Universidad del Bio Bío

Facultad de Ciencias Empresariales

Escuela de Ingeniería Civil Informática.

Desarrollo de aplicación de control de celda flexible de inspección visual mediante dispositivos Android.

Proyecto presentado en conformidad con los requisitos para optar al título de:

Ingeniería Civil en Informática.

Por:

Víctor M. Monsalves Rivas.

Profesor Guía:

Dra. Tatiana Gutiérrez Bunster.

Profesor C-Guía:

Luís Vera Quiroga.

Julio de 2017.

Concepción - Chile

Resumen

El proyecto titulado como “Desarrollo de Aplicación de Control de Celda Flexible e Inspección mediante dispositivos Android”, se presenta para dar conformidad a los requisitos exigidos por la Universidad de Bío-Bío en el proceso de titulación para la carrera de Ingeniería Civil Informática.

En este proyecto se desarrolló una aplicación para dispositivos móviles con sistema operativo Android para ejecutar el control de la estación de control e inspección visual del sistema de manufactura flexible (SMF) del Laboratorio de Sistemas Automatizados de Producción de la Universidad del Bio Bío (CIMUBB).

El sistema de manufactura flexible corresponde a un grupo de estaciones de trabajo equipadas e interconectadas por medio de un sistema de control y transporte, cuya función es la manufactura de piezas en madera de manera automatizada. La aplicación desarrollada trabaja en el control de una de estas estaciones correspondiente a la estación de inspección y control (Estación 3) cuya función es el control de calidad automatizada mediante visión artificial.

El sistema actual que administra el SMF (OpenCIM) se debe ejecutar en sistemas operativos anteriores a Windows XP por lo que se propuso administrar el SMF mediante dispositivos móviles, como consecuencia también surgió la necesidad de desarrollar una aplicación para Smartphone que ejecutara el proceso de control e inspección visual en el proceso de manufactura del laboratorio en la celda flexible de inspección (Estación 3).

La aplicación desarrollada de forma nativa involucra diversas técnicas como el procesamiento digital de imágenes, visión artificial, desarrollo en lenguaje de programación Java, comunicación inalámbrica, robótica y la integración entre estas. Esta posee funcionalidades como controlar el brazo-robot de la estación de inspección mediante smartphone; establecer las variables de ubicación, de comparación y de movimientos a ejecutar por el robot de los objetos inspeccionados mediante la cámara de un Smartphone.

La función principal del sistema es la captura de la región de interés del objeto a comparar, ejecutar la comparación con un objeto de referencia almacenado en la memoria del Smartphone que ejecuta la cámara, entregar el resultado de la comparación y decidir si la pieza fabricada a comparar cumple con el porcentaje mínimo de similitud aceptada para ser guardada en un almacén por el brazo robot, o en caso contrario ser rechazada y botada a un recipiente por este, por lo que el robot realiza una rutina de movimientos de acuerdo a la posición de captura, productos aceptados y/o productos descartados cumpliéndose así el proceso de inspección y control de calidad del SMF.

La aplicación se ejecuta mediante dos modos, uno es el modo Administrador es el que permite controlar el Smartphone que ejecuta la aplicación en modo Cámara situado sobre la pinza del robot y que ejecutará las ordenes de procesamiento de imágenes y de control de la celda flexible de inspección, la aplicación interactúa mediante comunicación a través de WI-FI entre los Smartphone, y comunicación Bluetooth AT entre el Smartphone y la celda flexible mediante un conversor Bluetooth-serial.

El aporte de este proyecto propone el reemplazo de los computadores existentes y que controlan los procesos de producción, de manera de integrar las funciones tecnológicas en un solo objeto, el Smartphone, que posee una constante actualización en hardware permitiendo que este dispositivo móvil se “incruste” en el propio robot, con la consiguiente disminución de equipamiento e independencia, además de facilitar a futuro agregar características propias de la tecnología IoT (Internet de las cosas).

Índice General

Índice Tablas

Índice Figuras

Nomenclatura.

