UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Ingeniería

Diseño de una red neuronal artificial para la identificación de dientes de ajo y su aplicación en el proceso automatizado de siembra

TESIS

Que para obtener el título de

Ingeniero Mecatrónico

P R E S E N T A

Francisco Muñoz Bustos

DIRECTOR DE TESIS

Ing. Juan Manuel Gómez González

Ciudad Universitaria, Cd. Mx., 2017

Agradecimientos

Quiero agradecer principalmente a mis padres Francisco y Marisela a mis hermanos Carolina, Mariana y David, a mis sobrinos Héctor Emiliano, Bruno Alejandro y Mena Muñoz la pequeña de la casa, por el apoyo brindado a lo largo de la carrera y sobretodo en el trayecto de la creación de este documento, fueron valiosísimos sus palabras de aliento, las observaciones y las sonrisas otorgadas, en los buenos y malos momentos.

Este año ha sido algo tortuoso por todas las cosas que han pasado, pero con el apoyo de la familia todo se puede, siempre hay una luz al final del túnel.

Agradecer a mis amigos Paul, Anita, Ruth, Diego, Lalito, Mauricio, Yiss, Carlos, Kevin, Marilú, Juanito, Licha, Susy y Vane, ellos al igual que mi familia me apoyaron en todo momento.

A mis profesores que van a lo largo de mi trayectoria escolar, que me han enseñado que el estudio es un estilo de vida y que, con esfuerzo y trabajo duro cualquier objetivo, por más difícil que éste sea es posible.

Al ingeniero Juan Manuel, por aguantarme como tesista, apoyarme a lo largo de los últimos semestres de la carrera, por sus observaciones y correcciones de este trabajo de grado.

Al doctor Jesús Manuel Álvarez López director suplente, por sus sabias palabras y observaciones las cuales hicieron mejorar este documento.

A mi alma mater la UNAM por ser, además de una de las mejores universidades en el mundo un espacio donde cosas maravillosas ocurren.

En fin, muchas gracias a todas las personas que de una u otra manera me han dejado algo de enseñanza en este sueño loco llamado vida.

In memoriam,
Epifanía Gómez Gómez
Karina Partida Herrera

Índice General

Índice de Figuras

Índice de tablas

Capítulo 1. Introducción

1.1 Planteamiento del problema

Identificar por medio de visión artificial el ápice de un diente de ajo y orientarlo con el fin de automatizar su siembra en bandejas, utilizando una máquina estacionaria.

1.2 Alcances

Los alcances del siguiente tema son el diseño de la Red Neuronal Artificial para la identificación del ápice de dientes de ajo, probar la eficacia de la misma y del sistema de orientación e implementarla en la máquina estacionaria, cabe aclarar que el diseño mecánico del sistema de orientación ya estaba propuesto por otro ingeniero que igual está trabajando en el proyecto.

1.3 Justificación del tema

El ajo (Allium sativum L.), al igual que la cebolla (Allium cepa L.), es una planta que tuvo su origen en el Asia central, sus propiedades terapéuticas y usos se conocen desde hace más de 3000 años, aunque algunos autores remiten su uso desde 4000 años antes de nuestra era. En la actualidad es un producto valorado por su sabor y su uso en la preparación de una infinidad de platillos, mientras que la investigación continúa para descubrir y afinar su utilidad con fines medicinales.

Después de la cebolla, de la familia botánica de las Alliaceae, el ajo es el segundo producto más importante por su uso en la alimentación. El principal uso del ajo es como saborizante o condimento en la cocina para preparar diversos platillos alrededor de del mundo, su principal consumo es en fresco al utilizar los dientes o bulbillos, además se usa deshidratado o procesado de diversas maneras.

A través de la historia de la humanidad el ajo ha sido motivo de creencias sobre sus propiedades medicinales, además de sus atributos mágico religiosos llegando a convertirse en uno de los recursos más empleados en la herbolaría alrededor del mundo [1].

En 2016 la Secretaría de Agricultura, Ganadería, Desarrollo Rural, Pesca y Alimentación (SAGARPA) informó que la producción de este cultivo aumentó en 28.9% respecto al 2015 [2].

Con base en estadísticas del Servicio de Información Agroalimentaria y Pesquera (SIAP), la dependencia federal indicó que en 2016 la producción de ajo alcanzó las 75 mil 266 toneladas, lo que representa un volumen adicional de 17 mil 159 toneladas en relación a lo reportado en 2015.

La SAGARPA explicó que este perenne se cultiva en 21 entidades del país y los principales estados productores son Zacatecas, Guanajuato, Puebla, Baja California y Sonora, estados que aportan el 87.1 por cierto de la producción nacional.

Zacatecas produce 42 mil 340 toneladas, que equivalen al 56.3 por ciento del total; Guanajuato, 12 mil 586 toneladas; Puebla, tres mil 969 toneladas; Baja California, tres mil 682 toneladas, Sonora, dos mil 969 toneladas y Veracruz con 666 toneladas.

Otras entidades que también producen este alimento son Aguascalientes, Nuevo León, Oaxaca, Querétaro, Durango, San Luis Potosí, Hidalgo, Guerrero, Jalisco, Tlaxcala, Baja California Sur, Coahuila y Michoacán, entre otras [2].

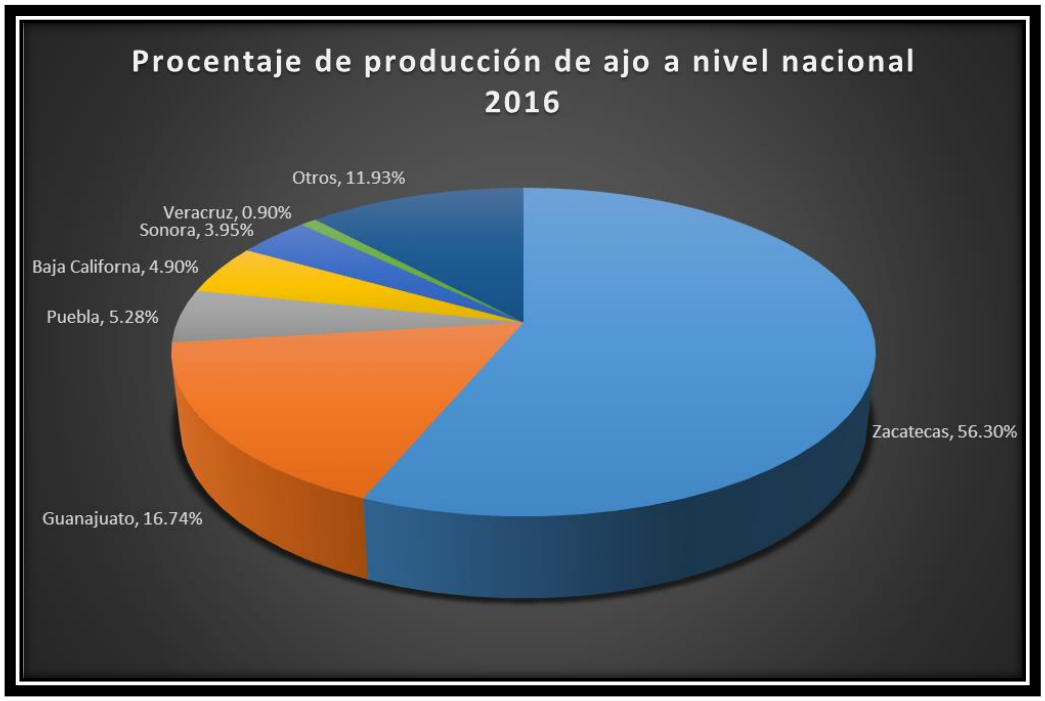

Figura 1.1 Principales estados productores de ajo en México en el año 2016

En la actualidad en el cultivo del ajo existen tres tipos de siembra [3]: a) manual, b) mecánica y c) mixta. La plantación manual es la más común en las zonas ajeras (Figura 1.2), está es una de las operaciones más caras y limitantes del cultivo, junto con la recolección, debido a la gran cantidad de mano especializada durante el proceso lo que eleva el costo de producción.

Según García [4], la siembra manual es una labor que requiere mano de obra con experiencia, para colocar la semilla a una profundidad y en una posición adecuada, ya que de la calidad de la siembra y de la semilla depende una buena germinación y desarrollo de la planta.

Cada trabajador se encarga de una línea colocando los dientes a intervalos regulares según el marco de plantación. A la cintura se ata un pequeño recipiente, una bolsa o paño doblado, donde almacena una cantidad de bulbillos, que el responsable de la cuadrilla le va suministrando.

La plantación manual no sólo es crítica por su costo, sino también por la disponibilidad de mano de obra durante la época adecuada. Los sistemas mecanizados por completo reducen el número de operarios a dos: el tractorista y un peón o jornal.

Figura 1.2 Siembra de ajo de manera manual

El sistema de plantación mixta se basa en la utilización de un bastidor, arrastrado por el tractor, donde los operarios se sitúan para colocar, de forma manual, los dientes sobre el terreno [5]. Esta forma de trabajo no es muy utilizada debido a que es necesaria una perfecta coordinación entre los plantadores, para evitar continuas interrupciones de la operación. En este sistema el operario, sentado en un asiento fijado al bastidor, coloca manualmente los dientes dispuestos con la "punta" hacia arriba. Los bulbillos se encuentran almacenados en compartimentos bien fijos al bastidor o bien atados al cuerpo del plantador.

Las máquinas seguramente nunca van a imitar a una buena tarea artesanal, sin embargo, la falta de mano de obra en tiempo y forma, prácticamente obliga a ir buscando soluciones que tiendan a mecanizar las diferentes operaciones, aunque se sacrifique precisión y productividad del cultivo.

Algunos equipos por lo general son de alto costo de inversión inicial y bajo costo operativo, sin embargo, por el hecho que solo son de uso exclusivo a este cultivo, el costo de amortización es alto. Sólo como ejemplo: adquirir una desgranadora/clasificadora de dientes, una plantadora y una cosechadora de tamaños medianos implica el costo aproximado de producción de 8 a 9 hectáreas, y el mismo debe ser amortizado en un tiempo no inferior a 10 años [5].

Los altos costos, aunados a la falta de disponibilidad de mano de obra calificada en los países desarrollados han impulsado la introducción de equipos semiautomáticos y automáticos para mecanizar esta labor pesada. Sin embargo, su desempeño, aunque satisfactorio, está por debajo de los niveles de calidad logrados con la siembra manual, o por sembradoras para otras semillas. La semilla del ajo es una de las más difíciles desde el punto de vista de la mecanización de siembra, debido a su forma, intervalo de variación en tamaño, y por su túnica y resina [3].

Como explica López *et al* [5], las máquinas de cangilones producen graves daños en los dientes, tanto en el ingreso del cangilón como a la salida en el tubo de descarga. Tienen dos tamaños de cangilones y es recomendada solo para plantaciones con fines industriales (Figura 1.3).

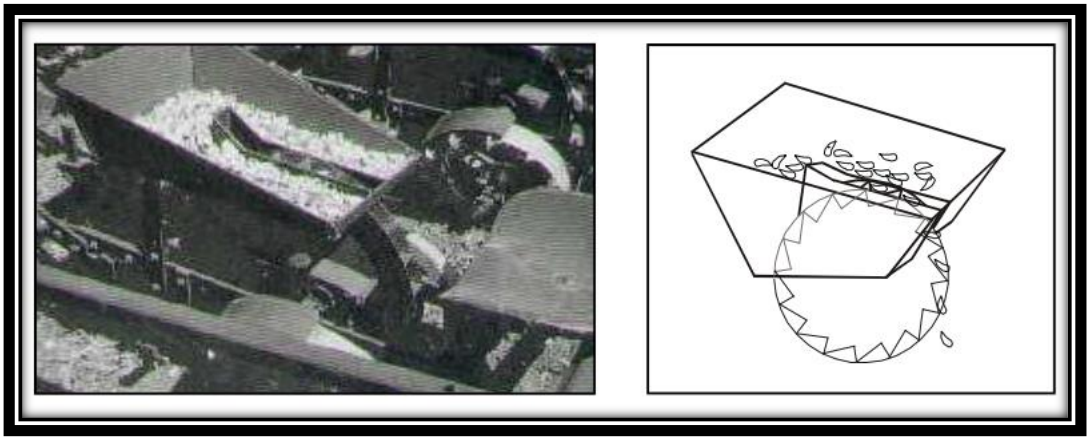

Figura 1.3 Máquina sembradora de ajos por medio de cangilones

Las máquinas de pinzas (intercambiables según el tamaño del diente), exigen que la semilla esté bien clasificada por tamaños (Figura 1.4).

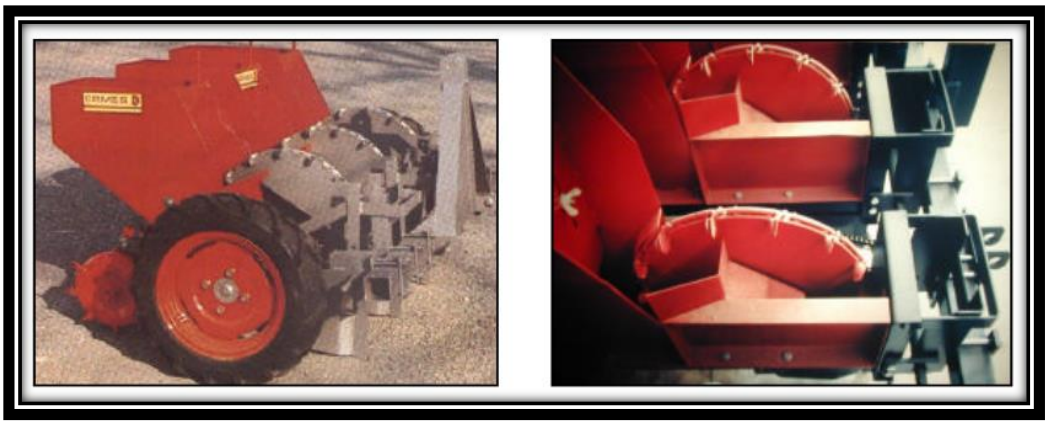

Figura 1.4 Máquina sembradora de dientes de ajo por medio de pinzas

Las máquinas neumáticas exigen que no haya catáfilas sueltas que obture los orificios de los platos de distribución, requiere de la toma de fuerza del tractor. Con el uso continuo manifiesta pérdidas de aire e impide sostener dientes grandes. Es poco práctica para el cambio de discos y costosa.

Todas ellas tienen problemas de distribución en la línea, ya sea por excesos de plantación, que traen aparejados, faltas de uniformidad y deformaciones en los bulbos por competencia (Figura 1.5).

Figura 1.5 Deformación del bulbo por competencia

Para evitar estos problemas que afectan la calidad y rendimiento del cultivo, se ha propuesto la combinación de la máquina plantadora con personal que realiza la corrección mediante arreglo manual de la separación de los dientes que han sido depositados en el suelo (Figura 1.6). Para cada línea de plantación se asigna un operario que, con un vástago de madera, mueve aquellos dientes que han caído excesivamente juntos.

Figura 1.6 Corrección manual de una plantadora mecánica

Los resultados de la experiencia mostraron que la eficiencia sin corrección manual fue del 80% (porcentaje de distribución deseado), y este se puede mejorar cuando se realiza la corrección manual al 90 %, valores muy similares a una plantación manual. El arreglo manual permite minimizar las pérdidas por bulbos deformes por alta densidad [5].

En la actualidad todavía no se ha resuelto el principal problema asociado a este tipo de plantación: el posicionado de los dientes en el terreno [3]. Según López, *et al* [5], la plantación o siembra es la etapa de mecanización más cuestionada por la posición de caída del diente en el suelo y por la distribución, sin embargo, la eficiencia de los equipos depende de varios factores, algunos propios de la máquina, otros corresponden a la semilla y al manejo general del cultivo.

De acuerdo con resultados experimentales obtenidos en los Campos Experimentales Zacatecas y Bajío del INIFAP, se ha demostrado que la posición de la semilla al momento de la siembra afecta la calidad de la planta y el rendimiento final del cultivo, por lo que, la semilla debe colocarse con la parte apical hacia arriba para facilitar la germinación y evitar la deformación de los bulbos (Figura 1.7), [6,7].

Figura 1.7 Deformación de una planta de ajo por una mala colocación de la semilla

Castaños [8] propone cambiar la técnica y tecnología de producción de ajo a bandejas, para su posterior trasplante en campo o invernaderos y sus variantes, mediante la utilización de máquinas sembradoras estacionarias que sean capaces de colocar los dientes de manera correcta en los alveolos de las bandejas, que previamente han sido llenadas de sustrato y se le ha practicado orificios donde se alojarán los dientes. Acto seguido la bandeja avanza para que los orificios sean cubiertos de sustrato y se les aplique un riego de auxilio para favorecer la brotación, que ocurrirá en una cámara de germinación con clima e iluminación controlados.

Después de un tiempo de adaptación al medio ambiente, las bandejas pasarán a invernaderos de producción de plántulas donde completarán su crecimiento y una vez que se alcance, de acuerdo a los requerimientos agronómicos y de las máquinas trasplantadoras a utilizar, se les llevará al lugar de trasplante para que completen su crecimiento hasta la cosecha. Por regla general, las plántulas que se van a trasplantar deben haber formado dos o más hojas verdaderas.

El siguiente tema busca mejorar el proceso de mecanización en la siembra de dientes de ajo, diseñando un sistema de identificación por medio de visión artificial y redes neuronales artificiales para la identificación del ápice de dientes de ajos, ya que, el diente de ajo es una semilla susceptible a la orientación en la que ésta es plantada; la parte apical de la semilla tiene que ser plantada orientada hacia arriba para el correcto desarrollo de la planta, si el diente no cae en esa posición la planta pude crecer con deformaciones que afecten el desempeño del cultivo o en el caso más extremo no desarrollarse.

1.4 Objetivos

1.4.1 General

El objetivo de este trabajo es el diseño de una red neuronal artificial para la identificación del ápice de dientes de ajos mediante visión artificial para automatizar su siembra en charolas utilizando una máquina sembradora estacionaria.

1.4.2 Específicos

1. Evaluar la eficacia de la red neuronal y del sistema de orientación.
2. Una vez identificado el ápice del diente de ajo, mandar una señal de control para el sentido de giro del actuador, el cual moverá la corredera del sistema de orientación haciendo que el diente de ajo caiga en la posición identificada por la RNA.
3. Procurar que la parte apical del diente caiga hacia arriba.

1.5 Estado del arte

Al revisar los avances tecnológicos sobre la aplicación de las redes neuronales artificiales en el campo de la agricultura y agroindustria, se encontraron redes neuronales aplicadas a identificar frutas, en búsqueda de sus respectivas formas, en defectos superficiales o en el tamaño [9], maduración por medio de la concentración de elementos volátiles [10] y monitoreo de los procesos de deshidratación de frutas [11].

Hablando de la semilla abordada en esta tesis, no se encontró algo relacionado con redes neuronales artificiales, el único trabajo publicado en ubicar el ápice de un diente de ajo por medio de visión artificial [12] Se propone un análisis digital de imágenes para identificar el ápice de un diente por medio de una cámara de vídeo y lámparas de iluminación; el algoritmo se divide en cuatro pasos 1) captura de la imagen; 2) detección del borde perimetral de los dientes de ajo; 3) cálculo de ángulos en el interior del borde de los dientes y localización del ápice bajo la hipótesis de que éste coincide con el ángulo más pequeño en el interior del borde; 4) identificación de la necesidad de reorientar el ápice.

