CONTROL DE NIVEL DEL LÍQUIDO DE BOTELLAS DE COCA COLA EMPLEANDO UNA BEAGLEBOARD

MARÍA MARGARITA GONZÁLEZ RAMÍREZ
JOSEPH FERNANDO LÓPEZ PARADA

UNIVERSIDAD PONTIFICIA BOLIVARIANA
FACULTAD DE INGENIERÍAS
ESCUELA DE INGENIERÍA ELECTRÓNICA
FLORIDABLANCA
2013

CONTROL DE NIVEL DEL LÍQUIDO DE BOTELLAS DE COCA COLA EMPLEANDO UNA BEAGLEBOARD

MARÍA MARGARITA GONZÁLEZ RAMÍREZ
JOSEPH FERNANDO LÓPEZ PARADA

PROYECTO DE GRADO

DIRECTOR:
Msc. JUAN CARLOS VILLAMIZAR RINCÓN

UNIVERSIDAD PONTIFICIA BOLIVARIANA
FACULTAD DE INGENIERÍAS
ESCUELA DE INGENIERÍA ELECTRÓNICA
FLORIDABLANCA
2013

Nota de Aceptación:

_____
_____
_____
_____
_____
_____

_____
Firma del presidente del jurado

_____
Firma del Jurado

_____
Firma del Jurado

Floridablanca, 9 de Abril de 2013

Agradecemos a Dios por darnos la vida, la fe y confianza necesaria para superar los momentos difíciles a lo largo de la carrera.

A nuestros padres por brindarnos amor, apoyarnos incondicionalmente, confiar en nosotros y darnos sus consejos que nos han convertido hoy en día en unas grandes personas.

A nuestros familiares más cercanos que se sentían orgullosos de lo que éramos y nos alentaban para seguir adelante.

A cada uno de nuestros amigos, por hacer parte de este camino. Con los cuales compartimos momentos maravillosos.

A cada uno de los profesores de la facultad que nos dieron su tiempo y conocimientos valiosos, algunos utilizados en el proyecto de grado. Otros, esperamos utilizarlos más adelante.

Joseph F. López
Ma. Margarita González

AGRADECIMIENTOS

El autor expresa sus agradecimientos a:

- Al profesor Juan Carlos Villamizar por su amabilidad, apoyo, dirección y supervisión a lo largo del proyecto.
- A la Ing. Leidy Olarte por facilitar los equipos necesarios para el desarrollo de nuestro proyecto de grado.
- A Tito Quintero por la elaboración de toda la carpintería del prototipo.

TABLA DE CONTENIDO

LISTA DE FIGURAS

Pág.

LISTA DE TABLAS

LISTA DE ECUACIONES

Pág.

LISTA DE ANEXOS

Pág.

GLOSARIO

GUI: Interfaz gráfica de usuario (GraphicalUser Interface).

HSV: Matiz Saturación Valor (Hue,Saturation,Value).

PLC: Controlador lógico programable (Programable LogicController).

RGB: Rojo Verde Azul (Red Green Blue).

RTU: Unidad de Terminal Remota (Remote Terminal Unit).

TCP: Protocolo de control de transmisión (Transmission Control Protocol).

RESUMEN GENERAL DEL TRABAJO DE GRADO

TITULO: **CONTROL DEL NIVEL DEL LÍQUIDO DE BOTELLAS DE COCA COLA EMPLEANDO UNA BEAGLEBOARD.**

AUTOR(ES): **JOSEPH FERNANDO LOPEZ PARADA.**
MARÍA MARGARITA GONZÁLEZ RAMÍREZ.

FACULTAD: **INGENERIA ELECTRONICA.**

DIRECTOR(A): **JUAN CARLOS VILLAMIZAR RINCON.**

RESUMEN

Se desarrollóla automatización del control de nivel de llenado para botellas de Coca-Cola, a través del procesamiento digital de imágenes. Para ello, se utilizó un sistema embebido en la Beagleboard, sobre el sistema operativo Ubuntu, Linux. Además, de la creación de un prototipo que simula las condiciones físicas de la planta de Bucaramanga.

El sistema embebido tiene implícito un programa que permite el procesamiento de la imagen, pudiendo determinar si el nivel que cuenta la botella es el adecuado. De lo contrario, emite una orden para su rechazo.

Este programa fue desarrollado en Python, utilizando las librerías de OpenCV para el reconocimiento y manejo de la cámara. A su vez, extensiones del mismo para entablar una comunicación con el PLC mediante el protocolo Modbus TCP.

Palabras claves: Procesamiento de Imagenes, automatización y sistemas embebido.

ABSTRACT OF THE THESIS PROJECT

TITTLE: **LIQUID LEVEL CONTROL OF COCA COLA BOTTLES, USING A BEAGLEBOARD.**

AUTHORS: **JOSEPH FERNANDO LOPEZ PARADA.**
MARÍA MARGARITA GONZÁLEZ RAMÍREZ.

FACULTY: **ELECTRONIC ENGINEERING.**

DIRECTOR: **JUAN CARLOS VILLAMIZAR RINCON.**

ABSTRACT

Automation was developed filling level control for Coca-Cola bottles, through digital image processing. For this, we used an embedded system in the BeagleBoard, on the Ubuntu operating system, Linux. In addition, the creation of a prototype that simulates the physical conditions Bucaramanga plant.

The embedded system has implied a program for image processing, and it can determine whether the level that has the bottle is right. Otherwise, issues an order for dismissal.

This program was developed in Python using OpenCV libraries for the recognition and management of the camera. In turn, extensions thereof to engage in communication with the PLC using Modbus TCP.

Keywords: Image Processing, automation and embedded system.

INTRODUCCIÓN

El proceso de control de calidad de llenado, cumple una función fundamental en las empresas embotelladoras y distribuidoras de bebidas, ya que permite clasificar la calidad del producto próximo a salir al mercado. Sin embargo, en la mayoría de las industrias nacionales afines a este sector, no se encuentra automatizado, trayendo consecuencias negativas tanto para el operario como para la producción.

En el caso específico de la compañía Coca-Cola, este procedimiento se realiza de forma manual, un operario visualiza la producción de botellas que va pasando por la banda transportadora, y de acuerdo a su criterio visual son retiradas del proceso. Por ello, se realizó la automatización de este proceso, a través del procesamiento digital de imágenes que evalúa la calidad de llenado, simulando las condiciones físicas de la fábrica de Coca-Cola en la ciudad de Bucaramanga.

Esto mediante la creación de un prototipo e implementación de un sistema embebido en la Beagleboard, ayudado de una cámara y un monitor VGA. La Beagleboard es la encargada del análisis y del procesamiento de la imagen capturada por la cámara. La pantalla permite visualizar los resultados obtenidos. Todo esto, montado sobre un prototipo encargado de la clasificación de las botellas.

OBJETIVOS

Dado el contexto de la empresa y las mejoras que puede significar tanto para el operario como la producción, surge el siguiente proyecto de grado que pretende:“Desarrollar un prototipo para la inspección del llenado de botellas de gaseosa carbonatada para la planta de Coca-Cola ubicada en la ciudad de Bucaramanga”.

Para alcanzar el objetivo general, se han planteado los siguientes objetivos específicos:

- Conocer el software de OpenCV para el procesamiento digital de imágenes.
- Desarrollar un programa que reconozca el nivel de llenado de las botellas.
- Realizar pruebas sobre la máquina para evaluar su veracidad y verificar las diferentes condiciones de trabajo a las que va ser sometida en el entorno industrial.

1. HARDWARE

1.1 CRITERIO DE SELECCIÓN

Para llegar a la implementación final de cada uno de los dispositivos hubo un criterio de selección, algunos desde el inicio del proyecto u otros porque presentaban inconvenientes que solo se pudieron solucionar con el cambio de los mismos.

Desde el inicio del proyecto se pensó en hacer un sistema embebido, con funciones parecidas a la de un ordenador pero con un menor consumo de energia.

Se debía escoger un dispositivo que contara con: una plataforma de desarrollo especializada en multimedia, alta frecuencia de funcionamiento (procesador) y mucha documentación para que su implementación no fuera tan tediosa. Además, que tuviera cuatro puertos USB: uno para la cámara, otro para el teclado, la tarjeta de adquisición y el ratón. La única tarjeta de desarrollo con las características anteriores, era la Beagleboard, aunque no fuera la más rápida del mercado.

A la Beagleboard se le tenía que conectar una cámara, pero en ese tiempo se contaba con una webcam genérica con la que se venía trabajando tiempo atrás. Aunque el funcionamiento era muy bueno presentaba un problema fundamental, no podía estar fija, su eje móvil permitía su movimiento en todos los ángulos y ejes posibles. Solo bastaba una pequeña ráfaga de viento para descalibrar la imagen. Por ello, se cambio la cámara por una que presentara base de fijación, solo existía una en el mercado: la Logitech c920 (ver figura 1).

Figura 1. Logitech c920.[1]

También se necesitaba un dispositivo de adquisición de datos, se pensó trabajar con la tarjeta MyDaq que tiene puerto de conexión USB, pero su programación era

[1] <http://www.ubergizmo.com/2012/01/logitech-c920-webcam-review/>[Consultado el 10 de Abril de 2013]

bastante compleja. Igualmente, en el laboratorio se contaban con dos PLC´s que podían servir como dispositivos de adquisición: uno de la marca SIEMENS y otro, de Schneider Electric. Por su facilidad de programación se escogió el segundo. Luego, se encontró una librería para comunicación con el autómata desde Python, denominada PyModbus (explicada más adelante), esta ratificó la elección del mismo.

El PLC debía contar con entradas y salidas digitales, para poder conectar los actuadores y los sensores. Los sensores solo se ocuparían de detectar el paso de la botella, por lo que debían ser digitales, autoreflectivos, tener una distancia mínima de detección de 30 cm porque la cámara se iba a encontrar a una distancia prudente de la botella para poder captar la imagen completa.

Por otro parte, los actuadores iban a ser dos, el variador de velocidad y la electroválvula neumática. El primero, tenía que trabajar en el mismo rango de potencia del motor para poderlo manipular fácilmente. La segunda, iba a ser la encargada de activar el cilindro neumático, que se seleccionó de doble efecto, ya qué el vástago era mucho más largo que los de simple efecto.

Por último, el motor y la banda transportadora se encontraban ya fabricados. Por lo que se hizo fácil su uso. La banda había sido diseñada para transporte de elementos pesados, era de color anaranjado y se deslizaba, cuando el motor se desplazaba a baja velocidad, lo que dificultaba el análisis y procesamiento de imágenes.Así mismo, el color de la banda interfería en el procesamiento de la imagen por ser de un color similar. Por esto, se cambio a color blanco, lo que solucionó los problemas anteriores.

El motor se encontraba de forma mecánica acoplado a la banda pero le faltaba torque, esto se debe a que él presenta una alta resistencia en sus devanados y el variador no puede realizar la compensación de tensión necesaria. Razón por la cual, hubo la necesidad de ponerle un juego de poleas para su correcto funcionamiento.

1.2 CARACTERÍSTICAS DE LOS DISPOSITIVOS

A continuación se presentan las características de fábrica más relevantes de los dispositivos utilizados en el proyecto.

HD Pro Webcam C910

- Grabación full HD 1080p.
- Videoconferencias HD 720p en pantalla panorámica ara los servicios de mensajería instantánea.
- Micrófonos integrados con tecnología Logitech RightSound.

- Audio estéreo que da una alta calidad y naturalidad.
- Imágenes de hasta 10 Mp.
- Foco Automático.
- Reconocimiento Facial para inicio de sesión en Windows.
- Lente de vidrio de Alta precisión Carl Zeiss.
- Corrección Automática de luminosidad mediante la tecnología RightLight 2.
- Efectos de video Premium con la función Logitech Video Effects.
- Clip universal para monitores LCD, CRT o Portátiles.
- Tecnología Logitech Fluid Crystal para obtener mejores imágenes estáticasy en movimiento.
- Compatibilidad con Windows XP, Vista, 7 y Mac OS X.

Beagleboard-xM

- Procesador Super-escalar ARM Cortex A8 con frecuencia de trabajo a 1 Ghz.
- 512 Mb LPDDR (Low Power Double Data Rate memory) RAM.
- S-video (salida TV).
- 4 Puertos USB 2.0.
- DVI-D (Digital Visual Interface).
- Conexión USB Ethernet 10/100.
- Entrada y Salida de audio estéreo.
- Puerto de Alimentación de voltaje de 5 V.
- Ranura para tarjeta de memoria MicroSD.
- Conector de expansión para LCD y cámara digital.
- Puertos de comunicaciones: I2C, I2S, SPI, RS-232 y JTAG.
- Soporta sistemas operativos como: Android, Angstrom, Fedora, Ubuntu, Gentoo, Arch Linux ARM, Windows CE y RISC OS.

Sensor Fotoeléctrico LS18D-40P

- Transistor PNP de salida.
- Tipo de reflexión difusa.
- Distancia de detección 0-40 cm.
- Tensión de alimentación 10-30 Vdc.
- Caída de Tensión menor o igual a 1.5 V.
- Detección de tiempo de retardo de 2 ms.
- Máxima corriente de salida 100mA a 24 Vdc.
- Dimensiones 18 mm de diámetro x 68 mm de largo.
- Temperatura de trabajo de -25 a 50 grados Celsius.

Telemecanique TWDLCAA40DRF

- Display incluido.
- Tiempo de scan de 1 us.

- Puerto de comunicación RS485.
- Expandible hasta 4 Módulos I/O.
- Conectividad Modbus y Modbus TCP.
- Tensión de Alimentación 100/240 Vac.
- 24 entradas y 16 salidas de tipo relé a 2 A.
- Memoria total disponible 3000 instrucciones.
- Diseñado para pequeños sistemas de control.
- Temperatura de trabajo ente 0-50 grados Celsius.
- Dimensiones 70mm profundidad x 90mm largo x 157 mm ancho.

Motor Siemens Trifásico

- Peso: 5.7 Kg.
- Potencia: 0.4 Hp.
- Velocidad: 1055 rpm.
- Torque nominal: 2.7 Nm.
- Tensión de Alimentación: 220/440 V.
- Temperatura de trabajo -15 a 40 grados Celsius.
- Corriente nominal: 1.6 A (220 V) ó 0.8 A (440 V).

Electroválvula Neumática Alibaba 4V120-06

- Peso: 163 g.
- Color: Blanco y Negro.
- Material: Resina y Plástico.
- Tipo: Dos Posiciones y Cinco Vías.
- Tensión de Alimentación: 120 Vac.
- Temperatura de operación: -5 a 50 grados Celsius.
- Forma del cableado: Alambre o conector directo de plomo.

Variador de velocidad Schneider ATV312H075M2

- Peso: 1.05 kg.
- Fases de salida: 3.
- Potencia Nominal: 1 Hp.
- Corriente Nominal: 4.8 A.
- Tensión Nominal: 208/240 Vac.
- Comunicación: Modbus y CANopen.

