UNIVERSIDAD DE PAMPLONA

Una universidad incluyente y comprometida

Con el desarrollo integral

IMPLEMENTACION DE UN SISTEMA DE SEGURIDAD BANCARIO EN SISTEMAS EMBEBIDO UTILIZANDO INTELIGENCIA ARTIFICIAL

Autor:
Andrés Emilio Marín Herrera

Director:
MSc. (C). Jesús Eduardo Ortiz Sandoval

Ingeniería Electrónica
Departamento de Ingenierías Eléctrica, Electrónica, Sistemas y Telecomunicaciones
Facultad de Ingenierías y Arquitectura
Universidad de Pamplona
Pamplona, 15 de diciembre de 2015

Para Ustedes

Emilio Marín Zuluaga, Gilma Herrera Cujiño,
Darling D Marín Herrera, Darwin D Marín Herrera

Eterna Gratitud

INDICE

ÍNDICE DE FIGURAS

INDICE DE TABLAS

.

PREFACIO

“Al fin y al cabo, somos seres que llegamos incluso a la luna sin tener alas”

Anónimo

AGRADECIMIENTOS

Primero que todo gracias a Dios que me ha brindado todas las oportunidades necesarias y que ha estado en todo momento en el camino acompañándome y guiándome. Sobre todo agradezco por haber me entregado uno de los regalos más grande de mi vida mi familia la cual sin su apoyo constante no había logrado nada.

Le agradezco a mi padre Emilio Marín Zuluaga y madre Gilma Herrera Cujiño, los cuales han sido un importante apoyo durante todos los instantes de mi vida; aun mas en este momento en cual es una culminación de una de la etapa académica universitaria, gracias a ellos supe como confrontar este gran reto de mi trabajo de grado los cuales no solo con palabras sino con enseñanzas de vida me instruyeron para poder lograr grandes retos.

También a mis dos hermanos Darling Dayana Marín Herrera y Darwin David Marín Herrera los cuales estoy orgulloso de tenerlos como hermanos, los cuales me han llenado de alegrías mi vida, siendo yo el hermano mayor puedo decir que de los hermanos mayores no solo se aprende también de los hermanos menores los cuales me han enseñado bastante, mi hermana su dedicación y tenacidad para hacer las cosas y mi hermano el cómo tomar las cosas con calma siempre encontrando una solución o un camino para lograrlo.

Agradezco al ingeniero Jesús Ortiz Sandoval por haber participado en este proyecto como director de trabajo de grado y su apoyo prestado para realización del mismo, también a todo el profesorado del programa de ingeniería electrónica el cual me formo académicamente como ingeniero electrónico.

Tambien al ingeniero Julio Cesar Ospino y Doctor Oscar Eduardo Gualdrón los cuales son profesionales íntegros lo cual fue un honor tenerlos como jurados de mi trabajo de grado.

RESUMEN

En este trabajo lo que se va desarrollar es un sistema de seguridad el cual tiene como propósito generar un estado de alarma evaluando posible eventos de atracos para así brindar una mayor seguridad al personal del banco los cajeros encargados de la recaudación de dinero; este proyecto hace provecho de las tecnologías conocidas como la inteligencia artificial la cual va a evaluar si presentan situaciones de peligro y visión artificial la cual estará encargada de extraer características que reflejen un posible evento de atraco para que así la inteligencia artificial evalué la situación, cuando se habla de una característica que reflejo de evento de atraco se quiere mencionar un acto involuntario como alzar la manos o el hecho de poner la manos en el pecho; también se encuentra presente la tecnología GSM que se tiene la tarea de la notificación de un posible evento de atraco por medio de una llamada y el envió de un mensaje de texto, todo este proyecto se encuentra implementado en la tarjeta de desarrollo Raspberry pi de forma embebida.

INTRODUCCIÓN

1.1 PROBLEMA

Debido a los problemas de seguridad que se presentan en algunos establecimientos en los cuales se trabajan con cajeros con la necesidad de mejorar la seguridad en los centros bancarios y prestar mayor seguridad, se observa la necesidad de crear un dispositivo por el cual por medio de visón artificial se capaz de reconocer patrones de atraco o por así decirlo de otra forma los gesto que se presentan en un cajero de banco cuando está en una situación de peligro y así generar una alarma silenciosa.

Se desarrolla este proyecto viendo la falta de seguridad que tiene los cajeros a la hora de que se presenta un atraco en una oficina sin la posibilidad de activar manualmente una alarma debido que el atacante los tiene inmóvil , lo que se busca con este proyecto es que cuando la persona se encuentra en una situación de riesgo y no pueda activar manualmente una alarma debido a la situación que se encuentra; el dispositivo por medio de inteligencia artificial reconozca la situación de peligro y active una alarma sin intervención directa del cajero brindándole un sistema de seguridad autónomo

1.2 OBJETIVOS

1.2.1 OBJETIVO GENERAL

- Implementar un sistema de seguridad bancario en sistema embebido utilizando inteligencia artificial

1.2.1 OBJETIVOS ESPECÍFICOS

- Seleccionar el dispositivo electrónico adecuado para el proyecto
- Implementar sistema de visión artificial en un sistema embebido
- Diseñar e implementar el sistema de adquisición de data fotográfica como soporte de entrenamiento

- Entrenar un sistema de inteligencia artificial

- Acoplar el sistema de la parte lógica con el hardware

- Validar el funcionamiento del sistema de seguridad

1.3 DISTRIBUCIÓN BIBLIOGRÁFICA

El libro se organiza mediante cinco capítulos concernientes a la Introducción, Marco Teórico, Diseño de Componentes, Programación, Resultados y Conclusiones.

Cada capítulo tiene los ítems correspondientes al tema de interés.

2. MARCO TEÓRICO

2.1 ANTECEDENTES

El sistema como un elemento de seguridad en la cual por medio de visión artificial para extraer patrones o eventos que den un indicio de un evento delictivo es decir un atraco, académicamente no se consigue información, pero viéndolo desde un punto de vista de un proyecto seccionado; como la implementación de una red neuronal y visión artificial en un sistema embebido, se pueden apreciar varios trabajos académicos en esta áreas los cuales podemos mencionar

CLASIFICACIÓN DE COMPUESTOS QUÍMICOS USANDO UN SISTEMA MULTISENSORIAL (NARIZ ELECTRÓNICA) DESARROLLADA SOBRE UN DISPOSITIVO HARDWARE (FPGA)

El trabajo consiste en la clasificación de los datos provenientes de una matriz de sensores de gases químicos por medio de una nariz electrónica pero esta no se encuentra implementada en un computador, se trabaja en dispositivo hardware, con el fin de obtener un sistema que sea más portable y que proporcione resultados en tiempo real a través de la adquisición, procesamiento y clasificación de las muestras. Para el desarrollo del sistema, se diseño un algoritmo en lenguaje VHDL (Very High Speed Integrated Circuit - Hardware Description Language) y fue implementado una FPGA (Field Programmable Gate Array) Spartan 3E de Xilinx. [1]

.

HOGAR INTELIGENTE POR CONTROL DE VOZ USANDO REDES NEURONALES.

Este artículo presenta el diseño de un hogar inteligente de bajo costo basado en el *Google Voice* API encadenada a una serie de redes neuronales de topología *feedforward-backpropagation*, las cuales brindan la interacción con la máquina de una manera amena simulando una conversación humana y dejando de lado órdenes monótonas, el estado del arte presenta las opciones tecnologías escogidas todas bajo plataforma Open Source para minimizar costos y hacer el conocimiento más globalizado y accesible al lector, también encontrarán la evolución del sistema en sí, los resultados obtenidos es la automatización total de un apartamento de 57 metros cuadrados con funciones interactivas basadas en la voz ya que se puede

encender o apagar cualquier luz del complejo habitacional haciendo una solicitud vocal direccionada al dispositivo con una respuesta instantánea del sistema, además de una serie de opciones útiles para la vida cotidiana.[2]

Desarrollo de un Sistema de Detección de Movimiento basado en Flujo Óptico en Raspberry Pi

El proyecto muestra una alternativa para desarrollar un sistema para la detección de movimiento, utilizando sistemas embebidos de bajo costo basados en microprocesador. Se realiza la implementación de un algoritmo para la detección de flujo óptico. El flujo óptico se caracteriza básicamente por la detección de movimiento en una secuencia continua de imágenes o en un video en particular. De esta manera, es posible que se detecte movimiento en una zona donde no debe existir actividad en un horario específico. El sistema que se va a desarrollar se basa en el uso de sistemas embebidos basados en microprocesador, en este caso, una tarjeta Raspberry Pi, el cual cuenta con una cámara de 5 Mega-pixeles, para la adquisición de imágenes o video, así como la instalación de Python y las librerías de OpenCV. Se analizan diversas opciones de algoritmos de flujo óptico para determinar el que puede ser implementado en la tarjeta mencionada. Una vez que el sistema detecta movimiento, se adquiere la imagen del lugar y es almacenada para un análisis posterior.[3]

Sistema de Autenticación Facial mediante la Implementación del algoritmo PCA modificado en Sistemas embebidos con arquitectura ARM.

La idea principal de este proyecto es implementar un sistema de acceso por medio de reconocimiento facial e implementado en un hardware de arquitectura abierta. El sistema de reconocimiento facial propuesto se basa en una plataforma embebida de bajo consumo comprende un ARM (Advanced RISC máquinas) módulo central de procesamiento, un módulo de adquisición de vídeo, un módulo de visualización y una interfaz de transmisión de datos periférica. El algoritmo a implementar en el procesamiento digital de imágenes es el Principal Component Analysis (PCA) es robusto, rápido y eficiente para llevar a cabo el reconocimiento facial. El sistema de reconocimiento facial basado en la plataforma embebida de bajo

consumo tiene la ventaja de bajo consumo de energía, alta velocidad de computación, la alta precisión de reconocimiento, amplio campo de aplicación y similares.[4]

CLASIFICACIÓN DE IMÁGENES MULTIESPECTRALES MEDIANTE REDES NEURONALES.

En este articulo lo que se propone es una red neuronal para llevar a cabo una clasificación supervisada de imágenes multiespectrales y.[5]

2.2 VISION ARTIFICIAL

Podríamos decir que la Visión Artificial (VA) describe la deducción automática de la estructura y propiedades de un mundo tridimensional posiblemente dinámico, bien a partir de una o varias imágenes bidimensionales del mundo. Las imágenes pueden ser monocromáticas (de niveles de gris) o colores, pueden provenir de una o varias cámaras e incluso cada cámara puede estar estacionaria o móvil. Las estructuras y propiedades del mundo tridimensional que queremos deducir en visión artificial incluyen no sólo sus propiedades geométricas, sino también sus propiedades materiales. Ejemplos de propiedades geométricas son la forma, tamaño y localización de los objetos. Ejemplos de propiedades de los materiales son su color, iluminación, textura y composición. Si el mundo se modifica en el proceso de formación de la imagen, necesitaremos inferir también la naturaleza del cambio, e incluso predecir el futuro. La entrada a un sistema de VA es una imagen obtenida por un elemento de adquisición, mientras que su salida es una descripción de la escena, la cual ha sido obtenida a partir de la imagen.

Por un lado, esta descripción debe estar relacionada de algún modo con aquella realidad que produce la imagen y, por el otro, debe contener toda la información requerida para la tarea de interacción con el medio ambiente que se desea llevar a cabo, por ejemplo mediante un robot. Esto es, la descripción depende en alguna forma de la

entrada visual y debe proporcionar información relevante y utilizable por el robot.

La visión artificial o visión por computador es la ciencia y la tecnología que **permite a las "máquinas "ver**, extraer información de las imágenes digitales, resolver alguna tarea o entender la escena que están visionando.

Actualmente, las aplicaciones de la visión artificial están muy extendidas y van desde el campo de la industria (contar botellas, comprobar defectos en una cadena de montaje, interpretar un TAC médico...) y el campo de la medicina (recuento y búsqueda de células), hasta los sistemas más complejos, que permiten a los robots orientarse en un entorno desconocido, pasando por el reconocimiento de patrones de la realidad aumentada, entre otras muchas aplicaciones.

2.2.1 LUZ VISIBLE, LUZ NO VISIBLE

El ojo humano ve una parte del espectro de toda la luz que ilumina el universo. El rango de luz que podemos ver lo denominaremos *luz visible*. Esto quiere decir que hay frecuencias de luz que no podemos ver pero que existen, como por ejemplo, los infrarrojos y los ultravioletas.

- Los **infrarrojos** son los "colores" no visibles al ojo humano que Están por debajo del rojo desde el punto de vista frecuencial.

- Los **ultravioletas** son los "colores" no visibles al ojo humano Que están por encima del violeta. [6]

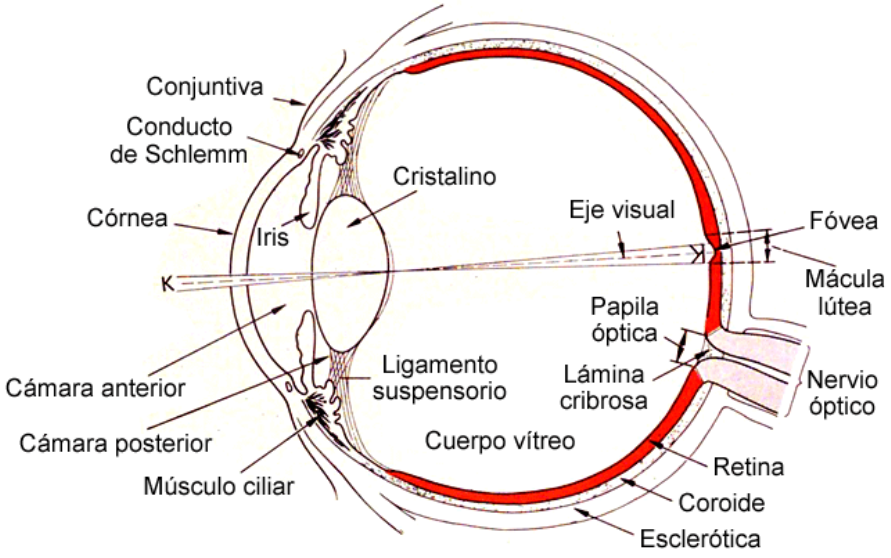


Figura 1. Ojo humano y sus partes.[6]

El hecho es que la mayoría de sensores de cámaras digitales son sensibles a la luz visible, pero también son sensibles (en diferente medida) a la luz infrarroja y/o a la ultravioleta. Ahora bien, la luz infrarroja no nos es útil para construir una imagen digital, puesto que nos da información de una frecuencia que no podemos ver y que, por lo tanto, no tiene una representación posible en un color.