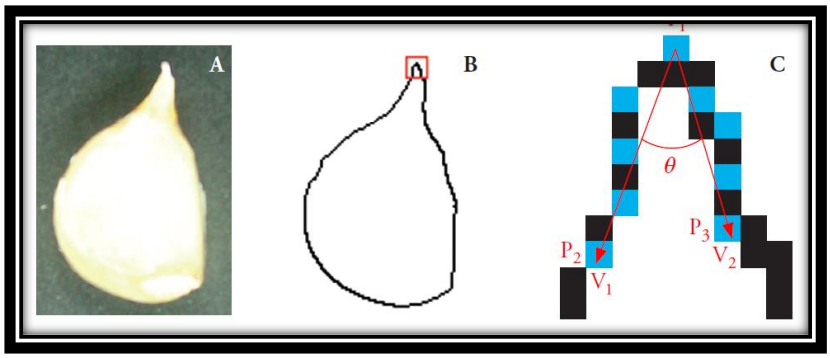

Figura 1.8 Detección del ápice del ajo [12]

Para el experimento se probaron 60 ajos (15 por cada tamaño de diente clasificado: chico, mediano, grande y jumbo), los autores identificaron dos posibles posiciones en que los ajos podían caer, los cuales fueron clasificados en posición lateral y posición frontal, haciendo un total de 360 imágenes.

Los resultados publicados mencionan que 83.3 % de las 360 imágenes de bordes de ajo se identificó correctamente. Para la posición frontal, el ápice se detectó correctamente en 80% de los casos, para la posición lateral 86.7%.

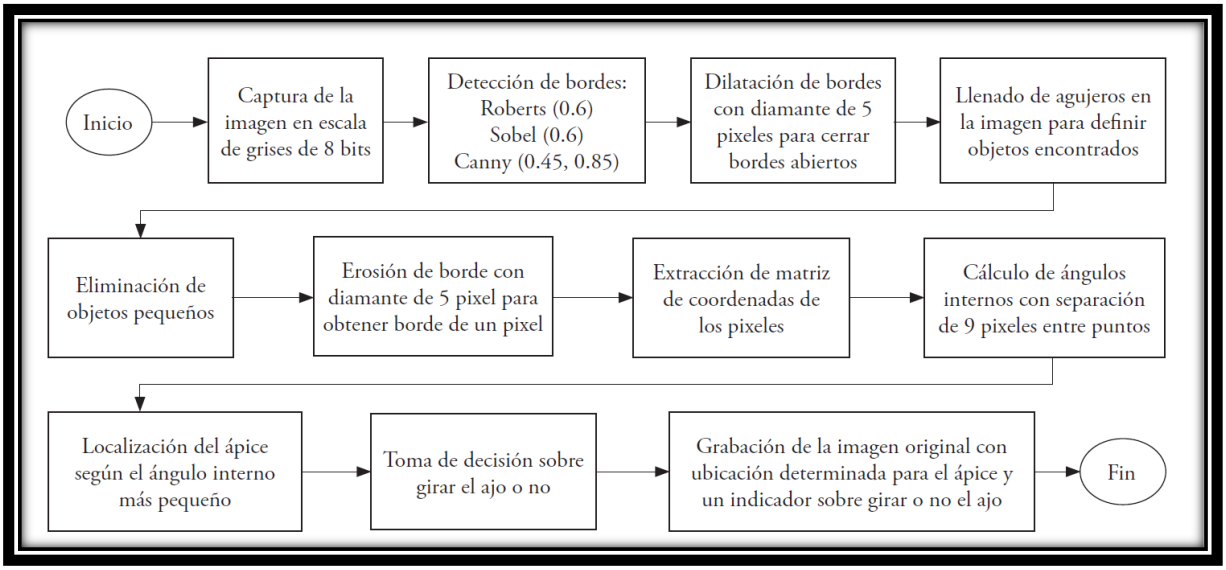

Figura 1.9 Diagrama de bloques usado en [12]

Capítulo 2. Marco Teórico

El cerebro humano es el sistema de cálculo más complejo que conoce el hombre. La computadora y el hombre realizan diferentes clases de tareas; así la operación de reconocer el rostro de una persona resulta una tarea relativamente sencilla para el hombre y difícil para el ordenador, mientras que realizar la contabilidad de una empresa es tarea costosa para un experto contable y una sencilla rutina para un ordenador básico [13].

La capacidad del cerebro humano de pensar, recordar y resolver problemas ha inspirado a muchos científicos intentar o procurar modelar en la computadora el funcionamiento del cerebro humano.

Las redes neuronales artificiales (*Artificial Neural Networks)* o (RNA) son sistemas, hardware o software, de procesamiento, que copian esquemáticamente la estructura neuronal del cerebro para tratar de reproducir sus capacidades. Las RNA son capaces así de aprender de la experiencia a partir de las señales o datos provenientes del exterior [13].

Se estima que el sistema nervioso de los animales vertebrados contiene alrededor de cien mil millones de neuronas que vistas al microscopio pueden presentar múltiples formas, aunque muchas de ellas presentan un aspecto similar muy peculiar (Figura 2.1), con un cuerpo o soma (de entre 10 y 80 micras de longitud), del que surge un denso árbol de ramificaciones (árbol dendrítico) compuesto por las dendritas, y del cual parte de una fibra tubular denominada axón (cuya longitud varia de las 100 micras hasta el metro en el caso de neuronas motoras), que también se ramifican en su extremo final para conectar con otras neuronas.

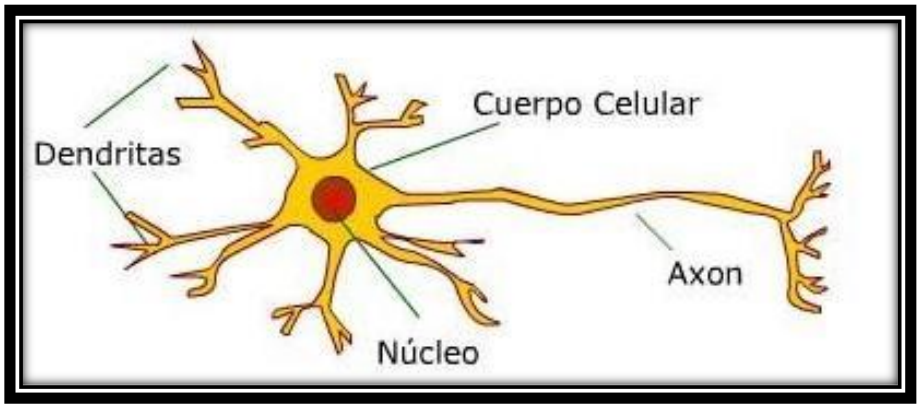

Figura 2.1 Estructura de una neurona biológica.

Desde un punto de vista funcional, las neuronas constituyen procesadores de información. Como todo sistema de este tipo, poseen un canal de entrada de información, las dendritas, un órgano de procesamiento, el soma o cuerpo celular y un canal de salida, el axón. En las interneuronas el axón envía la información a otras neuronas, mientras que en las neuronas motoras lo hace directamente al músculo. Existe un tercer tipo de neurona, las receptoras o sensoras, que, en vez de recibir la información de otras neuronas, la reciben directamente del exterior (tal sucede, por ejemplo, en los conos y bastones de la retina) [13].

La conexión de dos neuronas se denomina sinapsis. En el tipo de sinapsis más común no existe un contacto físico entre las neuronas, sino que éstas permanecen separadas por un pequeño vacío de unas 0.2 micras. En relación a la sinapsis, se habla de neuronas presinápticas (las que envían las señales) y postsinápticas (las que reciben las señales). Las sinapsis son direccionales, es decir, la información fluye siempre en un único sentido [13].

2.1 Marco histórico

Alan Turing (1912-1954) fue el primero en estudiar el cerebro como una forma de ver el mundo de la computación. Sin embargo, los primeros teóricos que concibieron los fundamentos de la computación neuronal fueron Warren McCulloch (1898-1969), un neurofisiólogo, y Walter Pitts (1923-1969), un matemático, quienes, en 1943, lanzaron una teoría acerca de la forma de trabajar de las neuronas. Ellos modelaron una red neuronal simple mediante circuitos eléctricos.

En 1949 Donald Hebb (1904-1985) escribe un importante libro: "La organización del comportamiento", en el que se establece una conexión entre psicología y fisiología. Fue el primero en explicar los procesos del aprendizaje (que es el elemento básico de la inteligencia humana) desde un punto de vista psicológico, desarrollando una regla de como el aprendizaje ocurría. Su idea fue que el aprendizaje ocurría cuando ciertos cambios en una neurona eran activados. También intentó encontrar semejanzas entre el aprendizaje y la actividad nerviosa. Los trabajos de Hebb formaron las bases de la Teoría de las Redes Neuronales.

Frank Rosenblatt (1928-1971) inventa el Perceptrón (1957). Esta es la red neuronal más antigua; utilizándose hoy en día para aplicación como reconocedor de patrones. Sin embargo, tenía una serie de limitaciones, por ejemplo, su incapacidad para resolver el problema de la función OR-exclusiva y, en general, era incapaz de clasificar clases no separables

linealmente. En 1959, escribió el libro Principios de Neurodinámica, en el que confirmó que, bajo ciertas condiciones, el aprendizaje del Perceptrón convergía hacia un estado finito (Teorema de Convergencia del Perceptrón).

Bernard Widrow (1929-) y Marcian Hoff (1937-) desarrollaron el modelo Adaline (ADAptative LINear Elements). Esta fue la primera red neuronal aplicada a un problema real (filtros adaptativos para eliminar ecos en las líneas telefónicas).

Marvin Minsky (1927-2016) y Seymour Papert (1928-2016). En el año de 1969 surgieron críticas que frenaron, hasta 1982, el crecimiento que estaban experimentando las investigaciones sobre redes neuronales. Minsky y Papert, publicaron un libro "Perceptrons." Probaron (matemáticamente) que el Perceptrón no era capaz de resolver problemas relativamente fáciles, tales como el aprendizaje de una función no-lineal.

James Anderson (1940-), desarrolló un modelo lineal, llamado Asociador Lineal, que consistía en elementos integradores lineales (neuronas) que sumaban sus entradas. Este modelo se basa en el principio de que las conexiones entre neuronas son reforzadas cada vez que son activadas. Anderson diseñó una potente extensión del Asociador Lineal, llamada Brain State in a Box (BSB).

Paul Werbos (1947-) en 1974 desarrolla la idea básica del algoritmo de aprendizaje de propagación hacia atrás (backpropagation); cuyo significado quedó definitivamente aclarado en 1985.

Stephen Grossberg (1939-) desarrolla en 1977 la Teoría de Resonancia Adaptada (ART). La Teoría de Resonancia Adaptada una arquitectura de red que se diferencia de todas las demás previamente inventadas. La misma simula otras habilidades del cerebro: memoria a largo y corto plazo.

John Hopfield (1933) Provocó el renacimiento de las redes neuronales con su libro: «Computación neuronal de decisiones en problemas de optimización» en 1985.

David Rumelhart (1942-2011) y Geoffrey. Hinton (1947 -) redescubren en el año de 1986 el algoritmo de aprendizaje de propagación hacia atrás (backpropagation). A partir de ese año, el panorama fue alentador con respecto a las investigaciones y el desarrollo de las redes neuronales [14].

2.2 Estructura de una Red Neuronal Artificial

Una RNA imita la estructura hardware del sistema nervioso, con la intención de construir sistemas de procesamiento de la información paralelos, distribuidos y adaptativos, que puedan presentar un cierto comportamiento "inteligente".

Una computadora es, en esencia, una máquina, construida en torno a una única CPU o procesador, que ejecuta de un modo secuencial un programa almacenado en memoria. Por el contrario, el cerebro no está compuesto por un único procesador, sino por miles de millones de ellos (neuronas), aunque muy elementales. Curiosamente, las neuronas son mucho más simple, más lentas y menos fiables que una CPU, y, a pesar de ello, existen problemas difícilmente abordables mediante una computadora convencional, que el cerebro resuelve eficazmente (reconocimiento del habla, visión de objetos inmensos en el ambiente natural, respuesta ante estímulos del entorno, etc.) [13].

Por lo tanto, la idea que subyace en los sistemas neuronales artificiales es que, para abordar el tipo de problemas que el cerebro resuelve con eficiencia, puede resultar conveniente construir sistemas que "copien" en cierto modo la estructura de las redes neuronales biológicas con el fin de alcanzar una funcionalidad similar.

Los elementos básicos de un sistema neuronal biológico son las neuronas, que se agrupan en conjuntos compuestos por millones de ellas organizadas en capas, constituyendo un sistema con funcionalidad propia. Un conjunto de estos subsistemas da lugar a un sistema global (el sistema nervioso, en el caso biológico). En la realización de un sistema neuronal artificial puede establecerse una estructura jerárquica similar. El elemento esencial de partida será la neurona artificial o elemento procesador, que se organizará en capas; varias capas constituirán una red neuronal; y, por último, una red neuronal (o conjunto de ellas), junto con las interfaces de entrada y salida, más los módulos convencionales adicionales necesarios, constituirán el sistema global de proceso [13].

2.2.1 Modelo general de una neurona artificial o elemento procesador

Se ocupará la siguiente notación [15] para diferenciar los siguientes elementos:
Escalares: Letras minúsculas en itálica: a, b, c.
Vectores: Letras minúsculas en **negritas**: **a, b, c**.

Matrices: Letras mayúsculas en **negritas**: **A, B, C**.
Una neurona artificial o elemento procesador con una sola entrada se muestra en la Figura 2.2.

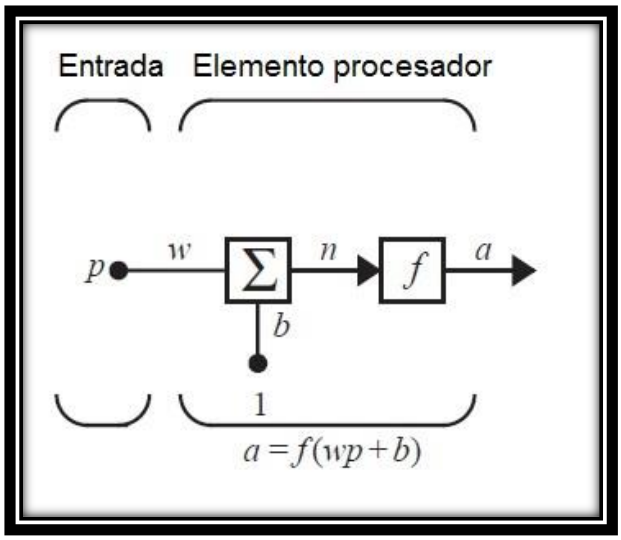

Figura 2.2 Modelo general de un elemento procesador

La entrada escalar p es multiplicada por el escalar de peso w para formar wp, este término es enviado a la suma. La otra entrada, 1, es multiplicada por un sesgo (bias) b, el segundo término de la suma. Esta suma proporciona la salida n, también conocida como $entrada\ neta$, este término se evalúa en la función de activación f, la cual produce la salida de la red a.

Si comparamos este modelo simple de un elemento procesador (neurona artificial) con una biológica, el peso w corresponde a la fuerza de la conexión sináptica, el cuerpo celular está representado por la suma y la función de activación, la salida de la neurona a representa la señal en el axón.

La salida de un elemento procesador puede calcularse como:

$$a = f(wp + b) \qquad \text{Ec.1}$$

El valor de la salida depende de la función de activación que se elige. El sesgo es una variable extra que proporciona a la red neuronal artificial un grado de libertad extra, excepto que éste tiene siempre un valor constante de 1, que puede o no estar dependiendo del tipo de red neuronal.

Tanto los valores de $w\ y\ b$ son parámetros ajustables. Generalmente la función de activación es elegida por el diseñador de la red y los parámetros $w\ y\ b$ se ajustan por la regla de entrenamiento que relaciona las entradas y salidas de la red. Se tienen varios tipos de funciones de activación para diferentes propósitos [15].

2.2.2 Funciones de activación

La función de activación o función de transferencia mostrada en la Figura 2.3 puede ser una función lineal o no lineal de n. Una función de activación se elige para satisfacer ciertas especificaciones del problema a resolver [15].

A continuación, se nombran las funciones de activación con más uso:

La función escalón "hardlimit" (Figura 2.3), obtiene como salida 0 si el argumento de la función es menor que 0, o 1 si el argumento es igual o mayor a 0. Se usa esta función de activación para solucionar problemas de clasificación con dos o más categorías.

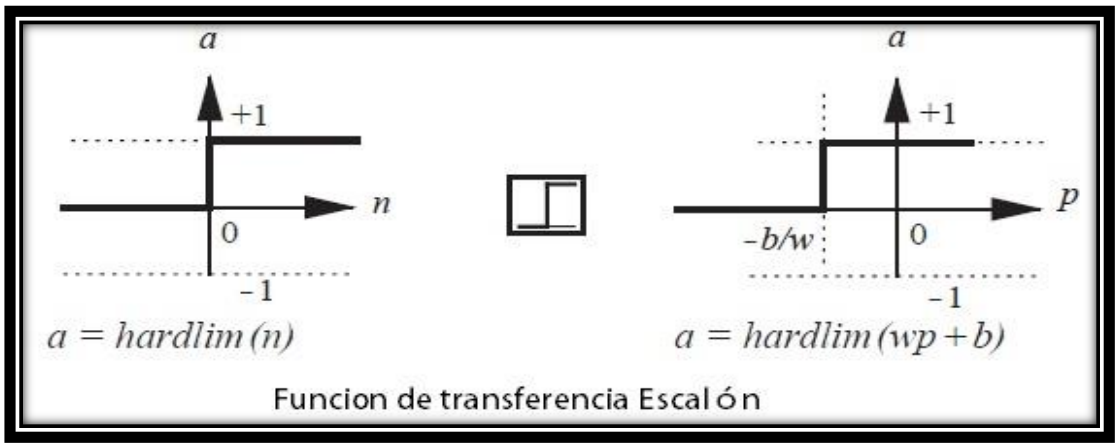

Figura 2.3 Función de activación Escalón

La salida en una función de activación lineal "purelin" (Figura 2.4) es igual a la entrada:

$$a = n \qquad \text{Ec 2}$$

Los elementos procesadores en una red ADALINE ocupan este tipo de función de activación.

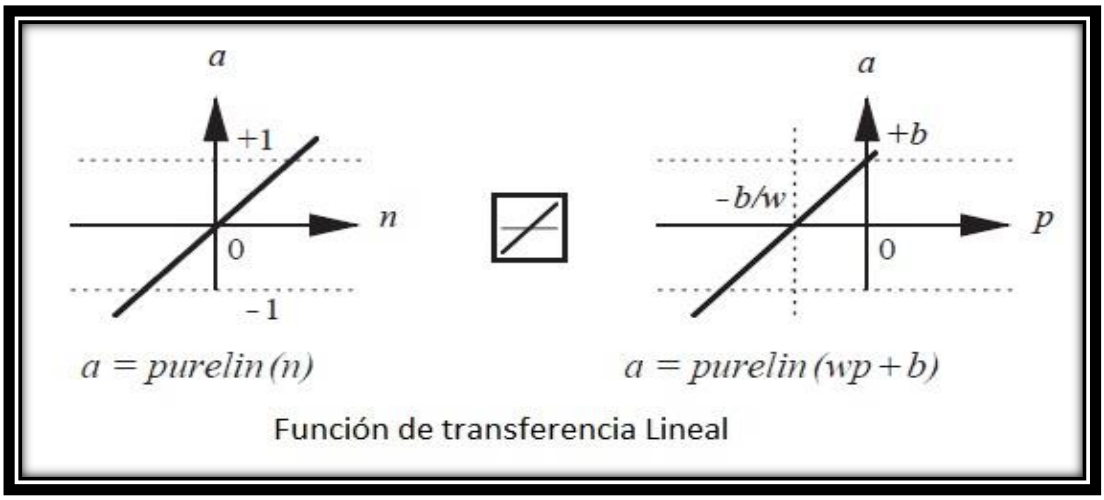

Figura 2.4 Función de activación Lineal

La función de activación logarítmica sigmoidal "logsig" (Figura 2.5).

