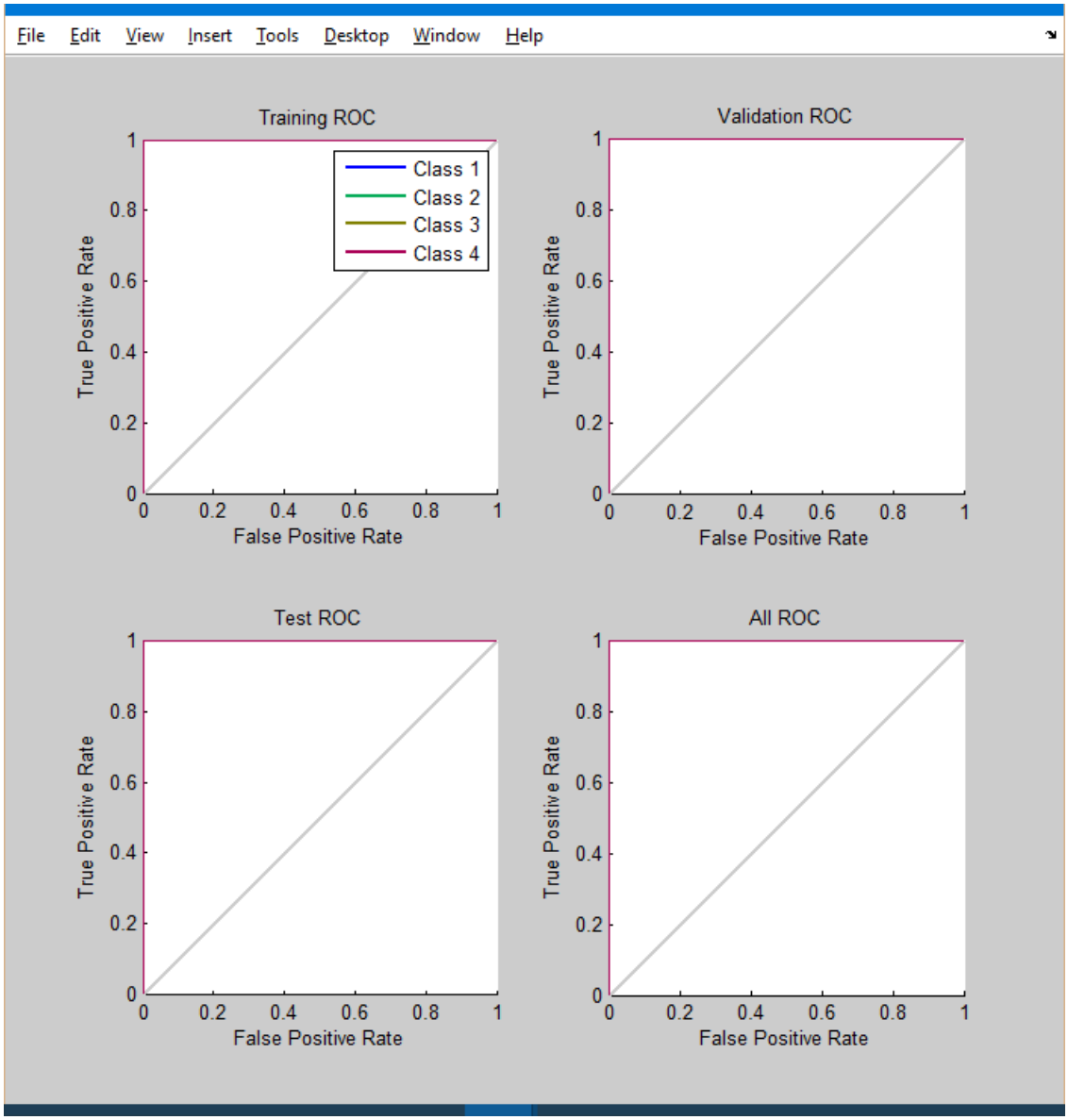


Figura 44: Curva ROC.

4.4. Resultados de pruebas

Se realizaron las pruebas en la interfaz gráfica ingresando las imágenes usadas en el entrenamiento para ver los valores obtenidos de salida en la red neuronal para ser comparados con los valores de salida establecidos anteriormente, también se observa la subsubregión detectada haciendo uso de las etiquetas definidas. Los resultados se

presentan a continuación en la Tabla 2 en el mismo orden que la Tabla 1 de la sección 4.2 con los valores de salida definidos.

Tabla 2: Resultados obtenidos de la red neuronal.