1.3 DIAGRAMA DE BLOQUES DEL PROCESO

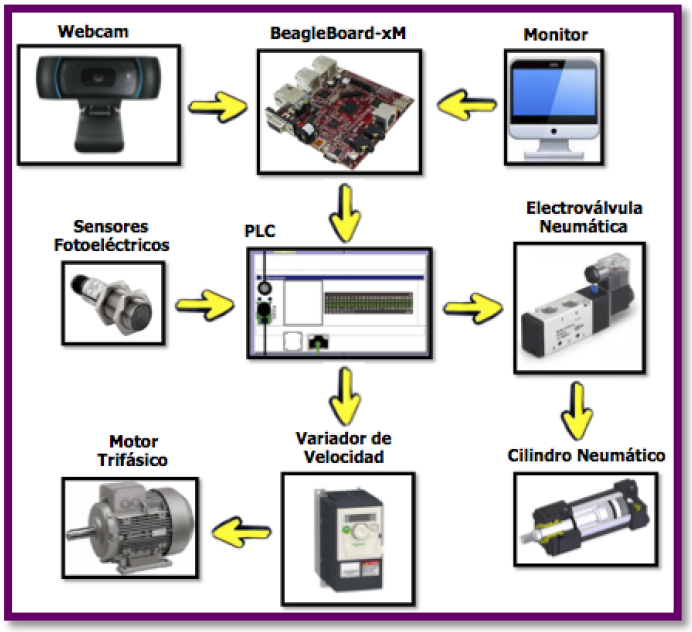

Figura 2. Diagrama de Bloques.[2]

En el proyecto se utilizaron diferentes dispositivos electrónicos para su funcionamiento. La Beagleboard es el núcleo central del proceso, en sus múltiples puertos de entrada/salida se encuentra conectado la cámara, una pantalla o dispositivo remoto para la visualización del mismo. Además, necesita estar conectada de modo permanente a Internet para establecer una conexión con el PLC, mediante el protocolo Modbus TCP. La comunicación entre estos dos dispositivos es bidireccional, el autómata envía constantemente el estado de sus entradas y de acuerdo a los datos obtenidos en el procesamiento, la Beagleboard envía una señal para la activación de los actuadores, conectados a él.

En las entradas del PLC se encuentran conectados dos sensores fotoeléctricos de tipo PNP. El primero, correspondiente a la detección de la botella para su procesamiento y el segundo, para su clasificación. Los dos actuadores que se encuentran conectados a sus salidas son: el variador de velocidad y la electroválvula neumática.

El variador de velocidad es el encargado de mantener una velocidad baja y constante, en el motor acoplado mecánicamente a la banda transportadora. Su fin

[2]Fuente Autor.

es que las botellas no caigan mientras son transportadas y puedan ser, analizadas posteriormente. Por otra parte, la electroválvula neumática activa el cilindro encargado de seleccionar las bebidas que no se encuentran con el nivel adecuado.

Por último, se encuentra una base lumínica, ubicada al frente de la cámara. Esta permite realizar un contraste entre el color del líquido y la superficie blanca, facilitando el desarrollo del software. -Esta metodología es utilizada en la industria de las bebidas-

1.4 DIAGRAMAS DE CONEXIONES FISICAS

Para el correcto funcionamiento del proyecto los dispositivos deben estar interconectados entre sí, de forma: eléctrica, mecánica y/o neumáticamente.

La primera conexión que se debe realizar es la del PLC (Figura 3) en sus entradas y salidas. En la entrada I0.1, se debe conectar el sensor encargado dedetectar el paso de la botella para capturar su imagen. En la siguiente entrada(I0.2), el sensor que detecta la botella para su clasificación. En la primera salida (Q0.2) va a ir conectado a una bobina (A1) que permitirá el encendido o apagado del motor de la banda transportadora. La salida Q0.3 (Y1), será la encargada de la activación de la electroválvula para el desplazamiento del vástago del cilindro neumático.

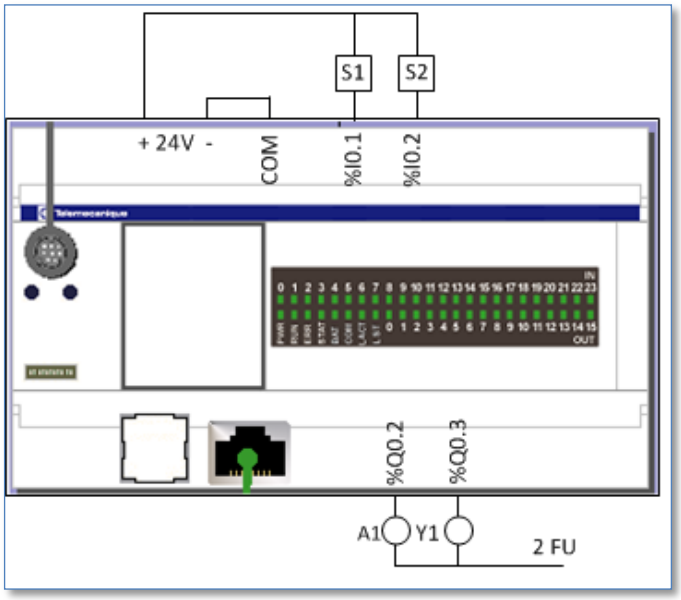

Figura 3. Conexión del PLC.[3]

[3]Fuente Autor.

Luego, debe hacerse la conexión del variador de velocidad (Figura 4) que permite mover la banda transportadora a una velocidad baja y constante. Los terminales U1, V1, W1, se conectan al motor trifásico, L1 y L2 a la fuente de alimentación,la entrada de control LI1 a un contacto normalmente abierto del relé auxiliar,que vadirectamente al PLC, esto permite el avance y frenado de la misma.

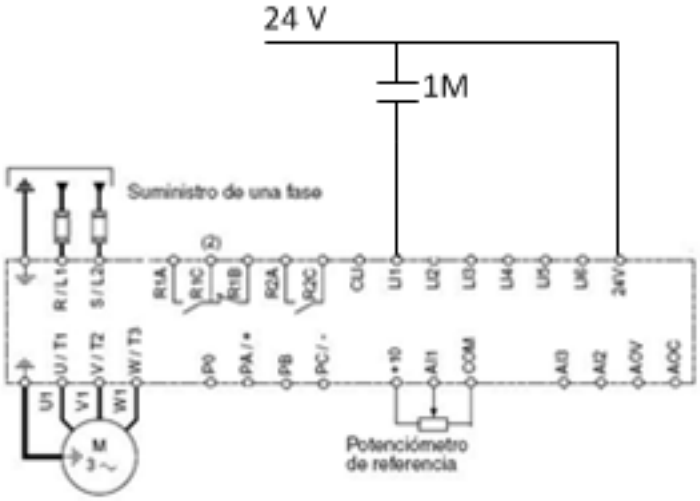

Figura 4. Conexión del Variador.[4]

Por último, se tiene un circuito neumático que cuenta con un cilindro de doble efecto y una electroválvula de 4/2. Cada vez que se active la bobina Y1 en el autómata,el cilindro realiza una expansión de la cámara que hace que la botella salga de la banda, lo que quiere decir que no cumple con el nivel de líquido adecuado.

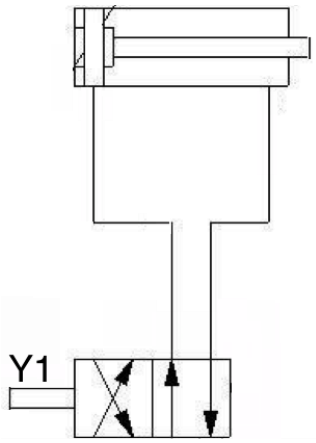

Figura 5. Circuito Neumático.[5]

[4]Fuente Autor.

[5]Fuente Autor.

2 SISTEMA EMBEBIDO

2.1 BEAGLEBOARD-XM

Es una tarjeta de desarrollo de bajo costo, producida por Texas Instruments en asociación con Digi-Key, con fines educativos. A través de ella se pensaba enseñar en las universidades las capacidades del Hardware y Software Libre. Además, cuenta con las herramientas mínimas para que el usuario experimente el poder del procesador.
Es expandible tanto en características como en interfaces. Así mismo, viene precargado el sistema operativo Angstrom.

Figura 6. Interfaz del Sistema Operativo Angstrom.[6]

Aunque Angstrom es una distribución de Linux bastante eficiente. En este proyectose usó otro distribución llamada Lubuntu,presentado como ventaja que se puede trabajar tanto en el ordenador como en sistemas embebidos.

Se pensó en dejar un computador para que hiciera el procesamiento, pero esta solución era costosa, por tal razón, hubo la necesidad de conseguir un sistema embebido de bajo costo, con alta velocidad de procesamiento. Como ya se habían hecho desarrollos sobre Ubuntu y los resultados habían sido buenos, se siguió trabajando sobre este.

2.2 LUBUNTU

6 Angstrom. <http://beagleboard.org/static/presentations/boston2011/beagleboard-101/> [Consultado el 5 de Abril del 2013]

Es unadistribución open source, rápido y liviano. Desarrollado por Mario Behling en Marzo del 2009, basado en Ubuntu, Linux. Este, utiliza una serie de aplicaciones livianas, enfocadas a la velocidad y el bajo consumo energético. Además, cuenta con bajos requerimiento de Hardware para que puedan ser empleado en dispositivos móviles, netbooks y antiguos computadores. Actualmente, existen instaladores para: Linux, Unix y Sistemas ARM.

Entre sus principales características se encuentran:

- Gestor de escritorio LXDE.
- LightDM que es un display manager.
- Chromium, navegador open-source.
- Administrador de ventanas Openbox.

Así mismo, tiene acceso a los repositorios del software de Ubuntu, a través de una aplicación llamada Lubuntu Software Center, dondese pueden descargar gran número de aplicaciones para ser ejecutadas en el sistema embebido.

Figura 7. Interfaz gráfica del Sistema Operativo Lubuntu.[7]

2.3INSTALACIÓN DE LUBUNTU

El proceso de instalación es muy sencillo. Solo se necesita de un ordenador con el sistema operativo Ubuntu, ya sea en una máquina virtual o en una partición.Además, se necesita una memoria microSD con una capacidad mínima de 8 GB, un lector de microSD y por supuesto la Beagleboard-xM.

Pre-requisitos:

[7]Fuente Autor.

Antes de comenzar la instalación del sistema operativo se deben tener en cuenta que el computador tenga todo el software necesario para realizar el procedimiento. Para ello,se escribe lo siguiente en el terminal:

```
margy9003@ubuntu:~$ sudo apt-get install wget pv dosfstools parted
```

Instalación de Lubuntu:

El primer paso es introducir el lector SD en el computador e identificar la memoria en la máquina virtual.

```
margy9003@ubuntu:~/ubuntu-11.10-r7-minimal-armel$ df -h
```

Después de haber escrito el código anterior se verá algo parecido, a lo que se muestra a continuación.Hay que tener presente que la memoria empleada es de 16 Gb, aparecereferenciada en la última línea como /dev/sdc1.

```
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1        19G  3.4G   15G  19% /
udev            494M  4.0K  494M   1% /dev
tmpfs           201M  816K  200M   1% /run
none            5.0M     0  5.0M   0% /run/lock
none            502M  152K  502M   1% /run/shm
none            100M   44K  100M   1% /run/user
.host:/         233G  139G   95G  60% /mnt/hgfs
/dev/sdb        5.2M   79K  5.1M   2% /media/margy9003/PHONE
/dev/sdc1        15G  2.8M   15G   1% /media/margy9003/OS
```

Posteriormente se bajará la última versión estable de Ubuntu,la cual se instalará en la Beagleboard. La descarga se demora alrededor de 10 min.

```
margy9003@ubuntu:~$ wget http://ynezz.ibawizard.net/beagleboard/oneiric/ubuntu-1
1.10-r7-minimal-armel.tar.xz
```

Luego, se verifica que la descarga haya sido exitosa. Para ello, se hace una suma de comprobación que genera un número de verificación.

```
margy9003@ubuntu:~$ md5sum ubuntu-11.10-r7-minimal-armel.tar.xz
```

Si el resultado concuerda, quiere decir que la operación fue exitosa.

```
58092de96f8934da74ea00e6a7b8eccf  ubuntu-11.10-r7-minimal-armel.tar.xz
```

Un fichero con la extensión .tar, que un formato de almacenamiento propio de Linux, semejante a los encontrados en los compresores de datos de Windows, con extensiones .rar y .zip. Lo que quiere decir, que antes de su instalación se deben

extraer los archivos que están dentro de él, para realizar este paso se debe estar en la raíz del directorio.

```
margy9003@ubuntu:~$ unxz -c ubuntu-11.10-r7-minimal-armel.tar.xz | tar xv
./ubuntu-11.10-r7-minimal-armel/
./ubuntu-11.10-r7-minimal-armel/initrd.img-3.2.0-psp6
./ubuntu-11.10-r7-minimal-armel/vmlinuz-3.2.13-x7
./ubuntu-11.10-r7-minimal-armel/armel-rootfs-201203291538.tar
./ubuntu-11.10-r7-minimal-armel/setup_sdcard.sh
./ubuntu-11.10-r7-minimal-armel/vmlinuz-3.2.0-psp6
./ubuntu-11.10-r7-minimal-armel/initrd.img-3.2.13-x7
margy9003@ubuntu:~$ cd ubuntu-11.10-r7-minimal-armel
margy9003@ubuntu:~/ubuntu-11.10-r7-minimal-armel$
```

Ahora, que se encuentra en el directorio indicado, se llama un comando para particionar y configurar esos archivos en la tarjeta micro SD. Es importante recordar el nombre de la unidad, para este caso es “sdc”.

```
margy9003@ubuntu:~/ubuntu-11.10-r7-minimal-armel$ sudo ./setup_sdcard.sh --mmc /
dev/sdc --uboot beagle_cx
```

Aquí, ya se instaló Lubuntu en modo de consola. Se extrae la tarjeta de modo seguro.

GUI:

Para comprobar que la instalación se hizo correctamente se inserta la tarjeta en el slot de la Beagleboard. Desde ahí, se procede a hacer la instalación de la interfaz gráfica.

La primera información que pide el sistema es el usuario y la contraseña. Estos son fijados por defecto en la instalación.

Usuario: Ubuntu.
Contraseña:temppwd.