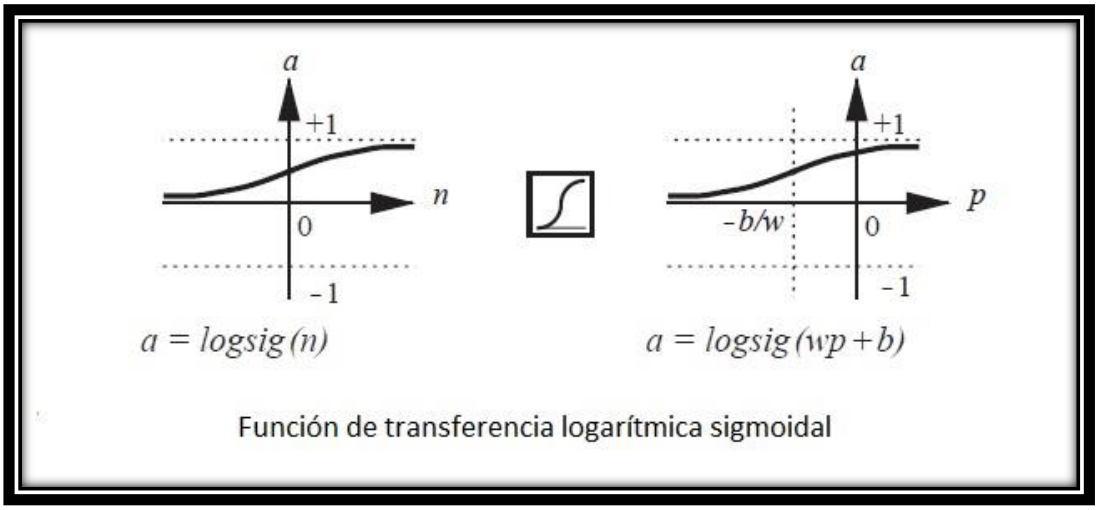

Figura 2.5 Función de activación logarítmica sigmoidal

Esta función de activación toma como entradas cualquier valor de $\infty > n > -\infty$ y lo transforma a un rango entre 0 y 1, de acuerdo a la siguiente expresión:

$$a = \frac{1}{1 - e^{-n}} \qquad \text{Ec 3}$$

Esta función se usa comúnmente en redes multicapas usando el algoritmo de retropropagación, en parte porque esta función es derivable.

2.2.3 Elemento procesador con entradas múltiples

Generalmente, un elemento procesador tiene más de una entrada. Un elemento procesador con R entradas se muestra en la Figura 2.6.

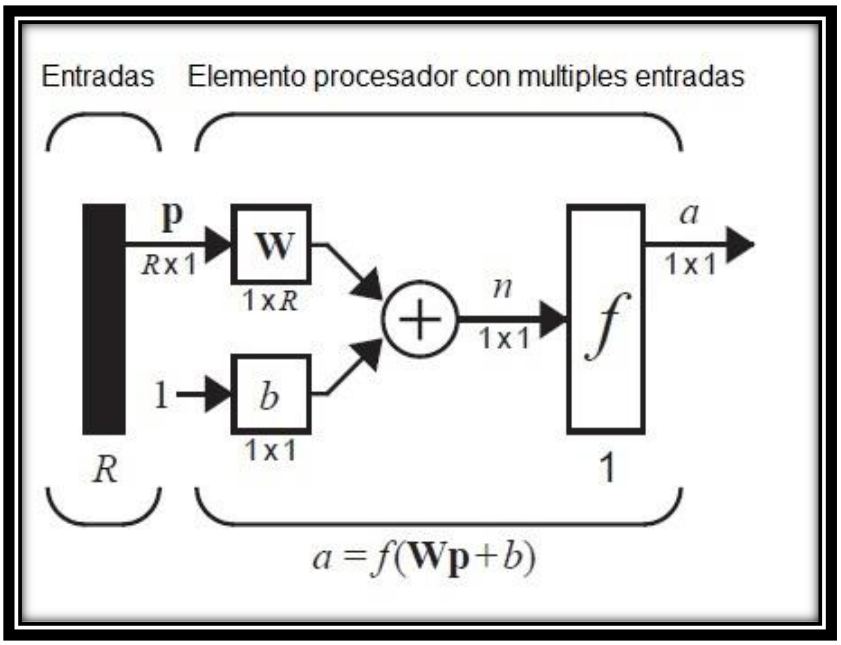

Figura 2.6 Elemento procesador con múltiples entradas

Las entradas individuales $p_1, p_2, \dots, p_R$ están multiplicadas por los elementos correspondientes $w_{1,1}, w_{1,2}, \dots, w_{1,R}$ de la matriz de pesos **W**. El valor del sesgo *b* se suma con la multiplicación de las entradas con los pesos formando la entrada neta *n*:

$$n = w_{1,1}p_1 + w_{1,2}p_2 + \cdots + w_{1,R}p_R + b \qquad \text{Ec 4}$$

Esta expresión puede ser escrita en forma matricial de la siguiente forma:

$$n = \mathbf{W}\mathbf{p} + b \qquad \text{Ec 5}$$

Ahora la salida de la red es:

$$a = f(\mathbf{W}\mathbf{p} + b) \qquad \text{Ec 6}$$

Afortunadamente, las redes neuronales artificiales pueden escribirse en forma matricial.

Para interpretar los subíndices de los elementos de la matriz de pesos se ocupará las siguientes reglas. El primer subíndice indica el número de neurona, el segundo subíndice representa el número de entrada que está siendo procesada [15].

2.3 Redes Monocapa

Comúnmente un solo elemento procesador, aún con múltiples entradas no suele ser suficiente para resolver el problema, los elementos procesadores pueden trabajar en paralelo, este tipo de arreglo se le conoce como "capa" [15].

Una red monocapa con *S* elementos procesadores se muestra en la Figura 2.7, con *R* entradas que se conectan a cada neurona a través de la matriz de pesos con *S* filas y *R* columnas.

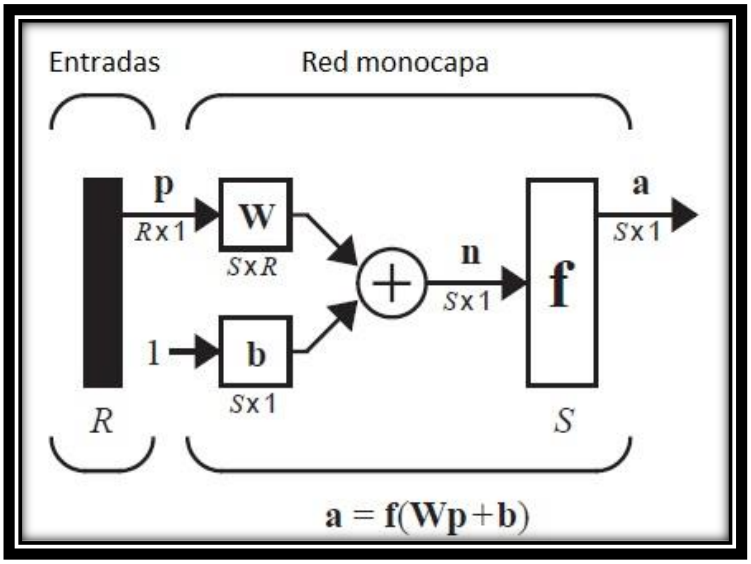

Figura 2.7 Representación de una red monocapa

Cada elemento del vector de entradas **p** está conectado a cada neurona a través de la matriz de pesos **W**. Cada neurona posee su valor de sesgo b_i, una suma y una función de activación *f* así como una salida a_i, formando en vector de salidas **a**.

Es común que el número de entradas sea diferente al número de neuronas en la capa $(R \neq S)$.

Cada elemento del vector de entrada se multiplica con los elementos de la matriz de pesos **W:**

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & \cdots & w_{1,R} \\ \vdots & \ddots & \vdots \\ w_{S,1} & \cdots & w_{S,R} \end{bmatrix}$$

Como se puede notar, las filas indican el número del elemento procesador, mientras que las columnas indican el número de entrada que está siendo multiplicado [15].

2.3.1 Perceptrón

En 1943, Warren McCulloch y Walter Pitts introdujeron una de los primeros elementos procesadores. La tarea principal de éste es sumar el producto de las entradas por los pesos y compararlo con un umbral para determinar la salida de la neurona. Cuando la suma es mayor o igual que el valor del umbral, la salida es "1", mientras que, si el valor de la suma es menor al umbral, la salida es "0". Demostraron que este elemento procesador era capaz de hacer algunas operaciones lógicas básicas. En comparación con las neuronas biológicas este tipo de elemento procesador tenía que ser diseñada en vez de entrenada ya que no había una regla de entrenamiento [15].

En la década de los 50s, Frank Rosenblatt y un grupo de investigadores desarrollaron una clase de red neuronal llamada perceptrón. Los elementos procesadores en esta red eran similares a las neuronas de McCulloch-Pitts. El descubrimiento de Rosenblatt fue la introducción de la regla de entrenamiento del perceptrón para la resolución de problemas. Él demostró que esa regla siempre converge a una solución si ésta existe, es decir encuentra los pesos correctos para solucionar el problema.

Desafortunadamente, la red perceptrón posee limitaciones. Estas limitaciones fueron publicadas en el libro *Perceptrons* de Marvin Minsky y Seymour Papert, quienes demostraron que la red perceptrón era incapaz de implementar ciertas operaciones lógicas básicas. No fue sino hasta la década de los 80s cuando esas limitaciones fueron superadas con la implementación del perceptrón multicapa y su respectiva regla de entrenamiento.

El perceptrón hoy día sigue siendo una red neuronal artificial muy importante por su capacidad y velocidad para resolver problemas de clasificación, además, entender la operación de un perceptrón facilita el entendimiento de redes más complejas [15].

La salida de la red está dada por:

$$\mathbf{a} = \boldsymbol{hardlim}(\mathbf{Wp} + \mathbf{b}) \qquad \text{Ec 7}$$

O bien

$$\mathbf{a} = \boldsymbol{hardlims}(\mathbf{Wp} + \mathbf{b}) \qquad \text{Ec 8}$$

El esquema general de un perceptrón se muestra en la Figura 2.8

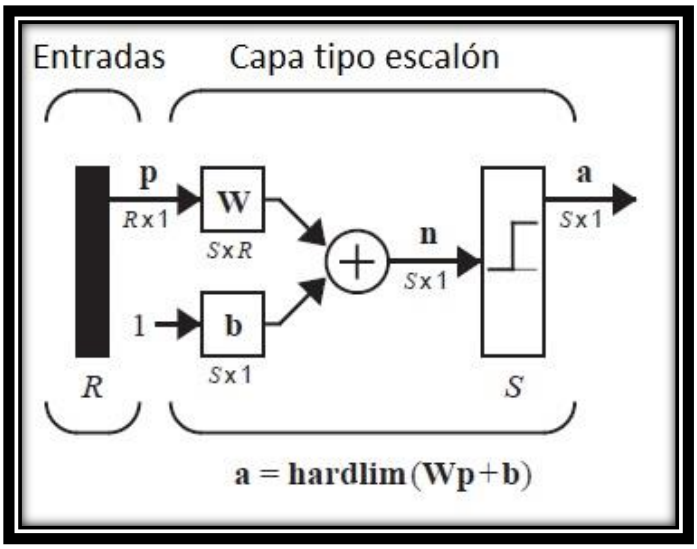

Figura 2.8 Esquema general de un perceptrón

Así, sí el producto de la matriz de pesos y el vector de entrada es mayor o igual a $-b_i$, la salida de la neurona será 1, sino 0. Se puede decir que cada neurona en la red divide el espacio de entradas en dos regiones [15].

2.3.2 ADALINE

La red ADALINE es muy similar a la del perceptrón excepto que ésta usa una función de activación lineal en vez de una función escalón. Ambas redes sufren la misma limitante: sólo pueden resolver problemas con separabilidad lineal [15].

La red ADALINE (ADAptive LInear NEuron) se muestra en la Figura 2.9

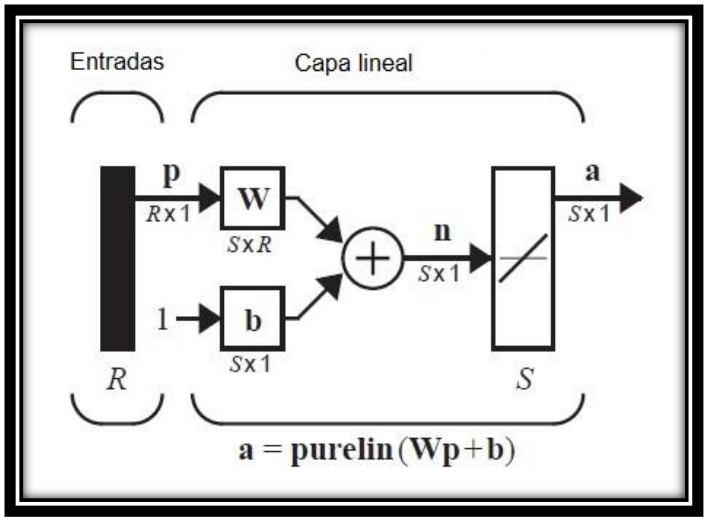

Figura 2.9 Red ADALINE

La salida de la red está dada por la siguiente expresión:

$$\mathbf{a} = \mathbf{purelin}(\mathbf{Wp} + \mathbf{b}) = \mathbf{Wp} + \mathbf{b} \qquad \text{Ec 9}$$

2.4 Redes Multicapa

Ahora consideremos el caso de una red multicapa, estas redes son mucho más poderosas que las redes monocapas ya que son capaces de resolver problemas de separabilidad no lineal, así como los problemas de aproximación de funciones.

Cada capa posee su propia matriz de pesos **W**, su vector de sesgos **b**, un vector de entradas netas **n** y un vector de salidas **a**. Para lograr distinguir este tipo de arquitecturas se usará la siguiente nomenclatura:

Se ocuparan superíndices para identificar las capas, así la matriz de pesos de la primera y segunda capa estarán representados por $\mathbf{W}^1, \mathbf{W}^2$ respectivamente.

A continuación, se muestra un ejemplo de cómo aplicar esta nomenclatura (Figura 2.10)

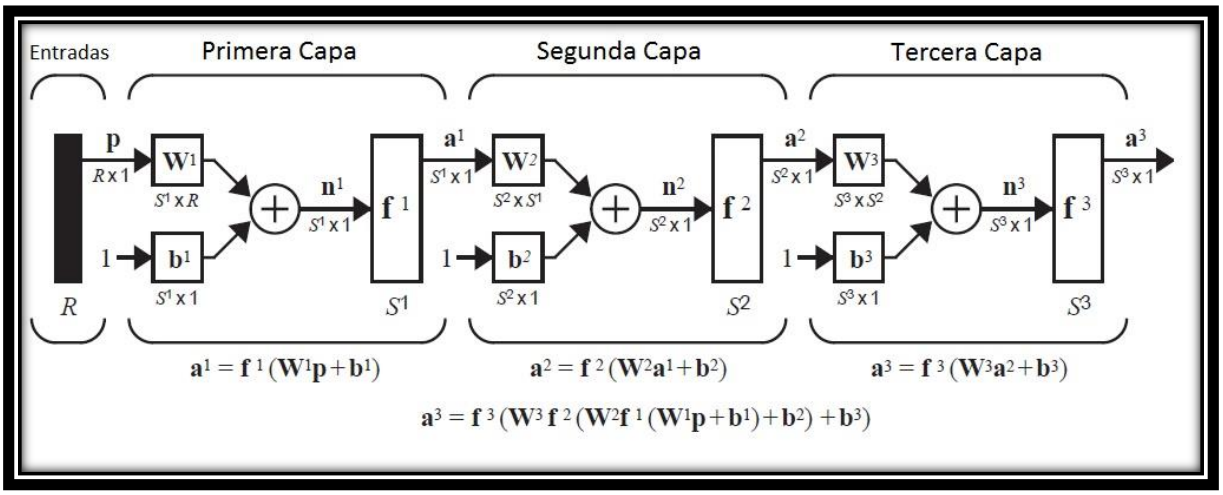

Figura 2.10 Red Multicapa

La última capa de la red se le conoce como capa de salida, las otras capas se conocen como capas ocultas, en la Figura 2.10 se muestran dos capas ocultas (capa 1 y capa 2) y una capa de salida (capa 3).

En la práctica las redes con 2 o 3 capas son más usadas que redes con 4 o más capas.

2.5 Reglas de entrenamiento

Por regla de entrenamiento entenderemos al procedimiento para modificar los pesos y sesgos de una red. (Éste también puede ser llamado algoritmo de entrenamiento). El propósito de estas reglas es entrenar la red para que realice una tarea. Existen varias reglas de entrenamiento para redes, pero todas ellas caen en tres categorías: a) aprendizaje supervisado, b) aprendizaje no supervisado y c) aprendizaje por reforzamiento.

En el aprendizaje supervisado, la red aprende con una serie de patrones de ejemplos (patrones de entrenamiento) de la forma:

$$\{\mathbf{p}_1, \mathbf{t}_1\}, \{\mathbf{p}_2, \mathbf{t}_2\}, \dots, \{\mathbf{p}_Q, \mathbf{t}_Q\}$$

Donde $\mathbf{p}_q$ es la entrada de la red y $\mathbf{t}_q$ es la salida deseada correspondiente (objetivo). Como las entradas van siendo procesadas por la red, la salida obtenida se compara con la salida deseada. La regla de entrenamiento trata de ajustar los pesos y los sesgos para que la salida de la red sea lo más parecido a la salida deseada.

El aprendizaje por reforzamiento es similar al entrenamiento supervisado, excepto que, en vez de presentar la salida deseada por cada entrada, se le otorga una calificación. Esta calificación es una medida del rendimiento de la red sobre una secuencia de entradas. Este tipo de entrenamiento es menos usado que el entrenamiento supervisado, siendo usado en aplicaciones de sistemas de control.

En el aprendizaje no supervisado, los pesos y los sesgos son modificados en respuesta únicamente de las entradas, aquí no hay salidas deseadas disponibles para la red. Como primer vistazo esto puede parecer impráctico. ¿Cómo podemos entrenar una red si no sabemos que es lo que tiene que hacer? Muchos de estos algoritmos se ejecutan en algún tipo de operación con clusters. Ellos aprenden a separar los patrones de entrada en un número finito de categorías, este tipo de aprendizaje se ocupa mucho en aplicaciones que involucren cuantificación de vectores [16].

2.5.1 Entrenamiento de Widrow-Hoff

Bernard Widrow comenzó a trabajar en Redes Neuronales Artificiales a finales de la década de los 50´s, al mismo tiempo que Frank Rosenblatt diseñaba el algoritmo de aprendizaje del perceptrón. En 1960 Widrow y su alumno Marcian Hoff, introdujeron la red ADALINE y su regla de entrenamiento la cual llamaron algoritmo LMS (Least Mean Square).