	Salida definida	Salida obtenida	Subsubregión detectada
1	1;0;0;0	1.0;0.0;0.0;0.0	2_1
2	0;1;0;0	0.0;1.0;0.0;0.0	2_2
3	0;0;1;0	0.0;0.0;1.0;0.0	2_3
4	0;0;0;1	0.0;0.0;0.0;1.0	2_4
5	1;1;0;0	0.5;0.5;0.0;0.0	2_5
6	0;0;1;1	0.0;0.0;0.5;0.5	2_6
7	1;0;1;0	0.5;0.0;0.5;0.0	2_7
8	0;1;0;1	0.0;0.5;0.0;0.5	2_8
9	1;1;1;1	0.25;0.25;0.25;0.25	2_9

Como se puede observar, comparando la columna de salida definida con la salida obtenida se obtienen valores menores, como se mencionó en el apartado 2.2.1.3 del capítulo 2 la suma de la salida obtenida debe ser 1, los valores definidos para la salida tuvieron que ser establecidos haciendo uso de 1 y 0 ya que son los únicos valores que pueden ser utilizados en una red neuronal para clasificación, en caso de tener varios unos en una salida definida se procede a distribuir la salida para cada uno definido de la siguiente manera:

$$S = \frac{1}{n}\,, \tag{11}$$

En donde S es la salida que se puede obtener para un uno definido, el valor n es el número de elementos 1 que existen en el vector de la salida definida, por ejemplo si se tiene la salida deseada 1;1;1;0 nuestra máxima salida posible es 0.33;0.33;0.33;0 ya que se tiene que:

$$S = \frac{1}{n} = 0.333\,, \tag{12}$$

Con los resultados anteriores se puede ver que el entrenamiento fue exitoso para los casos que fueron utilizados para entrenar, ahora se procede a realizar pruebas detectando una subsubregión pero desplazando la fotografía, para este caso se utilizó la subsubregión 6 o 2_6 como se puede ver en la Figura 45, las coordenadas en píxeles de su objeto más brillante son: (451, 210) y su desplazamiento en píxeles con respecto al punto de referencia: (195.5, 174).

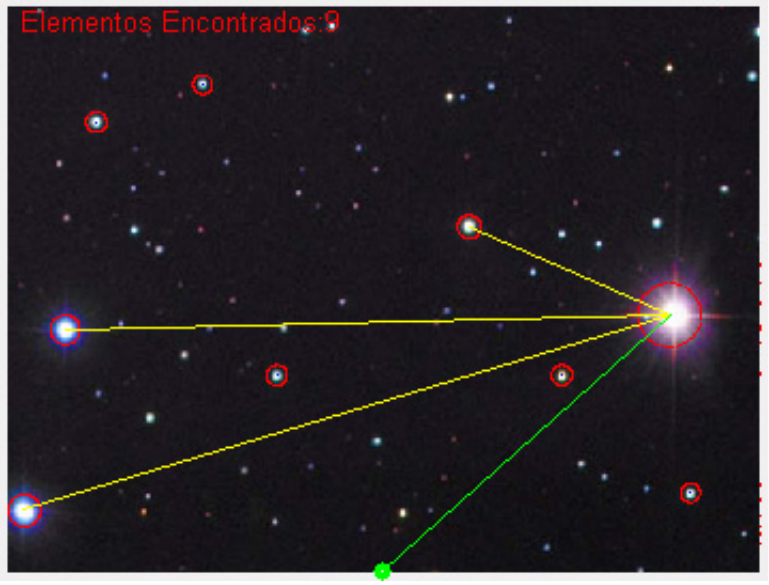
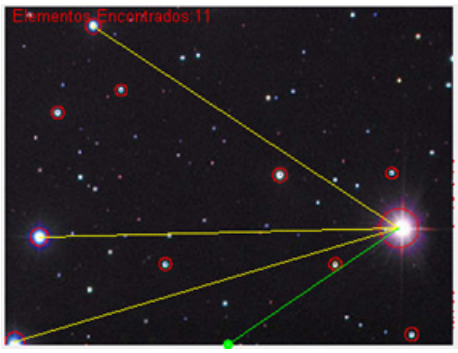
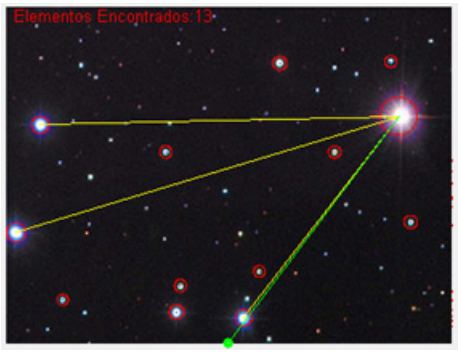
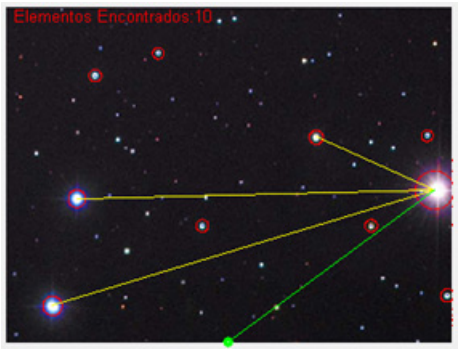
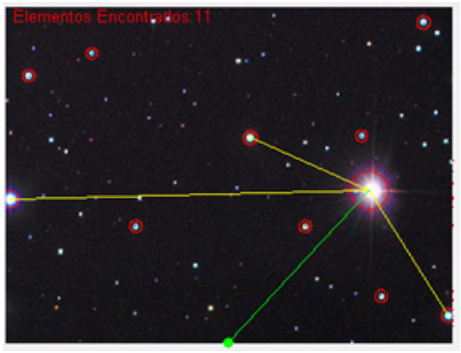


Figura 45: Imagen de prueba.