Ya como el usuario se encuentra como administrador se propone a hacer las actualizaciones del sistema operativo.

sudo apt-getupdate

Seguido, se actualizan los paquetes que ya se encuentran instalados y se borran todos los repositorios de los paquetes descargados.

sudo apt-get upgrade
sudo apt-get clean

Como el sistema embebido debe ser lo más ligero posible. No se instalan los programas recomendados por Ubuntu como son: juegos, gimp, office, etc.

```
sudo apt-get install –no-install-recommends lubuntu-desktop
```

Se actualiza los paquetes recién instalados.

```
sudo apt-get dist-upgrade
```

Por último, se limpian los archivos que quedaron como residuos después de la instalación.

```
sudo apt-get autoclean
sudo rm /var/cache/apt/archives/*.deb
```

Lo único que falta es reiniciar el sistema para observar a Lubuntu con interfaz gráfica.

```
sudo apt-getreboot
```

3 SOFTWARE

En este capítulo se describe detalladamente el software utilizado a lo largo del proyecto. El lenguaje de programación usado fue Python, su elección se debe a que puede importar las librerías de OpenCV, dedicadas al procesamiento de imágenes, y su vez, podía comunicarse con dispositivos periféricos, a través de librerías de comunicación como lo son:Pymodbus y Pyserial.

Es un lenguaje de programación open source de alto nivel, lo que significa que no tiene ningún costo su uso y distribución es gratuita, así sea para uso comercial. Su licencia es administrada por Python Software Foundation.

3.1 METODOLOGIA DEL PROCESO

En la Figura 8, se presenta el diagrama de flujo pertinente a la aplicación desarrollada en Python, usando las librerías de OpenCV,para obtener el nivel de llenado de una botella de Coca-Cola y su póstuma, clasificación.

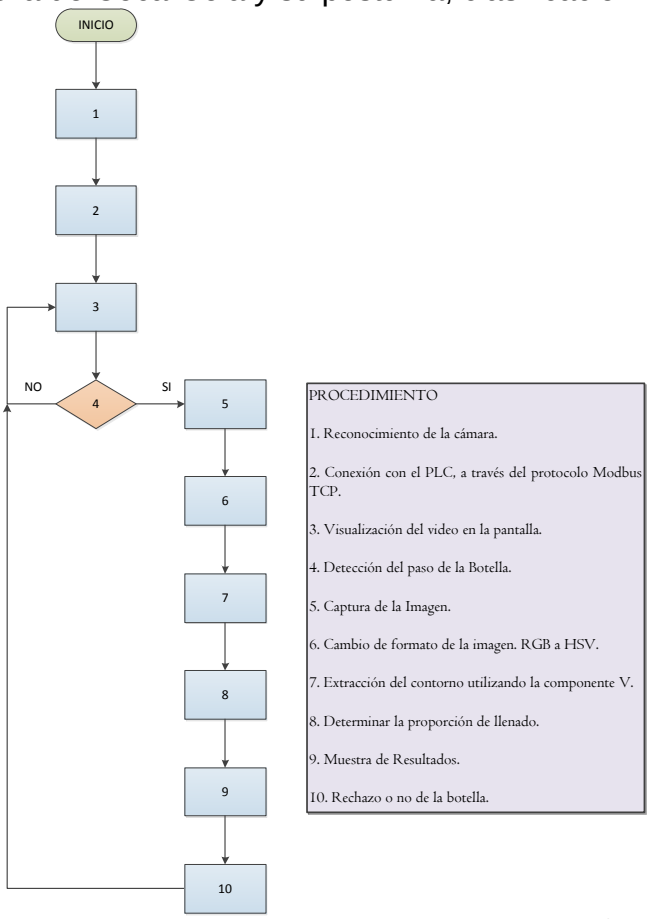

Figura 8. Diagrama de Flujo del Proceso.[8]

[8]Fuente Autor.

Al ejecutar el software escrito en Python, se reconoce el puerto donde se encuentra conectada la cámara(ver cv.CreateCameraCapture, pág.40)y se hace una lectura de ella (ver cv.QueryFrame, pág.40). Para obtener en la pantalla una secuencia de video del proceso. Hay que recordar que la cámara de video, referencia Logitech c920 se conecta por puerto USB a la tarjeta de desarrollo.

Luego, se entabla la comunicación con el autómatamediante el protocolo de comunicación Modbus TCP (ver ModbusClient, pág.36). Para ello, se le asigna la dirección IP a la que está conectado, y su puerto de comunicación por defecto, 502. El control de los actuadores se realiza con el PLC Telemecanique, ya que no se pudieron utilizar lospuertos de la Beagleboard debido al alto ruido que se presenta en el laboratorio.

Al controlador lógico programable también se encuentran conectados dos sensores fotoeléctricos (ver Figura 3), uno que detecta el paso de la botella por la cámara y otro que detecta la botella en la sección de rechazo (cilindro neumático).

Seguido, se lee de forma permanente el estado del sensor PNP que está ubicado frente a la cámara. Cuando se detecta un FALSE por el paso de la botella, la cámara hace una captura de la imagen con un tamaño de 640x480 pixeles (ver cv.SaveImage, pág.39), y un formato de color de RGB (ver Figura 9).

Figura 9. Imagen en formato RGB.[9]

Posteriormente, se cambia el formato de color de RGB a HSV (ver cv.CvtColor, pág.40), ya que este es menos susceptible al cambio de luminancia (ver Figura 10).

[9]Fuente Autor.

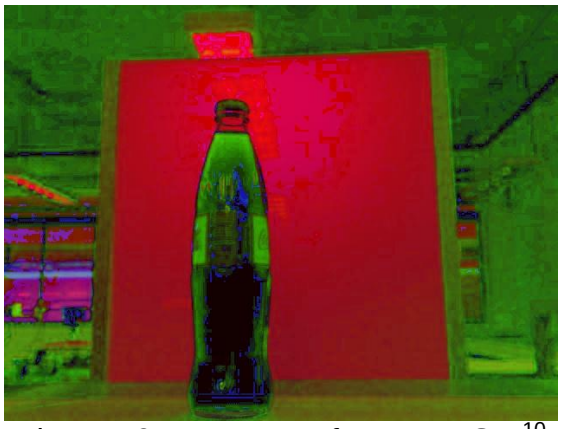

Figura 10. Imagen en formato HSV.[10]

Como se ve, en la figura 10, hay elementos de fondo que no son de interés. Lo único que interesa analizar es la botella y la cantidad liquido presente dentro de ella. Para ello se recorta la imagen mediante una región de interés (ver cv.SetImageROI, pág.47) a 190×369 pixeles, dejando como fondo la botella (ver Figura 11).

Figura11. ROI.[11]

El formato HSV cuenta con tres componentes Hue, Saturationy Value. Los dos primeros cambian respecto al nivel de iluminación y el tercero, permanece contante dentro de unos rangos definidos. Por esta razón se extrae la componente V (ver Figura 12), siendo la base para posteriores análisis (ver cv.Split, pág.48).

[10]Fuente Autor.

[11]Fuente Autor.

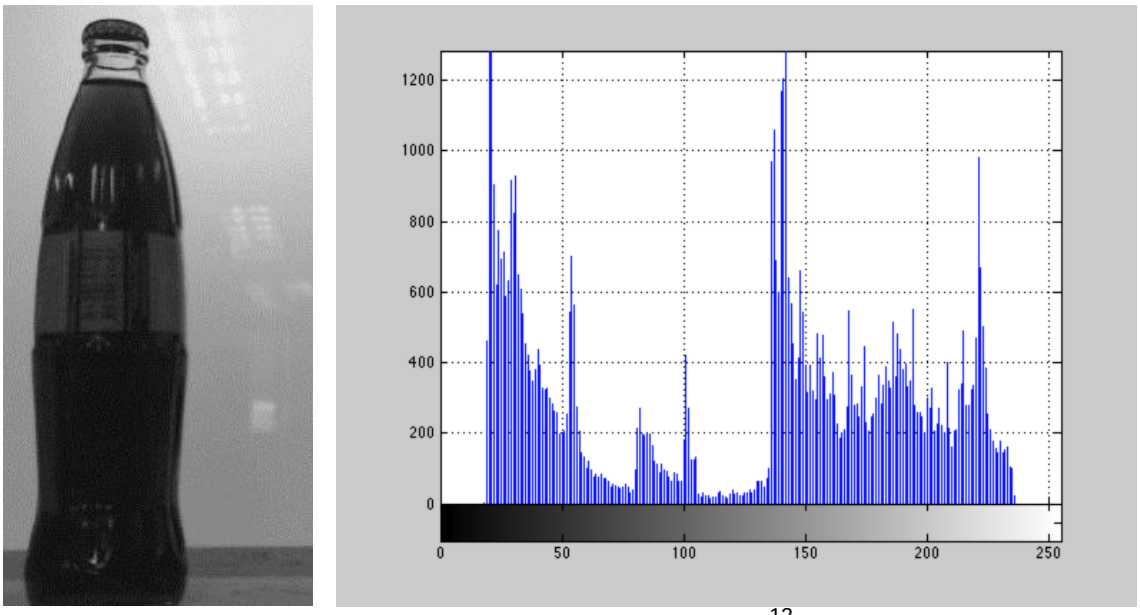

Figura 12. Componente V.[12]

Para obtener la región de interés con mayor facilidad se aplica un filtro de desenfoque de tipo MEDIAN, lo que hace que sus valores máximos en histograma se suavicen (ver Figura 13). Éste filtro usa un algoritmo eficiente que permite crear un suavizado más uniforme en un mínimo tiempo (ver cv.Smooth, pág.43), ignorando los valores atípicos de la imagen.

En la Figura 12, se aprecia que antes de aplicarse el filtro a la imagen, el valor máximo de los picos sobrepasa los 1200, en comparación con la imagen filtrada (Figura 13), sus valores máximos no alcanzan este valor. Siendo los picos menos pronunciados y permitiendo determinar fácilmente los valores óptimos para la segmentación.

Después,se tiene que realizar el proceso de segmentación. En este caso, separar el fondo de la imagen (color blanco) de la región de interés (líquido presente en la botella).

[12]Fuente Autor.

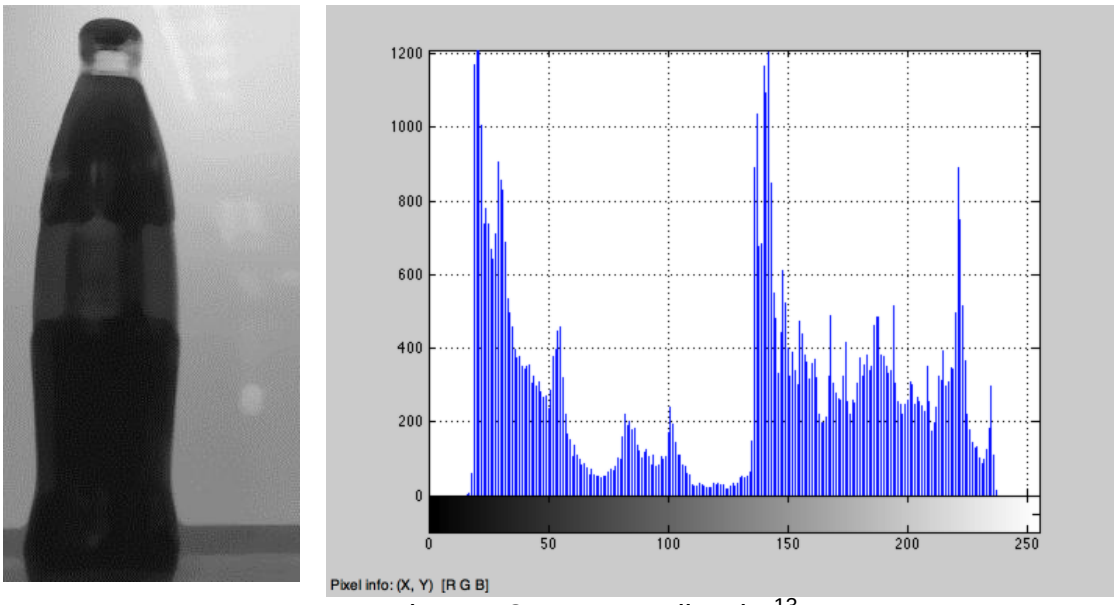

Figura 13. Imagen Filtrada.[13]

Al emplear la segmentación, se debe establecer un rango de clasificación, para su binarización (ver cv.InRageS, pág.46). Este rango esta dado por el histograma mostrado en la Figura 13. Como la región de interés es el liquido, se toma un rango donde se encuentren condensados la gran mayoría de los colores oscuros, pertenecientes al liquido de la botella.

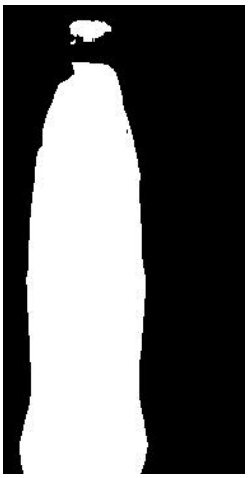

Figura 14. Imagen Binarizada.[14]

En la imagen binarizada aparecen dos regiones: una perteneciente a la tapa y la otra, al líquido. Es fácil distinguir que el líquido, es la imagen que mayor área tiene.

[13]Fuente Autor.

[14]Fuente Autor.

A esta región se le extrae el contorno para determinar el nivel de llenado de la botella (ver cv.FindContours, pág.41).

Para verificar el funcionamiento del algoritmo, se compara la imagen binarizada con su contorno (ver cv.DrawContours, pág.41). Esto se hace con el fin de mejorar los niveles de segmentación y corroborar que las dimensiones son las correctas. (ver Figura 15).

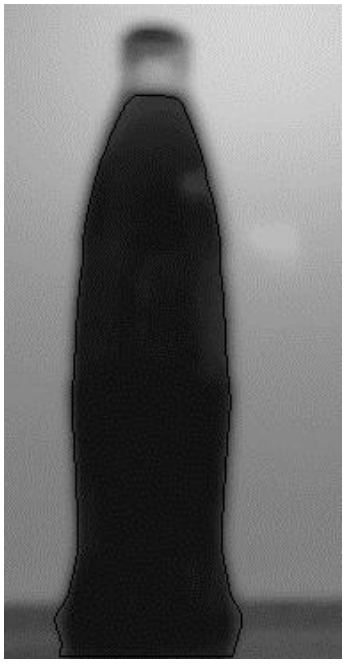

Figura 15. Dibujo del Contorno.[15]

Por medio del contorno se pueden obtener los puntos máximos y mínimos de la imagen (ver figura 15) estos puntos se toman con el fin de establecer el ancho y alto del contorno del líquido. Estos valores son de gran importancia para obtener la proporción llenado, que se determina con la siguiente formula:

$$Prop = \frac{y_{MAX} - y_{MIN}}{X_{MAX} - X_{MIN}}$$

Ecuación 1. Proporción.

La proporción es inherente a la distancia entre la botella y la cámara, lo que permite disminuir el porcentaje de error, a la hora de hacer la clasificación.

Una vez, obtenido este resultado se pregunta sí el valor esta entre el rango máximo o mínimo de tolerancia. Si sí, la botella no es rechaza al final del proceso.

[15]Fuente Autor.