- CIM: Computer Integrated Manufacturing (Manufactura Integrada por Computador).
- CIMUBB: Laboratorio de Manufactura Integrada por Computador de la Universidad del Bío Bío.
- UML: Unified Modeling Language (Lenguaje Unificado de Modelado).
- MER: Modelo Entidad Relación.
- LA: Layout.
- RF: Requerimiento Funcional.
- DE: Interfaz o dato de entrada.
- IS: Interfaz o dato de salida.
- DFD: Diagrama de Flujo de Datos.
- CU: Caso de Uso.
- BPMN: Bussiness Process Model Notation.
- IEEE: Institute of Electrical and Electronics Engineers (Instituto de Ingeniería Eléctrica y Electrónica).
- IU: Interfaz de usuario.
- GUI: Interfaz gráfica de usuario.
- ROI: Region of Interest (Región de Interés).
- ACL: Advanced Control Language (Lenguaje Avanzado de Control).
- ATS: Advanced Terminal Software (Software Terminal Avanzado).
- Tecnología IoT: Internet of Things (El internet de las cosas).
- ER-V: Modelo y versión del brazo robot flexible Scorbot empleado en este proyecto.
- OpenCV: Open Source Computer Vision.
- IVA: Inteligencia y Visión Artificial.
- PLC: Controlador lógico programable
- CNC: Control Numérico Computarizado.
- AC: Control Adaptativo.
- P&P: Manejo automatizado de los materiales.
- ASRS: Sistema de ensamble automatizado y robótico.
- CAPP: Planeación de procesos asistidos por computadora.
- GT: Tecnología en grupo.
- JIT: Producción justo a tiempo
- FMC: Manufactura celular.
- FMS: Sistemas de manufactura flexible.
- ERP: Planificación de Recursos Empresariales.

1 INTRODUCCIÓN.

En este mundo competitivo y globalizado, las empresas de manufactura luchan constantemente por alcanzar un puesto elevado en el mercado. En el área de manufactura de una empresa esto se logra mediante distintos factores en el proceso de manufactura como la calidad de los productos, el tiempo de producción, la diversidad de los productos y la flexibilidad en la producción, entre otros. Todo esto sumado a las nuevas exigencias del mercado, trayendo como consecuencia de que consolidarse como empresa no sea una tarea fácil, por lo que, actualmente los procesos automatizados es uno de los aspectos más complejos que se han interiorizado en las empresas de manufactura. En consecuencia, las empresas se tienen que adaptar y equipar con nuevas herramientas para mejorar la producción y calidad de los productos y servicios. Una de las herramientas más importantes es la tecnología, la que permite automatizar el proceso de manufactura naciendo así los sistemas de manufactura flexible (SMF).

Un sistema de manufactura flexible corresponde a un grupo de estaciones de trabajo equipadas e interconectadas por medio de un sistema de control y transporte, cuya función es la fabricación de piezas de manera automatizada. Comúnmente un SMF está compuesto por celdas flexibles de manufactura o estaciones de trabajo. Cada una incluye un conjunto de máquinas y dispositivos como por ejemplo brazos robóticos, PLCs, cinta transportadora, CNC, etc.

Los SMF son en la mayoría controlados por sistemas computacionales por lo que los computadores han sido integrados en la industria manufacturera naciendo así el concepto CIM (Computer Integrated Manufactured) que significa la integración de los computadores en un sistema de manufactura.

El Laboratorio de Sistemas Automatizados de Producción de la Universidad del Bio-Bío CIMUBB dispone de un Sistema de Manufactura Flexible organizados en tres estaciones de trabajo más un coordinador (Manager) simulando una planta de manufactura automatizada. Estas estaciones del SMF corresponden a la estación de almacenamiento y proveedor de materia prima, la estación de mecanizado, y a la estación de inspección o control de calidad, siendo la estación de control de calidad el lugar en donde termina el proceso de manufactura de un producto elaborado por el SMF y es donde operará la aplicación desarrollada en este proyecto con el objetivo de establecer el control de calidad de un objeto.

En el presente trabajo se pretende trabajar en la estación de inspección y control de calidad. Esta estación actualmente dispone de una celda flexible de inspección compuesta de un computador, de un brazo robótico, una cámara web, una prensa y de software para la inspección mediante visión artificial para el diagnóstico de las piezas manufacturadas por las demás estaciones del sistema de manufactura flexible.

El principal objetivo de este trabajo es replicar y optimizar el proceso de inspección visual integrando el uso de los dispositivos móviles a este proceso de manera de modernizar la plataforma tecnológica de la aplicación utilizada en este proceso reemplazando el uso de los computadores. Para ello se desarrolló una aplicación móvil de prototipo, que es capaz de controlar el brazo robótico de la estación de inspección del SMF mediante comunicación remota y de evaluar los productos elaborados en madera desde estación de mecanizado del SMF por medio de visión artificial de acuerdo a criterios de similitud e identificación de objetos o figuras. La aplicación móvil desarrollada puede operar en dispositivos móviles que funcionen bajo el sistema operativo Android 4.4.3 o superior.

El desarrollo de la aplicación involucra técnicas de procesamiento de imágenes, visión artificial, comunicación inalámbrica, robótica y la integración entre ellas, por lo que involucra distintos desafíos, que serán explicados en los siguientes capítulos de este documento.