Para realizar las pruebas la imagen fue desplazada hacia arriba, abajo, derecha, izquierda y hacia arriba a la izquierda para ver el desplazamiento de la imagen y del objeto más brillante y con esto obtener más datos sobre la ubicación además de la subsubregión en la que se encuentra la fotografía. Los resultados para estas pruebas se presentan a continuación en la Tabla 3.

Tabla 3: Resultados del sistema completo.

	**Dirección:** Arriba **Salida red:** 5.6e-6;2.48e-5;0.59;0.40 **Subsubregión detectada:** 2_6 **Coordenadas originales:** (451,210) **Nuevas Coordenadas:** (451,252) **Desplazamiento:** (0,+42) **Coordenadas elemento brillante con respecto a referencia:** (195,132) **Tiempo de procesamiento:** 13 segundos
	**Dirección:** Abajo **Salida red:** 3.75e-6;3.87e-6;0.70;0.29 **Subsubregión detectada:** 2_6 **Coordenadas originales:** (451,210) **Nuevas Coordenadas:** (451,125) **Desplazamiento:** (0,-85) **Coordenadas elemento brillante con respecto a referencia:** (195.5,259) **Tiempo de procesamiento:** 15 segundos
	**Dirección:** Izquierda **Salida red:** 1.1e-7;4.12e-8;0.509;0.49 **Subsubregión detectada:** 2_6 **Coordenadas originales:** (451,210) **Nuevas Coordenadas:** (493,210) **Desplazamiento:** (-41.5,0) **Coordenadas elemento brillante con respecto a referencia:** (237,174) **Tiempo de procesamiento:** 14 segundos

	**Dirección:** Derecha **Salida red:** 0.99;4.09e-4;1.4e-5;2.16e-9 **Subsubregión detectada:** 2_1 **Coordenadas originales:** (451,210) **Nuevas Coordenadas:** (417.5,210) **Desplazamiento:** (34,0) **Coordenadas elemento brillante con respecto a referencia:** (161.5,174) **Tiempo de procesamiento:** 13 segundos
	**Dirección:** Arriba-Izquierda **Salida red:** 5.2e-6;2.39e-5;0.6162;0.383 **Subsubregión detectada:** 2_6 **Coordenadas originales:** (451,210) **Nuevas Coordenadas:** (494.5,239) **Desplazamiento:** (-43,28) **Coordenadas elemento brillante con respecto a referencia:** (238.5,146) **Tiempo de procesamiento:** 15 segundos

Como se puede ver en las pruebas en la dirección hacia arriba, abajo, izquierda y arriba-izquierda funcionaron correctamente al hacer la detección de la subsubregión al tener las salidas más grandes en los últimos dos elementos de salida ya que la subsubregión 2_6 tiene como salida esperada 0;0;1;1 y obtenida 0;0;0.5;0.5 nos indica que los valores de salida más grande efectivamente se encuentran en los últimos dos elementos. Como se puede observar en el desplazamiento, los valores positivos en x indican un desplazamiento a la derecha, los negativos desplazamiento a la izquierda, los valores positivos en y indican un desplazamiento hacia arriba y los valores negativos un desplazamiento hacia abajo para una misma subsubregión. La subsubregión detectada en cada una de las pruebas indica la ubicación aproximada con respecto a la constelación de referencia con la cual fue entrenada la red neuronal.

Para el caso del desplazamiento a la derecha fracaso al determinar la ubicación al indicar la subsubregión 2_1, esto porque la salida fue 0.99;4.09e-4;1.4e-5;2.16e-9 y el valor más grande se encuentra en el primer elemento en lugar de estar distribuido en los dos últimos valores de la salida, este problema en la detección se debe, como se

vio en las matrices de confusión, que aunque el entrenamiento tuvo buenos resultados existen algunos valores no fueron clasificados correctamente por lo tanto la red no pudo generalizar correctamente este caso y se obtuvo un resultado diferente al esperado. En cuanto al desplazamiento este es correcto ya que se continúa siguiendo el objeto más brillante, aunque al final este dato no es de utilidad si no se puede detectar la subsubregión en la que uno se ubica.

Con esto se puede ver que la red neuronal tiene un buen rango de detección para determinar la ubicación de una subsubregión, para la prueba realizada se detectaron correctamente 4 de 5 distintas direcciones de una subsubregión, en otras palabras un 80 % de acierto aunque en algunos casos puede tender a fallos al no poder generalizar correctamente, ya que como se observó en la matriz de confusión algunos valores no pudieron ser clasificados correctamente así que este fallo también puede llegar a presentarse en otras subsubregiones.