De lo contrario, es rechazada apenas sea detectada por el segundo sensor. Repitiéndose el proceso de forma continua, hasta que el dispositivo sea apagado.

A continuación se explica de forma más profunda cada una de las librerías usadas en el programa.

3.2 PYTHON

Fue desarrollado por Guido van Rossum (Figura 16) a finales de los años ochenta en el instituto de investigación de Matemáticas y Ciencias de la Computación, Ámsterdam, Holanda. Su implementación comenzó 1989 como sucesor del lenguaje ABC para el sistema operativo Amoeba.

Figura 16. Creador de Python.[16]

Entre sus principales características se destacan:

- Potencia y rápidez, es capaz de realizar diferentes funciones en pocas líneas de código de forma rápida, gracias a su compilador optimizado. El código de Python corre mucho más rápido de lo necesario para la mayoría de las aplicaciones.
- Alta Adaptabilidad, puede integrarse con varios lenguajes de programación mediante librerías. En el caso de Java, a través de Jython. .NET, IronPython, etc.
- Multiplataforma, es compatible con la mayoría de sistemas operativos como: Windows, Unix, Linux, AmigaOS. Incluso, existen versiones que corren en .NET o JVM.

[16] Guido van Rossum. <http://www.maestrosdelweb.com/editorial/introspectiva-guido-van-rossum-python/> [Consultado el 4 de Abril de 2013]

- Amigable y de fácil uso, ya qué contiene gran documentación en su pagina web, tutoriales online y libros que permiten aumentar la productividad del usuario rápidamente.

Igualmente, cuenta con una gran variedad de bibliotecas y extensiones para realizar diversas tareas. Entre las más comunes se encuentran: TkInter (GUI), Pyserial (encapsula la información de forma serial), Pymodbus (Entabla comunicación por los protocolos Modbus), entre otras.

3.2.1. Pymodbus. Es una implementación del protocolo Modbus en python, para versiones mayores a la 2.3.

Características del cliente:

- Soporta Modbus, Modbus TCP, Modbus UDP y RTU.
- Permite comunicaciones síncronas y asíncronas.
- Lectura/Escritura directa sobre registros.

Esta librería es desarrollada para comunicar dispositivos de campo como PLC o variadores de velocidad a aplicaciones de oficina, con el fin de adquirir datos de forma remota; es basado en la arquitectura cliente servidor y es ampliamente usado en el sector industrial, especialmente por la empresa Schneider Electric.

3.2.1.1Comandos encargados de la comunicación Modbus TCP

- *Client.close()*. Cierra la conexión del socket subyacente.

- *Client.connect()*. Conecta con el servidor de Modbus TCP. Si la conexión se realiza con éxito devuelve el valor True, de lo contrario, False.

- *ModbusClient(host, puerto).* Establece una comunicación por medio del protocolo TCP. El Host es el destino de conexión, en este caso, una dirección IP. El puerto Modbus por defecto es 502.

- *Client.write_coil(dirección, valor)*. Esta función escribe en un registro interno del programador lógico programable. La dirección corresponde al registro de inicio. El valor es booleano, lo que quiere decir que puede ser: False o True.

- *Client.read_coils(dirección, conteo)*. Este comando tiene como objetivo leer un registro interno o múltiples registros. La dirección es donde comienza a leer. El conteo es el número de registros a leer desde la posición fijada.

3.2.1.2 Código Ejemplo para entablar una comunicación Modbus TCP

```
from pymodbus.client.sync import ModbusTcpClient

client = ModbusClient('127.0.0.1', 502)
client.connect()
client.write_coil(1, True)
result = client.read_coils(1,1)
print result.bits[0]
client.close()
```
[17]

3.3 OPENCV

Es una librería de visión por computador, open source, desarrollada por Intel que cuenta con más de quinientas funciones referentes al área de visión como: robótica, reconstrucción 3D, seguridad, inspección de productos, reconocimiento facial, entre otras.

Su lenguaje de programación es C, lo que permite la creación de sistemas complejos, robustos y de fácil uso para el usuario. Además de ser multiplataforma, cuenta con versiones para Windows, Mac OS X y Linux.

Se caracteriza por su: eficiencia, funcionalidad, calidad y gran rapidez de procesamiento, ya que sus algoritmos están basados en estructuras flexibles, acopladas con rutinas de bajo nivel, optimizadas para procesadores Intel (IPL).

OpenCV maneja varios tipos de datos para hacerlo más simple y uniforme. Los datos fundamentales son:

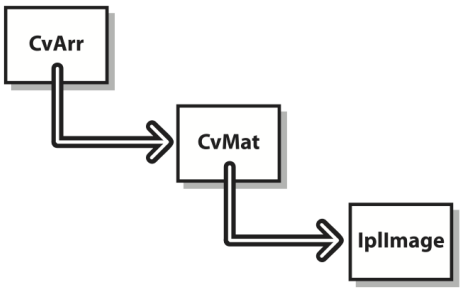

Figura 17. Tipos de Datos.[18]

[17]Código Modbus TCP. <http://code.google.com/p/pymodbus/> [Consultado el 4 de Abril de 2013].

[18]Kaebler, Adrian. Bradsky, Gary. Learning OpenCV: Computer Vision With The OpenCV Library. 1° Ed. O'reilly: 2008. ISBN: 978-0-596-51613-0.

Iplimage es la estructura básica donde se codifica las imágenes sin importar el formato de color o el número de canales. Seguida de CvMat, la estructura de Matriz que se maneja en esta librería. La última, más usada y compleja, CvArr es una clase abstracta de CvMat.

A su vez, OpenCV se encuentra subdividido en cinco librerías principales: CV, MLL, HighGUI, CXCORE y CvAux.La primera, contiene algoritmos relacionados al procesamiento de imágenes y de visión por computador. La segunda, comprende el aprendizaje de la máquina con clasificadores estadísticos. La tercera, abarca rutinas de entradas/salidas para almacenar imágenes y video. Por último, CXCoreson estructuras básicas para el procesamiento digital de imágenes.

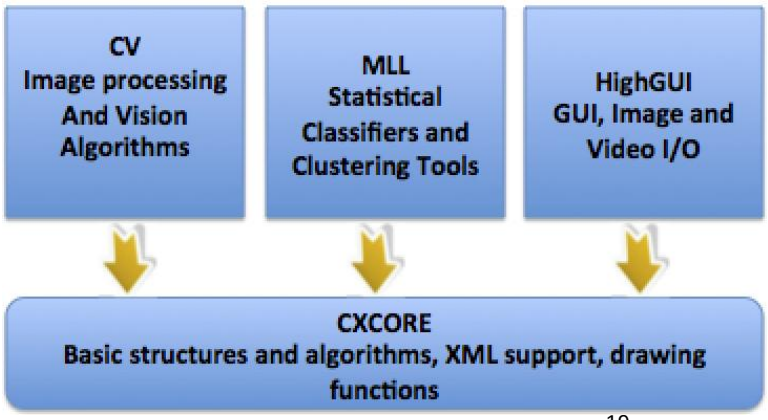

Figura 18. Librerías de OpenCV.[19]

3.3.1 HighGUI

Aquí se encuentran las funciones encargadas de interactuar con el sistema operativo, el gestor de archivos y los dispositivos periféricos. A su vez, permite crear una interfaz de usuario de alto nivel, lo que quiere decir: originar nuevas ventanas, mostrar en ellas imágenes o videos, manejar eventos, entre otros. Algunas de estas herramientas pueden ser redundantes para el usuario.

Esta librería se subdivide en tres más pequeñas: El hardware, sistema de archivos y la GUI. La primera se ocupa de la interacción de los dispositivos periféricos con el sistema operativo, su reconocimiento y obtención de la información necesaria de los mismos.

[19]Kaebler, Adrian. Bradsky, Gary. Learning OpenCV: Computer Vision With The OpenCV Library. 1° Ed. O'reilly: 2008. ISBN: 978-0-596-51613-0.

El gestor de archivos se encarga de cargar y guardar las imágenes. Su característica más relevante es que permite capturar video de la misma forma que se hace para leer la cámara, solo que se lee frame por frame. Así, como funciones universales para cargar y guardar imágenes.

La última parte es el sistema de ventanas, puede abrir una ventana e introducir en ella una imagen o video. Además, de responder a eventos del teclado o el ratón.

3.3.1.1Funciones Utilizadas

- *cv.NamedWindow(Nombre, Bandera)*. Crea una ventana que es utilizada para mostrar una imagen o un video.Si dos ventanas poseen el mismo nombre, no se muestra ninguna de las dos imágenes. El nombre de la ventana es el identificador de la misma, aparece como parámetro de la función.

- *cv.ShowImage(Nombre de la ventana, Imagen).*Muestra la imagen en una ventana definida. Si la ventana tiene de bandera CV_WINDOW_ AUTOSIZE se muestra en su tamaño original, sino es escalada en la pantalla a mostrar.

- *cv.WaitKey(Retardo)*. Espera por un evento en milisegundos, que sea oprimido una tecla o botón. Sí se cumple la condición en el tiempo predeterminado, se devuelve al código, de lo contrario se repite por un ciclo infinito (la mayoría de las veces se encuentra acompañado de un while).

  Si el argumento es igual a cero esperará de forma indefinida a que sea oprimida una tecla. Este evento es usual en la captura de video.

- *cv.LoadImage(Nombre del Archivo, bandera).*Carga una imagen desde un archivo y retorna un puntero con la imagen cargada. Los formatos soportados son los siguientes: BMP, DIB, JPEG, JPG, JPE, PNG, PBM, PGM, PPM, SR, RAS, TIFF, TIF, EXR y JP2.

  Las banderas definen el tipo de formato de color de la imagen, por lo tanto el número de canales. Son dos principalmente:
  - CV_LOAD_IMAGE_COLOR, carga una imagen a color de tres canales.
  - CV_LOAD_IMAGE_GRAYSCALE, carga una imagen a blanco y negro de un solo canal.

- *cv.SaveImage(Nombre del fichero, Imagen).*Guarda la imagen en una carpeta especificada. El tipo de extensión es escogido acorde a la profundidad de la imagen o las necesidades del usuario. Con esta función

solo se pueden guardar imagenes de 8 bits de un solo canal o de tres canales. No obstante, el usuario se puede ayudar de la función cv.CvtColor y cv.CvtScale para guardar la imagen.

- *cv.CreateCameraCapture(index)*. Cuenta con un identificador paraacceder a una cámara especifica y el codec de comunicación con el sistema operativo. Una vez el enlace sea creado, se inicializa la estructura CvCapture para leer la transmisión de video proveniente de la cámara. Actualmente se utiliza dos interfaces de cámara para Windows: Video for Windows y Matrox Imaging Library. Dos para Linux: V4L y Fire Wire.

  Sí solo se encuentra una cámara conectada al dispositivo el valor por defecto es cero, por otro lado se tendrá que buscar el de interés.

- *cv.QueryFrame(Estructura de Captura)*. Graba un marco, desde una cámara o archivo de video, lo descomprime y lo devuelve. La imagen que retorna no puede ser liberada ni modificada por el usuario.
  La estructura de captura es una variable que almacena la función cv.CreateCameraCapture().

3.3.2 CV

Esta librería contiene algoritmos relacionados al procesamiento de imágenes y de visión por computador. Las funciones que fueron utilizados se explican a continuación:

- *cv.CvtColor(fuente, destino, conversión)*. Se encarga de convertir una imagen fuente en otro formato de color manteniendo la misma cantidad de canales y el mismo tipo de datos. Las conversiones más significativas se encuentran en la Tabla 1.

Tabla 1. Conversiones de Formatos de Color.[20]

Conversión	Significado
CV_BGR2GRAY	Convierte de formato BGR a escala de grises.
CV_RGB2GRAY	Convierte de formato RGB a escala de grises.
CV_BGR2RGB	Convierte de formato BGR a RGB.
CV_RGB2BGR	Convierte de formato RGB a BGR.
CV_RGB2Luv	Convierte de formato RGB a Luv.
CV_BGR2Luv	Convierte de formato BGR a Luv.
CV_Luv2RGB	Convierte de formato Luv a RGB.

[20]Fuente Autor.

- *cv.DrawContours(img, contorno, color_externo, color_hueco, max_nivel, espesor, tipo de línea, offset).* Dibuja un contorno en la pantalla, con ayuda de ocho argumentos. El primero es la imagen en la cual se desea dibujar el contorno. Seguido del contorno obtenido mediante la función cv.FindContours. El color externo se refiere al color del que se quiera dibujar el contorno. Contrario, al color del hueco, cuando se determinan los contornos cualquiera que sea macado como un agujero será pintado en un color alternativo.

  El máximo_nivel denota la máxima jerarquía en la que debe ser recorrido el dibujo. Por ejemplo, si este valor es 0, debe recorrer el contorno de entrada y los próximos a él. El valor 1, significa que dibuje los contornos de entrada y sus hijos y así sucesivamente. Esto fue porque el contorno fue producido por el modo CV_RETR_CCOMP o CV_RETR_TREE, valiéndose así de valores negativos de igual significado a los positivos.

  El espesor y el tipo de línea no tienen nada de especial. Por último, con el offset el contorno se dibujará en un lugar diferente al cual fue definido.

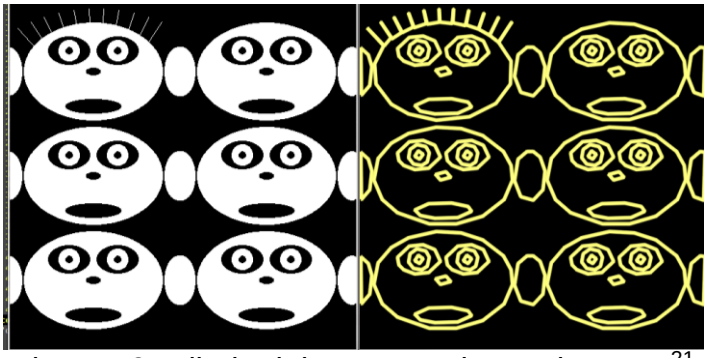

Figura 19. Dibujo del contorno de una imagen.[21]

- *cv.FindContours (fuente, almacenamiento memoria, Threshold, modo, método)*. Se aplica con el fin de tener un umbral fijo de un solo canal. Es usado típicamente para obtener una imagen de salida binaria o remover el ruido.La imagen fuente debe ser de 8-bits de un solo canal, lo que quiere decir, puede ser una imagen binaria o en escala de grises.

  El almacenamiento de memoria permite guardar de forma dinámica los contornos. Se crea con la función cv.CreateMemStorage(). Esta función toma como argumento el tamaño de un bloque, retorna el tamaño de los bloques dentro de él. Si el valor del argumento es cero, fija por defecto un

[21] Contorno. <https://compvis.readthedocs.org/en/latest/_images/contour1.png> [Consultado el 10 de Abril de 2013].

bloque de 64kB para ser usado. A su salida retorna un puntero con la dirección de la nueva memoria de almacenamiento.