El algoritmo LMS es más poderoso que el algoritmo de aprendizaje del perceptrón, mientras que en la regla de aprendizaje del perceptrón garantiza converger a una solución que separe correctamente los patrones de entrada, el resultado del entrenamiento puede ser sensible a ruido de patrones muy cercanos a la frontera de decisión. Los algoritmos LMS minimiza el error cuadrático medio, además de que trata de mover la frontera de decisión lo más lejano posible de los patrones de entrenamiento.

El algoritmo LMS ha encontrado mucho más usos prácticos que la regla de entrenamiento del perceptrón. Esto es especialmente cierto en áreas del procesamiento digital de señales. Por ejemplo, las líneas telefónicas más largas usan una red ADALINE para la cancelación de eco.

A causa del gran logro de algoritmo LMS en aplicaciones de procesamiento de señales y a la ausencia de un algoritmo para redes multicapas, Widrow dejo de trabajar en redes neuronales en los primeros años de la década de los 60´s y se dedicó de tiempo completo en procedimientos adaptativos de señales. Regresó al campo de las redes neuronales artificiales en los 80´s y comenzó la investigación del uso de redes neuronales artificiales en control adaptativo, usando temporalmente el algoritmo de Retropropagación, un descendiente del algoritmo LMS.

El algoritmo LMS es un ejemplo de entrenamiento supervisado, a la red se le muestran patrones de entradas y las salidas deseadas, en cada iteración la salida de la red es comparada con la salida deseada.

Este algoritmo ajustará los pesos y los sesgos de una ADALINE para minimizar el error cuadrático medio, donde el error es la diferencia entre la salida deseada y la salida de la red [16].

El algoritmo LMS es el siguiente [16]:

Sea

$$\mathbf{x} = \begin{bmatrix} {}_1\mathbf{w} \\ b \end{bmatrix}$$

Donde

$$_i\mathbf{w} = \begin{bmatrix} w_{i,1} \\ \vdots \\ w_{i,R} \end{bmatrix}$$

Es un vector compuesto por los elementos de la i-ésima fila de la matriz **W**

Similarmente, se incluye el peso del sesgo "1" como un elemento del vector de entradas.

$$\mathbf{z} = \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix}$$

Ahora la salida de la red se obtendrá de la siguiente forma

$$a = {}_1\mathbf{w}^T\mathbf{p} + b \qquad \text{Ec 10}$$

Sustituyendo los nuevos vectores se obtiene

$$a = \mathbf{x}^T\mathbf{z} \qquad \text{Ec 11}$$

El error es definido como

$$e = t - a \qquad \text{Ec 12}$$

El error cuadrático medio se define como

$$F(\mathbf{x}) = E[e^2] = E[(t-a)^2] = E[(t - \mathbf{x}^T\mathbf{z})^2] \qquad \text{Ec 13}$$

Desarrollando la expresión

$$F(\mathbf{x}) = E[t^2 - 2t\mathbf{x}^T\mathbf{z} + \mathbf{x}^T\mathbf{z}\mathbf{z}^T\mathbf{x}] \qquad \text{Ec 14}$$

$$F(\mathbf{x}) = E[t^2] - 2\mathbf{x}^T E[t\mathbf{z}] + \mathbf{x}^T E[\mathbf{z}\mathbf{z}^T]\mathbf{x} \qquad \text{Ec 15}$$

Que puede ser reescrita de la siguiente manera

$$F(\mathbf{x}) = c - 2\mathbf{x}^T\mathbf{h} + \mathbf{x}^T\mathbf{R}\mathbf{x} \qquad \text{Ec 16}$$

Donde

$$c = E[t^2]\,, \mathbf{h} = E[t\mathbf{z}] \;\; y \;\; \mathbf{R} = E[\mathbf{z}\mathbf{z}^T]$$

Aquí el vector **h** proporciona la correlación cruzada entre el vector de entradas y la salida deseada, mientras que **R** es la matriz de correlación de las entradas. Los elementos de la diagonal de la matriz son igual al error cuadrado medio de los elementos del vector de entradas.

Ahora obtendremos el punto estacionario, para ello se obtiene el gradiente de la función

$$\nabla F(\mathbf{x}) = \boldsymbol{\nabla}\left(\mathrm{c} + \boldsymbol{d}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x}\right) = \mathbf{d} + \mathbf{A}\mathbf{x} = -2\mathbf{h} + 2\mathbf{R}\mathbf{x} \qquad \text{Ec 17}$$

El punto estacionario de F(**x**) se obtiene igualando el gradiente a cero

$$-2\mathbf{h} + 2\mathbf{R}\mathbf{x} = 0 \qquad \text{Ec 18}$$

Por lo tanto, si la matriz de correlación es positiva definida habrá un único punto estacionario que será un mínimo:

$$\mathbf{x}^* = \mathbf{R}^{-1}\mathbf{h} \qquad \text{Ec 19}$$

El siguiente paso es diseñar el algoritmo para localizar el mínimo de la función. Si se puede calcular las matrices **h** y **R** se puede ocupar la ecuación anterior directamente. Si no se quiere calcular la inversa de **R**, se puede ocupar el algoritmo de pasos descendientes, con el cálculo de gradiente. Sin embargo, no es deseable calcular las matrices **h** y **R**. Por esta razón usaremos una aproximación del algoritmo de pasos descendientes, usando un gradiente estimado.

El algoritmo LMS puede ser escrito en forma matricial de la siguiente forma

$$\mathbf{W}(k+1) = \mathbf{W}(k) + 2\alpha\mathbf{e}(k)\mathbf{p}^T(k) \qquad \text{Ec 20}$$

$$\mathbf{b}(k+1) = \mathbf{b}(k) + 2\alpha\mathbf{e}(k) \qquad \text{Ec 21}$$

2.5.2 Retropropagación

El algoritmo de Retropropagación (Backpropagation) es una generalización del algoritmo LMS, éste es usado en el entrenamiento de redes multicapa. Como el algoritmo LMS, el algoritmo de Retropropagación es una aproximación del algoritmo de pasos descendientes. La diferencia entre el algoritmo LMS y el algoritmo de Retropropagación es únicamente la forma en la que se calculan las derivadas parciales. Para el caso de una capa el error es una función lineal explicita de los pesos de la red, y su derivada respecto a los pesos es sencilla de computar. En una red multicapa con una función de activación no lineal, la relación entre los pesos de la red y el error es más complicada. Para encontrar esa relación necesitaremos utilizar la regla de la cadena.

Los algoritmos del perceptrón y LMS fueron diseñados para entrenar redes monocapa, estas redes poseen la misma limitante, no pueden resolver problemas con separabilidad no lineal. Widrow y Rosenblatt conocían esto y propusieron redes multicapa que podían solucionar esas limitaciones, pero no fueron capaces de generalizar algún algoritmo de entrenamiento para este tipo de redes.

La primera referencia de una red multicapa y un algoritmo para entrenarla se encontró en la tesis de Paul Werbos en 1974. Ésta presentó un algoritmo en general para las redes neuronales, pero no tuvo mucha difusión en la comunidad que estudiaba las redes neuronales artificiales No fue sino hasta mediados de la década de los 80´s que el algoritmo de Retropropagación fue redescubierto y extensamente publicado por David Rumelhart, Geoffrey Hinton y Ronald Williams, David Parker y Yann Le Cun, popularizado en la publicación del libro *Parallel Distributed Processing*. El perceptrón multicapa entrenado con este algoritmo es el más usado en redes neuronales [17].

Como se explica en [17] la salida de la capa siguiente en una red multicapa se usa la siguiente expresión:

$$\mathbf{a}^{m+1} = \mathbf{f}^{m+1}(\mathbf{W}^{m+1}\mathbf{a}^{m} + \mathbf{b}^{m+1}) \; para \; m = 0,1, \dots , M-1. \qquad \text{Ec 22}$$

Donde M es el número de capas en la red. La primera capa recibe el vector de entradas:

$$\mathbf{a}^{0} = \mathbf{p}$$

La salida de la red es el resultado de la última capa:

$$\mathbf{a} = \mathbf{a}^{M}$$

El algoritmo de Retropropagación al igual que el algoritmo LMS usan el mismo índice de rendimiento: el error cuadrático medio. Al algoritmo se le presenta una serie de entradas y salidas deseadas con la forma

$$\{\mathbf{p}_1, \mathbf{t}_1\}, \{\mathbf{p}_2, \mathbf{t}_2\}, \dots , \{\mathbf{p}_Q, \mathbf{t}_Q\}$$

Donde $\mathbf{p}_q$ es la entrada a la red y $\mathbf{t}_q$ es la salida deseada, la red calcula la salida obtenida por cada entrada presentada, el algoritmo ajustará los parámetros de la red para minimizar el error cuadrático medio:

$$F(\mathbf{x}) = E[e^2] = E[(t-a)^2] \qquad \text{Ec 23}$$

Donde **x** es el vector de entradas y sesgos. Si la red está multiplicada por todas las salidas el error cuadrático medio puede generalizarse como

$$F(\mathbf{x}) = E[\mathbf{e}^T\mathbf{e}] = E[(\boldsymbol{t}-\boldsymbol{a})^T(\boldsymbol{t}-\boldsymbol{a})] \qquad \text{Ec 24}$$

Como en el algoritmo LMS, podemos aproximar el error cuadrático como

$$\hat{F}(\mathbf{x}) = \big(\mathbf{t}(\mathrm{k}) - \mathbf{a}(\mathrm{k})\big)^T\big(\mathbf{t}(\mathrm{k}) - \mathbf{a}(\mathrm{k})\big) = \mathbf{e}^{\mathrm{T}}(k)\mathbf{e}(k) \qquad \text{Ec 25}$$

Donde el valor esperado del error es sustituido por el error cuadrático en la iteración *k*.

El algoritmo de pasos descendientes con el error cuadrático medio es

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \alpha \frac{\partial \hat{F}}{\partial w_{i,j}^m} \quad \text{Ec 26}$$

$$b_i^m(k+1) = b_i^m(k) - \alpha \frac{\partial \hat{F}}{\partial b_i^m} \quad \text{Ec 27}$$

Ya que el error es una función indirecta de los pesos en las capas ocultas, necesitamos la regla de la cadena para calcular esas derivadas. Como repaso, tenemos una función *f* que es una función explicita de la variable *n*, podemos encontrar la derivada de *f* con respecto a una tercera variable *w*. La regla de la cadena queda como:

$$\frac{df(n(w))}{dw} = \frac{df(n)}{dn} \times \frac{\mathrm{d}n(w)}{\mathrm{d}w}$$

Usaremos ese concepto para encontrar las derivadas en las ecuaciones del algoritmo de pasos descendentes como:

$$\frac{\partial \hat{F}}{\partial w_{i,j}^m} = \frac{\partial \hat{F}}{\partial n_i^m} \times \frac{\partial n_i^m}{\partial w_{i,j}^m} \quad \text{Ec 28}$$

$$\frac{\partial \hat{F}}{\partial b_i^m} = \frac{\partial \hat{F}}{\partial n_i^m} \times \frac{\partial n_i^m}{\partial b_i^m} \quad \text{Ec 29}$$

El segundo término en cada una de las ecuaciones anteriores puede ser fácilmente calculada, ya que la entrada neta de la capa *m* es una función explicita de los pesos y los sesgos en esa capa:

$$n_i^m = \sum_{j=1}^{S^{m-1}} w_{i,j}^m a_j^{m-1} + b_i^m \quad \text{Ec 30}$$

Por lo tanto

$$\frac{\partial n_i^m}{\partial w_{i,j}^m} = a_j^{m-1} \quad \frac{\partial n_i^m}{\partial b_i^m} = 1$$

Ahora definiremos la sensibilidad de la capa *m*

$$s_i^m = \frac{\partial \hat{F}}{\partial n_i^m} \quad \text{Ec 31}$$

Las ecuaciones del algoritmo quedan hasta el momento como

$$\frac{\partial \hat{F}}{\partial w_{i,j}^m} = s_i^m a_j^{m-1} \quad \text{Ec 32}$$

$$\frac{\partial \hat{F}}{\partial b_i^m} = s_i^m \quad \text{Ec 33}$$

Ahora podemos expresar la aproximación del algoritmo de pasos descendentes:

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \alpha s_i^m a_j^{m-1} \quad \text{Ec 34}$$

$$b_i^m(k+1) = b_i^m(k) - \alpha s_i^m \quad \text{Ec 35}$$

Que de manera matricial se escribe

$$\mathbf{W}^m(k+1) = \mathbf{W}^m(k) - \alpha \mathbf{s}^m (\mathbf{a}^{m-1})^T \quad \text{Ec 36}$$

$$\mathbf{b}^m(k+1) = \mathbf{b}^m(k) - \alpha \mathbf{s}^m \quad \text{Ec 37}$$

Donde

$$\mathbf{s}^m = \frac{\partial \hat{F}}{\partial \mathbf{n}^m} = \begin{bmatrix} \frac{\partial \hat{F}}{\partial n_1^m} \\ \vdots \\ \frac{\partial \hat{F}}{\partial n_{S^m}^m} \end{bmatrix}$$

Ahora que hemos visto como calcular las sensibilidades, necesitamos aplicar otra vez la regla de la cadena para calcular la relación de la sensibilidad de la capa siguiente con respecto a la sensibilidad de la capa actual, de aquí viene el nombre del algoritmo Retropropagación.

Para derivar la relación entre las sensibilidades usaremos la siguiente matriz Jacobiana

$$\frac{\partial \mathbf{n}^{m+1}}{\partial \mathbf{n}^m} = \begin{bmatrix} \frac{\partial n_1^{m+1}}{\partial n_1^m} & \cdots & \frac{\partial n_1^{m+1}}{\partial n_{S^m}^m} \\ \vdots & \ddots & \vdots \\ \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_1^m} & \cdots & \frac{\partial n_{S^{m+1}}^{m+1}}{\partial n_{S^m}^m} \end{bmatrix}$$

Ahora necesitamos encontrar una expresión para esta matriz. Considerando los elementos *i,j* de la matriz

$$\frac{\partial n_i^{m+1}}{\partial n_j^m} = \frac{\partial\left(\sum_{l=1}^{S^m} w_{i,l}^{m+1} a_l^m + b_l^{m+1}\right)}{\partial n_j^m} = w_{i,j}^{m+1} \frac{\partial a_j^m}{\partial n_j^m} \qquad \text{Ec 38}$$

$$\frac{\partial n_i^{m+1}}{\partial n_j^m} = w_{i,j}^{m+1} \frac{\partial f^m(n_j^m)}{\partial n_j^m} = w_{i,j}^{m+1} \dot{f^m}\left(n_j^m\right) \qquad \text{Ec 39}$$

Donde

$$\dot{f^m}\left(n_j^m\right) = \frac{\partial f^m(n_j^m)}{\partial n_j^m}$$

Ahora la matriz Jacobiana puede escribirse como:

$$\frac{\partial \mathbf{n}^{m+1}}{\partial \mathbf{n}^m} = \mathbf{W}^{m+1} \dot{\mathbf{F}^m}(\mathbf{n}^m) \qquad \text{Ec 40}$$

Donde

$$\dot{\mathbf{F}^m}(\mathbf{n}^m) = \begin{bmatrix} \dot{f^m}(n_1^m) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \dot{f^m}\left(n_{S^m}^m\right) \end{bmatrix}$$

Ahora escribiremos la relación recurrente de la sensibilidad usando la regla de la cadena en forma matricial:

$$\mathbf{s}^m = \frac{\partial \hat{F}}{\partial \mathbf{n}^m} = \left(\frac{\partial \mathbf{n}^{m+1}}{\partial \mathbf{n}^m}\right)^T \frac{\partial \hat{F}}{\partial \mathbf{n}^{m+1}} = \dot{\mathbf{F}^m}(\mathbf{n}^m)(\mathbf{W}^{m+1})^T \frac{\partial \hat{F}}{\partial \mathbf{n}^{m+1}}$$

$$\mathbf{s}^m = \dot{\mathbf{F}^m}(\mathbf{n}^m)(\mathbf{W}^{m+1})^T \mathbf{s}^{m+1} \qquad \text{Ec 41}$$

Las sensibilidades se propagan hacia atrás a través de la red desde la última hasta la primera capa.

$$s^M \to s^{M-1} \to \cdots \to s^2 \to s^1$$

Solo hay un paso más para completar el algoritmo de Retropropagación. Necesitamos el punto inicial $\mathbf{s}^M$. Esta se obtiene de la última capa.

$$s_i^M = \frac{\partial \hat{F}}{\partial n_i^M} = \frac{\partial (\mathbf{t}-\mathbf{a})^T(\mathbf{t}-\mathbf{a})}{\partial n_i^M} = \frac{\partial \sum_{j=1}^{S^M}(t_j - a_j)^2}{\partial n_i^M} = -2(t_i - a_i)\frac{\partial a_i}{\partial n_i^M}$$

Ya que

$$\frac{\partial a_i}{\partial n_i^M} = \frac{\partial a_i^M}{\partial n_i^M} = \frac{\partial f^M(n_i^M)}{\partial n_i^M} = \dot{f}^M(n_i^M)$$

Que puede escribirse como

$$s_i^M = -2(t_i - a_i)\dot{f}^M(n_i^M) \quad \text{Ec 42}$$

Que en forma matricial

$$\mathbf{s}^M = -2\dot{\mathbf{F}}^M(\mathbf{n}^M)(\mathbf{t} - \mathbf{a}) \quad \text{Ec 43}$$

Como resumen el algoritmo de Retropropagación es:

1) Iniciar de manera aleatoria $\mathbf{W}^m\ y\ \mathbf{b}^m\ \forall m$ con valores pequeños.
2) Propagar las entradas a través de la red

$$\mathbf{a}^0 = \mathbf{p} \quad \text{Ec 44}$$

$$\mathbf{a}^{m+1} = \mathbf{f}^{m+1}(\mathbf{W}^{m+1}\mathbf{a}^m + \mathbf{b}^{m+1})\ para\ m = 0, 1, \dots, M-1 \quad \text{Ec 45}$$

$$\mathbf{a} = \mathbf{a}^M \quad \text{Ec 46}$$

3) Propagar las sensibilidades desde la última capa a la primera

$$\mathbf{s}^M = -2\dot{\mathbf{F}}^M(\mathbf{n}^M)(\mathbf{t} - \mathbf{a}) \quad \text{Ec 47}$$

$$\mathbf{s}^m = \dot{\mathbf{F}}^m(\mathbf{n}^m)(\mathbf{W}^{m+1})^T\mathbf{s}^{m+1}\ para\ \ m = M-1, \dots, 2, 1 \quad \text{Ec 48}$$

4) Por último, actualizar los pesos y los sesgos usando el algoritmo de pasos descendentes

$$\mathbf{W}^m(k+1) = \mathbf{W}^m(k) - \alpha\mathbf{s}^m(\mathbf{a}^{m-1})^T \quad \text{Ec 49}$$

$$\mathbf{b}^m(k+1) = \mathbf{b}^m(k) - \alpha\mathbf{s}^m \quad \text{Ec 50}$$

Las funciones de activación más usadas en el algoritmo de Retropropagación son la función Logarítmica Sigmoidal, Tangente Hiperbólica y Lineal:

Para la función de activación Logarítmica Sigmoidal

$$a = \frac{1}{1 + e^{-n}}$$

$$\dot{f}^M(n) = \frac{d}{dn}\left(\frac{1}{1+e^{-n}}\right) = \frac{e^{-n}}{(1+e^{-n})^2} = \left(1 - \frac{1}{1+e^{-n}}\right)\left(\frac{1}{1+e^{-n}}\right) = (1-a)(a)$$

$$\dot{f}^M(n) = (1-a)(a) \quad \text{Ec 51}$$

Para la función de activación Tangente Hiperbólica

$$a = \frac{e^n - e^{-n}}{e^n + e^{-n}}$$

$$\dot{f}^M(n) = \frac{d}{dn}\left(\frac{e^n - e^{-n}}{e^n + e^{-n}}\right) = 1 - \left(\frac{e^n - e^{-n}}{e^n + e^{-n}}\right)^2 = 1 - a^2$$

$$\dot{f}^M(n) = 1 - a^2 \qquad \text{Ec 52}$$

Para la función de activación lineal

$$a = n$$

$$\dot{f}^M(n) = \frac{d}{dn} n = 1$$

$$\dot{f}^M(n) = 1 \qquad \text{Ec 53}$$

Capítulo 3. Diseño de la Red Neuronal Artificial y descripción del sistema de identificación

3.1 Parámetros de diseño de la Red Neuronal Artificial

En la entrevista con el usuario se identificaron los siguientes requerimientos:

a) Identificar el ápice del ajo.
b) Orientar el ápice del ajo.
c) Identificar semillas de tamaño mediano y grande

Que traducidos a especificaciones:

Especificación	Unidad	Rango
Identificación	% de identificación	[75, 100]
Orientación	% de aciertos	[75, 100]
Tamaño de la semilla	cm	[3, 5]

Tabla 3.1 Especificaciones de la RNA

Los parámetros de diseño de la RNA son los siguientes:

- Imágenes binarias con codificación [-1,1] de 80 x 46 píxeles, dando un vector de entradas $\mathbf{p}_{3680\text{ x }1}$ elementos.
- Se ocuparon dos capas M=2, con 350 elementos procesadores en la capa oculta y 3 en la capa de salida.

$$S^1 = 350\,, S^2 = 3$$

Todas las funciones de activación serán tangentes hiperbólicas (sigmoidea).

$$f^1 = f^2 = a = \frac{e^n - e^{-n}}{e^n + e^{-n}}$$

Se ocupó el algoritmo de retropropagación como regla de entrenamiento, los parámetros de entrenamiento fueron:

Taza de aprendizaje	α=0.001
Error cuadrático medio	e<0.01
Épocas	Épocas>750

Tabla 3.2 Parámetros de entrenamiento de la RNA

Se manejó una taza de aprendizaje constante, con ésta el algoritmo no oscilaba, es decir el error cuadrático medio no subía y bajaba con cada época.