Por esta razón, la mayoría de cámaras digitales enfocadas a hacer fotografías o fotogramas llevan un filtro anti-IR, o sea anti-infrarrojos, para cortar todas las frecuencias por debajo del espectro visible que nos resultarían un ruido innecesario, y solo dejan pasar el rango de luz visible que queremos plasmar con la cámara.

2.2.2 IMAGEN

- **Matrices de píxeles:**

En el mundo digital, las imágenes se representan como una matriz bidimensional de píxeles en la que cada píxel puede adquirir **valores de color codificados** con tres parámetros (**R**: *red*, **G**: *green*, **B**: *blue*). A pesar de que ello se ha heredado del mundo de la imagen analógica, normalmente trabajaremos con imágenes de proporción 4 × 3 (cuatro unidades de anchura por tres unidades de altura); es el caso de resoluciones estándar como 640 × 480 px, 800 × 600 px y 1.024 × 768 px. Frecuentemente, utilizaremos imágenes de baja resolución

(entre 160 × 120 y 640 × 480 píxeles) como fuente de análisis de los procesos de visión artificial, ya que en la mayoría de casos no necesitaremos excesivo detalle en las imágenes para efectuar un análisis orientado a la visión por computador.

- **Bytes, bits y colores:**

Un píxel normalmente se expresa mediante tres números enteros (R, G, B), que representan los componentes rojo, verde y azul de todo color. Estos valores de R, G y B se suelen expresar en un rango de 8 bits, o sea, de valores entre 0 y 255.

- **Frecuencia de imagen (*frame rate*):**

La frecuencia de imagen (*frame rate*) hace referencia al número de imágenes por segundo. Es la medida de la frecuencia a la que un reproductor de imágenes muestra diferentes fotogramas (*frames*).

En informática estos fotogramas están constituidos por un número determinado de píxeles que se distribuyen a lo largo de una red de texturas. La frecuencia de los fotogramas es **proporcional** al número de píxeles que se han de generar y que inciden en el rendimiento del ordenador que los reproduce. La frecuencia de actualización de las imágenes oscila, en el entorno digital, entre los 15 y los 60 FPS (*frames* por segundo). El rango entre 25 y 30 FPS es el más común.

2.2.3 ADQUISICIÓN DE LA IMAGEN

Una imagen digital es producida por sensores digitales presentes en cámaras y otros dispositivos digitales que generan una imagen bidimensional (2D), es decir, un conjunto de $N \times M$ píxeles o colores o intensidades de un cierto valor que representan el **espacio** que queremos analizar. En el mundo de la interacción, podemos utilizar cámaras digitales de diferentes tipos (cámaras web, DV o USB). En el fondo, cada tipo de aplicación puede necesitar un tipo u otro de cámara según las necesidades. [6]

2.2.4 PROCESAMIENTO DE LA IMAGEN

Antes de extraer información directamente de la imagen, se acostumbra a hacer un procesamiento previo de la misma para conseguir otra que nos permita hacer el proceso de extracción de datos más sencillo y eficiente.

- **Escala de grises:**

En muchos casos, el color de la imagen no nos aportará ninguna información relevante para interactuar. Por lo tanto, en estas situaciones se acostumbra a descartar la información de color de la imagen transformándola en una **imagen de tonos de grises**.

Figura 2. Imagen de tonos de grises[6]

Binarización por umbral (*threshold binarization*):

El proceso de binarización por umbral parte de una imagen en tonos de grises y, a partir de un valor definible de umbral de intensidad de luz llamado *threshold,* la imagen se transforma en una imagen binaria, es decir, con píxeles blancos o píxeles negros. Si el píxel analizado tenía un valor de intensidad inferior al umbral, quedará en negro, y si su intensidad era más elevada que el umbral, quedará en blanco.

De alguna manera, esto nos permite descartar los tonos medios grises y facilita mucho la computación de ciertos algoritmos, puesto que otra vez hemos transformado la imagen en algo mucho más ligero y fácil de procesar. [6]

- **Velocidad y resolución de la imagen:**

Muchos de los procesos de visión artificial se basan en trabajar por cada píxel de la imagen internamente; por lo tanto, si con la binarización conseguimos 24 veces menos de CC-BY-NC-ND • PID_00184756 20 Visión artificial bits que procesar, los algoritmos correrán mucho más rápido. Así pues, un factor determinante en la velocidad de los cálculos que se generan en las aplicaciones de visión artificial es la resolución de la imagen en píxeles. Una imagen de 320 × 240 píxeles será procesada mucho más rápidamente que una imagen de 1.024 × 768 píxeles.

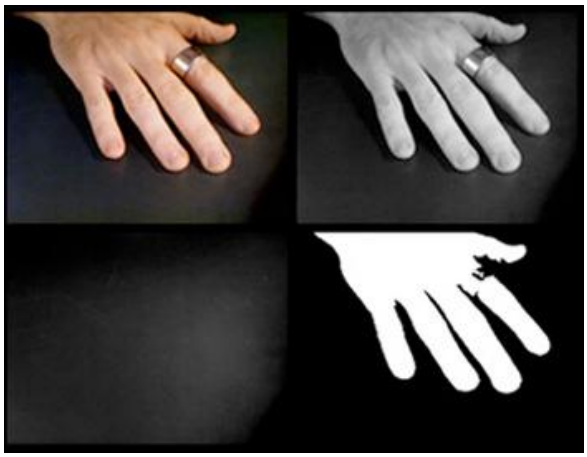


Figura 3. Imagen, imagen escala de grises, imagen binarizada.[6]

En esta imagen, podemos ver las tres representaciones hasta ahora mencionadas. Arriba a la izquierda, tenemos la representación en color del fotograma capturado por la cámara; arriba a la derecha, la transformación a blanco y negro de la misma imagen, y abajo a la derecha, la representación binaria de la imagen en blanco y negro. El umbral en esta binarización se ha fijado en un valor que permite distinguir fácilmente la mano (blanca) del fondo (negro).

- **Otras operaciones morfológicas:**

En muchas situaciones, una vez que hemos obtenido una imagen binaria (en blanco y negro), observamos que aparece **ruido** en la imagen, fruto de los cambios en la iluminación de ambiente y los pequeños reflejos de esta luz en los diferentes objetos y cuerpos en la escena. Este ruido suele consistir simplemente en píxeles que

aparecen caóticamente en la imagen y que pueden estorbar seriamente la capacidad de los algoritmos de visión para reconocer *blobs* y patrones gráficos. Para eliminar estos pequeños y molestos píxeles, hay una serie de algoritmos de "limpieza" de la imagen que nos ayudan a cumplir la tarea de obtención de una imagen binaria lo más neta posible.

- **Erosionar/dilatar (*Erode/dilate*):**

Los algoritmos de erosión y dilatación se suelen aplicar en serie. Sobre la imagen "ruidosa" se aplica el algoritmo de erosión, que contrae los contornos de todas las áreas blancas un número determinado de píxeles y que elimina completamente las áreas de píxeles blancos más pequeñas e irrelevantes. Una vez aplicado el algoritmo de erosión, se aplica el algoritmo de dilatación, que ayuda a recuperar la medida original de las áreas importantes y que expande los contornos de las áreas blancas tantos píxeles como sea necesario.

Figura 4. Imagen a la cual se le aplica la técnica de erosión y dilatación.[6]

2.3 REDES NEURONALES

El cerebro humano ha sido objeto de estudio durante décadas, buscando explicar los procesos cognitivos y de razonamiento que nos diferencia del resto de animales que habitan en el planeta tierra. Los científicos en los últimos años han buscado desarrollar herramientas que emulen la capacidad del ser humano de recordar, analizar, clasificar y razonar. A esta ciencia se le llama inteligencia artificial, y en una de sus ramas aparecen las Redes Neuronales ANN, esta herramienta fue la escogida para el desarrollo del proyecto.

Las Redes Neuronales Artificiales, ANN (Artificial Neural Networks) están inspiradas en las redes neuronales biológicas del cerebro humano. Están constituidas por elementos que se comportan de forma similar a la neurona biológica en sus funciones más comunes. Estos elementos están organizados de una forma parecida a la que presenta el cerebro humano.

Las ANN al margen de "parecerse" al cerebro presentan una serie de características propias del cerebro. Por ejemplo las ANN aprenden de la experiencia, generalizan de ejemplos previos a ejemplos nuevos y abstraen las características principales de una serie de datos.

- ***Aprender:*** adquirir el conocimiento de una cosa por medio del estudio, ejercicio o experiencia. Las ANN pueden cambiar su comportamiento en función del entorno. Se les muestra un conjunto de entradas y ellas mismas se ajustan para producir unas salidas consistentes.

- ***Generalizar:*** extender o ampliar una cosa. Las ANN generalizan automáticamente debido a su propia estructura y naturaleza. Estas redes pueden ofrecer, dentro de un margen, respuestas correctas a entradas que presentan pequeñas variaciones debido a los efectos de ruido o distorsión.

- ***Abstraer:*** aislar mentalmente o considerar por separado las cualidades de un objeto. Algunas ANN son capaces de abstraer la esencia de un conjunto de entradas que aparentemente no presentan aspectos comunes o relativos.

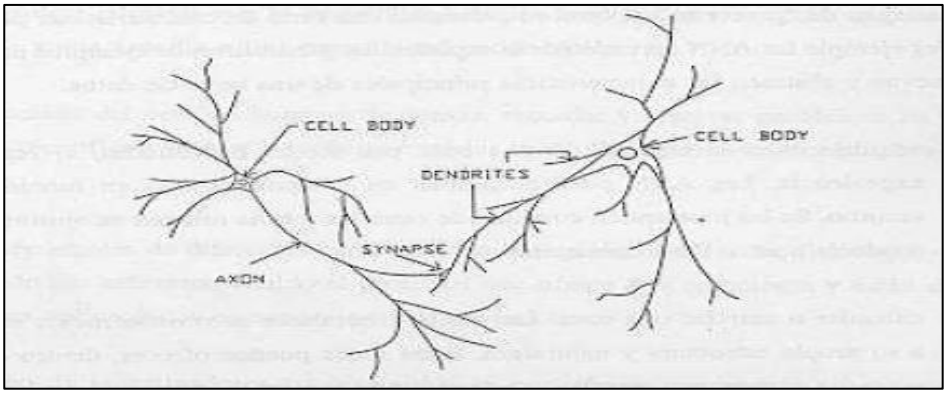


Figura 5. Neurona biológica.[7]

La neurona es la unidad fundamental del sistema nervioso y en particular del cerebro. Cada neurona es una simple unidad procesadora que recibe y combina señales desde y hacia otras neuronas. Si la combinación de entradas es suficientemente fuerte la salida de la neurona se activa. La Figura 8 muestra las partes que constituyen una neurona. El cerebro consiste en uno o varios billones de neuronas densamente interconectadas. El axón (salida) de la neurona se ramifica y está conectada a las dendritas (entradas) de otras neuronas a través de uniones llamadas sinapsis. La eficacia de la sinpasis es modificable durante el proceso de aprendizaje de la red.

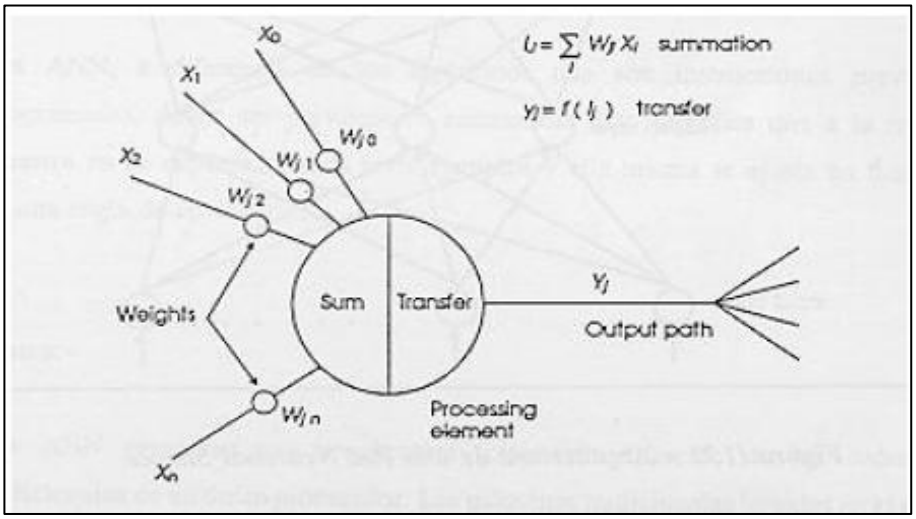


Figura 6. Neurona biológica.[7]

En las Redes Neuronales Artificiales, ANN, la unidad análoga a la neurona biológica es el elemento procesador, PE (*process element*).