Los resultados obtenidos se diferencian a los resultados obtenidos con respecto a otros sistemas existentes que tienen un objetivo similar, un claro ejemplo es un sitio web llamado Astrometry el cual posee un sistema muy desarrollado basado en web y software para Linux, este sistema después de ingresar una imagen, como resultado muestra la imagen resaltando los elementos detectados y una etiqueta del elemento detectado la cual se obtiene de una base de datos de la NASA [29-30].

Conclusiones

En este trabajo de tesis se propuso un método para ubicar un minisatélite basado en la comparación de datos obtenidos de un conjunto de fotografías. El método que se utilizó es diferente a otros métodos existentes ya que este hace uso de reconocimiento de patrones basado en redes neuronales.

Con el método desarrollado se obtuvo un 80 % de éxito en función de la comparación de los datos almacenados y así determinar la ubicación aproximada usando una fotografía "tomada" por el minisatélite en un tiempo de detección aproximado de 14 segundos recalcando que para este trabajo se usó una base de datos muy reducida por lo que la detección es mas rapida.

Para este trabajo se propuso un método con el cual podemos obtener información única para cada imagen midiendo la distancia entre los objetos brillantes presentes, mientras más distancias se midan en una fotografía y se generen más patrones en función de estas se pueden mejorar los resultados para determinar la ubicación.

En comparación con otros sistemas similares existentes se tiene que estos son exclusivos para detección de estrellas mediante el uso de bases de datos de la Unión Astronómica Internacional y la NASA las cuales contienen datos astronómicos, estos métodos son eficientes ya que pueden determinar elementos individuales (estrellas, clusters, etc), su principal desventaja es que la detección puede tardar entre 10 a 15 minutos [58].

Para el método propuesto, si se desearan mejorar los resultados, sería necesario generar más patrones con valores de distancia entre el elemento más brillante y los elementos circundantes (más de 3 elementos los cuales son usados actualmente) ya que al tener más distancias posibles entre estos elementos podemos caracterizar mejor las subsubregiones y aumentar la capacidad de detección en la fotografía. Además con esto sería necesario generar una cantidad mayor de patrones de entrenamiento lo que se traduce en un entrenamiento, detección y determinación de la ubicación más tardado además de que con el software MATLAB se tiene un limite el cual no permite tener una cantidad mayor a 1,000 neuronas en la capa oculta, con lo cual, haciendo uso de la

regla de la pirámide geométricas nos deja ver que tenemos un número limitado posible de neuronas de entrada y de salida que pueden ser utilizadas.

Para trabajo a futuro este sistema puede ser desarrollado para ser usado para determinar la ubicación de sondas espaciales u otros elementos que deseen ser ubicados dentro del sistema solar, ya que no existe variación en cómo se ven los elementos vistos desde una fotografía, sea tomada en una órbita terrestre, en órbita de otro planeta o en espacio interplanetario.

Bibliografía

[1] "Marcaban el rumbo el mar, el cielo y el viento — BIBLIOTECA EN LÍNEA Watchtower" Disponible en: http://wol.jw.org/es/wol/d/r4/lp-s/102003613#h=11. [Último acceso: 2015].

[2] "Unknown. (2006) Ancient Astronomers.". Disponible en: http://burro.astr.cwru.edu/stu/pre20th_ancients.html [Último acceso: 2015].

[3] "Steering by the stars" Disponible en: http://folwell.mpls.k12.mn.us/uploads/steering_by_the_stars.pdf [Último acceso: 2015].

[4] "The History of the Sextant" Disponible en: http://www.mat.uc.pt/ helios/Mestre/Novemb00/H61iflan.htm [Último acceso: 2015].

[5] "The History of Fingerprints,". Disponible en: http://onin.com/fp/fphistory.html. [Último acceso: 2015].

[6] N. Yalabık, "Medical Applications of Pattern Recognition ," April-2010. Disponible en: http://ocw.metu.edu.tr/pluginfile.php/5201/mod_resource/content/0/hibit10_presentation.pdf. [Último acceso: 2015].

[7] "GEOGRAFÍA - FÍSICA: Orientación y localización geográfica - 1ª parte," Disponible en: http://www.natureduca.com/geog_fisica_orienta1.php. [Último acceso: 2015].

[8] "Qué es el GPS y como funciona," Disponible en: http://www.informatica-hoy.com.ar/aprender-informatica/que-es-el-gps-y-como-funciona.php. [Último acceso: 2015].

[9] F. Y. Shih, Image processing and pattern recognition fundamentals and techniques. Piscataway, NJ: IEEE Press ; 2010.

[10] "Using the test set for training - Pattern Recognition Tools," Disponible en: http://www.37steps.com/4967/testset-for-training/. [Último acceso: 2015].

[11] E. Alpaydin, Introduction to machine learning, 2nd ed. Cambridge, Mass.: MIT Press, 2010.

[12] A. Pal and D. K. Dutta Majumder, "Approaches to supervised learning for pattern recognition," Jan-1994. Disponible en: http://www.dli.gov.in/rawdataupload/upload/insa/insa_2/20005a1c_1.pdf [Último acceso: 2015].