Se documentará todo el proceso de elaboración de este trabajo, incluyendo los elementos que forman parte de la elaboración de la aplicación, como las técnicas de comunicación, de visión artificial, y robótica, además de la especificación de requerimientos, análisis y diseño.

El contenido de este proyecto de título se ha presentado de la siguiente forma:

En el capítulo 2 se define la empresa y el área de estudio en donde se desarrolla el proyecto, además, se da una breve definición acerca del concepto CIM (Manufactura Integrada por Computador).

En el capítulo 3 se definen los objetivos del proyecto, la descripción de la problemática y la situación actual, la metodología y estándares de documentación y de desarrollo incluyendo las siglas y abreviaciones utilizadas a lo largo de este informe.

En el capítulo 4 se definen los alcances y limitaciones del proyecto, el objetivo general y los objetivos específicos de la aplicación, además de una descripción global del producto. Por último, se definirán los requerimientos específicos del sistema.

En el capítulo 5 se define la factibilidad del proyecto en el ámbito operativo, técnico y económico de manera de analizar la disponibilidad de los recursos necesarios para lograr cumplir con las metas y objetivos del proyecto.

En el capítulo 6 se definen los conceptos acerca del procesamiento digital de imágenes entregando una breve base teórica acerca del tema. Además, se explica el pipeline o secuencia de las funciones implementadas relacionadas al procesamiento de imágenes junto a las funciones utilizadas de la librería de visión artificial OpenCV.

En el capítulo 7 se da a conocer las características básicas del brazo robot Scorbot, utilizado como dispositivo actuador de la estación de inspección que permitirá el traslado de objetos como el Smartphone en el que operará la cámara y los objetos manufacturados en la estación de mecanizado. Además, se definen los comandos y programas del robot que se serán utilizados para la ejecución del sistema.

En el capítulo 8 comprende la etapa de análisis del sistema describiendo la estructura de los datos, los flujos de información, la interacción con los usuarios implicados en el sistema a través de diferentes diagramas, como son procesos de negocios futuros, diagrama de flujo de datos (DFD) y diagramas de casos de uso (CU).

En el capítulo 9 se define la fase del diseño de la aplicación, definiendo su diseño arquitectónico, su funcionalidad, y el diseño de la interfaz gráfica de usuario y de navegación.

En el capítulo 10 se detalla el diseño de las pruebas que se realizan al sistema y el resultado esperado de cada característica o función a probar, con la finalidad de demostrar su correcto funcionamiento y eficacia.

En el capítulo 11 describe el proceso de instrucción a los usuarios sobre el uso y funcionamiento de la aplicación móvil.

En el capítulo 12 se presentan las conclusiones del proyecto y trabajos futuros.

2 DEFINICIÓN DE LA EMPRESA O INSTITUCIÓN.

2.1 Descripción de la Empresa o Institución.

2.1.1 Antecedentes Generales de la Empresa o Institución.

Razón Social	: Universidad del Bio-Bío.
RUT	: 60.911.006-6
Representante Legal	: Héctor Gaete Feres.
Dirección	: Avenida Collao 1202, Concepción, Región del Bío -Bío, Chile.
Sitio WEB	: www.ubiobio.cl.
Teléfono	: +56 41 3111200.

2.1.2 Misión.

La Universidad del Bío-Bío, a partir de su naturaleza pública, responsable socialmente y estatal, tiene por misión, desde la Región del Biobío, aportar a la sociedad con la formación de personas integrales, a través de una Educación Superior de excelencia. Comprometida con los desafíos de la región y del país, contribuye a la movilidad e integración social por medio de; la generación y transferencia de conocimiento avanzado, mediante la docencia de pregrado y postgrado de calidad, la investigación fundamental, aplicada y de desarrollo, la vinculación bidireccional con el medio, la formación continua y la extensión. Asimismo, impulsa el emprendimiento y la innovación, el fortalecimiento de la internacionalización y el desarrollo sustentable de sus actividades, basada en una cultura participativa centrada en el respeto a las personas (Universidad del Bio-Bío, 2017).

2.1.3 Visión.

Ser reconocida a nivel nacional e internacional como una Universidad pública, responsable socialmente y regional que, comprometida con su rol estatal, desde la Región del Biobío, forma personas integrales de excelencia y aporta a través de su quehacer al desarrollo sustentable de la región y el país (Universidad del Bio-Bío, 2017).