Un elemento procesador tiene varias entradas y las combina, normalmente con una suma básica. La suma de las entradas es modificada por una función de transferencia y el valor de la salida de esta función de transferencia se pasa directamente a la salida del elemento procesador. La salida del PE se puede conectar a las entradas de otras neuronas artificiales (PE) mediante conexiones ponderadas correspondientes a la eficacia de la sinapsis de las conexiones neuronales. La Figura 9 representa un elemento procesador de una red neuronal artificial implementada en un computador.

Una red neuronal consiste en un conjunto de unidades elementales PE conectadas de una forma concreta. El interés de las ANN no reside solamente en el modelo del elemento PE sino en las formas en que se conectan estos elementos procesadores.

Generalmente los elementos PE están organizados en grupos llamados niveles o capas. Una red típica consiste en una secuencia de capas con conexiones entre capas adyacentes consecutivas.

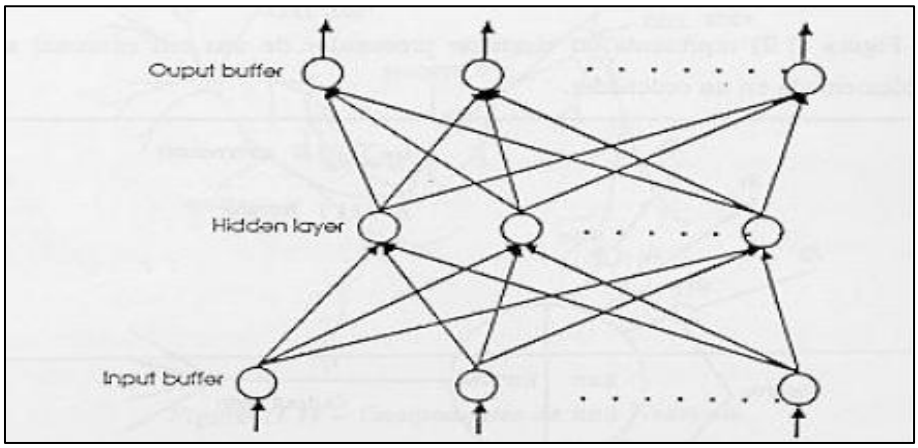


Figura 7. Arquitectura de una ANN.[7]

Existen dos capas con conexiones con el mundo exterior. Una capa de entrada, buffer de entrada, donde se presentan los datos a la red, y una capa buffer de salida que mantiene la respuesta de la red a una entrada. El resto de las capas reciben el nombre de capas ocultas. La Figura 7 muestra el aspecto de una Red Neuronal Artificial. [7]

2.3.1 ELEMENTOS DE UNA RED NEURONAL MULTICAPA

En el proyecto se implementó una red MLP, o una red multicapa, pero para el desarrollo del algoritmo es necesario comprender como es la estructura de este tipo de ANN, esta es una de las partes más importantes de la investigación.

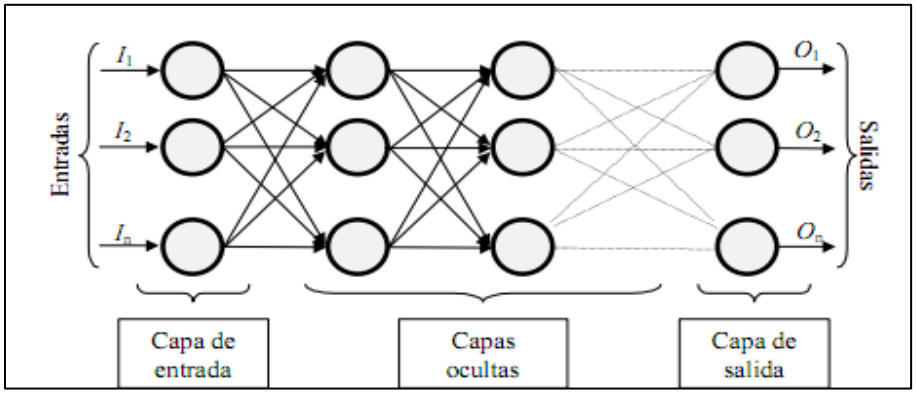


Figura 8. Estructura de una Red MLP.[8]

En la figura 8 está la representación pictográfica de una neuronal multicapa, la misma está constituida por neuronas interconectadas y arregladas en tres capas (esto último puede variar). Los datos ingresan por medio de la "*capa de entrada*", pasan a través de la "*capa oculta*" y salen por la "*capa de salida*". Cabe mencionar que la capa oculta puede estar constituida por varias capas.

La neurona trata a muchos valores de entrada como si fueran uno solo; esto recibe el nombre de *entrada global*. Por lo tanto, ahora nos enfrentamos al problema de cómo se pueden combinar estas simples entradas (*ini*1, *ini*2,..) dentro de la entrada global, *gini*. Esto se logra a través de la *función de entrada*, la cual se calcula a partir del *vector entrada*. La función de entrada puede describirse como sigue:

$$input_i = (in_{i1} \bullet w_{i1}) * (in_{i2} \bullet w_{i2}) * \ldots (in_{in} \bullet w_{in}) \quad \text{Ec. (1)}$$

En la ecuación 1 el símbolo * representa al operador apropiado (por ejemplo: máximo, sumatoria, producto, etc.), *n* al número de entradas a la neurona *Ni* y *wi* al peso. Los valores de entrada se multiplican por los pesos anteriormente ingresados a la neurona. Por

consiguiente, los pesos que generalmente no están restringidos cambian la medida de influencia que tienen los valores de entrada. Es decir, que permiten que un gran valor de entrada tenga solamente una pequeña influencia, si estos son lo suficientemente pequeños.

Algunas de las funciones de entrada más comúnmente utilizadas y conocidas son:

- *Sumatoria de las entradas pesadas:* es la suma de todos los valores de entrada a la neurona, multiplicados por sus correspondientes pesos.

$$\sum_i \left(n_{ij}\, w_{ij}\right), \quad \text{con } j = 1, 2, ..., n \qquad \textit{Ec. (2)}$$

- *Producto de las entradas pesadas:* es el producto de todos los valores de entrada a la neurona, multiplicados por sus correspondientes pesos.

$$\prod_j \left(n_{ij}\, w_{ij}\right), \quad \text{con } j = 1, 2, ..., n \qquad \textit{Ec. (3)}$$

- *Máximo de las entradas pesadas:* solamente toma en consideración el valor de entrada más fuerte, previamente multiplicado por su peso correspondiente.

$$\operatorname{Max}_j \left(n_{ij}\, w_{ij}\right) \quad \text{con } j = 1, 2, ..., n \qquad \textit{Ec. (4)}$$

Una neurona biológica puede estar activa (excitada) o inactiva (no excitada); es decir, que tiene un "*estado de activación*". Las neuronas artificiales también tienen diferentes estados de activación; algunas de ellas solamente dos, al igual que las biológicas, pero otras pueden tomar cualquier valor dentro de un conjunto determinado.

La *función activación* calcula el estado de actividad de una neurona; transformando la entrada global (menos el umbral, Θi) en un valor (estado) de activación, cuyo rango normalmente va de (0 a 1) o de (–1 a 1). Esto es así, porque una neurona puede estar totalmente inactiva (0 o –1) o activa (1). [8]

2.4 TARJETAS DE DESARROLLO

En el mercado actualmente se ofrece una gran variedad de tarjetas de diferentes fabricantes para diferentes tipos de prestaciones en las cuales se pueden mencionar la Raspberry pi, Beagle Bone Black, Tiva- c

- **Raspberry pi 2**

Esta tarjeta pose un el Broadcom BCM2836, un sistema en un chip que contiene quad-core ARM Cortex-A7 con punto flotante, corre a 900MHz, y un procesador de Video 4 GPU. El GPU provee de un GL ES 2.0 abierto, acelerador de hardware OpenVG, y decodificador de alto perfil 1080p30 H.264 y capaz de 1Gpixeles, 1.5Gtexteles/s o 24 GFLOPs de propósito general. Esto significa que la Raspberry Pi puede ser conectada a una TVHD, se pueden ver videos de calidad BlueRay, usando H.264 a 40MBits/s.

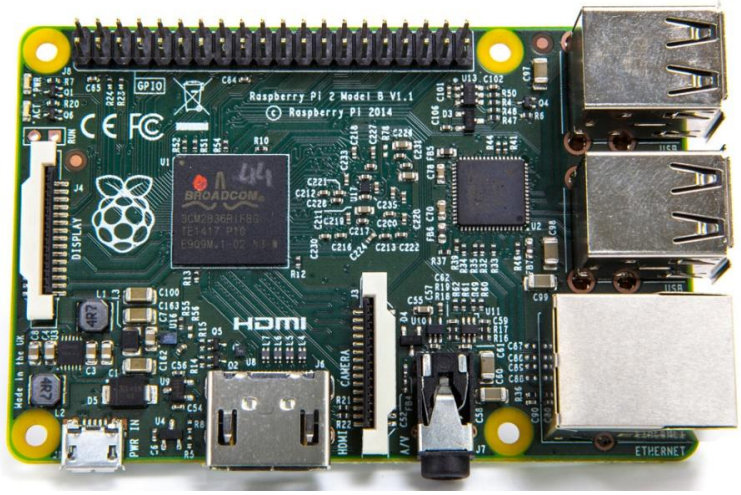


Figura 9. Tarjeta de desarrollo Raspberry pi 2. [11]

El más grande cambio en la Raspberry Pi 2 es una mejora en el procesador principal y un incremento de 512MB a 1 GB. La RPi2 aun utiliza microSD para su sistema operativo significando que cabe cualquier sistema operativo para la Pi2 en 4GB microSD pero también tiene soporte para tarjetas más grandes en capacidad.

Posee 40 pines header GPIO con espaciamiento de 0.1" en la Pi te proveen de 26 GPIO, UART, I2C, SPI así como fuentes de 3.3V y 5V. Los primeros 26 pines son idénticos a los del modelo B.

Dimensiones: 85mm x 56mm x 17mm

Características:

- 900 MHz BCM2836 ARMv7 Quad Core Processor SoC
- VideoCore IV GPU
- 1 GB RAM
- 4 x Puertos USB2.0 de hasta 1.2A de salida
- 40-pin GPIO Header
- Video/Audio Out atraves de 4-pole 3.5mm conector, HDMI, o Raw LCD (DSI)
- Almacenamiento: microSD
- 10/100 Ethernet (RJ45)
- Periféricos de bajo nivel: 27 x GPIO
- UART
- I2C bus
- Bus SPI con dos selectors de chip
- +3.3V
- +5V
- Tierra
- Requerimientos de energía: 5V @ 600 mA por MicroUSB o GPIO Header
- Soporte Windows 10, Debian GNU/Linux, Fedora, Arch Linux, RISC OS y Otros.

- **Raspberry pi B**

La Raspberry pi B es una versión anterior a la Raspberry pi 2 la diferencia más notable cambio con Raspberry Pi B es una mejora en el procesador principal y un incremento de 512MB a 1 GB.

Figura 10. Tarjeta de desarrollo Raspberry pi B. [11]

Características:

- **SoC :** BROADCOM BCM2835
- **CPU:** ARM11 ARMV6 700MHz
- **GPU:** BROADCOM VIDEOCORE IV 250MHz
- **MEMORIA RAM:** 512MB LPDDRR SDRAM 400MHZ
- **PUERTOS USB 2.0:** 2
- **GPIO** 26 PINES (SPI, I2C UART)
- **VIDEO**: HDMI 14 1920 X 1200
- **SALIDA DE AUDIO**: Conector de 3,5mm, HDMI
- **SALIDA DE VIDEO:** CONECTOR RCA(PAL Y NTSC), HDMI, Interfaz DSI para panel LCD
- **ALMACENAMIENTO**: MEMORIA SD (mínimo 4G)
- **ETHERNET**: ETHERNET 10/100MBPS (RJ-45)
- **ALIMENTACION:** 5V (Conector micro USB) I=700Ma
- **SISTEMAS OPERATIVOS:** Debían GNU/Linux, Fedora, Arch Linux, RISC OS y Otros.

- **Beagle Bone Black**

La Beagle Bone Black cuenta con un procesador ARM Cortex A8 de Sitara XAM3359AZCZ100 con una frecuencia de trabajo a 1GHz de Texas Instruments, proporciona un puerto micro HDMI, 512 MB de DRAM DDR3L, 4 GB de memoria interna flash y JTAG opcional. Más de 3 millones de operaciones de operaciones aritméticas por segundo y el vector de punto flotante.

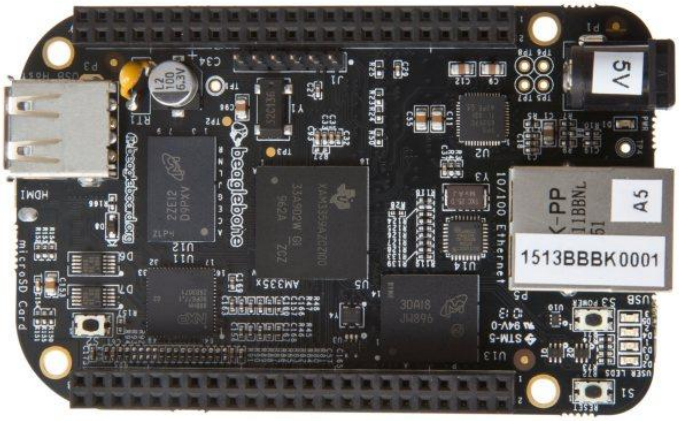


Figura 11. Tarjeta de desarrollo Beagle Bone Black. [10]

Características:

- **Procesador:** Sitara AM335x 1GHz ARM® Cortex-A8
- 512MB DDR3L RAM 606 Mhz
- 4GB 8-bit eMMC on-board flash storage
- Acelerador de graficas 3D
- NEON acelerador punto flotante
- 2x PRU 32-bit microcontrolador

Conectividad:

- USB client, para alimentación y comunicaciones
- USB host
- Ethernet
- HDMI resolución 1280x1024
- 2x 46 pin headers

Software Compatible:
Debian, Android, Ubuntu, Cloud9 IDE on Node.js w/ BoneScript library, entre otros

- **Tiva C Series TM4C1294**

Figura 12. Tarjeta de desarrollo Tiva C TM4C1294. [9]

Características:

- **Microcontrolador:** TM4C1294NCPDTI ARM® Cortex-M4 MCU con punto flotante
- **Velocidad:** 120 MHz
- **Memoria Flash:** 1 MB
- **Memoria RAM:** 256 KB
- **Timers:** 8 de 32 bit
- **Comunicación serial :** 10 PC, 8 UARTS, 4 QSPI, 2 CAN, EPI, USB
- 8 PWMs
- 2 ADC de 12 bit
- 2 x 40 pines para boosterpacks
- Puerto de conexión Ethernet
- 4 Leds de usuario
- 2 Botones de usuario

2.5 LENGUAJE DE PROGRAMACIÓN EN PYTHON

Python es un lenguaje de programación creado por Guido van Rossum a principios de los años 90 cuyo nombre está inspirado en el grupo de cómicos ingleses “Monty Python”. Es un lenguaje similar a Perl, pero con una sintaxis muy limpia y que favorece un código legible; Se trata de un lenguaje interpretado o de script, con tipado dinámico, fuertemente tipado, multiplataforma y orientado a objetos.