[13] B. Settles, "Active Learning Literature Survey," 26-January-2010. Disponible en: http://burrsettles.com/pub/settles.activelearning.pdf. [Último acceso: 2015].

[14] O. Chapelle, Semi-supervised learning. Cambridge, Mass.: MIT Press, 2006.

[15] K. R. Castleman, Digital image processing. Englewood Cliffs, N.J.: Prentice-Hall, 1979

[16] "¿QUÉ ES UNA IMAGEN?" Disponible en: http://narceaeduplastica.weebly.com/iquestqueacute-es-una-imagen.html. [Último acceso: 2015].

[17] "Imagen digital: conceptos básicos," Disponible en: http://platea.pntic.mec.es/ lgonzale/tic/imagen/conceptos.html. [Último acceso: 2015].

[18] "Tema 2. Preproceso (realzado y filtrado) de imágenes digitales." Disponible en: http://www.escet.urjc.es/ visiona/tema2.pdf. [Último acceso: 2015].

[19] "Image Processing Toolbox," Disponible en: http://www.mathworks.com/products/image/. [Último acceso: 2015].

[20] P. Cruz and A. Herrera, Inteligencia artificial con aplicaciones a la ingeniería. México: Alfaomega, 2010.

[21] F. Izaurieta and C. Saavedra, "Redes Neuronales Artificiales." Disponible en: http://www.uta.cl/charlas/volumen16/indice/ch-csaavedra.pdf. [Último acceso: 2015].

[22] J. lez and V. J. Hernando, Redes neuronales artificiales: fundamentos, modelos y aplicaciones. Wilmington: Addison-Wesley Iberoamericana, 1995.

[23] F. Casacuberta Nolla and E. Vidal Ruiz, "Aprendizaje Automático," Dec-2010. Disponible en: http://users.dsic.upv.es/ fcn/students/ml/t5aa2p.pdf. [Último acceso: 2015].

[24] "Neural Network Toolbox" Disponible en: http://www.mathworks.com/products/neural-network/. [Último acceso: 2015].

[25] "MATLAB GUI" Disponible en: http://www.mathworks.com/discovery/matlab-gui.html. [Último acceso: 2015].

[26] "Scaled conjugate gradient backpropagation" Disponible en: http://www.mathworks.com/help/nnet/ref/trainscg.html. [Último acceso: 2015].

[27] "Neural networks and deep learning." Disponible en: http://neuralnetworksanddeeplearning.com/chap3.html#introducing_the_cross-entropy_cost_function. [Último acceso: 2015].

[28] "Books by Robert Gendler," Disponible en: http://www.robgendlerastropics.com/. [Último acceso: 2015].

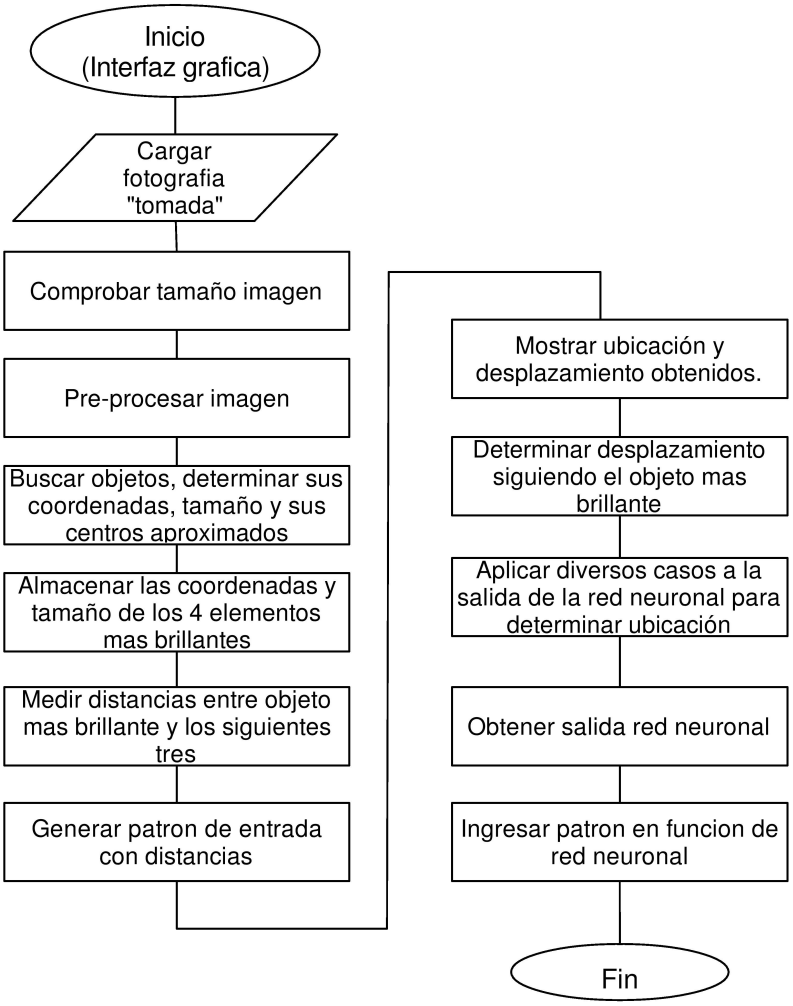
[29] "Astrometry.net" Disponible en: http://astrometry.net/ [Último acceso: 2015].