El entrenamiento de la red se detenía al llegar a, a) el error cuadrático medio fuera igual o menor a 0.01 o b) el número de épocas fuera igual a 750.

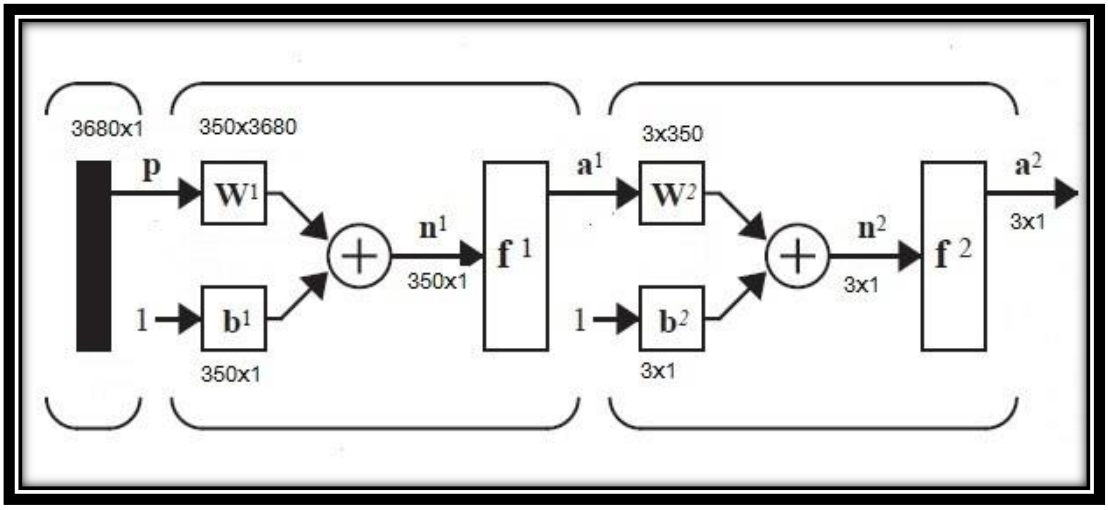

Figura 3.1 Parámetros de la Red Neuronal

- La salida de la red está dada de la siguiente forma

$[-1 \; -1 \; \; 1]^T$ $[-1 \; \; 1 \; -1]^T$ $[\; 1 \; -1 \; -1]^T$

Figura 3.2 Salidas de la RNA dependiendo de las imágenes de entrada

Algunas de las imágenes ocupadas en el entrenamiento se muestran en el Anexo B.

El programa para obtener la base de entrenamiento guarda dos bases de entrenamiento, una con una codificación [0,1] llamada P0 y otra con codificación [-1,1] llamada P1, realizando pruebas se vio que la codificación de la base de entrenamiento P1 obtenía mejores resultados que P0, por lo tanto, se trabajó con ella.

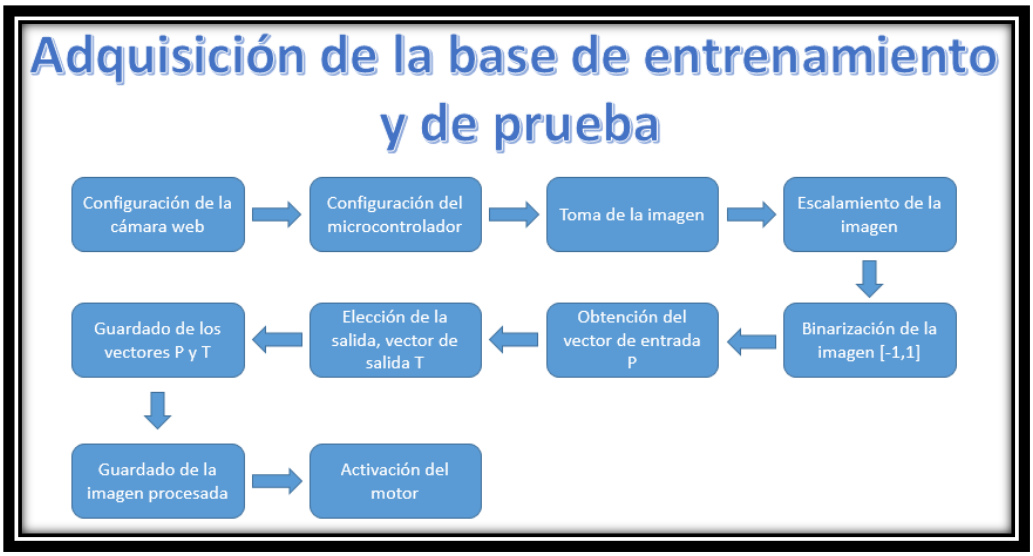

Figura 3.3 Diagrama de bloques del programa de adquisición de la base de entrenamiento y base de prueba

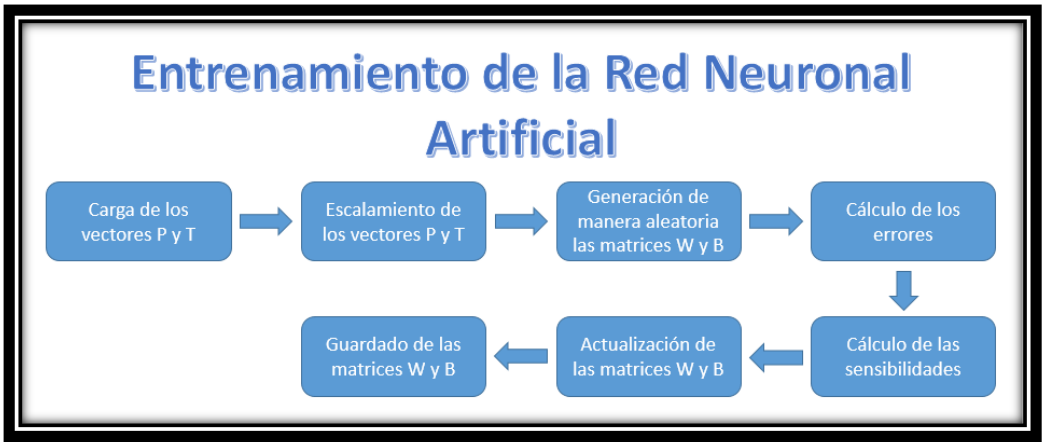

Figura 3.4 Diagrama de bloques del entrenamiento de la RNA

Se notó que los valores iniciales de las matrices de pesos W11, W21 y vectores de sesgos o polarizaciones b11 y b21 debían ser pequeños en el orden de 0.1 para que el algoritmo convergiera a una solución.

En la Figura 3.5 se puede ver como el error cuadrático medio disminuía en cada época de entrenamiento, llegando al error mínimo de 0.01; se puede ver que el error cuadrático medio decrecía de manera asintótica, reiterando que la taza de aprendizaje escogida fue la correcta para evitar oscilaciones en el entrenamiento.

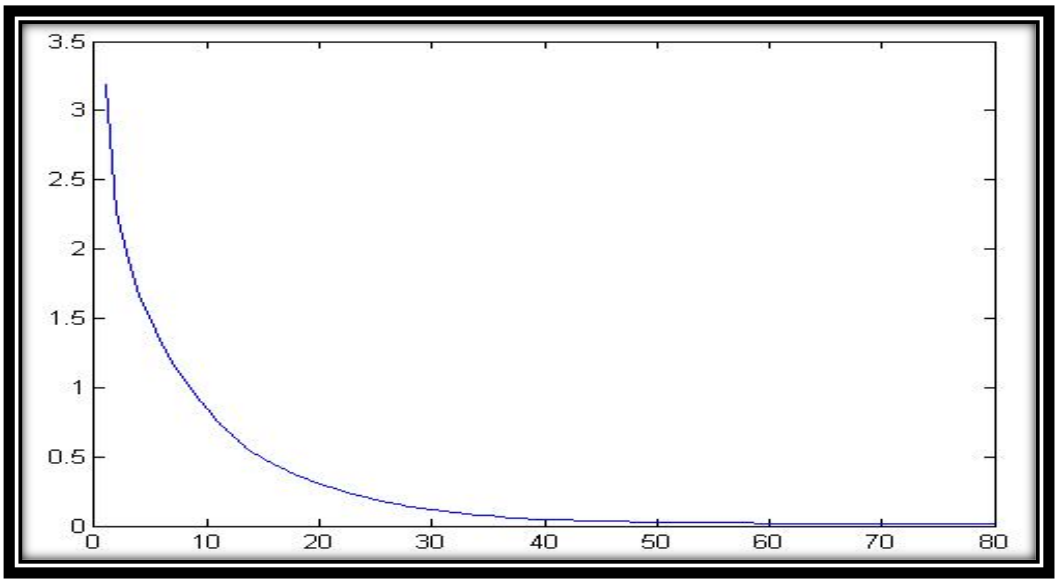

Figura 3.5 Gráfica del error cuadrático medio al momento de entrenar la red.

Se realizaron dos programas más para probar la eficacia de la RNA, el primero prueba la base de entrenamiento, el segundo prueba la RNA con 100 imágenes de prueba, los resultados se muestran el capítulo 5.1.1. Un tercer programa ayuda a medir la eficacia del sistema de orientación, los resultados de este experimento se muestran en el capítulo 5.1.2.

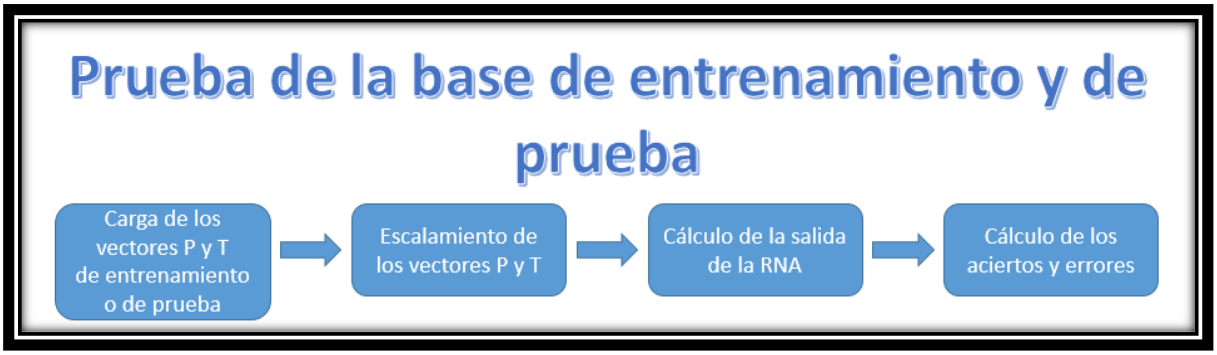

Figura 3.6 Diagrama de bloques del programa para medir la eficacia de la RNA

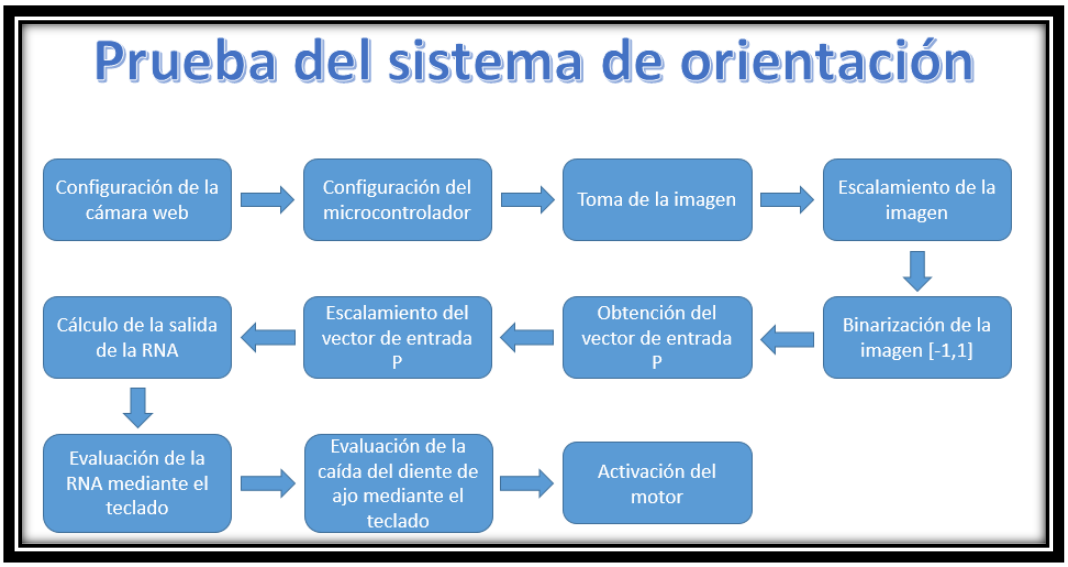

Figura 3.7 Diagrama de bloques del programa de prueba del sistema de orientación

3.2 Descripción del sistema de identificación

3.2.1 Sistema de visión artificial

El sistema de visión artificial consta de los siguientes elementos:

- Cámara Web Perfect Choice (Figura 3.8).

Figura 3.8 Cámara usada en el sistema de visión.

Característica	**Especificación**
Marca	Perfect Choice
Modelo	Cámara Web S-Vision
Alimentación	5 Vcc 150 mA
Resolución	SVGA 800 x 600 pixeles
Conector	USB
Compatibilidad	Windows® XP / Vista / 7 / 8/ 8.1
Rango de muestreo	30 fps (cuadros por segundo)
Sensor	CMOS

Tabla 3.3 Especificaciones de la cámara Fuente: PerfectChoice

- La base de la cámara se modificó para que ésta quedará fija en el sistema de control (Figura 3.9).

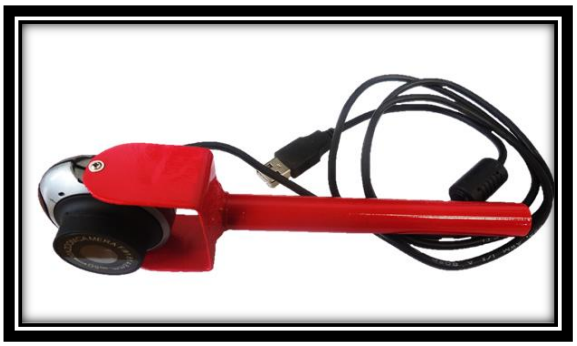

Figura 3.9 Modificación de la base de la cámara.

- Lámparas LED infrarrojas (Figura 3.10) de $12V_{DC}$, éstas se ocuparon para iluminar el rango de visión y así tomar imágenes claras y sin interferencia de sombras provocadas por luz visible.

Figura 3.10 Lámparas LED infrarrojas

3.2.2 Sistema de orientación

El sistema de orientación está conformado por el mecanismo de corredera (Figura 3.11) y el motor de $12V_{DC}$, el fin de este sistema es depositar el ajo dependiendo de la salida obtenida de la RNA.

Figura 3.11 Mecanismo de corredera del sistema de orientación.

3.2.3 Sistema de control

El sistema de control está conformado por el circuito de control y una placa para proyectos Arduino UNO (Figura 3.12), que sirve como interfaz entre la máquina y el circuito antes mencionado.

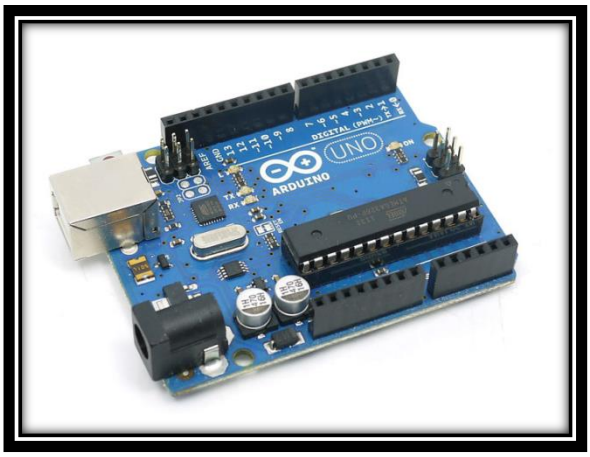

Figura 3.12 Arduino UNO

El circuito está conformado por dos botones, uno de encendido/apagado, el otro un botón de Stop con sus respectivos indicadores LED y una etapa de potencia conformada por un puente H L293D.

A partir de la salida obtendida por la RNA la placa Arduino UNO manda una señal de control al puente H, el cual mueve el motor en un sentido o en otro, o no moverse cuando no se detecte un diente de ajo. Si el botón de Stop está activado el programa de la RNA no funcionará hasta que éste se desactive.

Se diseñó la siguiente PCB, para dejar el circuito usado en el sistema de manera soldada y así evitar que algún componente se salga con la vibración de la máquina, también para dejar el trabajo de manera más profesional.