- **Lenguaje interpretado o de script**

Un lenguaje interpretado o de script es aquel que se ejecuta utilizando un programa intermedio llamado intérprete, en lugar de compilar el código a lenguaje máquina que pueda comprender y ejecutar directamente una computadora (lenguajes compilados). La ventaja de los lenguajes compilados es que su ejecución es más rápida. Sin embargo los lenguajes interpretados son más flexibles y más portables. Python tiene, no obstante, muchas de las características de los lenguajes compilados, por lo que se podría decir que es semi interpretado. En Python, como en Java y muchos otros lenguajes, el código fuente se traduce a un pseudo código máquina intermedio llamado bytecode la primera vez que se ejecuta, generando archivos .pyc o .pyo (bytecode optimizado), que son los que se ejecutarán en sucesivas ocasiones.

- **Tipado dinámico**

La característica de tipado dinámico se refiere a que no es necesario declarar el tipo de dato que va a contener una determinada variable, sino que su tipo se determinará en tiempo de ejecución según el tipo del valor al que se asigne, y el tipo de esta variable puede cambiar si se le asigna un valor de otro tipo.

- **Fuertemente tipado**

No se permite tratar a una variable como si fuera de un tipo distinto al que tiene, es necesario convertir de forma explícita dicha variable al nuevo tipo previamente. Por ejemplo, si tenemos una variable que contiene un texto (variable de tipo cadena o string) no podremos tratarla como un número (sumar la cadena “9” y el número 8). En otros lenguajes el tipo de la variable cambiaría para adaptarse al comportamiento esperado, aunque esto es más propenso a errores.

- **Multiplataforma**

El intérprete de Python está disponible en multitud de plataformas (UNIX, Solaris, Linux, DOS, Windows, OS/2, Mac OS, etc.) por lo que si no utilizamos librerías específicas de cada plataforma nuestro programa podrá correr en todos estos sistemas sin grandes cambios.

- **Orientado a objetos**

La orientación a objetos es un paradigma de programación en el que los conceptos del mundo real relevantes para nuestro problema se trasladan a clases y objetos en nuestro programa. La ejecución del programa consiste en una serie de interacciones entre los objetos.

Python también permite la programación imperativa, programación funcional y programación orientada a aspectos.[13]

2.3 Opencv

OpenCV es una biblioteca libre de visión artificial originalmente desarrollada por Intel. Desde que apareció su primera versión alfa en el mes de enero de 1999, se ha utilizado en infinidad de aplicaciones. Desde sistemas de seguridad con detección de movimiento, hasta aplicativos de control de procesos donde se requiere reconocimiento de objetos. Esto se debe a que su publicación se da bajo licencia BSD, que permite que sea usada libremente para propósitos comerciales y de investigación con las condiciones en ella expresadas.

Open CV es multiplataforma, existiendo versiones para GNU/Linux, Mac OS X y Windows. Contiene más de 500 funciones que abarcan una gran gama de áreas en el proceso de visión, como reconocimiento de objetos, calibración de cámaras, visión estérea y visión robótica.

3. DISEÑO DE COMPONENTES

3.1 SELECCIÓN DEL DISPOSITIVO

Se tiene en cuenta varios criterios a la hora de seleccionar el hardware a trabajar en los cuales esta:

- Que tenga vector de punto flotante para la implementación del a red neuronal
- Lenguaje de programación de alto nivel el cual permita la implementación de visión artificial
- Que cuente con modulo de cámara compatible con la tarjeta de desarrollo para la adquisición de imagen
- Características de las tarjetas como tipo de procesador y velocidades de trabajo
- Beneficio costo

Tiva C Series TM4C1294

Desventajas:

-no cuenta con sistema operativo en tiempo real.

-no cuenta con una plataforma versátil para implementación de códigos de alto nivel, como manejo de tratamiento de imágenes.

-no cuenta con modulo de cámaras, propios de la tarjeta.

-aunque tiene un ARM Cortex M4 con una frecuencia de 120Mhz que es alta comparada con otros microcontroladores del mercado y tarjetas, pero se queda corta de características para el trabajó con procesamiento de imágenes.

Ventajas:

-cuenta con vector de punto flotante, que es necesario para la implementación de la red neuronal.

-beneficio/costo la tarjeta tiene un precio de 50,000 pesos el cual es relativamente bajo para sus prestaciones.

Conclusión

-no sirve para la implementación del proyecto a trabajar.

Beagle Bone Black

Ventajas:

-Cuenta con un sistema operativo en tiempo Real sobre un microcontrolador a 32 bit base Linux.

-Cuenta con una plataforma versátil para implementación de código de alto nivel como el manejo de tratamiento de imágenes, lenguaje de programación Python y librerías de OpenCV.

-Tiene vector de punto flotante, que es necesario para la implementación de la red neuronal

-Posee un ARM Cortex A8 y con una frecuencia de un 1 GHz, con una memoria RAM de 512Mb

Desventajas

-Posee un modulo de cámara HD pero necesita tarjetas de acople y son realmente costosas y no se encuentran en el mercado nacional, siendo necesario realizar importación.

-Se puede trabajar con cámaras USB pero el precio de una cámara USB HD es costosa y tiene que tener soporte en Linux para poder usar

-Beneficio/costo no están bueno ya que cuando se requiere implementar el hardware necesario para el proyecto el costo de este extremadamente elevado triplicando el valor de otra opciones viables

Conclusión

-Con esta tarjeta se puede implementar el sistema lo que lo hace inviable es factor económico

Raspberry pi

Ventajas

-Cuenta con un sistema operativo en tiempo Real sobre un microcontrolador a 32 bit base Linux.

-Cuenta con una plataforma versátil para implementación de código de alto nivel como el manejo de tratamiento de imágenes, lenguaje de programación Python y librerías de OpenCV.

-Tiene vector de punto flotante, que es necesario para la implementación de la red neuronal

-Posee un ARM Cortex 11 con una Frecuencia de 700 MHz con una memoria RAM de 512 Mb

-Tiene un módulo de cámara HD de 5megapixeles con conexión CSI, no requiere módulos extras para su funcionamiento, también permite trabajar con cámaras USB compatibles con Linux

-Beneficio/Costo es realmente bueno ya que la Raspberry pi una de las opciones más económicas del mercado, llegando a un costo de la implementación del hardware del sistema a un tercio en comparación con la opción brindad por la Beagle Bone Black de Texas Instruments

Conclusión

Con esta tarjeta se implementara el sistema

3.2 CÁMARA DE LA RASPBERRY PI PICAMERA

Una de las partes más importantes del proyecto es la captura de imágenes, ya que con las imágenes se realizara una data fotográfica para extraer algunos patrones para el entrenamiento de la red y también la captura de la imagen para el funcionamiento del sistema para valorar si hay en el momento un patrón o un evento de atraco.

El modulo encargado de hacer las captura de imágenes es la cámara de la Raspberry pi; Es el sensor Omnivision 5647 tiene una resolución nativa de 5 mega píxeles y tiene una lente de foco fijo a bordo. En cuanto a las imágenes fijas la cámara es capaz de capturar imágenes estáticas de 2.592 x 1.944 píxeles y también es compatible con 1080p30, 720p60 y 640x480p60 / 90 en vídeo.

Se conecta a la interfaz en serie de la cámara del Raspberry Pi (CSI) conector de bus a través de un cable plano flexible,

Figura 13. Modulo de cámara de la Raspberry pi - PiCamera.

3.3 MODULO GSM/GPRS M95

Para poder generar una alarma silenciosa se necesita notificar el estado de activación de alarma por eso se usara el modulo GSM/GPRS M95; el modulo cuenta con un chip m95 Quad-band GSM/ GPRS de la empresa quectel el cual puede realizar el envío y la recepción de SMS, voz y datos la cual se comunica por medio del protocolo de comunicación serial, este modulo tiene niveles ttl de 5v es especial para uso de microcontroladores, pero para el caso de la Raspberry pi los niveles de comunicación son de ttl de 3,3v

Figura 14. Modulo GSM/GPRS M95.

Para poder realizar la comunicación con la Raspberry pi se debe realizar un acople de niveles la cual se diseño una tarjeta auxiliar para acoplar los niveles de tensión ttl del modulo GSM con los niveles de tensión de la tarjeta de desarrollo Raspberry pi.

3.4 SEÑALES DE COMUNICACIÓN

La señal de transmisión serial entre de la tarjeta auxiliar y la Raspberry pi se aplicaría un divisor de voltaje ajustado por la ecuación 5.

$$v = \frac{r1}{r1 + r2} * vin \;\; Ec(5)$$

Para la comunicación de la Raspberry pi y la tarjeta auxiliar la señal proveniente se conectara a un arreglo transistorizado para que normalice el nivel de la señal de 3,3 a 5v

3.4.1 DIVISORES DE VOLTAJE

Como se mencionó en el ítem anterior se tienen que diseñar un divisor de voltaje:

- Divisor de 5V a 3.3V que protegerá la Raspberry pi de posibles daños por sobrecarga

Para el divisor de 5V/3.3V se toma una resistencia de referencia con un valor de 10KΩ, el valor de la alimentación de entrada es de 5V y el voltaje 2 es de 3.3V aplicando la ecuación 6 obtenemos un valor de 20KΩ para la resistencia 2.

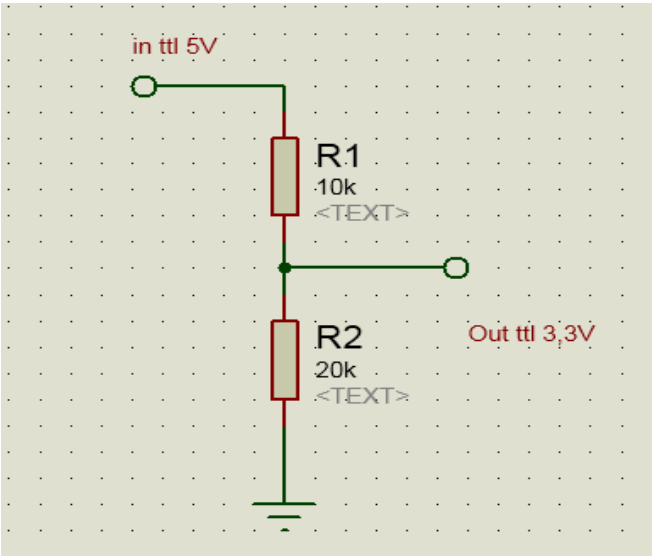


Figura 15. Divisor de voltaje 5V/3.3V

3.4.2 ARREGLO DE TRANSISTORES

Para poder normalizar la señal de salida de la Raspberry pi se conecta a un transistor en saturación con colector abierto. El cual permite un cero lógico cuando la trama que proviene de la Raspberry

por el puerto serial contiene un cero y la resistencia en pull up permite tener el un uno lógico de 5v cuando hay un 1 en la trama

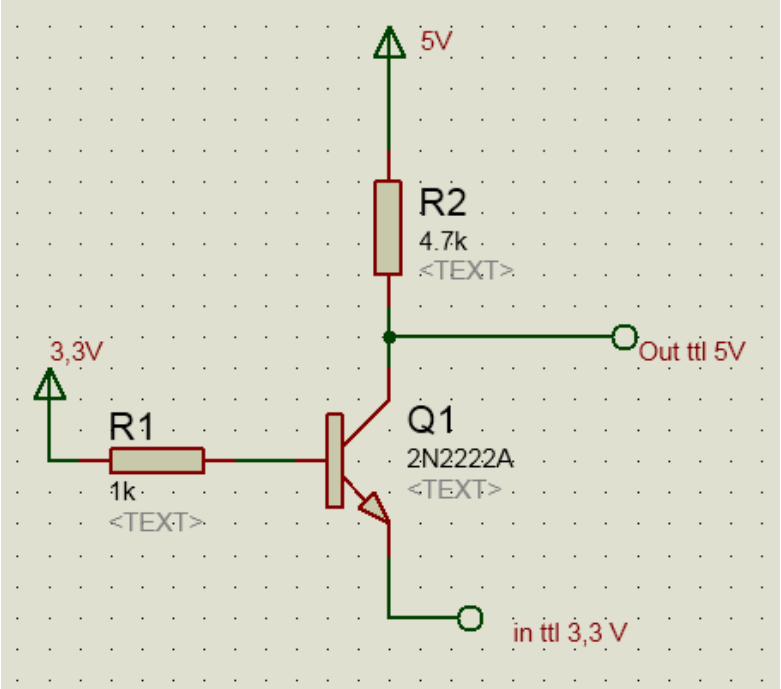


Figura 16. Acople para comunicación de 3,3V a 5V

El arreglo del transistor viene dado por el fabricante del modulo.