[30] "The FITS Support Office" Disponible en: http://fits.gsfc.nasa.gov/ [Último acceso: 2015].

Apéndice A

Códigos

A.1. Diagrama de flujo del sistema



A.2. Código interfaz gráfica

```
%Esta es la funcion que se lleva a cabo cuando se da click
%en el boton de identificar

function pushbutton2_Callback(hObject, eventdata, handles)
global File_Name Path_Name;

%http://www.mathworks.com/help/matlab/ref/imread.html
IM1 = imread([Path_Name, File_Name]); % Carga la imagen y la almacena en RGB

% http://www.mathworks.com/help/matlab/ref/imfinfo.html
info = imfinfo([Path_Name, File_Name]); %Obtener datos de la imagen
w = info.Width; % Obtener Largo de imagen
h = info.Height; % Obtener Ancho de imagen

%Aqui se ajusta la resolucion de la imagen para que sea a correcta
if w>512 && h>384
Imagen1 = imresize(IM1,[384 512]);
else
if w==512 && h==384
Imagen1 = IM1;
else
if w<512 && h<384
Imagen1 = imresize(IM1,[384 512]);
end
end
end

%-------------------------------------------------

IM2 = rgb2gray(Imagen1); %Se transforma la imagen de rgb a escala de grises

IM3 = imcomplement(IM2); %Se obtiene el incomplemento de la imagen para invertir
```

```
%los colores pero manteniendo la escala de grises.

IM4 = im2bw(IM3, 0.3); %Se tranforma a imagen a blanco y negro el valor 0.3
%es el umbral para la conversion mientras mayor es el numero mayor
%es la deteccion y transformacion de grises.

IM5 = imcomplement(IM4); %Se vuelve a invertir los colores,
%asi los elementos brillantes quedan de color blanco y lo demas de color negro.

RRR = bwareaopen(IM5, 15); %Eliminamos los puntos blancos menores o iguales
%a 15 pixeles para reducir la cantidad de elementos y aumentar
%la velocidad de operacion del programa
IM6 = bwboundaries(RRR);

axes(handles.axes2);

imshow(Imagen1) %Se muestra la imagen original en la interfaz

%Se muestran la cantidad de elementos encontrados en la imagen
text(10,10,strcat('\color{red}Elementos Encontrados:',num2str(length(IM6))))
hold on
title(File_Name)
hold on

for z = 1:length(IM6)  % Calculamos los centros aproximados de cada estrella
% y los almacenamos ademas de calcular un radio aproximado para marcar los
%objetos en la imagen y el radio representa el grado de brillo
ZZ{z}= cell2mat(IM6(z));
miny(z)=min(ZZ{1,z}(:,1));
maxy(z)=max(ZZ{1,z}(:,1));
promy = ((( maxy+miny)/2));
minx(z)=min(ZZ{1,z}(:,2));
maxx(z)=max(ZZ{1,z}(:,2));
promx = ((( maxx+minx)/2));
radio=((maxx)-promx);
end
```

```
rrvar=0;
for rr=1:length(radio)
radiomod(rr)=(radio(rr)+rrvar);
rrvar=rrvar+0.01;
end


for td=1:length(IM6)  %Graficamos cada una de los elementos encontrados
%con un circulo con su radio unico
for td2=1:79   %Disminuir para aumentar velocidad
ang=0:0.08:2*pi;
xp=(radiomod(td)+5)*cos(ang);
yp=(radiomod(td)+5)*sin(ang);
plot(promx(td)+xp,promy(td)+yp,'.r', 'markersize', 3);
end

hold on
end
hold on

for td_0=1:length(IM6)
plot(promx,promy,'.b', 'markersize', 4)
end

hold off

%-------------------------------------------------------------------------
%-------------------------------------------------------------------------
%En esta parte se crea una matriz en donde se almacenan las coordenadas y
%radio de cada elemento encontrado para manejar mas facil los datos

ca1=1;
for kk=1:length(IM6)
if (promx(1,kk)>=(0) && promx(1,kk)<=(512-1))
```

```
if (promy(1,kk)>=(0) && promy(1,kk)<=(384-1))
AREA1(ca1,1)= promx(1,kk);
AREA1(ca1,2)= promy(1,kk);
AREA1(ca1,3)= radiomod(kk);
ca1=ca1+1;
end
end
end

%Se acomodan los datos de a matriz de menor a mayor radio
AREA1_1 = sortrows(AREA1,3);
[m,n] = size(AREA1_1);

%Se almacenan las coordenadas y radios de los 4 elementos mas brillantes
%representados por el radio mas grande
max_x=AREA1_1(m,1);