El modo puede variar entre: CV_RETR_EXTERNAL, CV_RETR_LIST, CV_RETR_CCOMP o CV_RETR_TREE. Representa la forma como se buscará los contornos y la forma que se quiere mostrar al usuario.

- CV_RETR_EXTERNAL, devuelve los contornos más externos de la imagen. Así, que sí se tiene un contorno que encierra a otro, sólo se muestra el exterior.
- CV_RETR_LIST, retorna todos los contornos colocándolos en una lista. No muestra si se encuentran anidados entre sí y mucho menos, su jerarquía.
- CV_RETR_CCOMP, muestra todos los contornos, organizados en dos niveles de jerarquía. El orden que se da, es desde el más interno al externo.
- CV_RETR_TREE, se obtiene todos los contornos con su respectiva jerarquía. Así que se puede decir que un objeto se encuentra a un determinado número de niveles de jerarquía respecto a otro. Se muestra de forma más detallada y completa

En la Figura 20se muestra una imagen en escala de grises, con áreas definidas de color blanco y gris. A cada una de esas áreas se denomina contorno. Además se muestra la forma en que OpenCV numera estos.

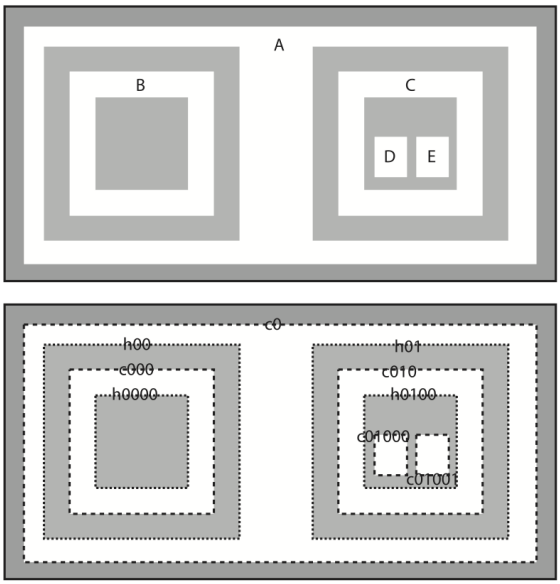

Figura 20. Ejemplo de Contornos.[22]

[22]Kaebler, Adrian. Bradsky, Gary. Learning OpenCV: Computer Vision with The OpenCV Library. 1° Ed. O'reilly: 2008. ISBN: 978-0-596-51613-0.

Acorde al modo escogido, el usuario va a apreciar los contornos extraídos de la Imagen. En la Figura 21, se muestra los diferentes modos.

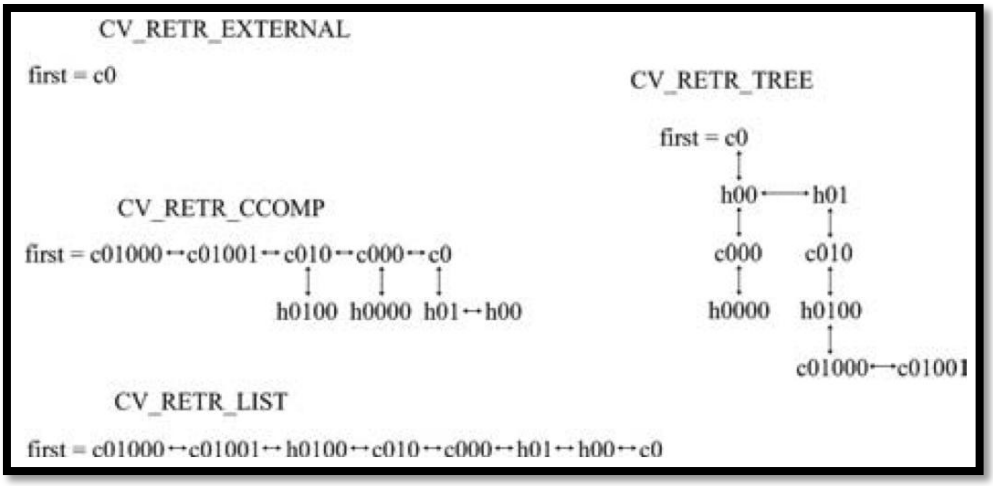

Figura 21. Modos de Contornos.[23]

El último argumento a tener en cuenta en la obtención del contorno, corresponde al método de aproximación. Existen cinco métodos diferentes:

- ✓ CV_CHAIN_CODE, entrega los contornos de salida en el código cadena de Freeman.
- ✓ CV_CHAIN_APPROX_NONE, traslada todo los puntos desde una cadena.
- ✓ CV_CHAIN_APPROX_SIMPLE, comprime segmentos de forma horizontal, vertical y diagonal. Excepto los puntos finales.
- ✓ CV_CHAIN_APPROX_TC89_L1, aproxima mediante el algoritmo Teh-Chin.
- ✓ CV_LINK_RUNS, método permitido únicamente para CV_RETR_LIST y enlaza los segmentos horizontales de primeros.

- *cv.GetSize(imagen).* Retorna las dimensiones de una imagen o un arreglo. Es de gran uso, a la hora de crear una nueva matriz o imagen del mismo tamaño.

- *cv.Smooth(fuente, destino, tipo de filtro, param 1, param2, param 3, param 4).* Es una operación de procesamiento de imágenes que se utiliza para reducir el ruido, disminuir la resolución de la imagen o corregir algunos artefactos de la cámara.

[23]Kaebler, Adrian. Bradsky, Gary. Learning OpenCV: Computer Vision With The OpenCV Library. 1° Ed. O'reilly: 2008. ISBN: 978-0-596-51613-0.

Esta función realiza un filtrado a la imagen fuente y es almacenada en el destino. Existen cinco maneras diferentes de hacer el filtrado: CV_BLUR_NO_SCALE, CV_BLUR, CV_GAUSSIAN, CV_MEDIAN y CV_BILATERAL. Sus características se muestran en la tabla a continuación:

Tabla 2. Tipo de Filtros.[24]

Smooth type	Name	In place?	Nc	Depth of src	Depth of dst	Brief description
CV_BLUR	Simple blur	Yes	1,3	8u, 32f	8u, 32f	Sum over a param1×param2 neighborhood with subsequent scaling by 1/(param1×param2).
CV_BLUR_NO_SCALE	Simple blur with no scaling	No	1	8u	16s (for 8u source) or 32f (for 32f source)	Sum over a param1×param2 neighborhood.
CV_MEDIAN	Median blur	No	1,3	8u	8u	Find median over a param1×param1 square neighborhood.
CV_GAUSSIAN	Gaussian blur	Yes	1,3	8u, 32f	8u (for 8u source) or 32f (for 32f source)	Sum over a param1×param2 neighborhood.
CV_BILATERAL	Bilateral filter	No	1,3	8u	8u	Apply bilateral 3-by-3 filtering with color sigma=param1 and a space sigma=param2.

CV_BLUR, es el filtrado más simple, el pixel de salida es la media de todos los pixeles que lo rodean. Soporta imágenes que tenga uno hasta cuatro canales y trabaja con imágenes de 8-bits o 32-bits punto flotante.

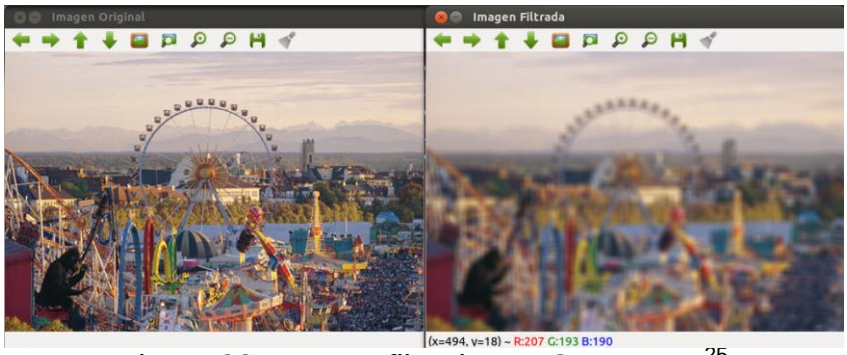

Figura 22. Imagen filtrada por CV_BLUR.[25]

CV_BLUR_NO_SCALE, tiene gran parecido con el filtro anterior, aunque no realiza una división para obtener una media y por lo tanto, no hay un

[24]Kaebler, Adrian. Bradsky, Gary. Learning OpenCV: Computer Vision With The OpenCV Library. 1° Ed. O'reilly: 2008. ISBN: 978-0-596-51613-0.

[25]Fuente Autor.

desborde. Este se puede aplicar a imágenes de un solo canal de 8-bits, 16-bits y 32-bits. A su vez, resulta más rápido que el CV_BLUR.

CV_MEDIAN, remplaza el valor de cada pixel por la media de sus pixeles vecinos. Puede trabajar con imágenes de un solo canal, tres o cuatro canales de 8-bits. Se diferencia del CV_BLUR ya qué puede ignorar los valores atípicos mediante la selección de puntos medios.

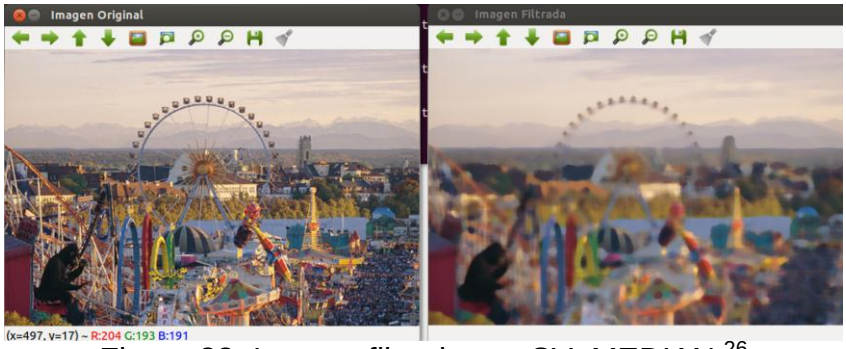

Figura 23. Imagen filtrada por CV_MEDIAN.[26]

CV_GAUSSIAN, es el más utilizado pero el más demorado. Este filtro hace la convolución de cada punto con el Kernel. Los dos primeros parámetros corresponden al ancho y el alto de la ventana de filtrado, el tercer parámetro es opcional e indica el valor de sigma. Si el tercer parámetro no es dado se determina mediante las siguientes ecuaciones:

$$\sigma_x = \left(\frac{n_x}{2} - 1 \right) \cdot 0.30 + 0.80, \quad n_x = \text{param1}$$

$$\sigma_y = \left(\frac{n_y}{2} - 1 \right) \cdot 0.30 + 0.80, \quad n_y = \text{param2}$$

Ecuación 2. Kernel del filtro Gaussiano.

[26]Fuente Autor.

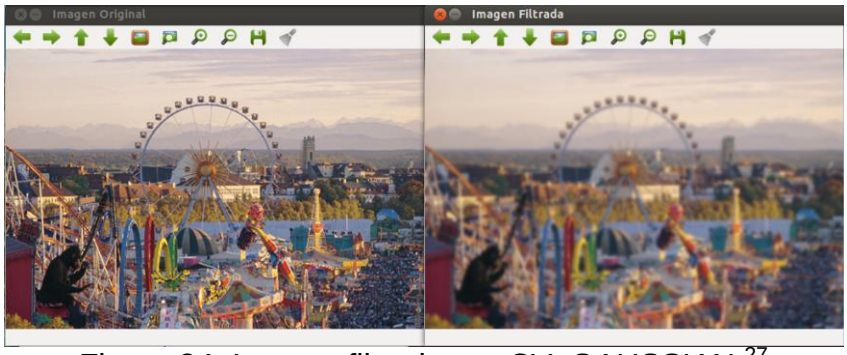

Figura 24. Imagen filtrada por CV_GAUSSIAN.[27]

CV_BILATERAL, este filtro ayuda a disolver el ruido pero también suaviza los bordes. El construye una media ponderada de cada píxel y sus vecinos. Cuenta con dos parámetros. El primero, es el ancho del kernel Gaussiano equivalente al sigma del filtro anterior. El segundo, es el ancho del kernel Gaussiano en el dominio del color, definido por el rango de intensidades. Entre más grande sea el segundo valor, mayor será el rango de colores utilizado en el suavizado.

Figura 25. Imagen filtrada por CV_BILATERAL.[28]

3.3.3 CXCORE

Esta librería contiene las estructuras básicas, consideradas por OpenCV para procesamiento de imágenes y la visión por computador. Las funciones que fueron utilizadas se explican a continuación:

- *cv.InRangeS().*Esta función revisa que los pixeles de la imagen se encuentra dentro de un rango de color especifico. Por ejemplo, en la figura 19 se determina un rango en el que se encuentte la plastilina de color amarillo para que puede ser detectada.

[27]Fuente Autor.
[28]Fuente Autor.

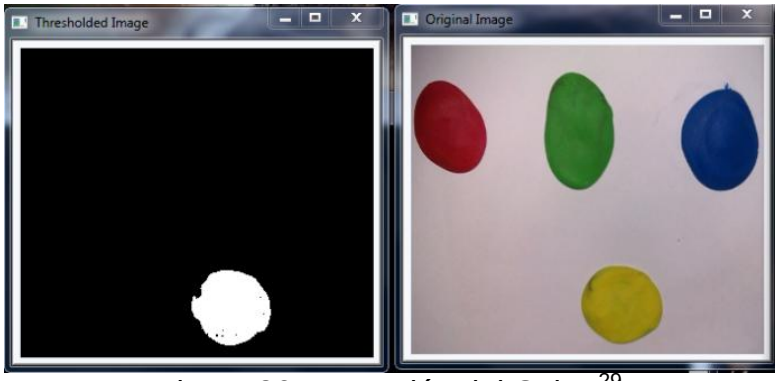

Figura 26. Detección del Color.[29]

- *cv.CloneImage(imagen).*Esta función crea una copia exacta de la imagen fuente, incluyendo el encabezado, la region de interés y los datos.

- *cv.Rect(x, y, ancho, alto)*. Realiza un rectángulo de las dimensiones deseadas, su principal uso se encuentra en la extracción de una región de interés.

- *cv.SetImageROI(imagen, rectángulo ROI)*. Selecciona una región de interés dentro de una imagen dada, mediante un rectángulo. Si los parámetros asignados al rectángulo no son iguales a la imagen total, la región es localizada. Los parámetros del rectángulo están dados en la función anterior.