3.5 TARJETA AUXILIAR

El PCB de la tarjeta auxiliar y se diseñó en el software PROTEUS con su aplicación ARES, la tarjeta tiene dos capas y se respeta la distancia mínima entre los caminos de comunicación, también tiene sus respetivas polígonos de tierra y se diseño con tecnología SMD por que no se contaba con gran espacio en chasis de protección del sistema, donde se almacenara el sistema completo ya que no se quiere sobre dimensionar innecesariamente el tamaño por eso se trato de realizar lo más pequeño posible.

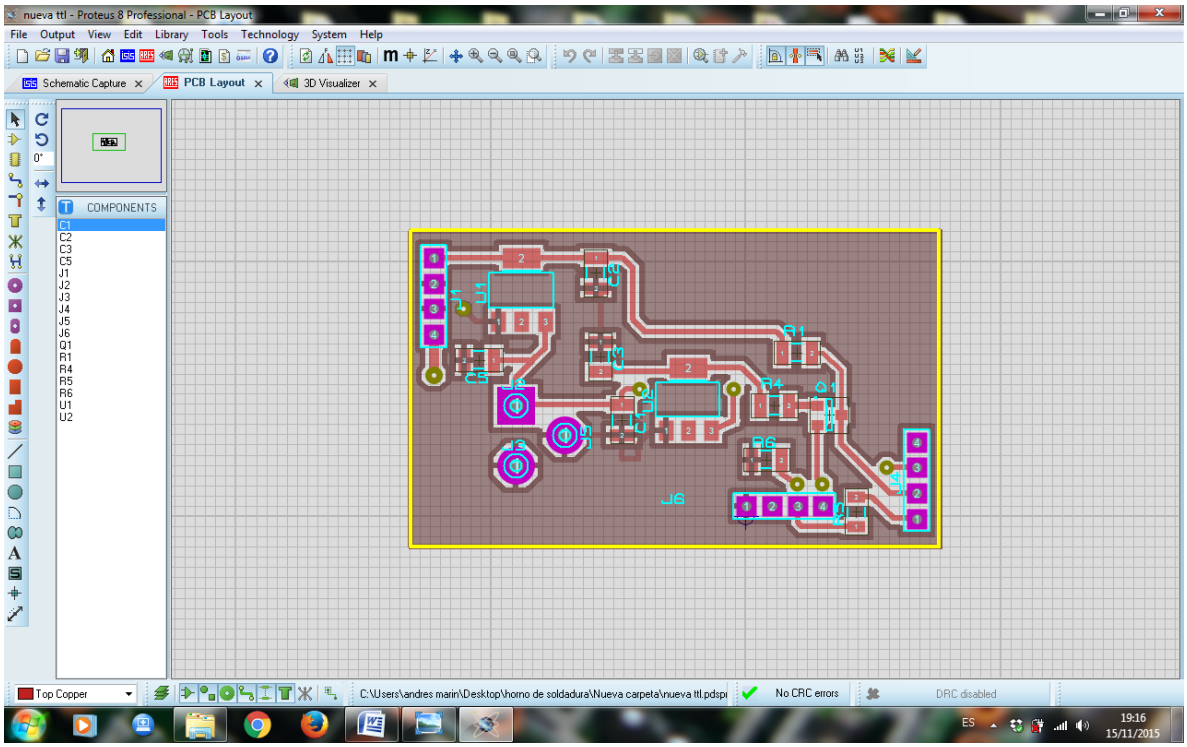


Figura 17. Vista frontal PCB tarjeta auxiliar.

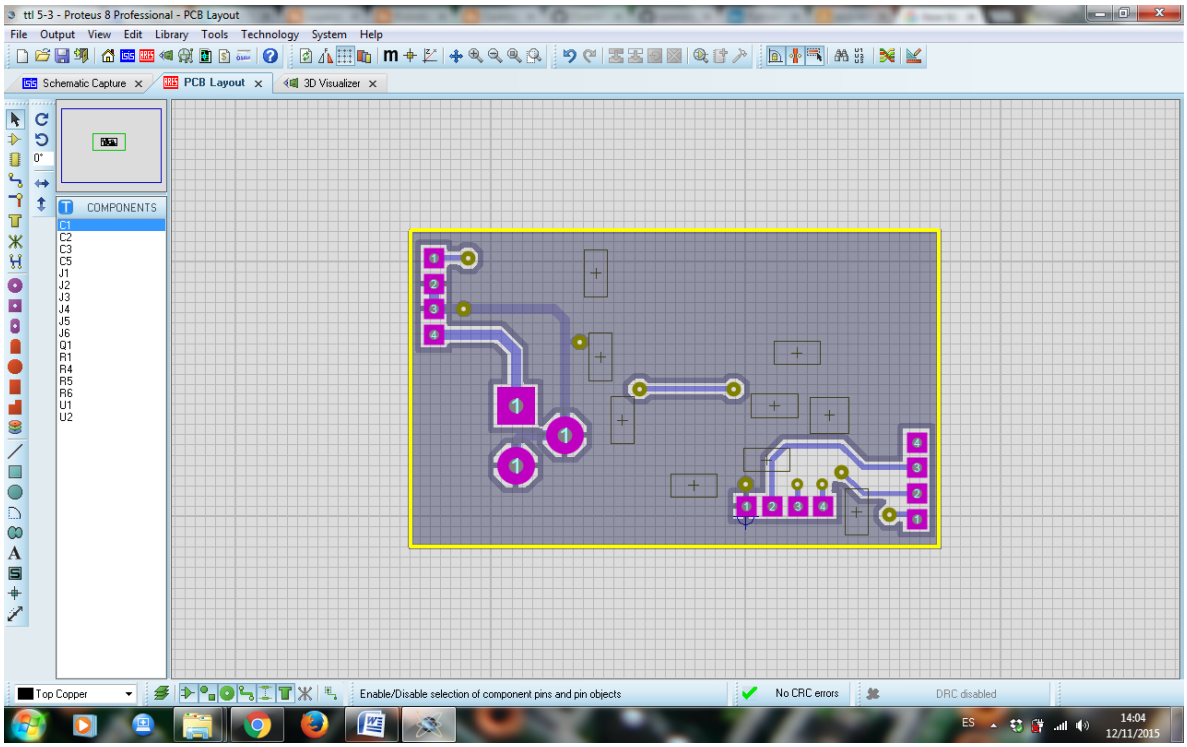


Figura 18. Vista trasera PCB tarjeta auxiliar

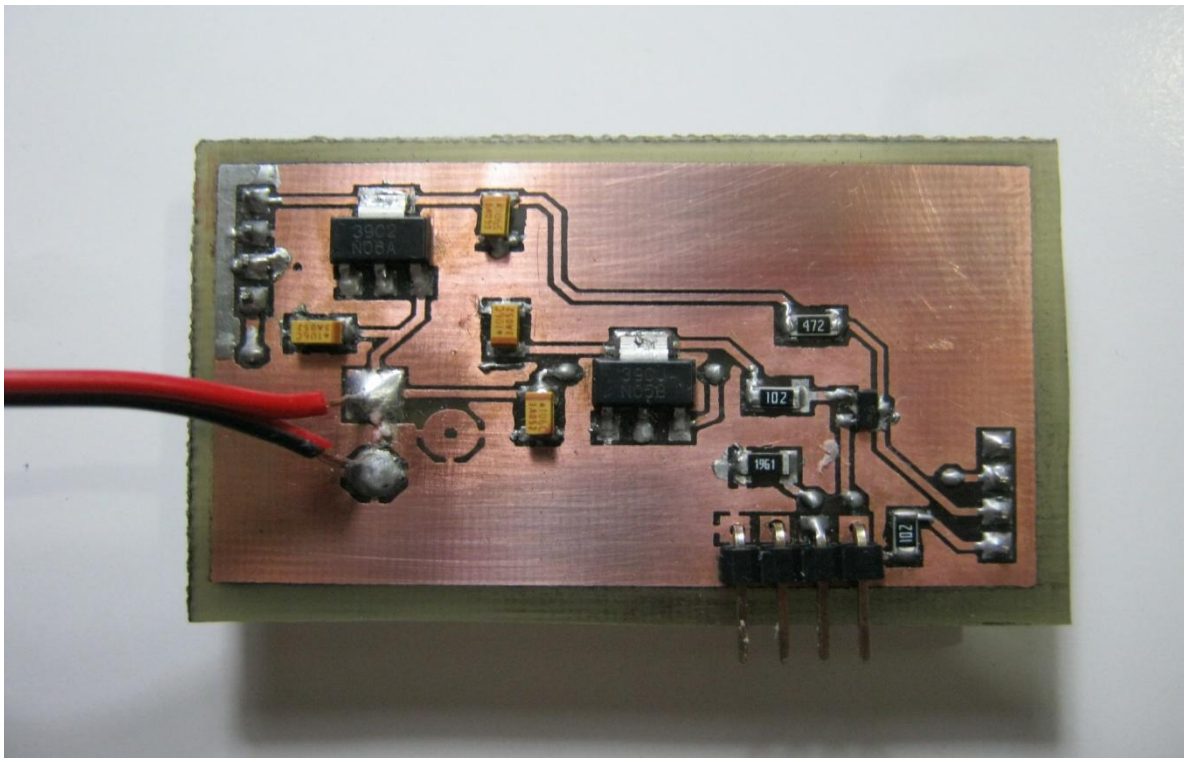


Figura 19. Circuito impreso cara posterior

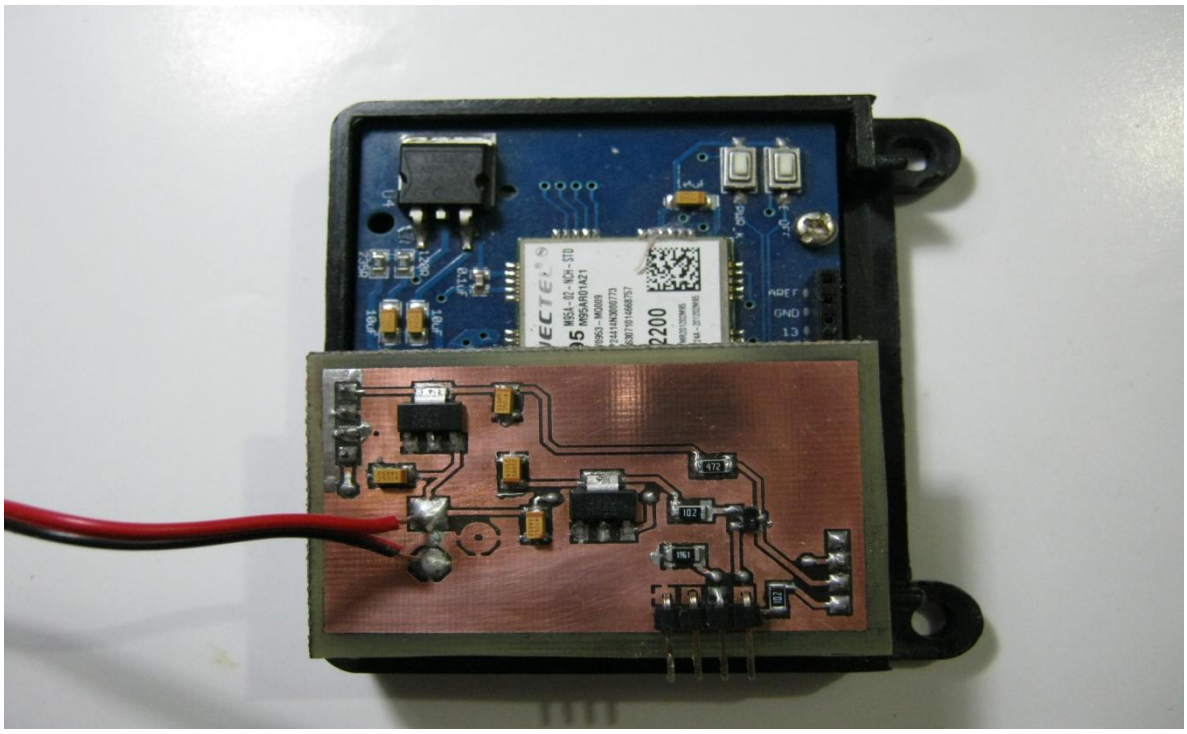


Figura 20. Circuito impreso acoplado al modulo GSM.