2.1.4 Organigrama general de la empresa.

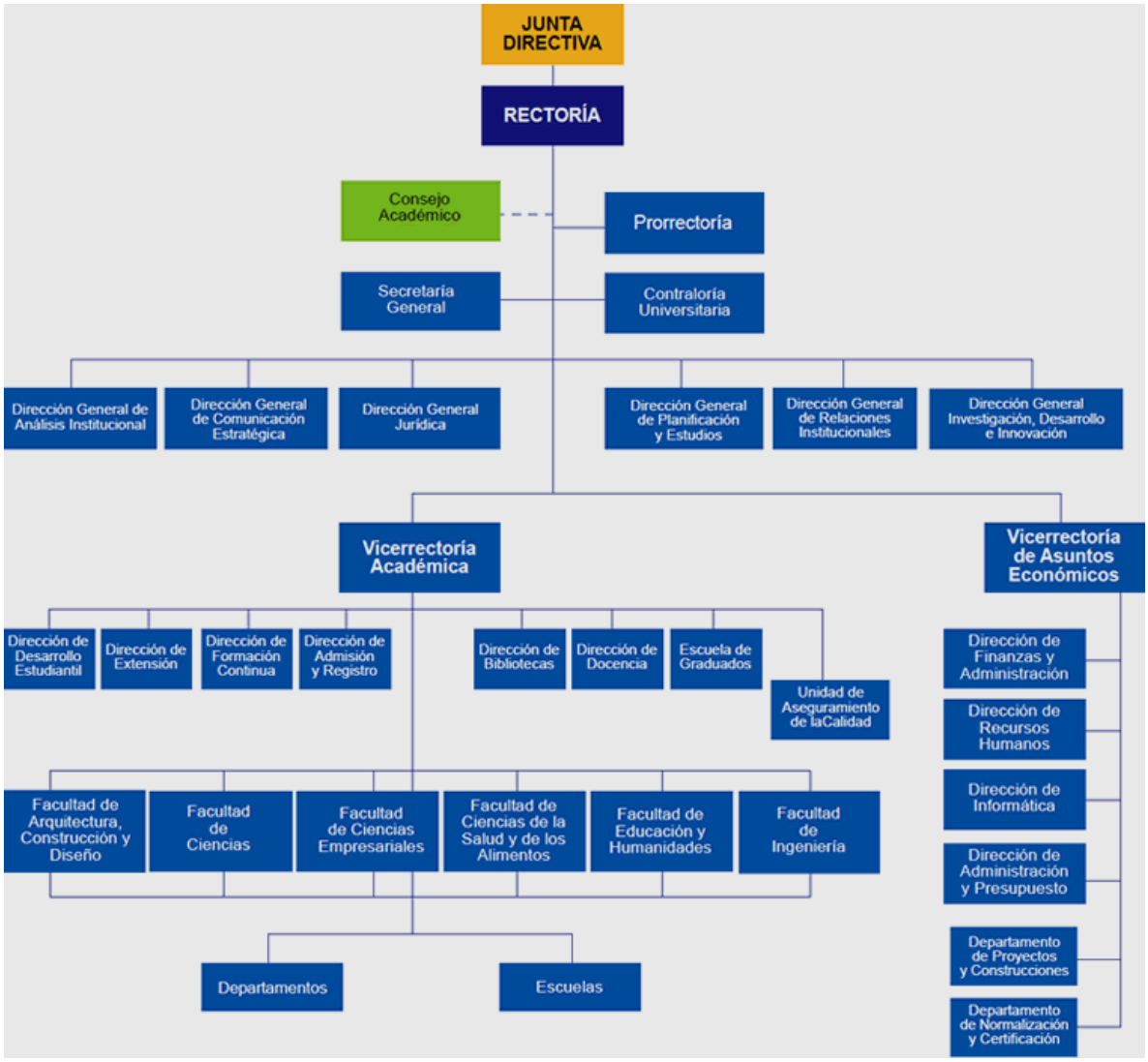


Figura 1. Estructura orgánica de la Universidad del Bio-Bío (Universidad del Bio-Bío., 2017).

2.2 Descripción del área donde se realiza el proyecto.

Laboratorio de Sistemas Automatizados de Producción de la Universidad del Bío -Bío.

2.2.1 Datos del área.

Unidad Responsable	: Facultad de Ingeniería.
Director del laboratorio	: Christian Aguilera Carrasco.
Jefe del laboratorio	: Luís Vera Quiroga.
Contacto	: Luís Vera Quiroga.
Teléfono	: +56 41 2731137
E-mail	: lvera@ubiobio.cl – luisvera.cimubb@gmail.com
Sitio WEB	: www.ubiobio.cl/cimubb.