max_y=AREA1_1(m,2);
rad_x=AREA1_1(m,3);
max_x1=AREA1_1(m-1,1);
max_y1=AREA1_1(m-1,2);
rad_x1=AREA1_1(m-1,3);
max_x2=AREA1_1(m-2,1);
max_y2=AREA1_1(m-2,2);
rad_x2=AREA1_1(m-2,3);
max_x3=AREA1_1(m-3,1);
max_y3=AREA1_1(m-3,2);
rad_x3=AREA1_1(m-3,3);

%Se crea una matriz en donde se almacenan los datos de los puntos anteriores
puntos = [max_x max_y rad_x;max_x1 max_y1 rad_x1;max_x2 max_y2 rad_x2;max_x3 max_y3 rad_x

%Calculamos las distancias entre el objeto mas brillante y los otros tres puntos
%y se almacenan en una matriz usando la matriz anterior
for tx=1:4
DistA1(1,tx)=abs(sqrt( (( puntos(tx,1)- max_x )^2) + (( puntos(tx,2)- max_y )^2)
));
```

```
x4=[max_x,puntos(tx,1)];
y4=[max_y,puntos(tx,2)];
hold on
plot(x4,y4,'Color','y') ;
end

for var1=5:16
DistA1(var1)=0;
end

%-------------------------
%Se convierte vector columna a vector fila la matriz con las distancias
IMP2 = DistA1';

%Mostrar las distancias en la interfaz grafica
set(handles.edit5, 'String', IMP2(1,1));
set(handles.edit6, 'String', IMP2(2,1));
set(handles.edit7, 'String', IMP2(3,1));
set(handles.edit8, 'String', IMP2(4,1));

%Se evalua el vector fila en la funcion de la red neuronal
NOUT =  red15(IMP2);

NOUT2=NOUT;

[fila,colum]=max(NOUT);

for a=1:4

[fil(a),col(a)]=max(NOUT2);
NOUT2(col(a))=0;
end


%------------Determinar desplazamiento-----------------
%En esta parte se calcula la distancia del objeto mas brillante al
```

```
%centro de el sistema de referencia establecido para calcular el
%desplazamiento

%if (NOUT(1,1)>=0.9)

plot(256,384,'green.-', 'markersize', 20); %punto (0,0) en imagen
puntos1=sortrows(puntos,3);
[m,n] = size(puntos1);
xpuntos1=puntos1(m,1);
ypuntos1=puntos1(m,2);
distcen=sqrt( (( xpuntos1- 256 )^2) + (( ypuntos1- 384 )^2)  );
axes(handles.axes2);
hold on

xc=[xpuntos1,256];
yc=[ypuntos1,384];
hold on

plot(xc,yc,'Color','g') ;
set(handles.edit10, 'String', distcen);
hold on

xs=256;
ys=384;
plot(xs,ys,'.r', 'markersize', 3, 'Color','w') ;

%Desplazamiento en px
despx=(xpuntos1-256);
despy=(384-ypuntos1);

%Mostrar desplazamiento en interfaz
set(handles.edit20, 'String', despx);
set(handles.edit21, 'String', despy);

%Coord en px elemento mas brillante
```

```
set(handles.edit15, 'String', xpuntos1);
set(handles.edit16, 'String', ypuntos1);

%end

%-----------------------------------
%En esta parte se coloca una etiqueda en la interfaz grafica
%para indicar la ubicacion en funcion a la salida de la red neuronal

if (NOUT(1,1)>=0.9)
set(handles.edit17, 'String', '2_1');
end
%-----------------------------------
if (NOUT(2,1)>=0.9)
set(handles.edit17, 'String', '2_2');
end
%-------------------
if (NOUT(3,1)>=0.9)
set(handles.edit17, 'String', '2_3');
end
%-------------------
if (NOUT(4,1)>=0.9)
set(handles.edit17, 'String', '2_4');
end
%-------------------
if (NOUT(1,1)>=0.35) && (NOUT(2,1)>=0.35)
set(handles.edit17, 'String', '2_5');
end
%-------------------
if (NOUT(3,1)>=0.35) && (NOUT(4,1)>=0.35)
set(handles.edit17, 'String', '2_6');
end
%-------------------
if (NOUT(1,1)>=0.35) && (NOUT(3,1)>=0.35)
set(handles.edit17, 'String', '2_7');
end
```

```
%--------------------
if (NOUT(2,1)>=0.35) && (NOUT(4,1)>=0.35)
set(handles.edit17, 'String', '2_8');
end
%--------------------
if (NOUT(1,1)>=0.2) && (NOUT(2,1)>=0.2) && (NOUT(3,1)>=0.2) && (NOUT(4,1)>=0.2)
set(handles.edit17, 'String', '2_9');
end
%--------------------
set(handles.edit11, 'String', col(1));
set(handles.edit12, 'String', col(2));
set(handles.edit13, 'String', col(3));
set(handles.edit14, 'String', col(4));

set(handles.edit1, 'String', NOUT(1,1));
set(handles.edit2, 'String', NOUT(2,1));
set(handles.edit3, 'String', NOUT(3,1));
set(handles.edit4, 'String', NOUT(4,1));

clear
clc
```