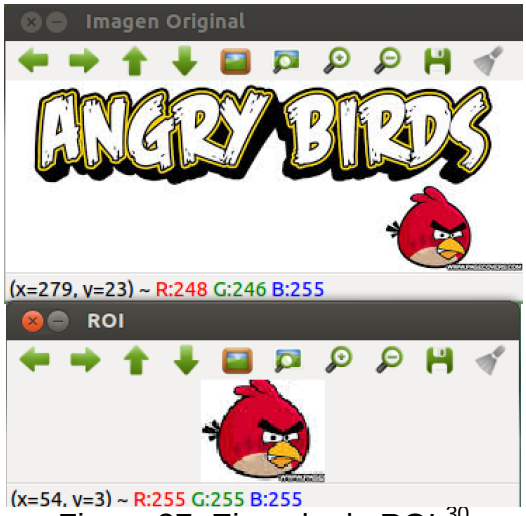

Figura 27. Ejemplo de ROI.[30]

[29] Detección de Color. <http://8a52labs.wordpress.com/tag/thresholding-images-in-opencv/> [Consultado el 10 de Abril de 2013]

[30]Fuente Autor.

- *cv.Split(fuente, destino0, destino1, destino2, destino3)*. Existen casos que no es conveniente trabajar con una imagen multicanal o con todos los canales de la misma. En este caso, se que separa la imagen multicanal en varias imágenes de un solo canal. Esta función copia los canales de la imagen fuente en las imágenes: destino0, destino1, destino2 y destino3 de forma organizada.

  Se debe tener en cuenta que la imágenes de destino deben tener el mismo tamaño de la imagen fuente y ser de un solo canal.

  Si la imagen fuente tiene menos de cuatro canales o canales que no son de interés se debe poner como argumento NONE.

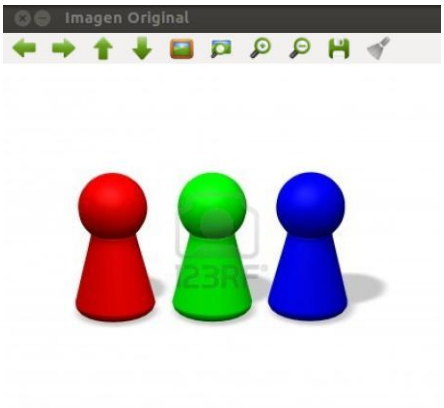

Figura 28. Imagen Original.

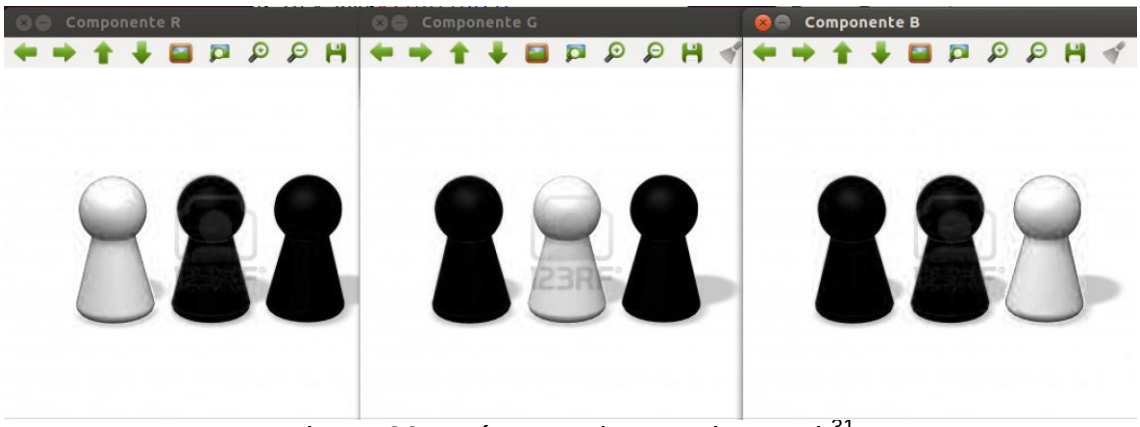

Figura 29. Imágenes de un solo canal.[31]

[31]Fuente Autor.

4. EVALUACIÓN DEL PROTOTIPO

4.1 VELOCIDAD DE PROCESAMIENTO

La primera prueba que se realizó, fue medir la velocidad de procesamiento tanto en el ordenador como en el sistema embebido, del programa escrito en Python.

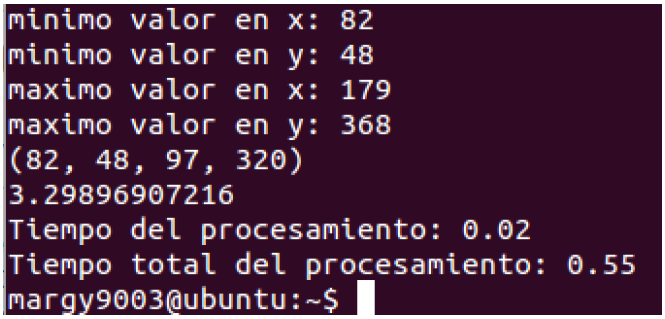

Figura 30. Velocidad de procesamiento en el ordenador.[32]

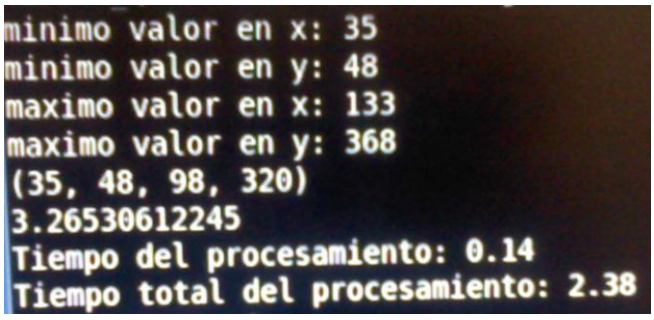

Figura 31. Velocidad de procesamiento en la Beagleboard.[33]

El tiempo del procesamiento (ver Figura 31-32), corresponde al tiempo que demora el programa en hacer todo el procesamiento de imagen, desde su captura hasta la obtención del nivel del líquido. El tiempo total de procesamiento, incluye el tiempo que demora el programa en la inicialización de todos los dispositivos periféricos. Hay que tener en cuenta que los dispositivos, solo se inicializan una vez, cuando se ejecuta el programa.

Se puede apreciar que el tiempo de procesamiento es más rápido en el ordenador (20 ms) que en el sistema embebido (140 ms). Aunque el tiempo sea mayor, es lo suficientemente bajo para aplicarlo en la industria. Disminuyendo ampliamente los costos, de la compra de un ordenador como núcleo central.

[32]Fuente Autor.

[33]Fuente Autor.

4.2 MÉTODO DE CONTROL ESTADÍSTICO DE PROCESOS

El “Control Estadístico de Procesos” nació a finales de los años 20 en los laboratorios Bell. Su creador fue W.A. Shewhart quien lo describió en su libro: “Economic Control of Quality of Manufactured Products” (1931). Después, Edwards Deming aplicó este método durante la segunda guerra mundial en Estados Unidos, mejorando exitosamente la calidad de la producción de armamento y otros productos de importancia estratégica.

En la manufactura, cuando se fabrican dos productos, no son exactamente iguales, las características no son uniformes y presentan variaciones. Puesto que el proceso de fabricación se ve afectado por un gran número de factores sometidos a una variabilidad, por ejemplo: variación de temperatura, humedad ambiental, repetitividad propia de la máquina, entre otras. Inciden en él e inducen a una variabilidad de las características del producto fabricado.

Estas características pueden deteriorar la calidad para el consumidor final o decrementar el margen de ganancias para la empresa. Además, presenta un clara ventaja frente a los métodos de calidad como la inspección, que detectan problemas ya cuando el producto se encuentra terminado, cuando ya es demasiado tarde. El control estadístico de procesos (SCP) mantiene la variabilidad de los productos finales dentro de unos límites aceptables, de tal manera que tanto los consumidores finales como la empresa obtengan un alto nivel de aceptación.

Su principal uso se encuentra en el proceso de fabricación. Se necesita de herramientas estadísticas sencillas. Lo que permite, que personal no calificado, lo pueda aplicar fácilmente para mejorar la calidad del producto. Cabe resaltar que es un método repetitivo, debe ser aplicado constantemente cuando se elabora un producto, para aumentar así el conocimiento de la variabilidad del producto dentro del proceso de la fabricación.

Si el proceso se encuentra operando de manera que existen pequeñas oscilaciones de todos estos factores, pero ninguno de ellos tiene un efecto preponderante frente a los demás, es esperable que la característica de calidad del producto fabricado se distribuya a la ley normal. Al conjunto de estos factores se le denomina causas comunes. Por el contrario, si la máquina tiene un desperfecto o las características de la materia prima cambian abruptamente la distribución ya no es normal y tiene una causa que es especial o asignable a alguno de los parámetros mencionados anteriormente.

4.2.1 FUNDAMENTOS ESTADÍSTICOS

Para el entendimiento de este método se deben recordar los siguientes conceptos:

La Distribución Normal o Campana de Gauss, es una función de probabilidad gaussiana posee una forma de acampanada, por ello su nombre. Como se aprecia en la Figura 32, esta gráfica depende de dos factores, la media (μ) y la desviación estándar (σ). Además, de ser simétrica respecto a la media.

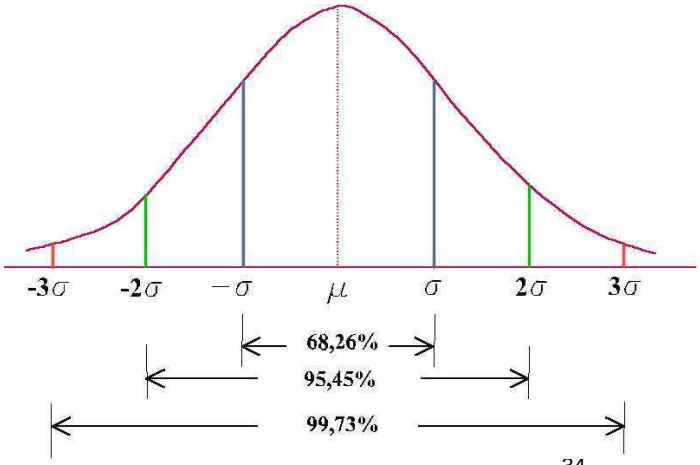

Figura 32. Campana de Gauss.[34]

Los múltiplos de la desviación estándar (σ) se ubican entorno a la media. Acorde, a la posición donde se encuentra ubicado, está contenido un porcentaje específico de la población.

Teorema del Límite Central, fija que "si una variable aleatoria (VA) se obtiene como suma de muchas causantes independientes, siendo cada una de ellas de poco importancia respecto al conjunto, entonces su distribución asintótica normal"[35]

$$X = [x_1, x_2, x_3 \ldots . x_n]$$

Ecuación 3. Muestras.

Los datos obtenidos tienen una media y una varianza entonces:

$$X \to N\left(\sum_{i=1}^{n} \mu \sqrt{\sum_{i=1}^{n} \sigma^2}\right)$$

Ecuación 4. Teorema del Límite Central.

[34]<http://www.imagenesi.net/campana-de-gauss/campana-de-gaus-2/>[Consultado el 12 de Junio de 2013]

[35]<http://web.cortland.edu/matresearch/controlprocesos.pdf>[Consultado el 12 de Junio de 2013]

Distribución de las medias muestrales, Si X es un variable aleatoria N(μ, σ) de la que se extraen muchas muestras de tamaño n, entonces las medias muestrales de distribuyen según otra ley normal[36]:

$$\overline{x_m} \propto N\left(\mu, \frac{\sigma}{\sqrt{n}}\right)$$

Ecuación 5. Distribución de las medias muestrales.

4.2.2 APLICACIÓN DEL CONTROL ESTADÍSTICO

Para la puesta en marcha del control estadístico se debe tener en cuenta dos etapas.

La primera, es una etapa de ajuste en la cual se debe tener una persona experta que verifique el proceso, esta persona se encarga de sacar unas muestras aleatorias del proceso previamente a las que se le aplica el proceso estadístico, se halla la media y la desviación estándar (ver Tabla 3, pág. 52), con las siguientes ecuaciones:

$$\bar{X} = \frac{\sum x_i}{N}$$

Ecuación 6. Media.

$$\sigma = \sqrt{\frac{(\bar{x} - x_i)}{n}}$$

Ecuación 7. Desviación estándar.

Luego, se calcula el límite superior e inferior de control de la siguiente forma:

$$L_s = \bar{X} + k\sigma$$
$$L_i = \bar{X} - k\sigma$$

Ecuación 8. Límites.

Aquí, surge una hipótesis de que la distribución de las observaciones es normal, donde k puede ser desde 2 hasta 3.09, es decir que los niveles de aceptación se encuentre entre el 95,45% hasta el 99.8%.

[36]<http://web.cortland.edu/matresearch/controlprocesos.pdf>[Consultado el 12 de Junio de 2013]

En la segunda etapa, los parámetros de interés son sometidos a un tratamiento estadístico y se verifica que su comportamiento este dentro de la campana. Si los resultados no muestran una distribución normal, hay que intervenir en el proceso para descubrir las causas de alteración. Antes de validar este proceso se debe verificar que esté ajustado y sino hay que repetir de nuevo este procedimiento.

Una vez, terminado este procedimiento se realiza un gráfico de control (ver Figura 33), es una técnica de diagnóstico y supervisión de procesos de producción que permite identificar inestabilidad o circunstancias anormales. En él se tiene en cuenta:

- La media
- El límite superior.
- El límite inferior.
- Las piezas del producto.

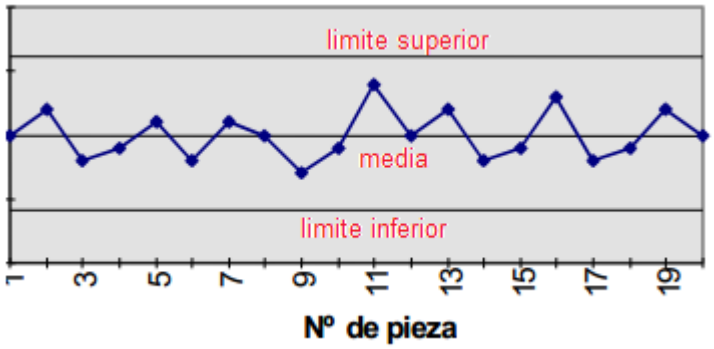

Figura 33. Gráfico de Control.

Este método de prueba se analiza detenidamente para verificar si está de acuerdo la hipótesis de que la variabilidad del proceso se debe solo a un sistema de causas aleatorias o si por el contrario existen causas asignables para su verificación, que lleven a tomar medidas correctivas.

4.3 AJUSTE DEL PROCESO

Para poder establecer los valores de: la media, la desviación estándar y los límites de rechazo (ver Tabla 3), se utilizó el programa Microsoft Excel, ya que es de fácil uso y cuenta con gran aceptación en el área de la ingeniería. Para ello, se sacaron treinta muestras aleatorias con botellas de 350 ml, con diferentes niveles de llenado en un ambiente de iluminación controlada. Para mejor visualización, se organizaron de acuerdo al nivel: las diez primeras botellas corresponden a las de bajo nivel de llenado, las diez siguientes cumplen con el estándar y las últimas, están por encima del nivel adecuado.