3.6 SISTEMA CON HARDWARE IMPLEMENTADO

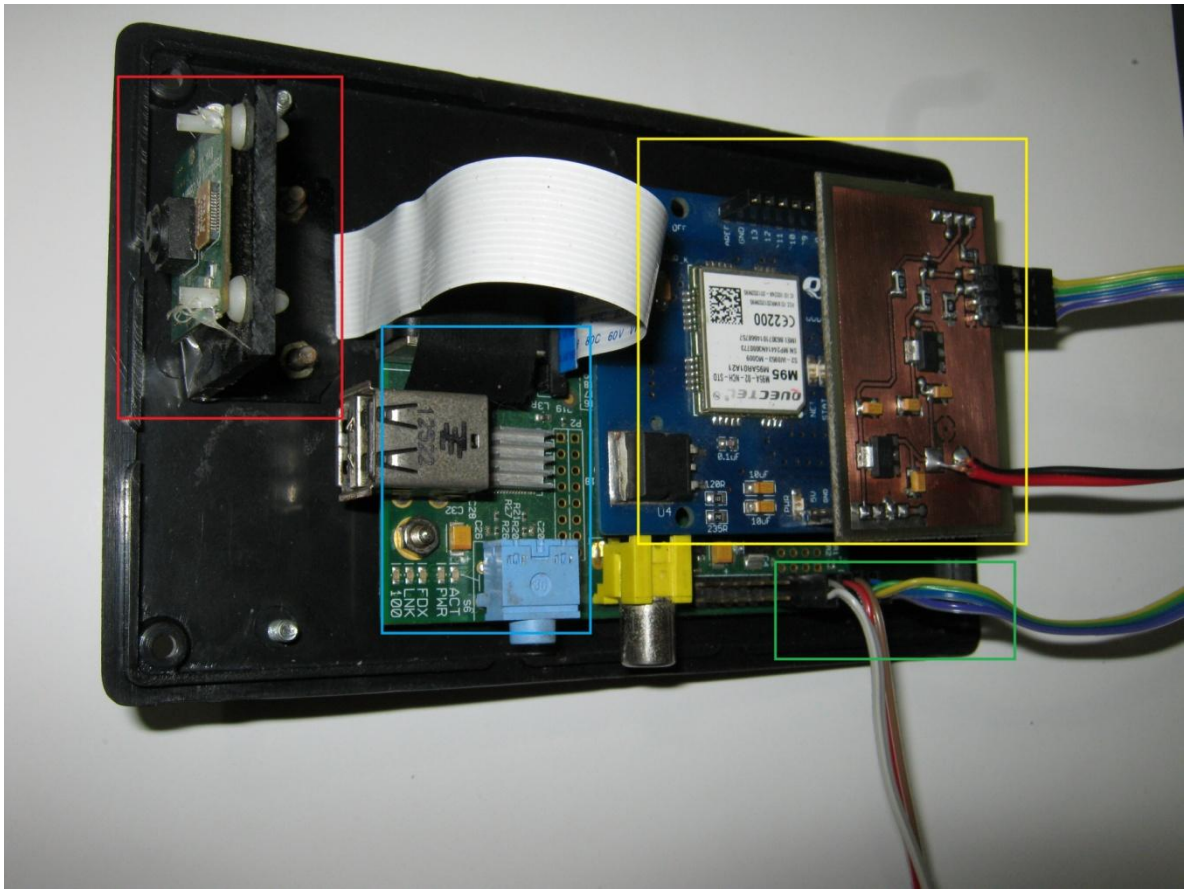


Figura 21.Imagen donde se muestran todo el sistema acoplado

En la imagen anterior podemos observar todo el sistema con sus componentes acoplados.

En el recuadro rojo se puede observar el modulo de cámara la camera py la cual es la encargada de hacerla captura de las imágenes la cual se comunica por medio de una cinta plana con la Raspberry pi por CSI (camera serial interfaz).

En el recuadro azul se encuentra la tarjeta de desarrollo la Raspberry pi es la encargada de hacer el procesamiento de la información realizar las acciones correspondientes.

El recuadro amarillo es el modulo GSM M95 con la tarjeta de auxiliar para los acoples de tensión ttl de la comunicación serial de 5v-3.3v, este modulo GSM se comunica con la Raspberry pi por medio del protocoló serial a 9600 baudios.

El recuadro verde son los pines GPIO de la Raspberry pi, entre los cuales están los pines de Tx y Rx de la comunicación serial con el modulo GSM también están dos GPIO uno como entrada con un pulsador a tierra para el apagado del sistema y uno como salida para un LED indicando el encendido del sistema y también la activación de la alarma.

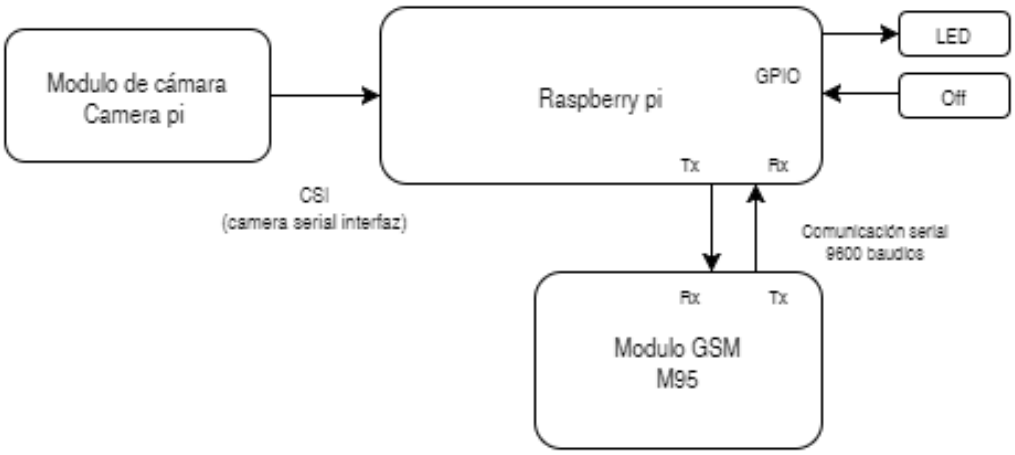


Figura 22. Diagrama de bloques del hardware del sistema

4. PROGRAMACIÓN Y DESARROLLO

4.1 CARACTERÍSTICAS DEL SISTEMA DE SEGURIDAD

El sistema realiza tres procesos muy importantes que consta de visón artificial, inteligencia artificial y notificación del estado del alarma, el primer proceso visión artificial lo que realiza es una captura de una imagen en una matriz de pixeles la cual se aplica técnicas de tratamiento de imágenes en el lenguaje python para así extraer características o patrones.

En la parte de la inteligencia artificial se implementa una red neuronal la cual va clasificar en dos clases los patrones obtenidos la clase 1 que representa que hay un evento importante un patrón de atraco y la otra clase dos la cual no se presenta ningún evento.

Notificación se realiza el caso que la red neuronal de un como salida clase 1 quiere decir un estado de alarma, la notificación se realiza por medio de tecnología GSM realizando una llamada a un numero asignado y mandando un mensaje de advertencia.

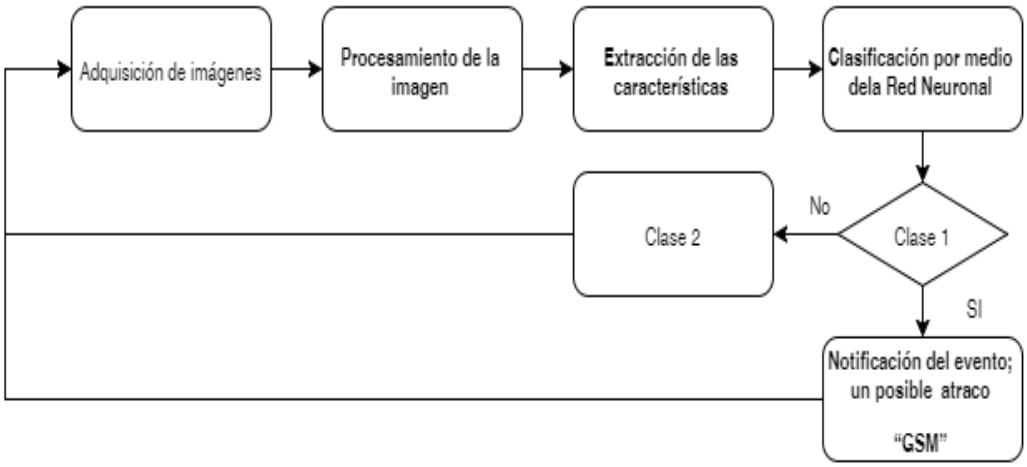


Figura 23. Diagrama de Flujo del sistema de seguridad

La figura 22 muestra el diagrama de flujo del sistema de seguridad. En él se pueden distinguir los diferentes procesos que se realizan.

4.1.1 CONFIGURACIÓN DEL HARDWARE

El sistema se trabaja con la cámara de la Raspberry pi la Picamera la cual es la encargada de entregar la matriz de pixeles para el posterior procesamiento de imágenes y extracción de características o patrones la cámara está configurada con una resolución de 640x480 y a 32 fotogramas por segundo.

La Raspberry pi tiene el sistema operativo Raspbian el cual tiene por defecto instalado el idle de python en cual se desarrollo el programa que implementa todo el sistema.

El modulo GSM es el que se encarga de la notificación de la detección de un posible evento de atraco realizando una llamada y el envió de un mensaje de texto, se comunica con la tarjeta de desarrollo la Raspberry pi por medio del protocolo de comunicación serial la cual está configurada a una velocidad de 9600 baudios y se le envían instrucciones por medio de comando AT.

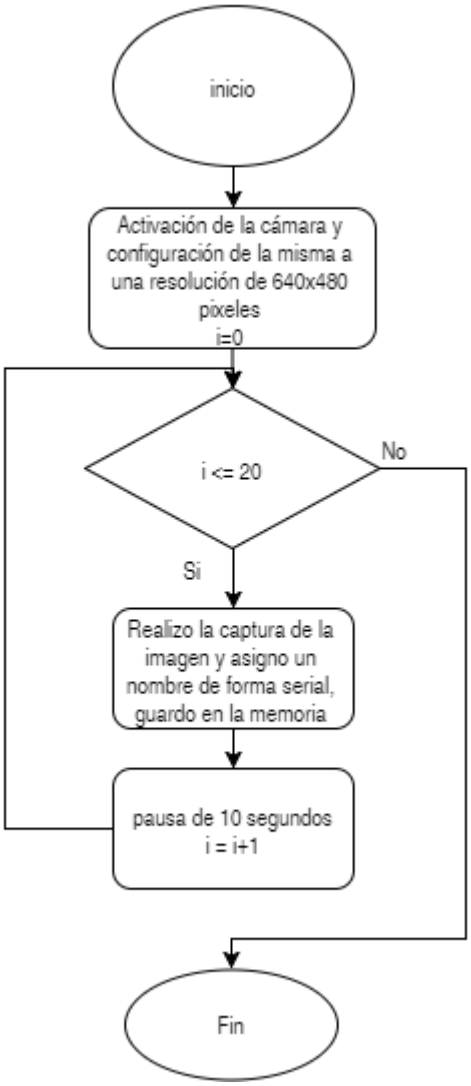
4.1.2 CONFIGURACIÓN DEL SOFTWARE

Para el proceso de adquisición de las datas que servirá como soporte para el entrenamiento de la red neuronal se utilizó un script el cual realizaba capturas de imágenes con la cámara en intervalos de tiempo la cual captura diferentes posturas de cajeros después por medio de otro script se realiza la lectura de una en una las imágenes para realizar un procesamiento de imágenes y con este sacar los patrones y luego guardarlo en un vector para sí usarlo como la data de entrada de red neuronal a entrenar.

El entrenamiento de la red se hace por medio del software Matlab el cual cuenta con una toolbox ANN que permite trabajar con redes neuronales artificiales multicapa ; luego su implementación se realiza en la tarjeta de desarrollo Raspberry pi en lenguaje python

4.2 CAPTURA DE DATOS Y EXTRACCIÓN DE PATRONES

El modulo de cámara se encarga de tomar las capturas de imágenes a una resolución de 640x480 la cual toma una serie de fotos en intervalos de tiempo en diferentes posiciones que se encuentra el cajero, unas posiciones con las manos levantadas y manos abajo entre otras; todo esto se realiza por medio de un script de python el cual a su vez le asigna en serie un nombre designado a la fotos y las almacena en la memoria de la Raspberry para su posterior tratamiento con otro script.



Algoritmo 1. Captura de imagen

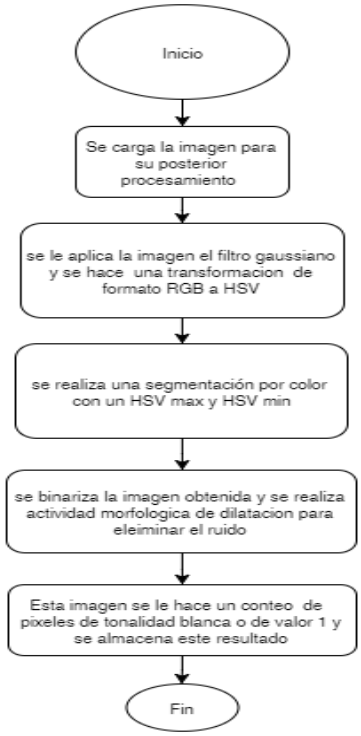
A una señal para su tratamiento se le puede mirar desde diferentes ángulos, algoritmos complejos, extracción de parámetros, potencia, energía, mediana, promedio y muchas más herramientas que calculan la información vital de la señal.

MATLAB necesita de estos patrones para poder crear la red neuronal, este proceso se realiza mediante un algoritmo propio para tal fin.

Para la extracción de las patrones se utilizo un script el cual consiste en leer las imágenes una por una y realizarles un tratamiento de imágenes; el proceso se realiza de la siguiente forma se toma la imagen se carga en una variable queda guarda como una matriz de pixeles a la cual se le aplica un filtro para reducir el ruido él un filtro gaussiano que consiste en difuminar la imagen después se realiza una cambio en formato de la imagen RGB a HSV (Hue, Saturación, Value – Matiz, Saturación, Valor),se realiza después una segmentación o separación de pixeles en los cuales se establecen unos rangos mínimos y máximos de HSV se obtiene una imagen solo con valores entre estos dos rangos y después realizamos una binarización generando una imagen en blanco y negros, aplicamos a la imagen obtenida de la binarización técnicas de morfológicas para minimizar el ruido presentado en la imagen para ser exacto se aplica la técnica de dilatación, después de esto lo que se realiza es el conteo de numero de pixeles de tonalidad blanca para obtener cuantos hay en la imagen este valores son los que se almacenan estos son la data de entrada para nuestra red, las imágenes también se clasifican en dos clases son las clase 1 donde se encuentran las manos alzadas y clase 2 cuando no se encuentran estas.