2.2.2 Organigrama del área.

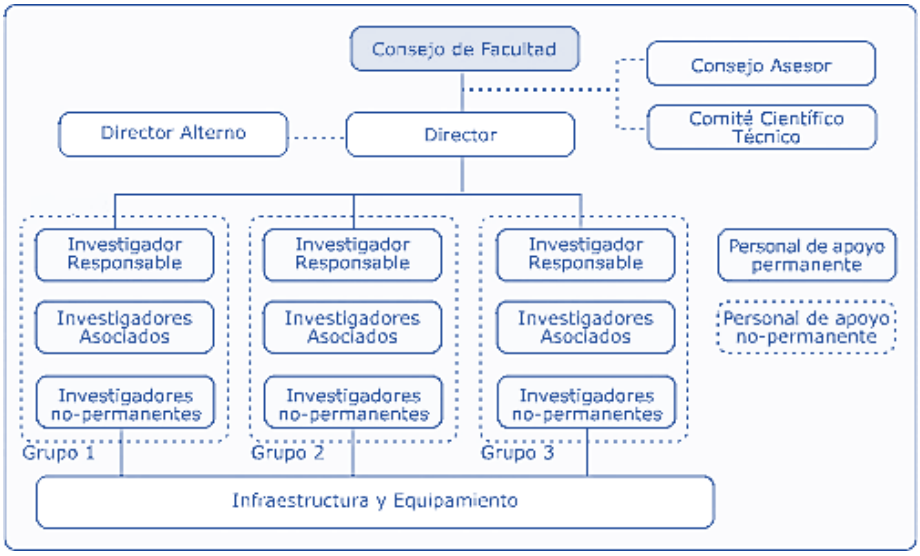


Figura 2. Organigrama del laboratorio CIMUBB, (Laboratorio CIMUBB., 2017).

2.2.3 Descripción del área.

Este laboratorio pertenece a la Facultad de Ingeniería de la Universidad del Bio-Bío se creó en el año 2000 gracias al financiamiento del Ministerio de Educación del Estado de Chile y de la Universidad del Bio-Bío, como Laboratorio de Sistemas Automatizados de Producción CIMUBB.

Actualmente, este laboratorio es utilizado en el ámbito educativo en la impartición de asignaturas electivas, todas ellas enfocadas a la enseñanza de las tecnologías ligadas a la automatización de la producción industrial. El laboratorio reúne a académicos y estudiantes de niveles superiores de diferentes departamentos, realizándose investigaciones de alto nivel.

El laboratorio tiene como misión "conocer, desarrollar, adaptar, aplicar" las técnicas de automatización de manufactura.

El laboratorio se encuentra constituido por sistemas de control de alta calidad, un conjunto de máquinas a escala (robots, CNC, cinta transportadora, etc.), y estaciones de trabajo CAD/CAM/CAE. Todo eso con la finalidad de emular una cadena productiva completa, desde el almacenamiento de materias primas hasta el mecanizado y el control de calidad, todo de manera automatizada y computarizada integrando todo el ciclo del producto bajo el concepto de CIM (CIMUBB, 2016).

Las principales líneas de trabajo del laboratorio son:

- Procesamiento digital de imágenes y señales.
- Automatización de la producción.
- Robótica.
- CAD/CAM/CAE.

La Figura 3 describe la organización del sistema de manufactura asistido por computador del laboratorio CIMUBB.

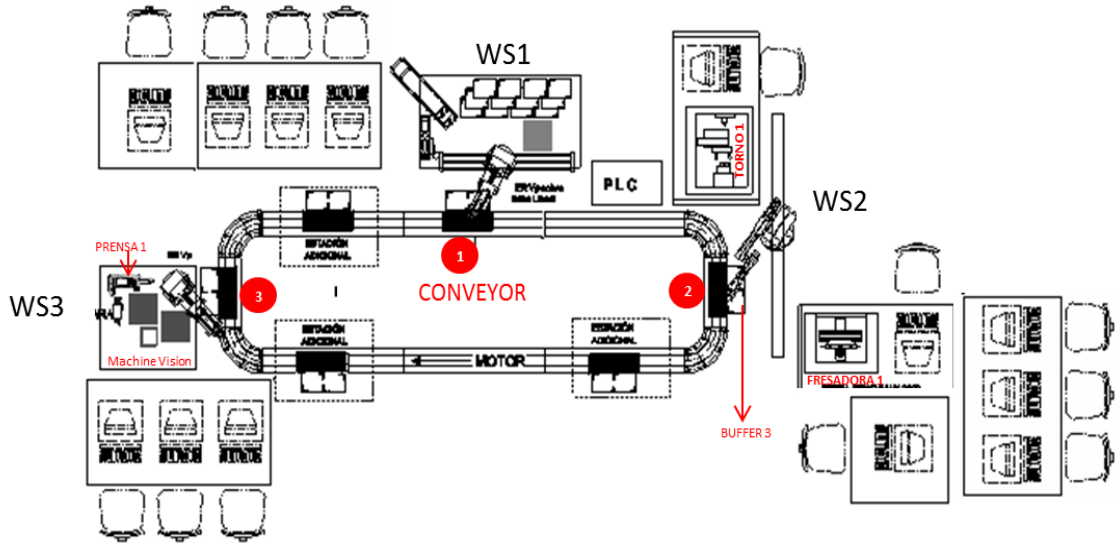


Figura 3. Layout del SMF del laboratorio CIMUBB (Laboratorio de Sistemas Automatizados de Producción CIMUBB, 2016)