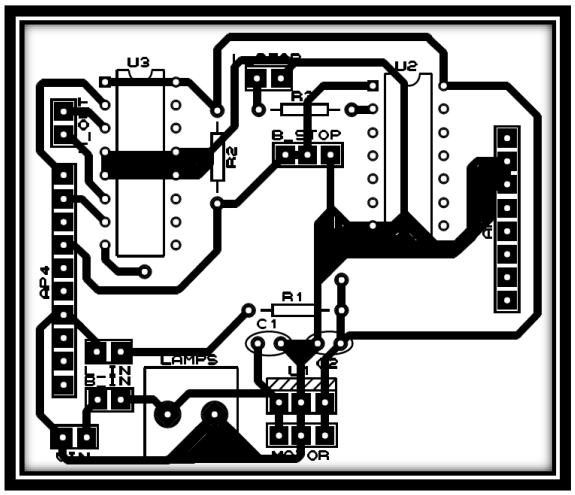

Figura 3.13 PCB del circuito usado

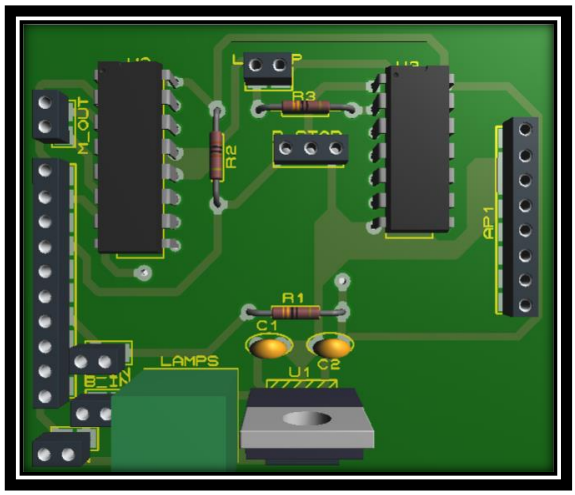

Figura 3.14 Vista 3D de la PCB

Se ocupó un software de simulación de circuitos para el diseño de la PCB, el fin de este diseño fue que el circuito embonara sobre la placa Arduino UNO ahorrando así espacio y evitando conexiones innecesarias (Figura 3.13 y 3.14).

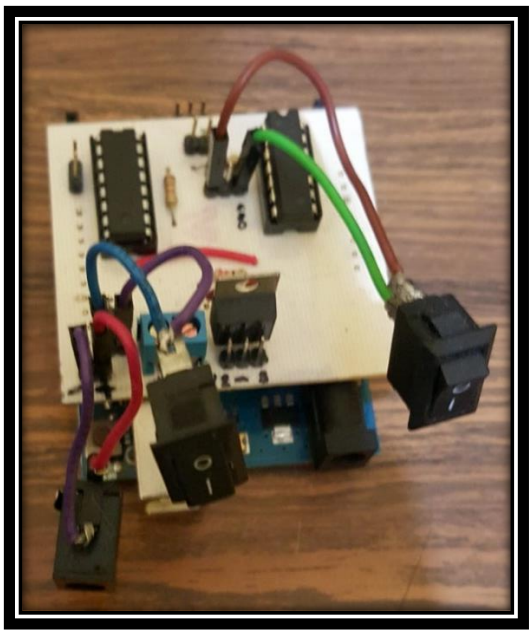

Figura 3.15a Vista frontal de la PCB

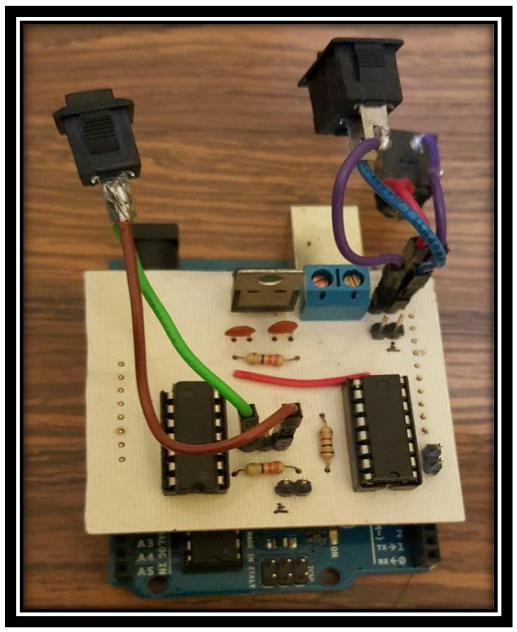

Figura 3.15b Vista Trasera de la PCB

Se diseñó una caja para proteger al circuito, ésta se hizo de MDF cortado por láser (Figura 3.16).

Figura 3.16 Caja protectora del circuito

El circuito consta de dos botones, el botón con el led rojo es el que detiene la comunicación con el Arduino y el programa, el segundo es el botón de encendido, por la parte trasera de la caja salen todos los cables necesarios para el control del sistema, el circuito es alimentado por un cargador de $12V_{DC}$.

En la figura 3.17 se muestra el sistema completo.

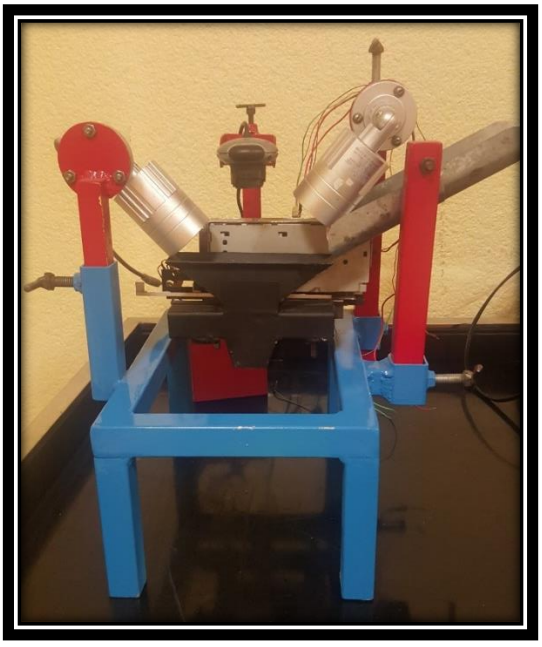

Figura 3.17 Sistema para la identificación del ápice de un diente de ajo.

3.3 Diseño de la interfaz gráfica

Como parte final del diseño de la red, se diseñó una interfaz. La interfaz creada se puede ver en las figuras 3.18 y 3.20.

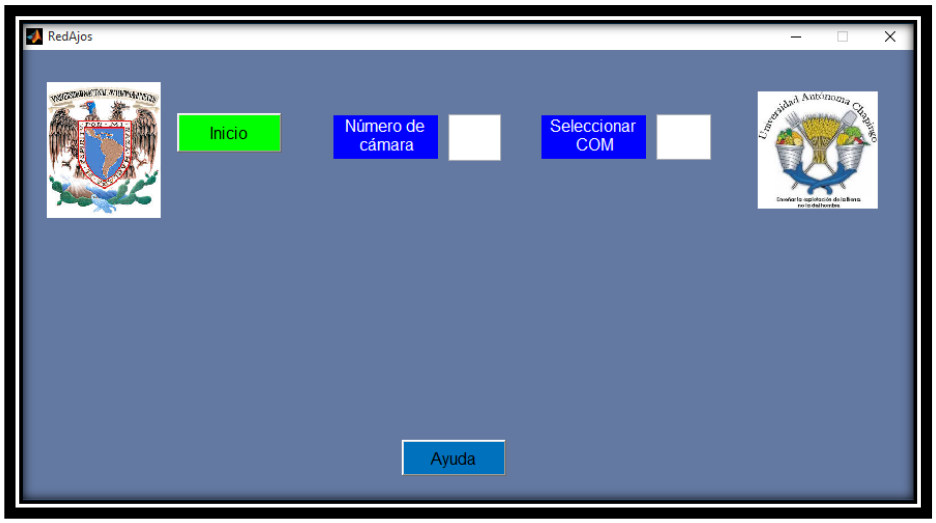

Figura 3.18 Interfaz de inicio.

Al momento de ejecutar la interfaz se pedirá el número de identificador de la cámara, en este el caso de la computadora usada es el 2, el segundo parámetro es el número del puerto COM en el cual está conectado Arduino. Si alguno de estos valores faltara al momento de hacer clic en el botón de inicio, se le envía un mensaje de error al usuario pidiendo que los llene.

Figura 3.19 Mensaje de error mostrado al faltar algún valor en los cuadros de texto.

Sí todo es correcto, la interfaz cambia de aspecto como se muestra a continuación.

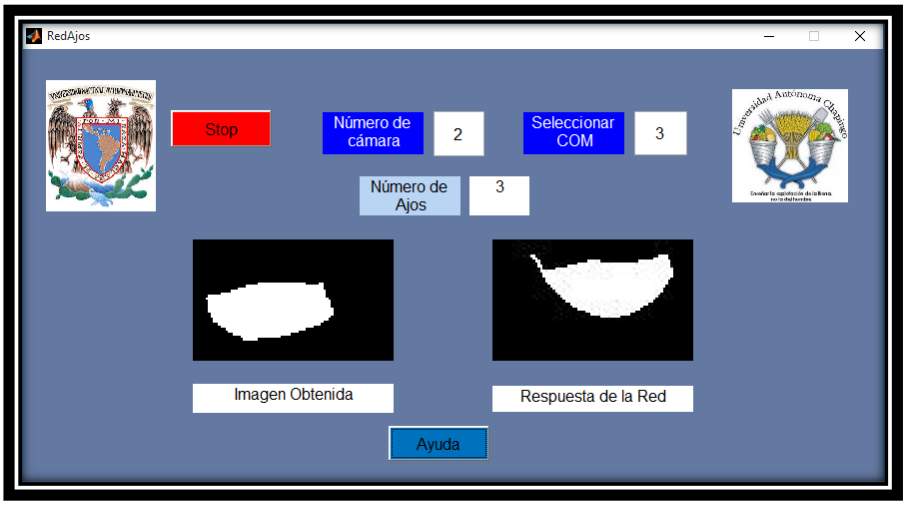

Figura 3.20 Interfaz después de haber dado click en el botón Inicio.

En ésta, las cajas de texto de "Número de cámara" y "Seleccionar COM" se deshabilitan, mostrando únicamente el valor que ha sido proporcionado, se muestra, además, el número de fotografías tomadas en la etiqueta "Número de Ajos", la imagen obtenida por la cámara y la respuesta calculada por la red neuronal se muestran al usuario.

En caso de que se quiera suspender el ciclo que hace trabajar a la RNA existen dos formas, dando clic en el botón Stop de la interfaz gráfica o accionando el switch de stop del circuito.

Cuando se detiene el ciclo de identificación de dientes de ajo, la interfaz que se despliega es la siguiente.

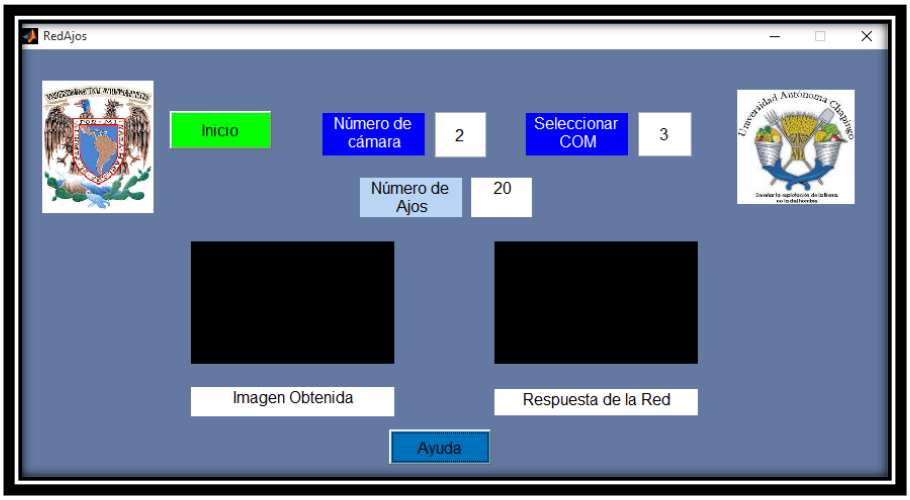

Figura 3.21 Interfaz después de detener el proceso.

Al momento de hacer clic en el botón de ayuda, se abrirá el archivo ayuda.pdf; el cual muestra como ocupar la interfaz y como resolver los problemas que se pudieran llegar a presentar (Anexo A).

A continuación, se presenta el diagrama de bloques del sistema completo.

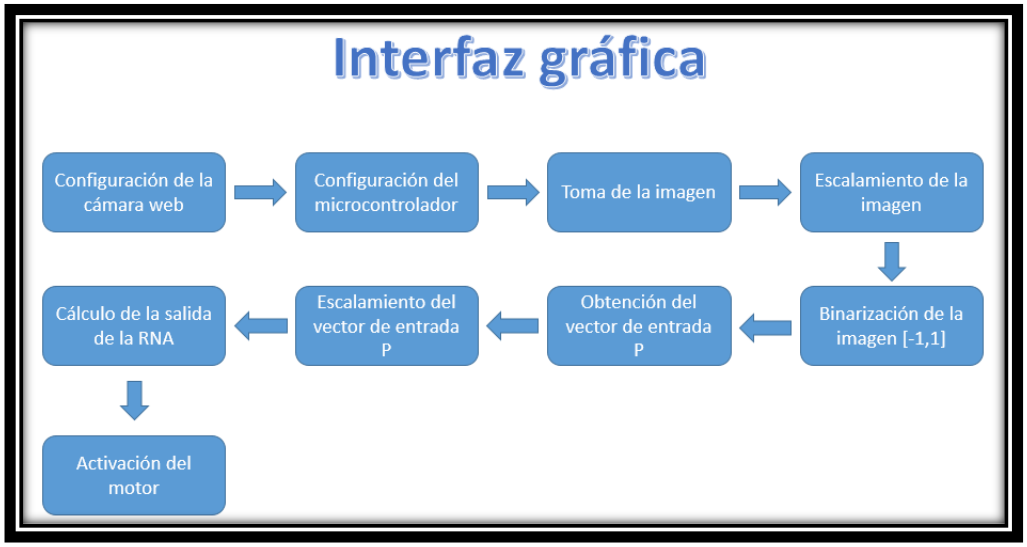

Figura 3.22 Diagrama de bloques del sistema de identificación

Capítulo 4. Materiales y métodos

Para medir la eficacia del sistema de identificación se implementaron tres tipos de experimentos, el primero sólo mide la eficacia en que la RNA identifica los dientes de ajo, el segundo mide la eficacia del sistema completo, es decir si la RNA identifica los ajos y caen bien, el último experimento es una corrida de prueba para documentar el funcionamiento de la máquina sembradora de diente de ajos en general.

4.1 Materiales

En cada uno de los tres experimentos se ocuparon los siguientes materiales:

- Ajos
- Computadora marca Toshiba, Intel core i5, 2.6 GHz ,4GB RAM.
- Placa Arduino UNO.
- Sistema de identificación.
- Máquina estacionaria sembradora de ajos.
- Bandeja con capacidad de 6 ajos con sustrato.
- Circuito de control.
- Matrices de pesos y sesgos calculados.

Figura 4.1 Ajos usados en las pruebas.

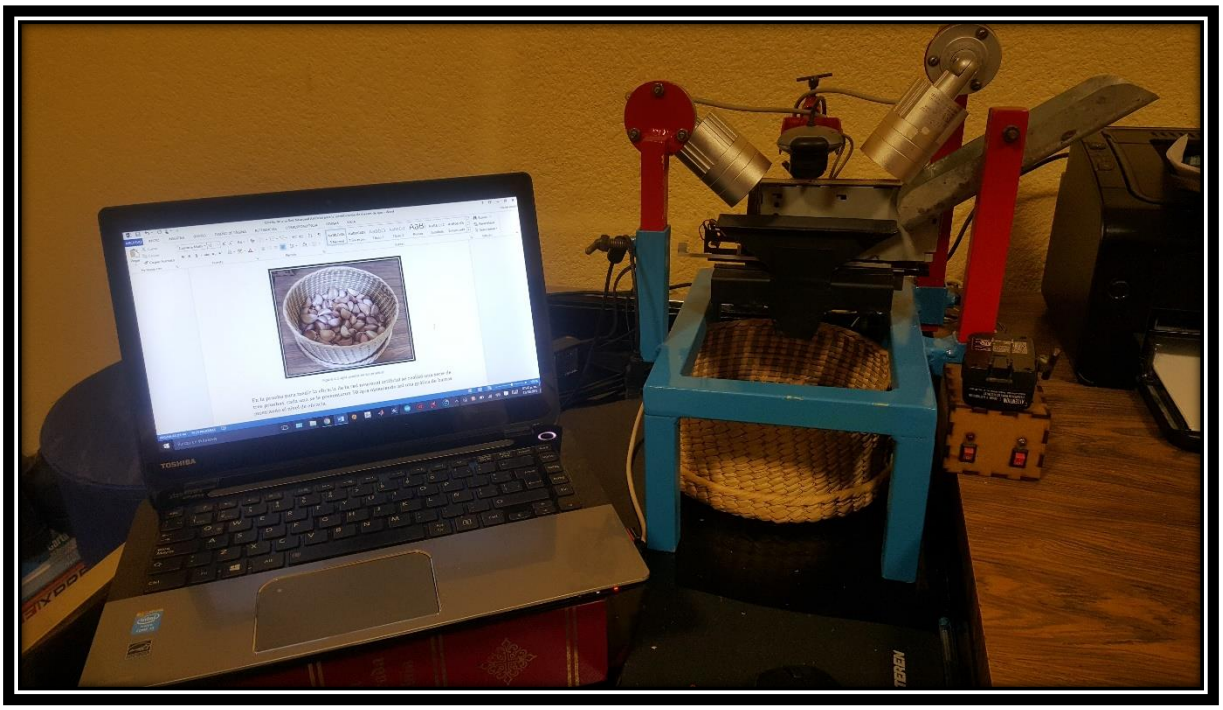

Figura 4.2 Materiales ocupados en el primer experimento

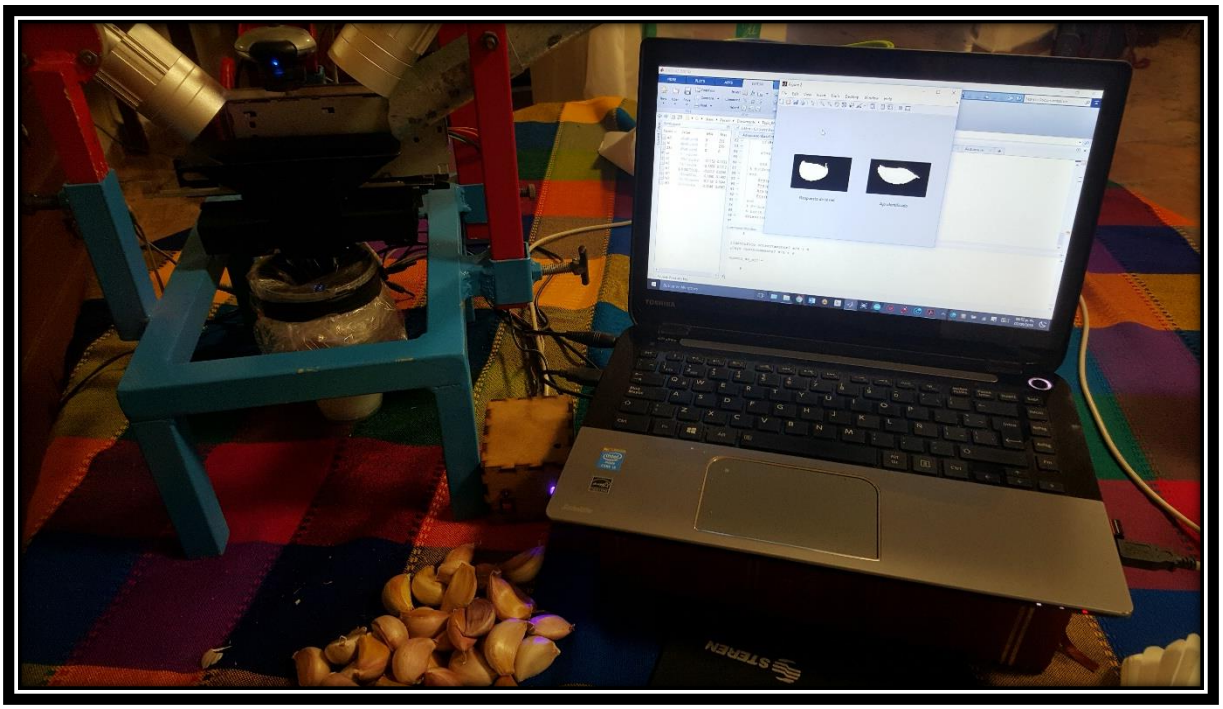

Figura 4.3 Materiales ocupados en el segundo experimento.