Figura 24. Foto que pertenece a la Clase 1

Figura 25. Foto que pertenece a la Clase 2



Algoritmo 2. Extracción de patrón para entrenamiento de la red neuronal

Figura 26. Procesos de tratamientos a las imágenes

De izquierda a derecha y de la parte superior 1.Filtro gaussiano, 2.RGB a HSV, 3.Binarización, 4. Dilatación, 5.Contorno en las regiones donde se encuentra piel.

4.3 RED MLP EN MATLAB

MATLAB cuenta con la toolbox ANN que permite trabajar con redes neuronales artificiales multicapa. En un algoritmo sencillamente se asigna con que dígitos, vectores o matrices la red va a entrenar y con cuales va a validar. Queda a disposición del usuario configurar cuantas neuronas va a utilizar, función de activación de cada una de las capas, iteraciones entre neuronas, etc. Una vez la toolbox ha finalizado el proceso en una estructura llamada net entrega toda la información de la red, y de ahí es precisamente donde se obtienen los pesos y los umbrales para llevarlos al código fuente.

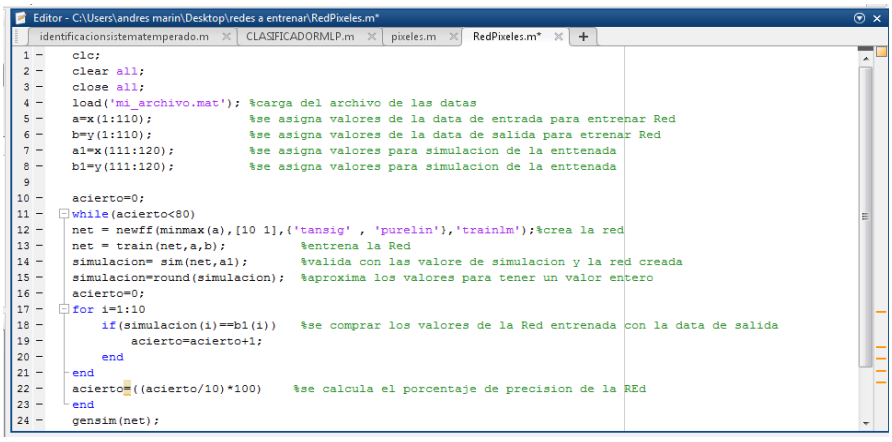


Figura 27. Algoritmo Red MLP por partes

En la figura se implementa lo explicado en la parte superior, con los comentarios se va explicando que sucede en cada parte del código

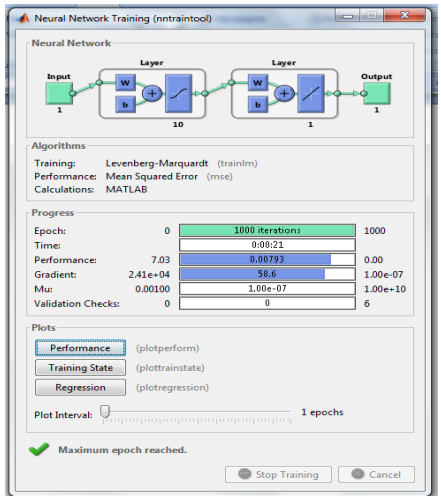


Figura 28. Entrenamiento de la red neuronal

En la figura 27 se observa la toolbox de redes neuronales de MATLAB durante el entrenamiento. Una vez que él ha encontrado los pesos que garantizan por lo menos el 80% de efectividad, la toolbox se detiene y envía los resultados de la red, la red tuvo una efectividad del 100%.

4.3.1 PESOS DE LAS NEURONAS Y UMBRALES

Una vez entrenada la red, el toolbox envía una estructura donde se encuentran los valores de los pesos en la capa oculta, capa de salida y los umbrales respectivos. Esos valores son los que son importantes pues en base a ellos es que se diseña el programa en python para la clasificación de las imágenes.

Capa oculta	Pesos	Umbrales
Neurona 1	0.00063690537948609988	*-34.595241588554458*
Neurona 2	0.00069383250867322444	*-31.472217865096731*
Neurona 3	0.0010999582029513032	*-28.358692664989629*
Neurona 4	0.00097936454544321461	*-25.270443619169349*
Neurona 5	0.00086179536153380219	*-22.221131470931649*
Neurona 6	-0.0010563193415692575	*19.06371374365898*
Neurona 7	0.00067153887485281287	*-15.929847459764478*
Neurona 8	-0.0023416798687257157	*12.815464992205477*
Neurona 9	0.00039426913567160561	*-9.6994310010046743*
Neurona 10	-0.00029932600039820809	*6.5946223228846508*

Tabla 1. Valores de los pesos y umbrales de las neuronas de la capa oculta

Capa de salida	Pesos	Umbral
Neurona 1	-0.53955587362185076	0.59996233174378866
	-0.041548972792197852	
	-0.027133929511937636	
	-0.025541960559497409	
	-0.35495366295232977	
	-0.0014669233771898696	
	0.19521173396125052	
	-0.20279202274184874	
	-0.43636828431351316	
	-0.26397726179189807	

Tabla 2. Valores de los pesos y umbral de la capa de salida.

Estos son los valores que se usaran en código python para la implementación de la red neuronal en la Raspberry pi la cual será la encargada de realizar la clasificación de las imágenes.

4.3.2 RESULTADOS DE ENTRENAMIENTO

En el entrenamiento de la red se puede simular para saber la efectividad de esta misma, con la función sim de Matlab se hacen una comparación de la salida de la red con la data de salida correspondiente para verificar la efectividad de la red.

Se tienen 10 muestras de las 2 clases, estas son aleatorias y se pasan por la red clasificadora. Ella entrega un resultado y de ahí se evalúa su efectividad.

Nombre de la Foto	Resultado RN	Resultado Teórico
FotoRED320	Clase 2	Clase 2
Foto RED19	Clase 1	Clase 1
Foto RED20	Clase 1	Clase 1
FotoRED619	Clase 1	Clase 1
FotoRED319	Clase 2	Clase 2
FotoRED220	Clase 1	Clase 1
FotoRED420	Clase 2	Clase 2
FotoRED620	Clase 1	Clase 1
FotoRED219	Clase 1	Clase 1
FotoRED520	Clase 2	Clase 2

Tabla 3. Resultados validación en Matlab

Este proceso se puede hacer con todas otras muestras y va a dar el mismo porcentaje de efectividad del 100% presente en la tabla 3, lo que confirma que la red neuronal quedo bien estructurada y hace un buen proceso clasificador.

4.4 ESTRUCTURA DEL PROGRAMA PRINCIPAL EN PYTHON

La estructura del programa principal como se había mencionado con anterioridad consta de tres procesos importantes todos ejecutados en uno solo script implementado en la Raspberry pi escrito en el lenguaje de programación python y se hace uso de la Liberia OpenCV.

Para recordar lo tres procesos consisten: 1.vision artificial, 2. inteligencia artificial, 3. notificación del estado de alarma.

Visión artificial

Ahora en el programa principal la cámara se trabaja como video a la resolución de 640x480 y 32 fotogramas por segundo, en la cual se captura una imagen esta imagen es cargada a una variable como una matriz de pixeles por medio de la librería de OpenCV en lenguaje de programación python se realiza una serie de tratamientos el primero es para reducir el ruido es un filtro conocido como filtro gaussiano el cual difumina la imagen obtenida, posteriormente se realiza una trasformación a la imagen de RGB a HSV para hacer una segmentación con uno valores mínimos y máximos HSV para así realizar una separación de pixeles de tonalidades especificas, luego a esta imagen obtenida se le realiza una binarización la cual lo que hace es pasar los pixeles de obtenidos con valor de 1 y el resto de la imagen con un valor de cero así obteniendo una imagen es blanco y negro, para eliminar el ruido y mejorar la imagen que se obtuvo se realiza una técnica de morfología conocida como dilatación, ya después de todo esto se realiza un conteo de pixeles de valor de 1 con esto valor es el patrón que luego se introducirá para que nuestra red implementada en código la clasifiqué.

Red Neuronal

En la red neuronal se tiene las 10 neuronas de la capa oculta y la neurona de la capa de salida, la función de activación para las primeras es Tansig y para la de salida es Purelin.

Una vez que llega el valor del patrón extraído en el anterior proceso este se pasa por las 10 primeras neuronas donde se multiplican por sus pesos, se le suma el umbral y se calcula la función Tansig, la salida de cada neurona de la capa oculta es una entrada de la neurona de salida. En la neurona de salida las señales provenientes de la capa oculta se multiplican por los pesos se suma el umbral y la salida al ser Purelin, la entrada es la misma salida

Todo lo anterior se implementa en código python el cual se realiza con gran versatilidad ya que la Raspberry pi cuenta con punto flotante y python tiene librerías matemáticas que permiten el uso directo de la función exponencial, se realiza una función la cual permita calcular el valor de la función tangente hiperbólica sigmoidea

La función tangente hiperbólica sigmoidea (Tansig) se muestra en la figura 28.

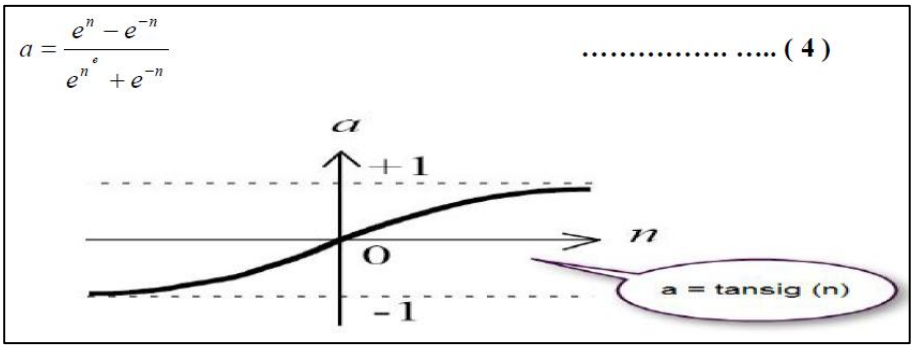


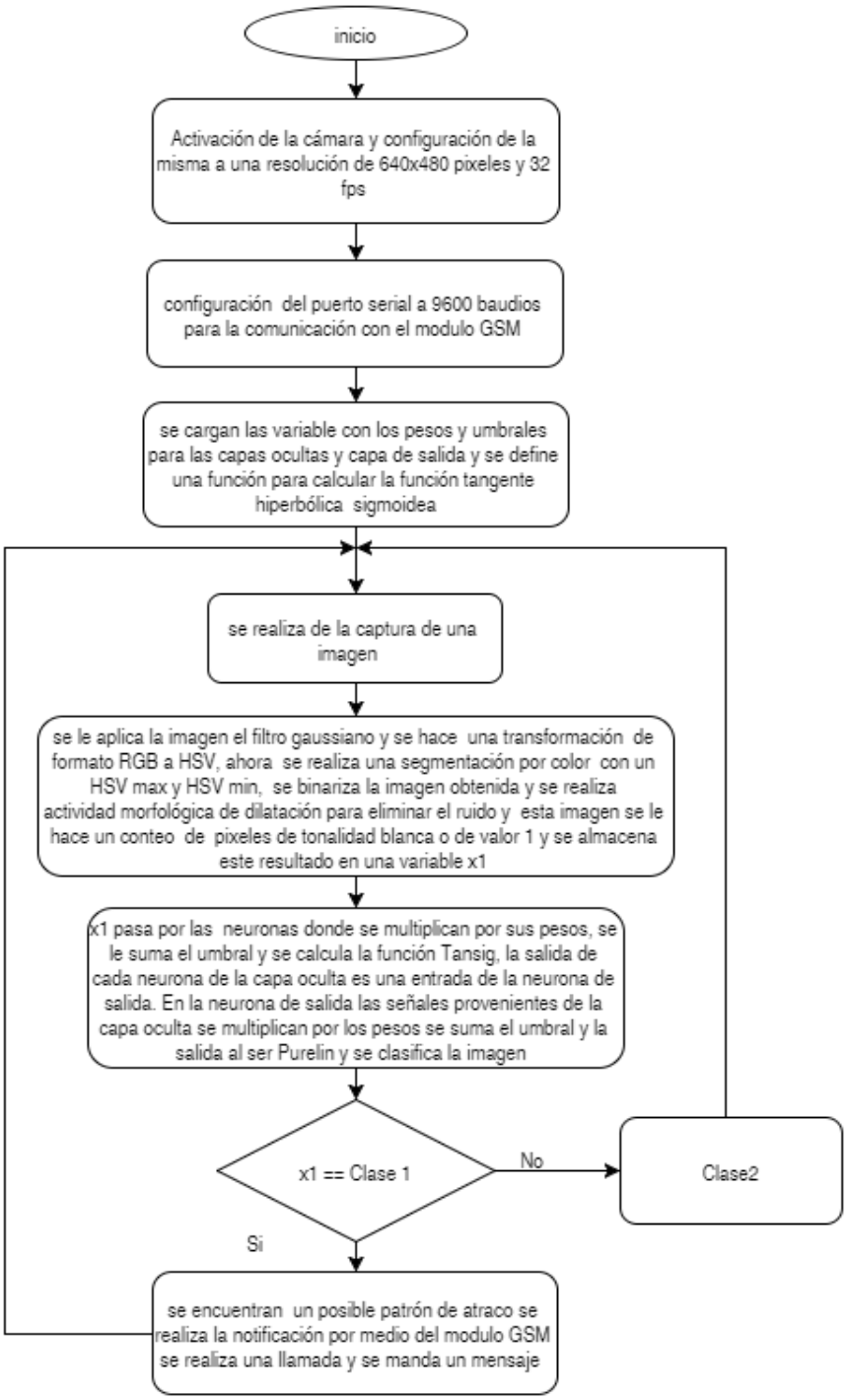
Figura 29. Función tangente hiperbólica sigmoidea [12]

Lo que realiza la red en este proceso es la clasificación de la imagen por medio del patrón obtenido en dos clases, **clase 1** significa que se presenta un posible evento de atraco y la **clase 2** no hay ningún evento

Notificación del estado de alarma

Cuando se obtiene un clasificación en una imagen de clase 1 significa que puede haber un posible evento de atraco, se necesita hacer una notificación al personal encargado; esta activación debe ser silenciosa para no genera más inconvenientes o repercusiones negativas al cajero por eso se trabajo con tecnología GSM el cual realiza una llamada y el envió de un mensaje notificando el estado del sistema, el modulo GSM se comunica con la tarjeta de desarrollo Raspberry pi con el protocolo de comunicación serial y con comandos AT

Si la clasificación de la red fue de clase 2 significa que no hay ningún evento y el programa realizara de nuevo todo el proceso como un bucle infinito repitiendo las operaciones anteriores, también lo hará cuando haya notificado el evento de clase 1



Algoritmo 3. Programa principal

5. ANÁLISIS Y RESULTADOS

5.1 ANALISIS Y RESULTADOS

En esta parte del libro se presentan los resultados del proyecto en su parte final, los inconvenientes, porcentajes de error, validaciones y alternativas de desarrollo que se tuvieron en cuenta para implementar el proyecto completamente.