Donde:

- WS1 (Estación 1): Corresponde a la estación de Almacenamiento Automatizado, almacén de materia prima, productos terminados y bandejas de transporte. Su función es proveer de materia prima al proceso de manufactura.

- WS2 (Estación 2): Estación que involucra el Proceso de Manufactura Automatizada, Torno y Fresadora de control numérico para procesos de mecanizado de materia prima. Su función es convertir materia prima en productos terminados.

- WS3 (Estación 3): Estación de Control de Calidad Automatizada, dispone de computador, cámara y software para inspección mediante Visión Artificial para el diagnóstico de las partes manufacturadas. Su función es control.

- WS0 (Manager): Estación de Gestión, Control y Planificación de la manufactura, computador y software para realizar control del proceso de manufactura por computador, planificación y coordinación de las tareas de las estaciones de trabajo. Su función es planificación y coordinación.

En este proyecto corresponde trabajar en el área de trabajo de robótica y del control de calidad mediante procesamiento digital de imágenes y señales en la estación de trabajo WS3 como nos indica en la Figura 3. El objetivo del laboratorio es integrar los dispositivos móviles para la ejecución de los procesos que involucran a la robótica y el control de calidad mediante visión.

La estación WS3 actualmente dispone de un brazo robot Scorbot ER-V junto al controlador, un computador, una webcam Intel con resolución de 320x240 pixeles y una prensa de banco como se muestra en la Figura 4.

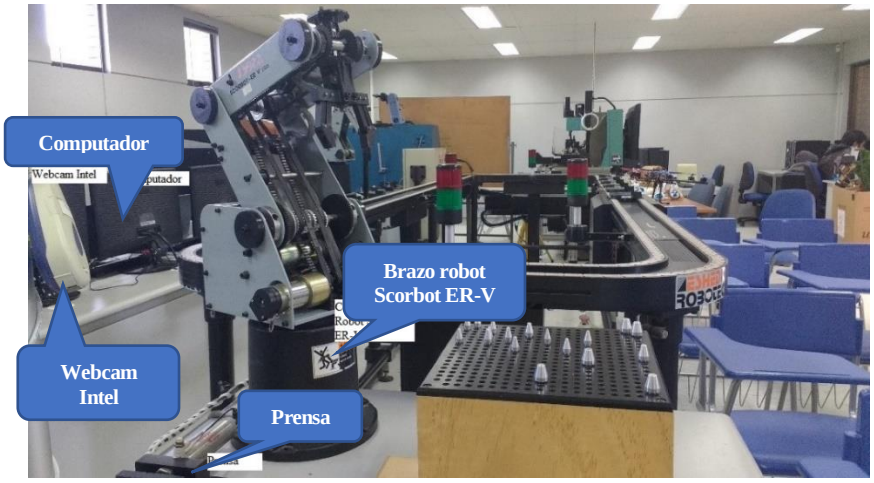


Figura 4. Estación 3 (WS3) del sistema de manufactura flexible.

La persona a cargo de la supervisión del proyecto es Don Luis Vera Quiroga jefe del Laboratorio de Sistemas Automatizados de Producción, Ingeniero de Soporte y Mantenimiento del Laboratorio CIMUBB.

2.3 El Concepto CIM.

El concepto CIM se define según John W. Bernard como "La integración de las computadoras digitales en todos los aspectos del proceso de manufactura". También se menciona que tiene que ver con proporcionar asistencia, automatizar, controlar y elevar el nivel de integración en todos los niveles del proceso de manufactura, con el propósito de minimizar los gastos, añadir valor agregado y generar riqueza en todos los aspectos.

Desde la década de los 1940's gracias al desarrollo de las tecnologías de las computadoras y de los sistemas de control para las maquinarias, la automatización se ha acelerado de manera significativa hasta nuestros días.

Las metas principales de la automatización en instalaciones de manufactura son:

- Integrar diversas operaciones de manera de mejorar la productividad.
- Incrementar la calidad y uniformidad del producto.
- Minimizar los tiempos del ciclo del producto.
- Reducir los costos de mano de obra.