A.3. Función red neuronal

```
%Funcion la cual contiene la red neuronal generada
%por el toolbox de MATLAB, este archivo contiene todos los pesos obtenidos
%para la red neuronal ademas de las funciones de activacion y normalizacion de
%la salida de la red.

function [y1] = myNeuralNetworkFunction(x1)
%MYNEURALNETWORKFUNCTION neural network simulation function.
%
% Generated by Neural Network Toolbox function genFunction, 21-Dec-2015 11:23:04.
%
% [y1] = myNeuralNetworkFunction(x1) takes these arguments:
```

```
%   x = 16xQ matrix, input #1
% and returns:
%   y = 4xQ matrix, output #1
% where Q is the number of samples.

%#ok<*RPMT0>

% ===== NEURAL NETWORK CONSTANTS =====

% Input 1
x1_step1_remove = [1 5 6 7 8 9 10 11 12 13 14 15 16];
x1_step1_keep = [2 3 4];
x1_step2_xoffset = [43.661;126.15;43.661];
x1_step2_gain = [0.0048132576368349;0.0066113516908532;0.0047141015276046];
x1_step2_ymin = -1;

% Layer 1
b1 = [3.1866407891210899;-2.5706644760212307;-3.375065187307249;-0.92484582446781538;
-0.76509006145430369;-0.24119730396075881;2.9720884557107436;-0.53555920171135574;
-2.3774918822969706;3.1598610852253382];
IW1_1 = [-0.4018672199933071 2.1137527341109479 3.8999627434097799;
2.3816231308636819 2.25844639521635 -1.3174952095698007;
2.5865766104824863 2.7749611369797522 1.1396813207745593;
2.9311463077513107 -3.3824096176304241 -2.2082614430830421;
-4.0943613335247377 -1.5775334239410346 0.69901376464923848;
-2.1156097841983623 -0.86186191726091632 -2.3966898861621453;
1.3264222890901056 -1.2576472018615699 1.4813252906404617;
-3.8978704934162201 4.989771942637125 -0.49083468513923711;
-3.0054542679695202 3.3750690092428104 1.4592980991060043;
1.4142656278388037 -1.3990330824981534 2.2170129321242582];

% Layer 2
b2 = [-1.1115840757393713;2.1349535401134969;0.22167171355448481;0.55926104538114974];
LW2_1 = [-1.6256739361019112 -2.0437212616225517 -2.9220017980560287 -2.2009598087900311
-3.1975489122888847 -2.3337797505238442 -1.2814971791641847
-4.06334597679336 -1.4191025513501998
```

```
−2.3841344524293953; −4.2187570948440314 −1.6921053588331219
0.30806964471911347 0.5918399129437959
−2.7827081017822342 0.85351471590566097
1.5467604846401377 2.6776190398438331 5.8197462854349427 1.1735955779846259;
3.3804740498132353 1.3802264238872297
3.5298120719026782 2.7226264860115634 2.5008938019054172
−0.98994433452247255 0.0093258710565233809 −4.3527166486567532
−1.2152112930844658 −1.3103885144662333;
2.080783889670045 3.1870503805322188 0.019247636976391688
−1.654942219570783 1.5297398100425637
0.64971553200940591
−0.21432812007419996 4.7879657718152968 −1.3465440120083261 −0.27882963188458054];

% ===== SIMULATION ========

% Dimensions
Q = size(x1,2); % samples

% Input 1
xp1 = removeconstantrows_apply(x1,x1_step1_keep,x1_step1_remove);
xp1 = mapminmax_apply(xp1,x1_step2_gain,x1_step2_xoffset,x1_step2_ymin);

% Layer 1
a1 = tansig_apply(repmat(b1,1,Q) + IW1_1*xp1);

% Layer 2
a2 = softmax_apply(repmat(b2,1,Q) + LW2_1*a1);

% Output 1
y1 = a2;
end

% ===== MODULE FUNCTIONS ========

% Map Minimum and Maximum Input Processing Function
function y = mapminmax_apply(x,settings_gain,settings_xoffset,settings_ymin)
```

```
y = bsxfun(@minus,x,settings_xoffset);
y = bsxfun(@times,y,settings_gain);
y = bsxfun(@plus,y,settings_ymin);
end

% Remove Constants Input Processing Function
function y = removeconstantrows_apply(x,settings_keep,settings_remove)
if isempty(settings_remove)
y = x;
else
y = x(settings_keep,:);
end
end

% Competitive Soft Transfer Function
function a = softmax_apply(n)
nmax = max(n,[],1);
n = bsxfun(@minus,n,nmax);
numer = exp(n);
denom = sum(numer,1);
denom(denom == 0) = 1;
a = bsxfun(@rdivide,numer,denom);
end

% Sigmoid Symmetric Transfer Function
function a = tansig_apply(n)
a = 2 ./ (1 + exp(-2*n)) - 1;
end
```