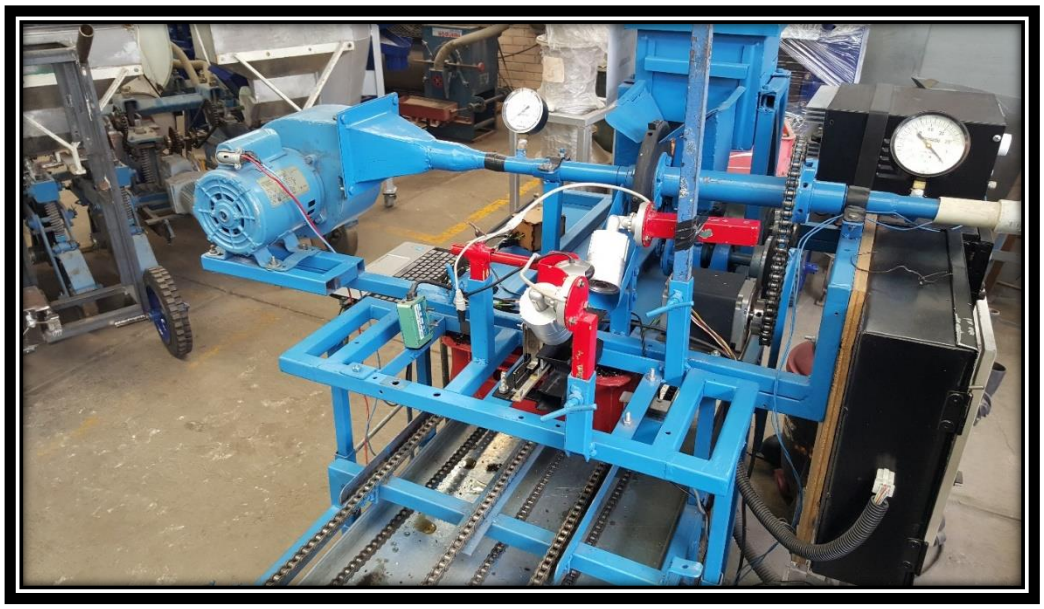

Figura 4.4 Materiales ocupados en el experimento con la máquina sembradora de dientes de ajo.

Figura 4.5 Bandeja usada durante el experimento de la máquina sembradora de diente de ajos.

4.2 Métodos

Para el primer experimento, el cual consta en el cálculo de la eficacia que tiene la RNA, se realizó un programa, donde se le presenta la base de prueba, la cual consiste en 100 imágenes que la RNA nunca vio, al final se obtiene el porcentaje de aciertos y fallos que tuvo la red al identificar la base de prueba. Esto se realiza con 4 bases de datos distintas.

Para el segundo experimento, se medirá la eficacia del sistema de orientación, usando el programa explicado con el diagrama de bloques de la Figura 3.7, se ocuparon 50 ajos por prueba, haciendo un total de tres pruebas.

El programa muestra la imagen obtenida y según la salida calculada muestra una imagen parecida a las mostradas en la Figura 3.2, esto para facilitar la evaluación de la red.

Un tercer y último experimento se realizó con la máquina sembradora de diente de ajos, ésta se encuentra en las instalaciones de la Universidad Autónoma Chapingo, en el taller del Departamento de Ingeniería Agrícola (DIMA), para ver como trabajaba el sistema de visión y orientación con los demás sistemas que integran la máquina.

El primer pasó fue sincronizar la interfaz de usuario con la máquina, después el proceso comenzaba, la bandeja entraba por el lado posterior de la máquina, el transportador, compuesto por una serie de cadenas, llevaba la bandeja hasta el sensor de presencia, la bandeja continuaba avanzando hasta que otro sensor encargado de detectar un agujero en el sustrato mandaba una señal que accionaban el giro del motor del sistema de alimentación, el diente de ajo se dejaba caer en el sistema de visión, la cámara tomaba la fotografía y la RNA procesaba los datos de entrada, tomando la decisión de accionar el motor como se ha descrito en el capítulo 3, posterior a esto, se checaba si el diente de ajo fue correctamente identificado y el sistema de orientación dejaba el ápice del ajo apuntando hacia arriba.

Capítulo 5. Resultados y análisis de resultados

5.1 Resultados

5.1.1 Primer experimento

Del experimento número 1 se obtuvieron los siguientes resultados

Prueba	Aciertos [# ajos]	Aciertos [%]	Fallos [# ajos]	Fallos [%]
1	84	84	16	16
2	89	89	11	11
3	88	88	12	12
4	84	84	16	16
Promedio		86.3		13.7

Tabla 5.1 Resultados obtenidos del primer experimento

A continuación, se presentan algunas imágenes obtenidas durante el experimento (Figuras 5.1)

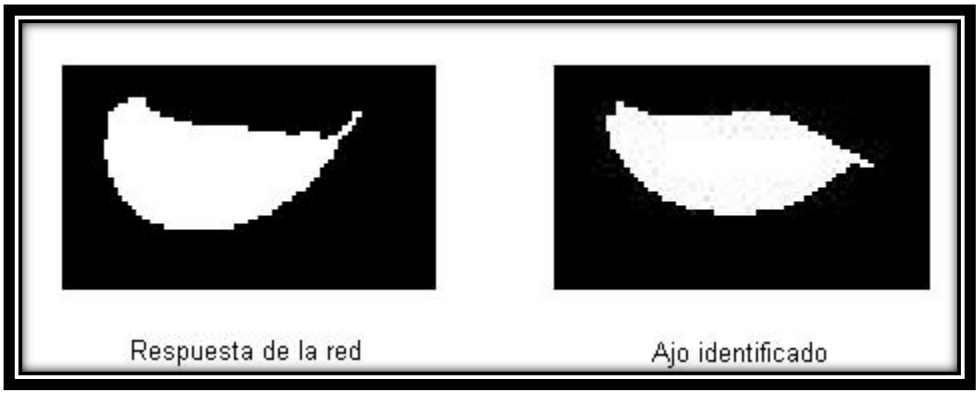

Figura 5.1 a) Ajo con el ápice apuntando hacia la derecha

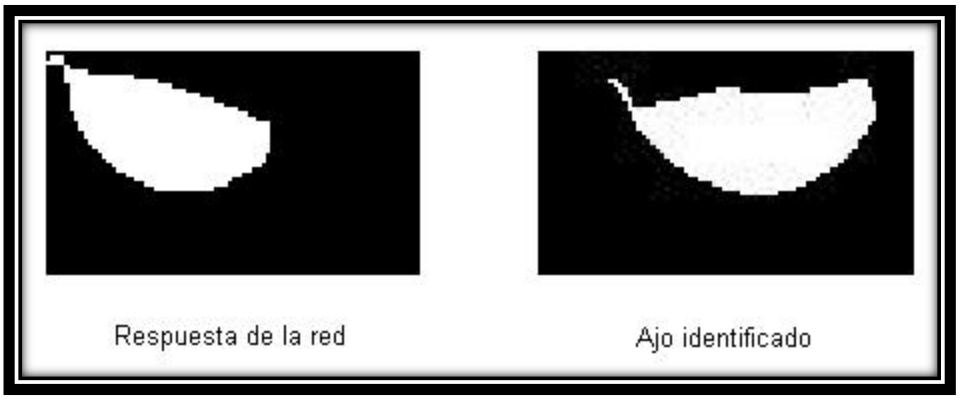

Figura 5.1 b) Ajo identificado con el ápice apuntando hacia la izquierda

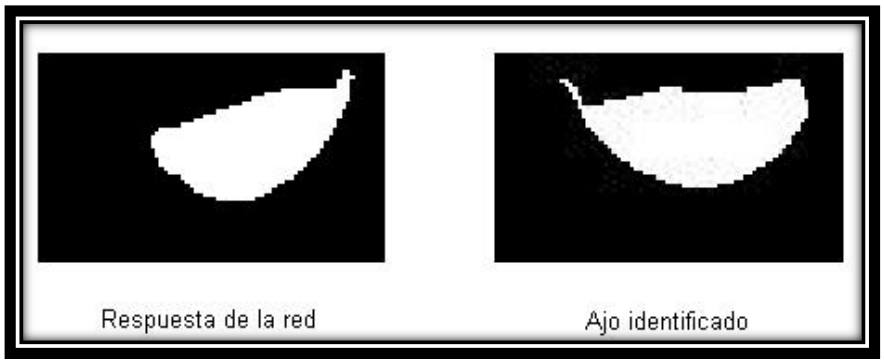

Figura 5.1 c) Ajo no identificado

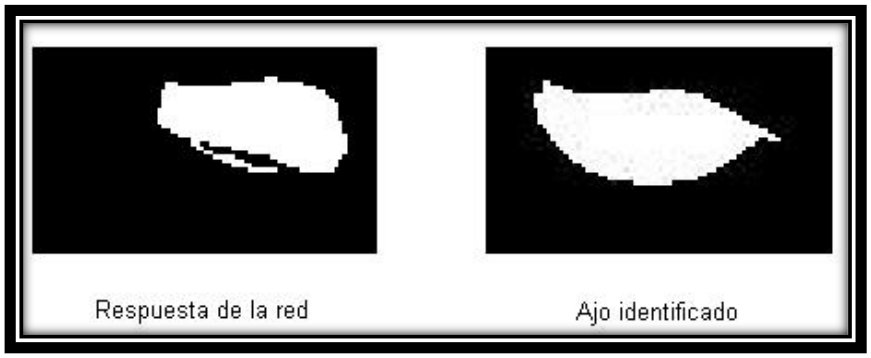

Figura 5.1 d) Ajo identificado, aunque la imagen no esté bien definida

5.1.2 Segundo experimento

Los resultados obtenidos del segundo experimento son los siguientes:

Prueba	Aciertos RNA[%]	Fallos RNA[%]	Aciertos Sistema de Orientación [%]	Fallos Sistema de Orientación [%]
1	82	18	78	22
2	84	16	82	18
3	80	20	78	22
Promedio	82	18	79.3	20.7

Tabla 5.2 Resultados obtenidos del segundo experimento

En las Figuras 5.2 se muestran los casos obtenidos durante el experimento, éstos son 4, 1) la red identificaba el diente de ajo y lo depositaba bien, 2) la red no identificaba el ajo y lo depositaba mal, 3) la red identificaba el diente, pero caía mal y 4) la red no identificaba el diente, pero caía bien.

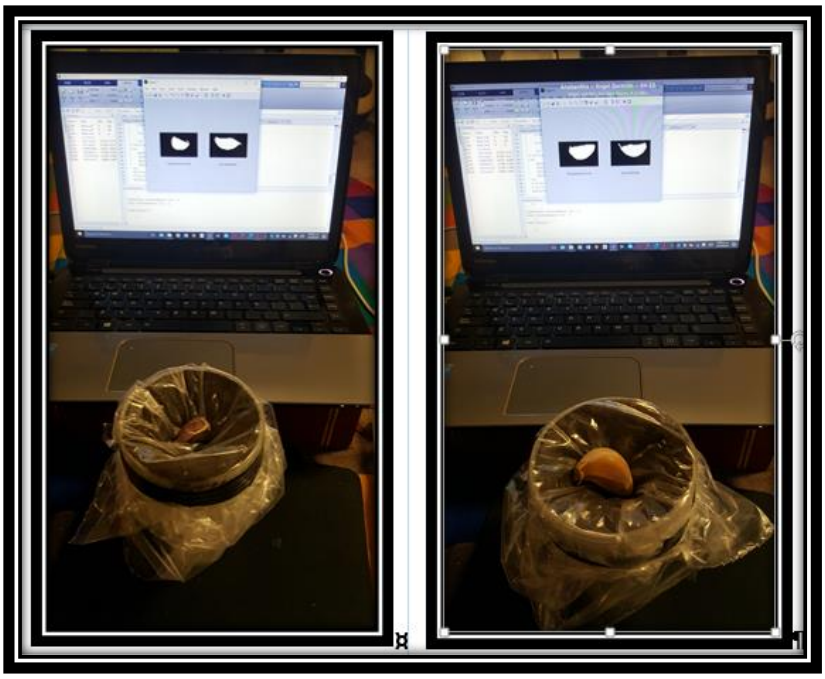

Figura 5.2 a) Casos 1 y 2 obtenidos en el segundo experimento.

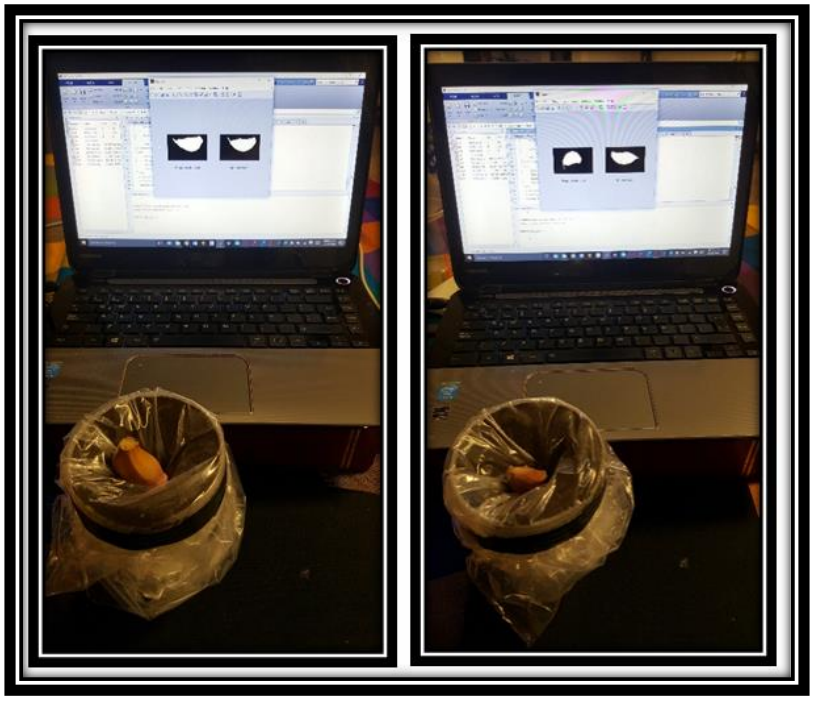

Figura 5.2 b) Casos 3 y 4 obtenidos en el segundo experimento.

5.1.3 Tercer experimento

Total de ajos	Aciertos	Fallos	Eficacia
30	24	6	80%

Tabla 5.3 Resultados del tercer experimento

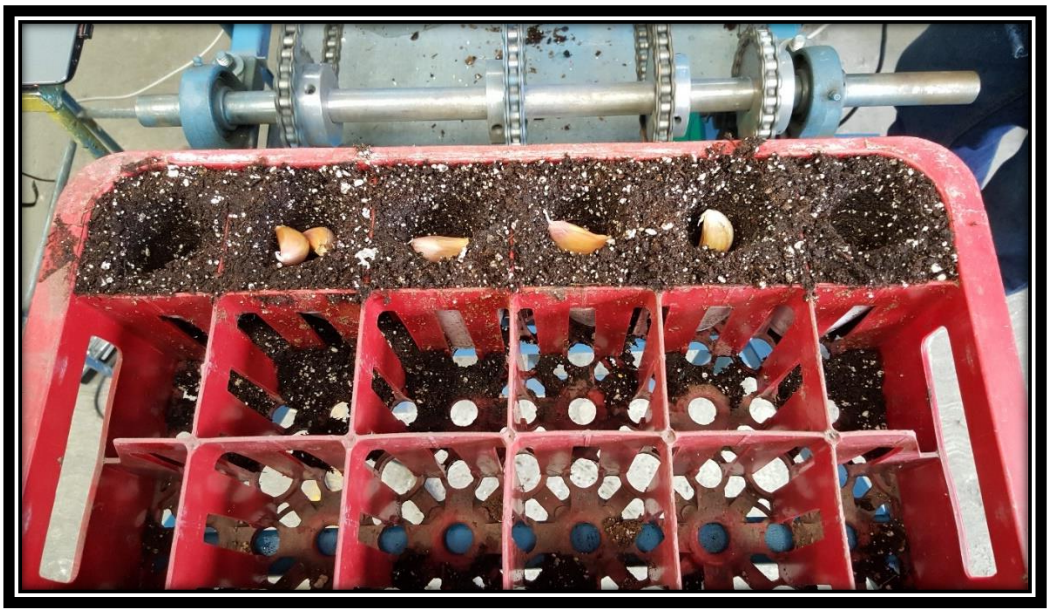

Figura 5.3 a) Resultados obtenidos

Figura 5.3 b) Resultados obtenidos

Figura 5.3 c) Resultados obtenidos

Figura 5.3 d) Resultados obtenidos

5.2 Análisis de los resultados

Del primer experimento se obtuvo una eficacia del 86.3% superando la eficacia mínima pedida por el usuario y cercano a la taza de identificación del experimento explicado en el estado del arte con 83.3% de eficacia en el reconocimiento de los dientes de ajo.

Como se pueden observar en las Figura 5.1 a) y 5.1 b) son las salidas obtenidas cuando la red identificaba la dirección del ápice del diente de ajo, en la Figura 5.1 c) la red tuvo problemas al identificar el diente de ajo, este problema puede ser resuelto presentando a la red un mayor número de patrones de entrada y así hacerla más robusta. En el caso de la Figura 5.1 d) se demuestra el poder que tienen las redes neuronales artificiales de obtener una respuesta, aunque la entrada de ésta esté o no bien definida.

Para el segundo experimento se obtuvo un promedio general del 82% de eficacia de la RNA y 79.3% de eficacia del sistema de orientación; éste tiene menos eficiencia que la RNA porque, en algunos casos, la red identificaba el diente de ajo y éste caía mal o viceversa, esto se esperaba ya que la caída del diente depende de su momento de inercia y no todos los ajos probados tenían el mismo momento, en los casos en que el diente caía en una posición diferente a la calculada por la RNA fue porque éste continuaba girando, otra razón de esta diferencia es que ningún sistema tiene una eficiencia de 1 o 100%, en otras palabras los dos sistemas nunca tendrán el mismo número aciertos.

Para el tercer experimento los resultados obtenidos demostraron una eficacia del 80%, lo que reitera que la RNA y el sistema de orientación trabajan con una eficacia superior a la pedida por el usuario.

Capítulo 6. Conclusiones y trabajo a futuro

6.1 Conclusiones

La Red Neuronal Artificial diseñada para la identificación del ápice de dientes de ajo demostró tener la eficacia mínima pedida en los requerimientos del proyecto (75%), ésta se podría mejorar presentando más imágenes de entrenamiento o probando otro tipo de regla de entrenamiento, pero recordemos que las RNA difícilmente obtienen una eficacia de 100%, ya que son métodos de solución aproximados.

También la eficacia de reconocimiento de la RNA varía entre un 82% y un 86.3%, esto por las múltiples formas de los dientes de ajo, la manera en la que éstos caen y la iluminación, en algunos casos, el diente tenía zonas más oscuras que otras provocando que la imagen binaria tuviera huecos, en algunos casos la RNA los identificaba y en otro no, esto se explica por las características que tienen las RNA a generalizar los patrones de entrada y asociarlos con algún patrón que aprendió en el entrenamiento.