Se escogió trabajar con límites de rechazo de la media mas dos desviaciones estándar, ya que es el valor mínimo recomendado.

Tabla 3. Establecimiento de los límites.

BOTELLAS	PROPORCIÓN	MEDIA	DESVIACIÓN	$\mu+\sigma$	$\mu+2\sigma$
1	2,92	2,78	0,1	2,88	2,98
2	2,59				
3	2,67				
4	2,72				
5	2,85				
6	2,8			2,69	2,59
7	2,8				
8	2,83				
9	2,84				
10	2,83				
11	2,96	3,03	0,04	3,06	3,1
12	3,07				
13	3,04				
14	3,03				
15	3,01				
16	3,03			2,99	2,95
17	3,06				
18	3,03				
19	3,07				
20	2,98				
21	3,2	3,23	0,04	3,28	3,32
22	3,22				
23	3,29				
24	3,23				
25	3,25				
26	3,18			3,19	3,15
27	3,17				
28	3,23				
29	3,28				
30	3,29				

4.4 CONTROL DEL PROCESO

Para determinar la veracidad de los límites se hicieron varias pruebas a diferentes horas del día en el transcurso de una semana. Se tomaron treinta muestras aleatorias con botellas de diversos niveles de llenado. Como se dijo anteriormente, las diez primeras se encuentran por debajo, las diez siguientes en el nivel adecuado y la última decena, por encima del nivel. Los datos obtenidos se muestran a continuación.

Tabla 4. Prueba realizada a las 10:00 el 30 de Mayo.

BOTELLAS	PROPORCIÓN	RECHAZO
1	2,78	SI
2	2,86	SI
3	2,68	SI
4	2,85	SI
5	2,78	SI
6	2,71	SI
7	2,77	SI
8	2,91	SI
9	2,83	SI
10	2,62	SI
11	3,01	NO
12	3,02	NO
13	3,03	NO
14	3,02	NO
15	3,06	NO
16	3,05	NO
17	3,02	NO
18	3,07	NO
19	3,05	NO
20	3,03	NO
21	3,17	SI
22	3,21	SI
23	3,18	SI
24	3,25	SI
25	3,20	SI
26	3,23	SI
27	3,33	SI
28	3,24	SI
29	3,20	SI
30	3,31	SI

Tabla 5. Prueba realizada a las 11:18 el 30 de Mayo.

BOTELLAS	PROPORCIÓN	RECHAZO
1	2,63	SI
2	2,86	SI
3	2,88	SI
4	2,66	SI
5	2,87	SI
6	2,64	SI
7	2,74	SI
8	2,82	SI
9	2,86	SI
10	2,62	SI
11	3,01	NO
12	3,04	NO
13	3,09	NO
14	3,02	NO
15	3,05	NO
16	3,06	NO
17	3,08	NO
18	3,01	NO
19	3,08	NO
20	3,08	NO
21	3,18	SI
22	3,24	SI
23	3,27	SI
24	3,24	SI
25	3,19	SI
26	3,28	SI
27	3,24	SI
28	3,20	SI
29	3,25	SI
30	3,17	SI

Tabla 6. Prueba realizada a las 14:00 el 30 de Mayo.

BOTELLAS	PROPORCIÓN	RECHAZO
1	2,88	SI
2	2,77	SI
3	2,66	SI
4	2,85	SI
5	2,71	SI
6	2,87	SI
7	2,91	SI
8	2,78	SI
9	2,82	SI
10	2,87	SI
11	3,00	NO
12	3,09	NO
13	3,07	NO
14	3,03	NO
15	2,96	NO
16	3,08	NO
17	3,07	NO
18	3,05	NO
19	3,06	NO
20	3,08	NO
21	3,18	SI
22	3,24	SI
23	3,23	SI
24	3,25	SI
25	3,27	SI
26	3,24	SI
27	3,28	SI
28	3,24	SI
29	3,16	SI
30	3,23	SI

El resto de las pruebas se encuentran consignadas en el Anexo B.

Se escogió una tabla aleatoria y se realizó un gráfico de control para supervisar el proceso de control de calidad.

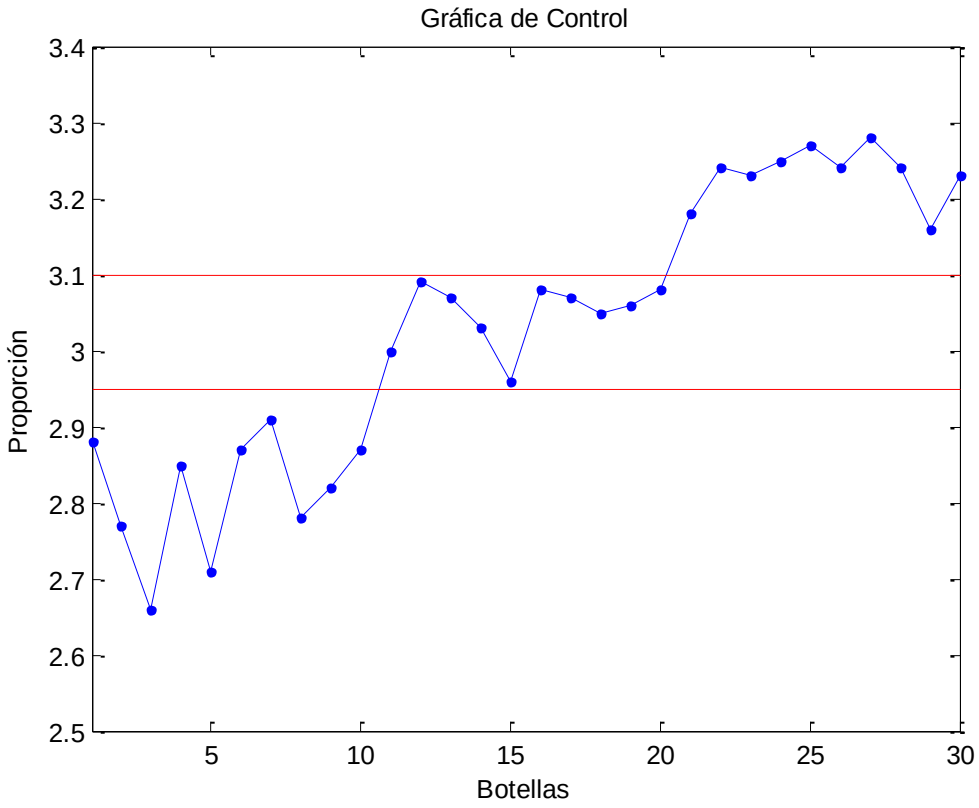

Figura 34. Gráfico de Control Real.

Se observa una línea quebrada irregular que muestra las fluctuaciones del nivel de llenado de las botellas a lo largo del tiempo. Esta fluctuación es esperable y natural del proceso debido a que las botellas tienen un diferente nivel de llenado. Entre la muestra once hasta la veinte, los valores se mueven alrededor de un valor central (media), debido a que las botellas cuentan con un nivel adecuado. Pero la mayor parte del tiempo, aparecen valores demasiado alejados de la media por lo que se mantienen fuera de los límites de control, ya que su nivel no es el correcto

5 INTERFAZ GRÁFICA DE USUARIO

Conocido también con el nombre de GUI, permite que el usuario interaccione con dispositivos electrónicos mediante un conjunto de elementos gráficos como: íconos, botones, ventanas y acciones. Su fin es brindar un entorno visual sencillo que proporcione la comunicación entre el sistema operativo y la Beagleboard, remplazando la línea de comandos.

Las acciones se realizan de forma directa, facilitando la interacción de usuario-Beagleboard. Entre los programas especializados para desarrollar GUI´s en Python se encuentran: PyQt,Tkinter y PyGame.

5.1 PyQt

Desarrollado inicialmente por una compañía Noruega llamada Trolltech, queluego, fue adquirida por Nokia y renombrada como:“Qt DevelopmentFrameworks”. En Mayo de 1995, este producto fue lanzado al mercado.

Entre las aplicaciones más famosas creadas con Qt están: Skype, VLC, Opera, Mathematica, entre otras.

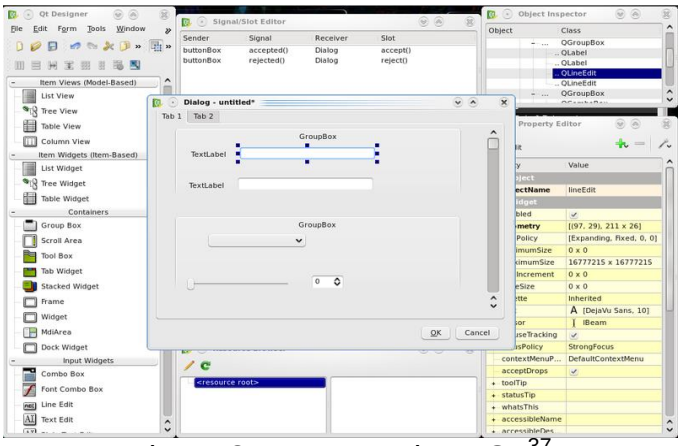

Figura 35. Entorno de PyQt.[37]

Presenta tres grandes ventajas frente a sus rivales:

- Multiplataforma.
- Gratuito para código abierto.
- Extensa librería con clases y herramientas para la creación de aplicaciones.

[37] <http://en.wikipedia.org/wiki/File:Qt_Designer_4_4_3.png> [Consultada el 20 de Mayo de 2013].

5.2 INSTALACIÓN DE PyQt

Para la instalación de PyQt4, que es el software elegido para realizar la interfaz grafica, es recomendable usar Python 2.7 debido a la estabilidad y compatibilidad que posee con los paquetes de PyQt4.

Primero, se debe instalar el gestor de paquetes. El Instala, elimina, actualiza y descarga automáticamente los paquetes que se le indiquen, así como sus dependencias, y opcionalmente, recomendaciones y sugerencias

```
sudo apt-get install aptitude
```

Luego, se procede a buscar el paquete de PyQt4.

```
aptitudesearch qt4
```

Aparecerá una lista de resultados que contienen el parámetro “Qt4”, se elige el paquete, acorde a la versión de Python instalada, para este caso 2.7.

```
sudo apt-get install python2.7-qt4
```

Para comprobar que el paquete ha sido instalado correctamente, se ingresa a Python.

```
python
```

Aquí se muestra la versión de Python, año de creación, entre otros datos. Se escriben las siguientes líneas de código para comprobar que tenga asociado las librerías de PyQt4.

```
import PyQt4
exit()
```

Ahora se descarga PyQt4 de la pagina oficial:
www.riverbankcomputing.co.uk/software/pyqt/download

Es necesario descargar también el modulo SIP, encargado de la iniciación, modificación y finalización de sesiones interactivas de usuario, donde intervienen elementos multimedia como el video.

http://www.riverbankcomputing.co.uk/software/sip/download

Una vez descargado los dos paquetes se descomprimen.

```
tar -zxvf PyQt-x11-gpl-4.10.1.tar.gz
tar -zxvf sip-4.14.6.tar.gz
```

La instalación debe realizarse en el orden señalado. Para hacer la configuración, se debe entrar al directorio del archivo .tar.

```
python configure.py
make
sudo make install
```

Para verificar su correcta instalación, se ingresa a Python y se escribe lo siguiente:

```
python
from PyQt4 import QtCore, QtGui
exit()
```

Instalación de Qt-Designer

Qt4-Designer es un programa que permite realizar una GUI de forma más amigable, modo diseñador, en vez de ser escrita directamente en código. Después de hacer el entorno grafico, se introduce un comando que transforma lo colocado en código.

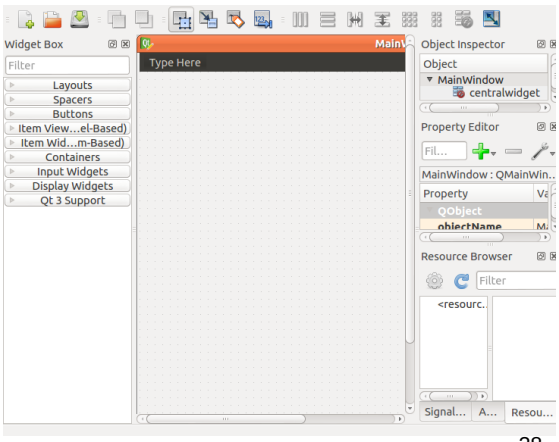

Figura 36. Entorno de Qt-Designer.[38]

Para su instalación se busca qt4-designer, de igual forma como se había hecho con Qt4, seguido de la instalación del paquete.

```
aptitude search qt4-designer
sudo apt-get install qt4-designer
```

[38]Fuente Autor.

5.3 GUI CREADA

Se creó una interfaz grafica de usuario para proporcionar un entorno visual sencillo que permita interactuar directamente con el programa desarrollado en Python.

Figura 37. GUI Creada.[39]

Funcionamiento de la GUI

Después de colocar en marcha la aplicación aparecerá una ventana (ver Figura 34), con espacios visiblemente en blanco. En ellos se mostrará la imagen capturada por la cámara (c920), las imágenes resultantes del procesamiento de imágenes(ver Figura 35), la proporción de cada una de las botellas analizadas y el número de admitidas y/o rechazadas.

Figura 38. GUI en Funcionamiento.[40]

[39]Fuente Autor.

[40]Fuente Autor.

El recuadro de la izquierda corresponde al visor de la cámara, el cual se actualiza constantemente con la imagen capturada. Existe un pequeño retardo en la colocación de la imagen.

Las dos imágenes del centro, muestran los resultados obtenidos, después del procesamiento de imagen aplicado a la botella. La imagen binarizada corresponde al proceso de segmentación, mientras que la otra imagen al contorno de la botella.

A la derecha, se encuentra una lista que contiene el historial de las proporciones determinadas para cada botella. Seguido, de un pequeño inventario de las botellas admitidas y rechazadas.

El botón de START es el encargado de colocar todo el proceso en funcionamiento. Lo que significa principalmente:

- Entablar la comunicación entre el PLC y la Beagleboard mediante ModbusTCP.
- Activar la banda transportadora.
- Importar las librerías de OpenCV correspondiente a la captura de la secuencia de video de la camara web.

Además, contiene el código interno encomendado de realizar el procedimiento, desde la captura de a imagen hasta la obtención del nivel de llenado (ver Figura 8).

En cambio, el botón de STOP es el responsable de finalizar el proceso. Lo que quiere decir:

- Cerrar la comunicación con el autómata.
- Detener el motor que se encuentra acoplado a la banda transportadora.
- Cerrar la ventana del programa.

La interfaz representa un entorno más cómodo, amigable y sencillo para el usuario, teniendo las mismas funcionalidades al alcance.

ANEXO A - PROGRAMACIÓN DEL AUTÓMATA

En la Figura 32 se muestra la programación implementada en el PLC. Entre las variables empleadas se encuentran: entradas, salidas y registros internos.