El significativo avance en los procesos industriales ha logrado conectar e integrar los procesos productivos que antes eren ejecutados de manera separada en un solo sistema integrado, permitiendo la comunicación de las diferentes áreas de la empresa y que cada área dependa de otra y así sucesivamente. Cuando esta integración se establece, entonces se ha implementado un sistema CIM, si fuera en el área de procesos de negocios se trataría de un sistema ERP (Enterprise Resources Planning, Planificación de Recursos Empresariales). Que hoy es considerado como el sistema de gestión más eficiente en materia de integración.

En la industria manufacturera, los computadores son utilizados en el control y optimización de los procesos de manufactura, manejo de materiales, ensambles, inspección y pruebas automatizadas de los productos, control de inventarios y numerosas otras actividades administrativas (Sistemas CIM y ERP para la Industria, 2016).

La manufactura integrada por computador ha sido efectiva debido a su capacidad de:

- Otorgar una rápida respuesta a los cambios en la demanda y la consiguiente modificación de los productos, respondiendo a las exigencias del mercado nacional e internacional.

- Optimizar el uso de los recursos tales como: materiales, personal, reducción de costos de inventarios, por ejemplo.
- Mejorar el control de la producción, y la administración de toda la operación de manufactura.
- Asegurar que el sistema de gestión obtenga productos de alta calidad a bajo costo. (Kalpakjian & Schmid, 2002)

Aplicaciones principales de los computadores en la manufactura (necesariamente incluyen CIM):

a) Control numérico computarizado (CNC).
b) Control adaptativo (AC).
c) Robots industriales.
d) Manejo automatizado de los materiales (P&P).
e) Los sistemas de ensamble automatizado y robótico. (ASRS)
f) Planeación de procesos asistidos por computadora (CAPP).
g) Tecnología en grupo (GT).
h) Producción justo a tiempo (JIT).
i) Manufactura celular (FMC).
j) Sistemas de manufactura flexible (FMS).
k) Sistemas expertos.
l) Inteligencia y Visión artificial (IVA).

Bajo el contexto de la Manufactura Integrada por Computadora se presentan cuatro tópicos principales que son:

- **Computadores industriales.** Son máquinas de procesamiento diseñadas principalmente para aplicaciones industriales que son de mayor velocidad de procesamiento, mayor número de puertos de entrada y salida, mayor resistencia en ambientes hostiles en comparación con los computadores personales. Pero la principal desventaja de estos es su costo debido a sus características.

- **Dispositivos de Control de Procesos en Redes (PLC).** Son dispositivos muy utilizados en el control de los procesos, que, aunque individualmente no tienen mucha capacidad de conexión, pueden conectarse en red y de esta manera lograr el control automático de un proceso mayor. En el sistema de manufactura del laboratorio como se muestra en la Figura 3 de la Sección 2.2.3 el dispositivo PLC corresponde a la unidad que controla la cinta transportadora, ubicada en el centro del laboratorio, su función es trasladar y controlar el paso de los materiales a las distintas etapas del proceso de manufactura.

- **Control e Inspección de Calidad.** Es una de las aplicaciones más importantes de los computadores en el área de producción. El control de calidad mediante sistemas de visión artificial permite la inspección total de los productos que pasan por la línea de producción, con una alta velocidad y precisión.

- **Sistemas CAD/CAM.** Los sistemas CAD son aplicaciones computacionales para asistir el desarrollo, modificación, presentación y documentación de los diseños de ingeniería y productos a manufacturar. También utilizan los computadores para planear, programar, controlar y administrar las operaciones de una planta manufacturera. La utilización de los sistemas CAD/CAM permiten la sistematización de gran parte de las operaciones desde la etapa de diseño hasta la elaboración del producto final (Sistemas CIM y ERP para la Industria, 2016).

A raíz de la constante evolución de la tecnología y de su potencial para mejorar los métodos productivos, algunos expertos han llegado a una visión de la fábrica del futuro en el que la intervención directa del humano en la manufactura será casi nula y se espera que quede limitado a la supervisión, mantenimiento y actualización de las máquinas, computadores y software, asumiendo que esta intervención no sea presencial, sino de forma virtual (Martinez, 2016).

3 DEFINICIÓN DEL PROYECTO.

El objetivo de este capítulo es la presentación del proyecto y del tema a tratar en sí. Se pretende definir como se realizó el proyecto dejando bien claro cuáles fueron los objetivos del proyecto como también definir conceptos iniciales correspondientes al desarrollo de éste.

3.1 Origen del tema.

Los procesos automatizados en las industrias son cada vez más importantes, permitiendo realizar procesos reiterativos, en forma organizada y programada con anticipación, aplicándose a diversas tareas repetitivas con el fin de agilizarlas. Una de las técnicas utilizadas para automatizar los procesos en la industria es la visión artificial.