El sistema de orientación también demostró ser apto para realizar su tarea, logrando una eficiencia del 80%, como se explicó en el análisis de los resultados esto se debió a que los dientes de ajos, al momento de caer seguían girando producto de la inercia rotacional, logrando así describir los casos explicados en el capítulo 5.1.2.

Hablando del funcionamiento de los sistemas de visión y orientación trabajando con la máquina sembradora de dientes de ajo, éstos funcionaron correctamente como lo demuestran las pruebas, el único problema encontrado fue la sincronización de todos los sistemas restantes (sistema de alimentación y sistema de transporte de la bandeja), actualmente esos dos sistemas y el sistema de visión se sincronizan por tiempo haciendo que algunas veces los ajos no caigan en el agujero o caigan dos dientes de ajo, algunas soluciones a estos problemas se plantean en el trabajo a futuro del proyecto.

La creación de la interfaz gráfica de usuario tiene la finalidad de hacer amena la interacción del sistema completo de visión con el usuario, mostrando en cada iteración la imagen tomada por la cámara y la respuesta obtenida por la RNA, así como el conteo de los dientes de ajos procesados hasta ese momento; el diseño de la interfaz es sencillo y si hubiera problemas al momento de operarla, el manual de usuario pudiera

ser de gran ayuda, sino fuera suficiente, con el correo electrónico disponible en el manual pueden contactar al diseñador.

Como conclusión final, quedó demostrado que las RNA son una muy buena herramienta para resolver problemas asociados a reconocer imágenes y que, la finalidad de este tipo de proyectos es generar una tecnología, de buena calidad y sobre todo diseñado y hecho por mexicanos para resolver problemas de índole nacional e internacional.

6.2 Trabajo a futuro

Como se mencionó en las conclusiones, para mejorar la eficacia de la red, presentar más imágenes a la red para hacerla más robusta y lograr sincronizar los sistemas de visión y orientación con los sensores dispuestos en la máquina actualmente, especialmente en el sensor que detecta el agujero en la bandeja, el cual mandaría la señal al sistema de alimentación y al sistema de visión y orientación.

Rediseñar el sistema de orientación para minimizar el giro del diente de ajo, cuando éste esté cayendo, así mismo migrar la RNA a otro sistema que no necesite ocupar la computadora ni mucho menos el programa usado, esto con la finalidad de hacerlo portable y ocuparlo en lugares donde una computadora no está disponible.

No depender de una computadora para el funcionamiento de la RNA, trasladando ésta a un microcontrolador y así lograr hacer un equipo más portable.

Bibliografía

[1] H. Reveles M, V. Velásquez R y L. A. Bravo G. *Tecnologías para cultivar ajo en Zacatecas*, Zacatecas México, Instituto Nacional de Investigaciones Forestales, Agrícolas y Pecuarias. INIFAP, Libro Técnico N° 1, 3-5p, 2009.

[2] Secretaria de Agricultura, Ganadería, Desarrollo Rural, Pesca y Alimentación (SAGARPA), *Aumenta producción de ajo "Hecho en México" 28.9 por ciento en 2016*, (2017, 17 marzo) http://www.sagarpa.gob.mx/Delegaciones/zacatecas/boletines/Paginas/2017B097.aspx

[3] S. J. Cabrera M. y H. Serwatowski R. *Mecanización de la siembra del ajo*. México, Acta Universitaria, Vol. 6, Nº 1. Junio, p. 67-76. 1996.

[4] A. C. García R. *El Ajo. Cultivo y Aprovechamiento.* Madrid, España. Ediciones Mundi-Prensa pp: 53-74 1990.

[5] A. López M., J. Burba L.y S. Lanzavechia, *Análisis sobre la mecanización del cultivo de ajo*, Mendoza, Argentina. Estación Experimental Agropecuaria La Consulta, Instituto Nacional de Tecnología Agropecuaria (INTA), Proyecto Ajo/INTA 104, 2012.

[6] J. Castellanos Z., P. Vargas-Tapia, J. Ojodeagua L., Alcanzar- Gonzalez, F. Mendez S., E. Alvarez-Sanchez, and A. Gardea A. *Garlic productivity and profitability as affected by seed clove size, planning density and planning method.* HortScience 39:1272-1277, 2004.

[7] H. Reveles M., V. Velásquez R. y L.A. Bravo G. *Tecnología para cultivar ajo en Zacatecas*, Zacatecas México, Instituto Nacional de Investigaciones Forestales, Agrícolas y Pecuarias. INIFAP, Libro Técnico N° 11.,63-64 p. 2009

[8] C. Castaños M., *Horticultura, Manejo Simplificado.* México Universidad Autónoma Chapingo, 2000.

[9] P. Constante Prócel N. y, A. Gordón Garcés M. "Diseño e implementación de un sistema de visión artificial para clasificación de al menos tres tipos de frutas." Tesis de Maestría, Escuela Politécnica Nacional Quito, 2015.

[10] C. Durán Acevedo M. y D. Baldovino Mercado R. "Sistema de monitoreo para detectar la madurez de productos agro-industriales a través de un sistema de olfato electrónico." Revista Colombiana de Tecnologías de Avanzada. vol. 1 no.13. 2009.

[11] N. Rojas J. y V. Vásquez V. "Predicción mediante Redes Neuronales Artificiales (RNA) de la difusividad, masa, humedad, volumen y sólidos en yacón (Smallantus sonchifolius) deshidratado osmóticamente." Scientia Agropecuaria, vol. 3 no.3. 2012

[12] N. Saldaña R., J. Serwatowski H., R.A. Aguilera H. A. Saldaña R.,O.A. Martínez J. y C. Gutiérrez V. "Localización del ápice del ajo mediante técnicas de análisis digital de imagen" Agrociencia, vol 50, no.2 , pp. 215-225,febrero-marzo 2016.

[13] B. Martín del Brío, A. Molina S. "Fundamentos de las Redes Neuronales Artificiales" en *Redes Neuronales y Sistemas Borrosos*, México 3° Edición. Alfaomega. 2007, pp:4-12.

[14] H. Hernández J., E. Hernández A., (2017, 13 Enero). Introducción a las Redes Neuronales [En línea] Disponible en: https://www.uaeh.edu.mx/docencia/P_Presentaciones/huejutla/sistemas/redes_neuronales/introduccion.pdf

[15] M. Hagan T., H. Demuth B.,M. Beale H.,O. De Jesus. "Neuron Model and Network Architectures" in *Neural Network Design*, Estados Unidos de América. 2° Edición, 2014, pp: 2-2 – 2-12.

[16] M. Hagan T., H. Demuth B.,M. Beale H.,O. De Jesus. "Widrow-Hoff learning" in *Neural Network Design*, Estados Unidos de América. 2° Edición, 2004, pp: 10-2 – 10-9.

[17] M. Hagan T., H. Demuth B.,M. Beale H.,O. De Jesus. "Backpropagation" in *Neural Network Design*, Estados Unidos de América. 2° Edición, 2004, pp: 11-2 – 11-14.

Anexos

Anexo A Manual de Usuario

El siguiente documento es para guiar al usuario del programa en caso de alguna duda o error que se llegara a presentar al momento de ejecutar la interfaz.

Elementos de la interfaz de inicio

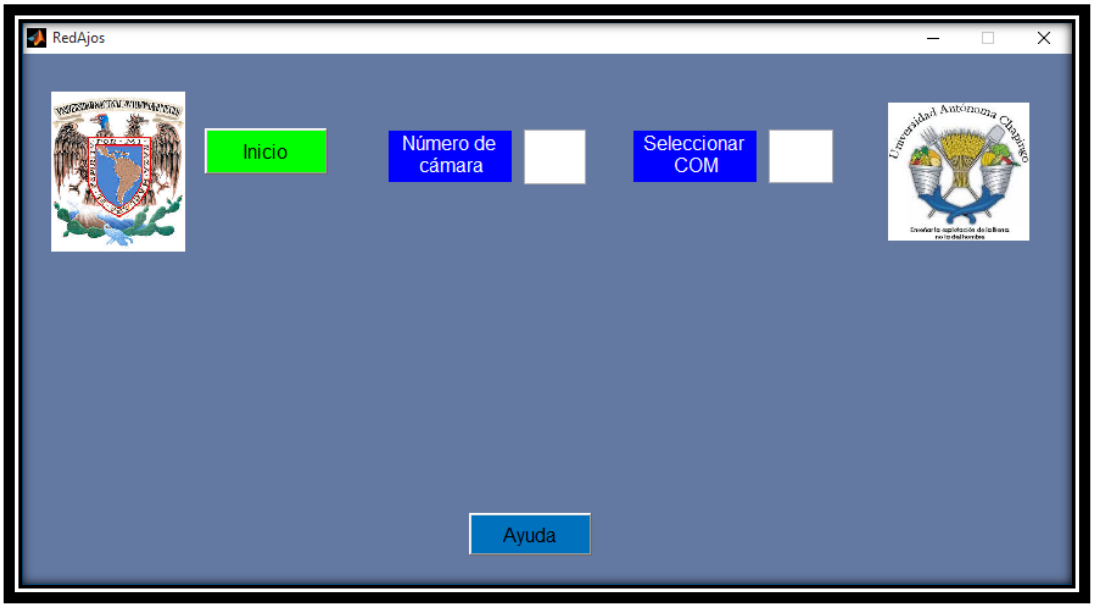

Figura 1 Interfaz de inicio del programa

La interfaz de inicio está compuesta por dos imágenes mostrando los escudos de las dos universidades participando es este proyecto, la Universidad Nacional Autónoma de México (UNAM) y la Universidad Autónoma Chapingo (UACh), el botón de Inicio, dos etiquetas, dos cuadros de diálogo y el botón de ayuda.

Para que el programa inicie es necesario llenar los cuadros de diálogo, el número de cámara depende del número de cámaras web conectadas en el equipo, La interfaz identifica las cámaras conectadas al equipo desde 1 hasta el número de cámaras conectadas, sí en el equipo en el cual correrán el programa sólo se conectará la cámara usada en el dispositivo de identificación de ajos, colocar el número "**1**". Si el equipo llegará a tener una cámara web incluida, el indicador de la cámara que se colocará en la interfaz será "**2**".

En el cuadro de diálogo "Seleccionar COM", se tiene que poner el número de puerto COM que la computadora habilitó para la comunicación con el microcontrolador Arduino UNO, los pasos para conocer en qué puerto COM está conectado se muestran a continuación:

1) Conectar el microcontrolador Arduino UNO a uno de los puertos USB de su computadora.
2) Abrir la ventana de Sistemas.

    2.1) Dar clic derecho en "Este Equipo".

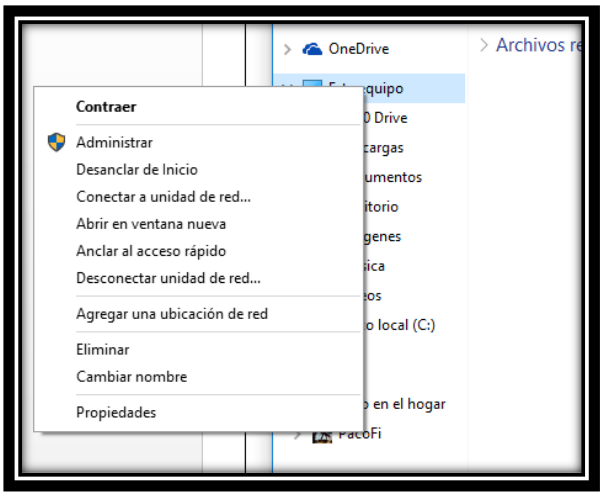

Figura 2 Paso 2.1

2.2) Dar clic izquierdo en Propiedades, se abrirá la siguiente ventana.

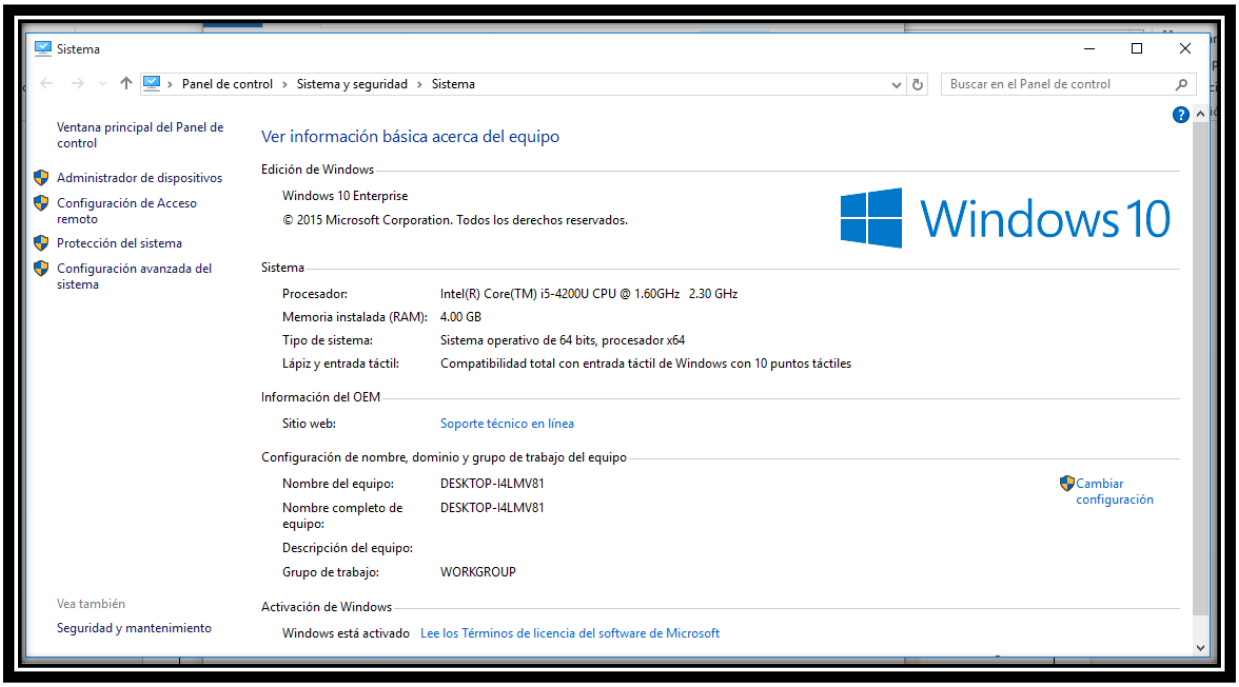

Figura 3 Ventana de Sistema

2.3) Hacer clic izquierdo en Administrador de dispositivos; ahí aparecerán los dispositivos conectados al equipo.

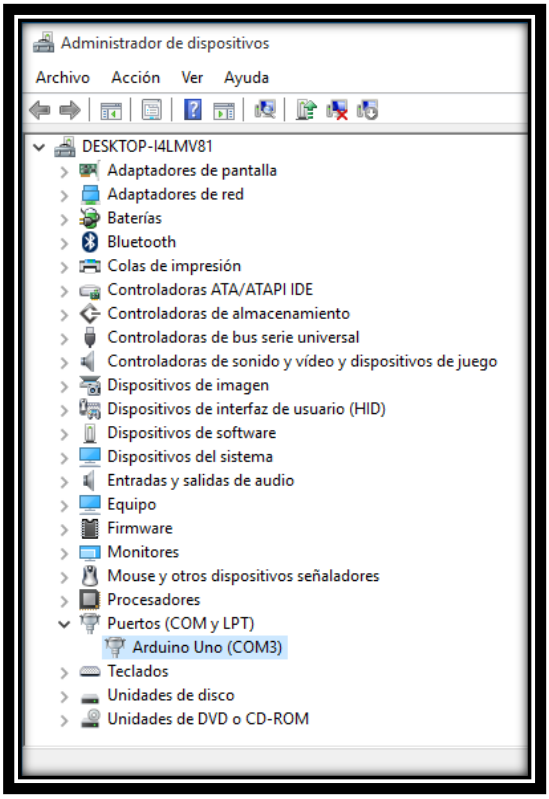

Figura 4 Administrador de dispositivos.

Al hacer clic en la pestaña Puertos (COM y LTP), se mostrarán los dispositivos conectados. El número de puerto COM del Arduino UNO viene dado entre paréntesis (COM3). En caso de la casilla de la interfaz sólo es necesario colocar el número "**3**".

Con estos datos se tiene todo para llenar los cuadros de diálogo de la interfaz, si llegará falta uno de éstos, se manda un mensaje de error pidiendo al usuario revisar los datos proporcionados (Figura 5).

Figura 5 Mensaje de error dirigido al usuario.

Sí los datos son correctos, la interfaz cambia a la siguiente (Figura 6).

El botón de Inicio cambia a Stop, los cuadros de diálogos se bloquean, permitiendo sólo la visualización de los datos anteriormente proporcionados y se hacen visibles los elementos que permiten la visualización de las imágenes.

La imagen de la izquierda muestra la fotografía tomada por la cámara, la imagen de la derecha la salida calculada por la red, se muestra además el número de fotografías tomadas durante el proceso.

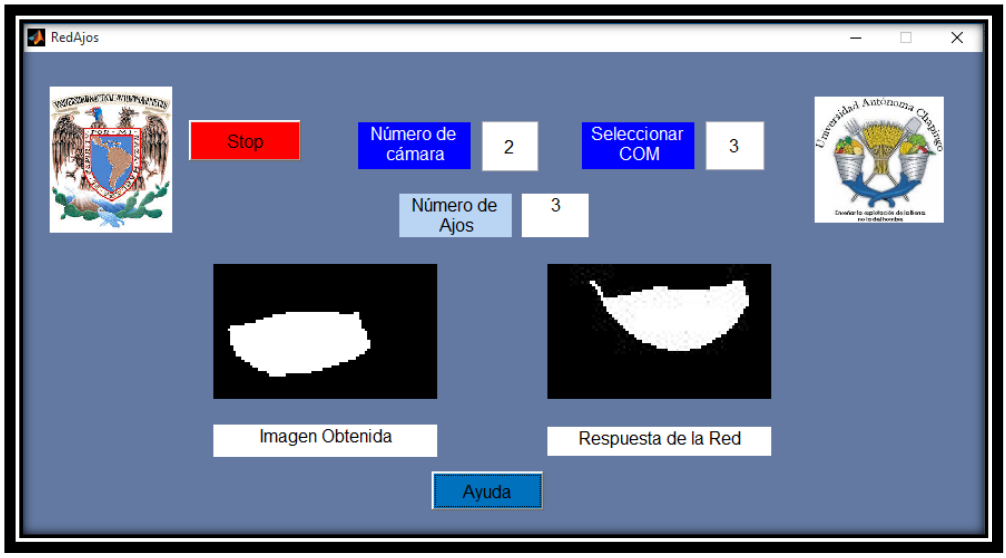

Figura 6 Interfaz de la RNA

Sí se desea detener el programa existen dos formas:

- Haciendo clic en el botón Stop.
- Activando el switch de Stop del circuito, el indicador de Stop del circuito se encenderá para dar a conocer al usuario dicha condición.

Al momento de parar el programa el botón de Inicio vuelve a ser visible, y los cuadros de diálogos se habilitan nuevamente.

Si se detiene el programa activando el switch y éste se mantiene activado el programa no funcionará hasta que se desactive.

Cualquier otro problema que se presentará y no estuviera explicado aquí favor de contactar en el siguiente correo electrónico:

Francisco Muñoz Bustos.

franciscombfi@gmail.com

Anexo B Base de entrenamiento de la RNA

Algunas imágenes usadas como base de entrenamiento en la RNA