Asegurado de que la información que se pierde es mínima y no afectara en nada el proceso de clasificación de la red neuronal.

5.2 VISIÓN ARTIFICIAL Y RED NEURONAL EN PYTHON

Antes de unir todos los programas y hacer una sola validación es necesario probar el componente principal del proyecto que es la red clasificadora y la visión artificial para extraer el patrón con el cual va a clasificar la red para asegurar de que esté funcionando correcto.

La prueba para poder verificar que el algoritmo está funcionando correctamente es relativamente simple. Se trabaja con unas fotos que se habían dejado para la simulación de la red. Este significa que se evaluaran con fotos que no fueran parte del entrenamiento que la red no conoce que son totalmente nuevas; lo que se realiza en el script de Python es que se realiza el proceso del tratamiento de imágenes el cual extrae el patrón y el script luego lo pasa por la red neuronal la cual hace su proceso de clasificación de la foto en las dos clases conocidas y si el resultado de la clasificación es clase1 hace la correspondiente notificación realiza una llamada y envía un mensaje.

Si la red neuronal en su entrenamiento fue optimo, se encontrara en la capacidad de clasificar las nuevas fotos que se capturen por la cámara a pesar de que estas sean totalmente nuevas y en diferentes circunstancias por la naturaleza propia de la red neuronal de tener la capacidad de clasificar

El script como tal alcanza las 112 líneas de programación y el tiempo de ejecución de una validación es un promedio de 530 ms y 8.6%

Raspberry pi	Consumo
RAM	8,6%(de 512Mb)
Líneas de código	112
Tiempo de ejecución	530ms

Tabla 4. Recursos utilizados por la Raspberry pi

En la tabla 4 se encuentran los resultados de tiempo de ejecución, líneas de códigos del programa, recursos de memoria de la Raspberry pi en la validación del script del sistema de seguridad.

5.2.1 LIMITANTES

Una de las mayores limitante del sistema es que se realiza unos tratamientos de imágenes y se encuentran sujetados a sus limitantes como son a una ambiente bien iluminado pero este se resuelve presentado o una adecuada iluminación en el recinto que se encuentre y con técnicas incluidas en el sistema para eliminar el ruido presente por la iluminación, el sistema la iluminación adecuado para el sistema debe contar con dos bombillas F48T12 de 39w de 2500 lúmenes, 6500 k de color designado es Daylight ; la otra limitante es más una condición de funcionamiento la cual se requiere para que el sistema trabaje adecuadamente es el de una distancia establecida y fijada entre el sistema y cajero en el encargado de recaudar el dinero, la distancia adecuada es 1 metro (±20cm)

5.2.2 RESULTADOS Y PORCENTAJE DE ERROR

Los resultados de la clasificación cuando se evalúa las fotos se imprime en pantalla mostrando la clase que pertenece la foto y se realiza una llamada si pertenece a clase 1 estos fueron los resultados obtenidos con las fotos de la simulación las cuales son diez y fueron escogida aleatoriamente.

Nombre de la Foto	Clases de las Muestras	Resultado obtenido
FotoRED320	Clase 2	Clase 2
Foto RED19	Clase 1	Clase 1
Foto RED20	Clase 1	Clase 1
FotoRED619	Clase 1	Clase 1
FotoRED319	Clase 2	Clase 2
FotoRED220	Clase 1	Clase 1
FotoRED420	Clase 2	Clase 2
FotoRED620	Clase 1	Clase 1
FotoRED219	Clase 1	Clase 1
FotoRED520	Clase 2	Clase 2

Tabla 5*. Resultados validación con datas.*

El porcentaje de error obtenido fue del 0%, las 10 fotos de muestras aleatorias que se les realizo el tratamiento y se tomaron sus patrones obtenidos y la red implementada en python dio una clasificación positiva, sin importar el orden de las muestras

Con este resultado es de esperar que al unir todos los algoritmos el resultado sea igual de efectivo que al hacerlo solamente con fotos de muestras sino con video y captura de imágenes en tiempo real.

5.3 SISTEMA DE SEGURIDAD RESULTADOS

La primera prueba se hace en tiempo real la cámara está captando en video con una resolución 640x480 a 32 fotogramas por segundo y se pasa por la red la cual evalúa las imágenes obtenidas si es clase 1 realiza la llamada y en el envió del mensaje y si es clase 2 se repite el proceso.

En las pruebas realizadas cuando se sostiene las manos elevadas marca clase 1 en todas las pruebas y cuando se encuentran abajo marca clase 2 en todos los casos. Por esto se puede afirmar que se ha obtenido un error del 0%.

Figura 30. Sistema de seguridad realizando una validación

Figura 31.imagen del resultando de la validación

Como se ilustra en la figura 30 se alcanza apreciar un contorno de color azul y unos recuadros de color verde en la imagen obtenida en la validación sobre las aéreas detectadas rostro y manos.

Durante la validación se pudo apreciar un lag en las imágenes resultantes, hay que aclarar que el lag es del video de las imágenes resultantes y no del proceso de la cámara el cual ocurre en un tiempo menor a un segundo, este lag se presume que es debido a la interfaz grafica ofrecida por python ya que se miro en las características de las tarjeta, esta maneja del vector de punto flotante es de doble precisión para ver las característica.

http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0301h/index.html

5.4 ANÁLISIS ECONÓMICO

Para poder justificar la viabilidad del proyecto hay que tomar el sistema de seguridad por zonas los cuales están a cargo de vigilar zonas restringidas y tecnología GSM su valor oscila entre 900.000 pesos hasta los 3,000.000 de pesos y compararlo por aparte con el Sistema de seguridad diseñado.

A continuación en la ***Tabla 6*** aparecen los costos del proyecto.

COMPONENTE	**PRECIO**
TARJETA Raspberry pi	148.500
Modulo de cámara Picamera	96.600
Modulo GSM M95	92.000
Memoria sd 8Gb	20.000
Caja de protección	6.000
Fuete de alimentación	30.000
TOTAL	375.150

Tabla 6. Inversión del proyecto

SISTEMA DE SEGURIDAD POR ZONAS	**SISTEMA DE SEGURIDAD DISEÑADO**	**DIFERENCIA**	**PORCENTAJE DE AHORRO**
900.000	375.150	524.850	58.31%

Tabla 7. Comparación sistemas del sistema de seguridad

En la tabla anterior se puede apreciar que sistema de seguridad desarrollado representa un ahorro mayor al del 50% solo comparando con la versión más económica del sistema de seguridad por zonas.

6. CONCLUSIONES

6. CONCLUSIONES FINALES

Se seleccionó la tarjeta de desarrollo Raspberry pi debido que es una herramienta muy poderosa y versátil la cual permite diferentes entornos de programación y plataformas teniendo a la mano un tarjeta de desarrollo que entrega la facilidad de interconectar trabajos de electrónica, con un sistema operativo para el trabajo de proyectos complejos; Como es el caso del lenguaje Python con la Liberia OpenCV que permiten la implementación de visión artificial y inteligencia artificial en un sistema embebido y también cuenta con vector de punto flotante.

Otro factor importante es el bajo costo en el cual se puede adquirir comparado con otras tarjetas en el mercado y sin olvidar mencionar la tecnología que ofrecen permite el desarrollo de nuevos proyectos que solo hubieran sido posible realizando una gran inversión o destinar una computadora en el proceso haciendo inviables los proyectos.

Uno de los mayores inconvenientes que se encontró en el proyecto fue el ambiente el cual al no ser un ambiente totalmente controlado permitiría la introducción de ruido en el tratamiento a variables indeseados los cuales se podrían trabajar con otra técnicas de tratamientos, pero los cuales presentan inconvenientes son técnicas relativamente nuevas y no se consigue gran documentación para su implementación y requieren un alto costo de rendimiento de un procesador.

Por esto se opta por un tratamiento no tan complejos para el tiempo de ejecución no sea muy grande y así poder hacer el programa en tiempo real; también se debe considerar que se está trabajando con un potente ARM cortex a 32 bit, pero el cual se ve corto con algunos procesos en el tratamiento de imágenes, estos procesos a un se ven cortos en algunos computadores portátiles.

Esto no quiere decir que en la Raspberry no se pueda implementar tratamiento de mayor complejidad lo que se requiere dar entender es que en este proyecto el factor tiempo jugaba un valor muy importante imposibilitando usar varias técnicas, Pero un proyecto donde este no juego un papel principal se podrían llevar a cabo trabajos de mayor complejidad.

Se realizó la implementación visión artificial y de una red neuronal en un sistema embebido gracias a las prestaciones de Raspberry y la versatilidad y la optimización de la librería para visión artificial OpenCV y de más librerías requeridas en el lenguaje Python, llegando a permitir realizar un sistema de seguridad utilizando inteligencia artificial y visión artificial.

Se implementó el sistema de adquisición de data fotográfica para el soporte de entrenamiento de la red, por medio de un script de python el cual toma una serie de fotos con el modulo de cámara de la Raspberry pi en intervalos de tiempo con diferentes posturas del cajero entre ellas manos arriba y manos abajo, las fotos se nombra automática y se guardan en la memoria de la Raspberry pi, se adquirieron 120 fotos para el entrenamiento de la red la imágenes tienen una resolución de 640x480 pixeles

El entrenamiento de la red neuronal se realizo con la data fotográfica obtenida y por medio del software matemático MATLAB, la red que se entreno fue una MLP (MultiLayer Perceptron) la cual después de haber realizado el entrenamiento de la red y simular obteniendo un 100% de efectividad, después de realizado el entrenamiento de la red neuronal se implemento en la Raspberry pi en código de programación python

Las validación del sistema fue exitosa en la cual el sistema realiza una clasificación de las imágenes obtenidas, si el cajero se encuentra con las manos arriba o manos en el pecho da una clasificación de clase 1 la cual hace a una notificación por medio del modulo GSM y manos abajo da una clasificación de clase 2, se obtuvo un 100% de efectividad del sistema.

BIBLIOGRAFIA

[1] Ortiz Sandoval Jesús Eduardo, “CLASIFICACIÓN DE COMPUESTOS QUÍMICOS USANDO UN SISTEMA MULTISENSORIAL (NARIZ ELECTRÓNICA) DESARROLLADA SOBRE UN DISPOSITIVO H,” pp. 13–21, 2013.

[2] Ms. C. Esperanza, I. S. Cesar, Calderón, and I. Milton, “SMART HOME CONTROL BY VOICE USING NEURAL NETWORKS,” vol. 1, pp. 16–20, 2015.

[3] E. S. Blanco, A. R. Francisco, and R. A. Edgar, “Personal de la Revista,” *Desarro. un Sist. Detección Mov. basado en Flujo Óptico en Raspberry Pi*, vol. 4, pp. 65 – 76, 2015.

[4] C. M. Andrés Ernesto López Sandoval and Martínez, “la mecatronica en mexico,” *Sist. Autenticación Facial Median. la Implementación del Algoritm. PCA Modif. en Sist. embebidos con Arquit. ARM*, vol. 4, pp. 53–64, 2015.

[5] J. A. Triñanes, J. Torres, and A. T. C. Hernández, “Clasificación de imágenes multiespectrales mediante redes neuronales,” *Rev. Teledetec.*, pp. 1–5, 1994.

[6] E. M. i García, *Visión Artificial*, 1st ed. 2012.

[7] X. O. Basogain, “Redes Neuronales Artificiales Y Sus Aplicaciones,” *Med. Intensiva*, vol. 29, no. 1, pp. 13–20, 2005.

[8] D. J. Matich, “Redes Neuronales: Conceptos Básicos y Aplicaciones.,” *Historia Santiago.*, p. 55, 2001.

[9] TEXAS INSTRUMENTS. ARM Cortex-M4F based MCU TM4C1294 Connected LaunchPad. [En línea]. http://www.ti.com/tool/ek-tm4c1294xl**.** [citado el 25 de Octubre de 2015].

[10] TEXAS INSTRUMENTS. BeagleBone Black Developmentacional. [En línea]. http://www.ti.com/tool/beaglebk. [citado el 25 de Octubre de 2015]

[11] RASPBERRY PI. Raspberry pi 2 . [En línea]. https://www.raspberrypi.org/blog/raspberry-pi-2-on-sale/. [citado el 25 de Octubre de 2015]

[12] KHALIL, Rafid Ahmed, AL-KAZZAZ, Sa'ad Ahmed. Digital Hardware Implementation of Artificial Neuron Model Using FPGA. En: Department of Electrical Engineering, University of Mosul, Mosul, Iraq.

[13] R. González Duque, "Python para todos," *Web B.*, p. 108, 2000.