De las tres entradas utilizadas, dos se encuentran conectadas a sensores fotoeléctricos. La primera entrada (I0.0), detecta el paso de la botella para capturar su imagen, posteriormente realizar su procesamiento. La entrada siguiente (I0.1), detecta la misma para su clasificación. Por último, la entrada I0.2 detecta que el vástago del cilindro neumático se encuentre accionado para efectuar un correcto rechazo.

Además, cuenta con dos salidas. La Q0.2, acciona la banda transportadora para el desplazamiento de las botellas. La Q0.3, activa la electroválvula neumática para el movimiento del cilindro.

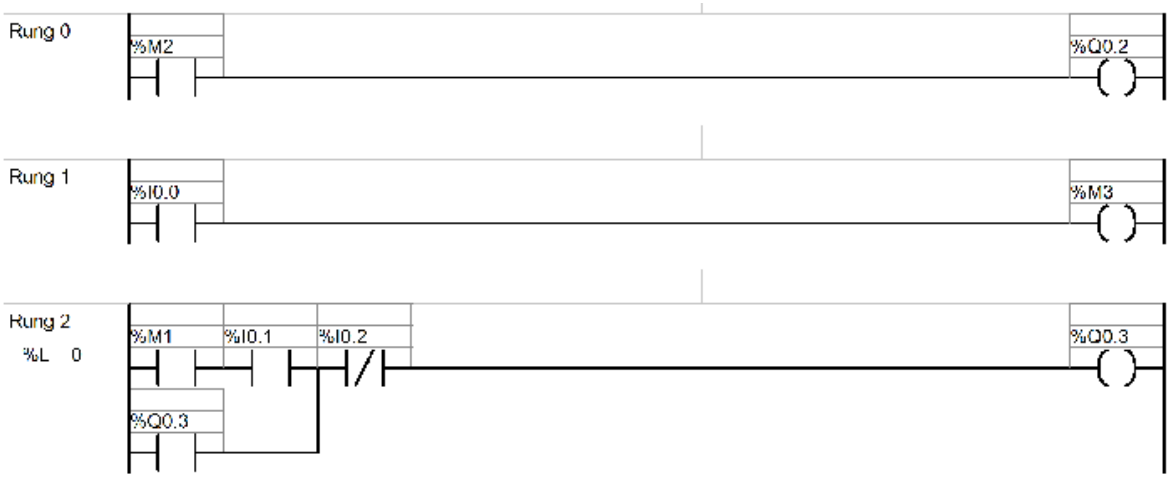

Figura 39. Programación interna del PLC.[41]

Se puede ver que se emplea de forma significativa los registros internos del autómata. Esto se debe a que la librería Pymodbus, solo puede modificar los registros internos, no sus salidas directamente. Mediante ellos, se almacena el estado de las variables para ser utilizadas, luego en el PLC o en Python.

En la línea numero dos se encuentra un circuito de apagado. Este, es el responsable de realizar el descarte de la botella cuando su nivel no es el adecuado. Una vez, el vástago haya alcanzo su extensión total es detectado por el sensor magnético, qué desactiva inmediatamente la bobina, haciendo que el mismo se contraiga.

[41]Fuente Autor.

ANEXO B – PRUEBAS

Tabla 7. Prueba realizada a las 09:26 el 1 de Junio.

BOTELLAS	PROPORCIÓN	RECHAZO
1	2,90	SI
2	2,66	SI
3	2,70	SI
4	2,88	SI
5	2,55	SI
6	2,83	SI
7	2,81	SI
8	2,67	SI
9	2,67	SI
10	2,86	SI
11	3,08	NO
12	3,03	NO
13	3,00	NO
14	3,07	NO
15	3,03	NO
16	3,06	NO
17	3,07	NO
18	3,04	NO
19	3,06	NO
20	3,01	NO
21	3,22	SI
22	3,15	SI
23	3,17	SI
24	3,21	SI
25	3,22	SI
26	3,27	SI
27	3,23	SI
28	3,23	SI
29	3,20	SI
30	3,31	SI

Tabla 8. Prueba realizada a las 15:15 el 5 de Junio.

BOTELLAS	PROPORCIÓN	RECHAZO
1	2,81	SI
2	2,67	SI
3	2,57	SI
4	2,65	SI
5	2,79	SI
6	2,81	SI
7	2,68	SI
8	2,75	SI
9	2,82	SI
10	2,77	SI
11	2,98	NO
12	2,97	NO
13	2,97	NO
14	3,05	NO
15	3,01	NO
16	2,98	NO
17	3,00	NO
18	3,03	NO
19	3,02	NO
20	2,95	NO
21	3,27	SI
22	3,16	SI
23	3,22	SI
24	3,11	SI
25	3,19	SI
26	3,20	SI
27	3,13	SI
28	3,14	SI
29	3,28	SI
30	3,21	SI

Tabla 9. Prueba realizada a las 15:00 el 6 de Junio.

BOTELLAS	PROPORCIÓN	RECHAZO
1	2,70	SI
2	2,71	SI
3	2,76	SI
4	2,73	SI
5	2,84	SI

6	2,77	SI
7	2,63	SI
8	2,59	SI
9	2,74	SI
10	2,90	SI
11	3,00	NO
12	3,02	NO
13	3,03	NO
14	3,07	NO
15	2,94	SI
16	3,03	NO
17	3,97	NO
18	3,07	NO
19	2,98	NO
20	2,99	NO
21	3,22	SI
22	3,23	SI
23	3,16	SI
24	3,22	SI
25	3,25	SI
26	3,14	SI
27	3,23	SI
28	3,27	SI
29	3,25	SI
30	3,24	SI

Tabla 10. Prueba realizada a las 17:00 el 6 de Junio.

BOTELLAS	PROPORCIÓN	RECHAZO
1	2,63	SI
2	2,71	SI
3	2,84	SI
4	2,80	SI
5	2,86	SI
6	2,86	SI
7	2,79	SI
8	2,78	SI
9	2,91	SI
10	2,75	SI
11	3,05	NO

12	3,04	NO
13	2,98	NO
14	3,02	NO
15	3,10	NO
16	3,05	NO
17	3,03	NO
18	3,05	NO
19	3,01	NO
20	2,98	NO
21	3,22	SI
22	3,28	SI
23	3,27	SI
24	3,26	SI
25	3,22	SI
26	3,27	SI
27	3,21	SI
28	3,29	SI
29	3,21	SI
30	3,30	SI

ANEXO C – INSTALACIÓN DE OPENCV

En este Anexo se describe de forma detallada el procedimiento para la instalación de OpenCV.

Primero se debe instalar el paquete que contiene las librerías de C

```
sudo apt-get install libcv-dev
```

Además, se instalan otras dependencias para su adecuado funcionamiento

```
sudo apt-get install build-essential checkinstallcmakepkg-configyasm.
```

Seguido, se instalan paquetes para la lectura y escritura de los diferentes formatos de imagen.

```
sudo apt-get install libtiff4-dev libjpeg-devlibjasper-dev.
```

Después, se instala los paquetes para la captura de video, su codificación y decodificación.

```
sudo apt-getinstalllibavcodec-devlibavformat-devlibswscale-dev
libdc1394-22-dev libxine-dev libgstreamer0.10-dev libgstreamer-
plugins-base0.10-dev libv4l-dev
```

Se instala las dependencias necesarias para construir los contenedores de Python.

```
sudo apt-getinstall python-dev python-numpy
```

La biblioteca para desarrollar interfaces gráficas por usuario es GTK. Opcionalmente, se puede instalar Qt y activarlo póstumamente. Se instalan las dos bibliotecas para no perder sus utilidades.

```
sudo apt-getinstall libqt4-dev libgtk2.0-dev
```

Ahora, se procede a realizar la instalación de OpenCV. Para ello, se debe descarga el paquete de su pagina oficial (*http://opencv.org/downloads.html*).

Una vez, descargado el paquete, se descomprime el archivo .tar.

```
tar-xvf OpenCV-2.4.0.tar.bz2
```

Se ingresa al directorio de descarga.

```
cd OpenCV-2.4.0
```

Se crea un directorio llamado build.

```
mkdirbuild
```

Se ingresa al directorio recién creado.

```
cd build
```

Se realiza una configuración usando CMake, de los paquetes instalados anteriormente.

```
cmake-D WITH_QT = ON-D WITH_XINE = ON-D WITH_OPENGL = ON-D
WITH_TBB = ON-D BUILD_EXAMPLES = ON ..
```

La bandera –D WITH_QT = ON, corresponde al modulo de HighGUI usado en Qt. Luego se efectúa la compilación del software.

```
make
```

Si no se presentan errores, se procede a hacer la instalación de OpenCV.

```
sudo makeinstall
```

Por último, se debe asegurar que los programas realizados se puedan vincular a las bibliotecas de OpenCV en el tiempo de ejecución. Por esto se añade la siguiente línea, al final del archivo /etc/ld.so.conf:

/Usr/local/lib

La única forma de llevar a cabo este proceso es mediante la línea de comandos.

```
sudo sh -c "echo '/usr/local/lib' >> /etc/ld.so.conf"
```

Se configura los enlaces dinámicos en tiempo de ejecución.

```
sudo ldconfig
```

Para verificar la correcta instalación del programa y sus librerías, se ejecuta un programa de prueba. Se debe ingresar al directorio OpenCV-2,4,0/build/bin y ejecutar:

```
./Opencv_test_core
```

Debe aparecer lo siguiente:

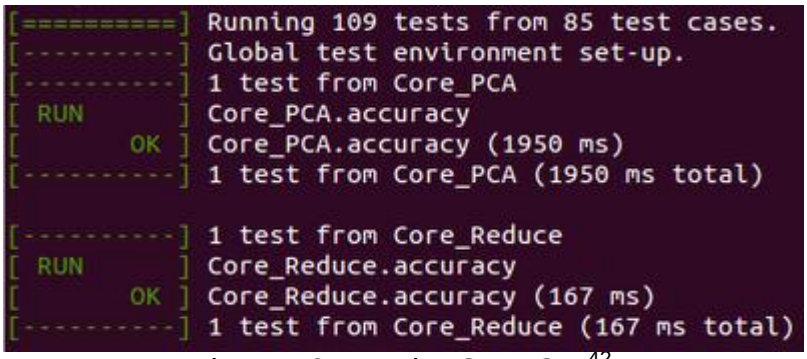

Figura 40. Prueba OpenCV.[42]

[42]Fuente Autor.

6 CONCLUSIONES Y RECOMENDACIONES

- Se creó e implemento un algoritmo en python, para determinar el nivel de llenado de una botella de gaseosa carbonatada, por medio de las librerías de opencv dedicadas al procesamiento digital de imágenes que son unas de las más completas que existen actualmente.

- Se desarrolló un prototipo experimental, robusto, capaz de inspeccionar el llenado de las botellas de Coca-Cola y simular las condiciones físicas de la planta, ubicada en la ciudad de Bucaramanga. Este, se elaboró bajo un sistema embebido de bajo costo, empleando software libre.

- Para el monitoreo del proceso se desarrolló dos interfaces: una plana y otra gráfica que se pueden acceder desde una pantalla o un terminal de acceso remoto. La primera, demoró 0.2s en realizar el procesamiento de la imagen, mientras que la segunda demoró 0.45s. Lo que quiere decir, solo la primera cumple con la condición del tiempo de trabajo para la aplicación, ya qué la producción de la planta, es de una gaseosa por segundo.

- El proceso de puesta a punto de la maquina de inspección, se basó en el método de control estadístico de procesos. Las muestras fueron recolectadas a diferentes horas del día, debido a que la planta de embotellado trabaja durante las 24 horas. Finalmente, se obtuvo un nivel de aceptación del 1 en 400 botellas lo que infiere un porcentaje muy pequeño de error, que es apto para la implementación en la planta.

- Por ahora, no se ha tenido acceso al proceso real, pero en un futuro se piensa hacer las pruebas en la planta de embotellado, ya qué el gerente esta interesado en ponerlo a funcionar e implementarlo a nivel nacional, puesto que los productos comerciales no han cumplido los resultados esperados.

- El procesamiento de imágenes tiene gran aplicación en el área control de calidad. Aquí se ha presentado una aplicación especifica: “Control de nivel de llenado”, para una empresa de la región, con lo que se sirve como preámbulo para la realización de futuros proyectos que se encuentran en lista de espera.

- Como recomendación, se deja una plataforma robusta al sector academico e industrial, capaz de determinar el control de llenado de una botella de Coca-Cola, impulsando la investigación en esta rama específica,en la escuela de Ingenería Electrónica.

- Con el desarrollo de este equipo, se deja abierta la posibilidad del

mejoramiento del mismo, implementando un algoritmo de mayor fiabilidad y complejidad como lo es el Espacio de Hering. Aumentado así, notablemente el desempeño del equipo y permitiendo una mejor adaptabilidad a las necesidades de la industria.

7 BIBLIOGRAFÍA

- ✓ Lutz, Mark. Learning Python. 4° Ed. O'reilly: 2009. ISBN: 978-0-596-15806-4.
- ✓ Kaebler, Adrian. Bradsky, Gary. Learning OpenCV: Computer Vision With The OpenCV Library. 1° Ed. O'reilly: 2008. ISBN: 978-0-596-51613-0.
- ✓ Beagleboard. <http://beagleboard.org/hardware-xm> [Consultado el 1 Abril de 2013].
- ✓ Sensor Fotoeléctrico. <http://oceancontrols.com.au/PES-007.html> [Consultado el 10 de Abril de 2013].
- ✓ Electroválvula Neumática. <http://www.alibaba.com/product-gs/745530421/SMC_Solenoid_Valve.html> [Consultado el 3 de Abril de 2013].
- ✓ PLC telemecanique. <http://products.schneider-electric.us/products-services/productdetail/?event=datasheet&partnumber=TWDLCAA24DRF&countrycode=us> [Consultado el 2 de Abril de 2013].
- ✓ Cámara web Logitech c920. <http://www.logitech.com/es-es/webcam-communications/articles/7507> [Consultado el 3 Abril de 2013].
- ✓ Variador de Velocidad de Schneider Electric. <http://schneider.thomasnet.com/item/variable-frequency-drives2/atv312-variable-frequency-drive/atv312h075m2>[Consultado el 3 de Abril de 2013].
- ✓ Pymodbus. <https://pymodbus.readthedocs.org/en/latest/library/sync-client.html [Consultado el 4 de Abril de 2013].
- ✓ Libreria HighGUI. <http://www.cognotics.com/opencv/docs/1.0/ref/opencvref_highgui.htm>[Consultado el 4 de Abril de 2013].
- ✓ Contornos. <http://stackoverflow.com/questions/8830619/difference-between-cv-retr-list-cv-retr-tree-cv-retr-external>[Consultado el 4 de Abril de 2013].