Se puede definir el concepto de Visión Artificial como un campo de la Inteligencia Artificial, que, mediante la utilización de técnicas adecuadas, permite obtención, procesamiento y análisis de la información obtenida a través de imágenes digitales.

La visión artificial aplicada a la industria manufacturera abarca la informática, la óptica, electrónica, mecánica y la automatización industrial. Las aplicaciones de visión artificial aplicada a la producción, integran sistemas de captura de imágenes, dispositivos de entrada/salida y redes de computadores para el control de equipos destinados a la fabricación, por ejemplo, brazos robóticos.

En un sistema de visión artificial suelen haber tres componentes principales que son: Sensores (cámara), procesadores (aplicación/programa) y actuadores (máquinas, robot, etc.).

Gracias a la visión artificial se puede realizar lo siguiente:

- Automatizar procesos repetitivos realizadas por personas.
- Realizar controles de calidad de productos que no era posible verificar por métodos tradicionales.
- Realizar inspecciones de objetos sin contacto físico.
- Realizar la inspección del 100% de la producción (calidad total) a gran velocidad.
- Reducir el tiempo de ciclo en procesos automatizados.
- Realizar inspecciones en procesos donde existe diversidad de piezas con cambios frecuentes de producción.

3.2 Objetivos del Proyecto.

3.2.1 Objetivo General.

Desarrollar una aplicación de visión artificial para la estación de inspección y control de calidad de un sistema de manufactura flexible mediante la comparación de figuras o patrones de los productos elaborados en madera y el control de la celda flexible de inspección, integrando los dispositivos móviles a los sistemas de producción y de control de calidad, con el objetivo de modernizar la plataforma tecnológica utilizada el Laboratorio de Sistemas Automatizados de Producción (CIMUBB) de la Universidad del Bio-Bío.

3.2.2 Objetivos Específicos.

- Estudiar la situación actual de la organización (Laboratorio CIMUBB) en relación a su estructura organizacional y en el ámbito del desarrollo de las aplicaciones de visión.
- Determinar los requerimientos de la aplicación, del usuario y del sistema.
- Explicar las características y el funcionamiento del robot Scorbot ER-V, como también la conexión inalámbrica (Bluetooth) entre el robot y un dispositivo móvil.

- Investigar, probar y seleccionar algoritmos que permitan dirigir el brazo robot de forma remota.
- Investigar el estado del arte de las aplicaciones que implementen visión artificial y/o por computador en el ámbito del procesamiento y comparación de imágenes.
- Investigar, probar e implementar algoritmos utilizando la librería OpenCV para la comparación de objetos y patrones con el fin de controlar un brazo robot Scorbot ER-V para seleccionar un producto según algún criterio de calidad de éste de manera automatizada.
- Desarrollar la aplicación para dispositivos móviles con SO Android e integrarlo al sistema CIM del laboratorio CIMUBB de la Universidad.
- Realizar pruebas de la aplicación y sus resultados.

3.3 Descripción del problema.

3.3.1 Situación actual.

En el laboratorio CIMUBB actualmente está implementando un sistema para controlar celdas flexibles (con robots Scorbot) en cada estación de trabajo, que interactúan mediante comunicación serial (ver sección 7).

También existen una serie de programas y comandos en lenguaje de programación ACL que permite controlar los movimientos del robot, a ciertas posiciones dentro del plano cartesiano ya establecidos por los usuarios del laboratorio.

En el ámbito de la visión artificial y del procesamiento digital de imágenes, en el laboratorio CIMUBB se imparten asignaturas a los alumnos de las carreras de Ingeniería en Automatización, en Electrónica e Ingeniería en Maderas, correspondiente a este ámbito, con el fin de desarrollar aplicaciones de escritorio para el reconocimientos de objetos y el control de calidad, desarrollados en el lenguaje de programación C#, en el IDE Visual Studio, por lo tanto en el laboratorio ya existen aplicaciones de visión artificial que permiten definir la calidad y similitud entre productos fabricados en el laboratorio.

Todos estos procesos se llevan a cabo en la estación 3 del sistema de manufactura CIM (ver Figura 3. Layout del SMF del laboratorio CIMUBB), correspondiente al área de inspección y control de calidad, de manera integrada en un computador de escritorio, el que ejecuta aplicaciones para la coordinación de tareas y control de calidad, está conectado a la celda flexible mediante un puerto serial, también incluye una cámara para la visión artificial.

Actualmente surge la necesidad de implementar las funcionalidades de control de calidad en una aplicación para dispositivos móviles (Android) con el objetivo de que en un corto plazo ser integrada a un Sistema de Ejecución de Manufactura Flexible implementado en Android, desarrollado por Jorge Sánchez y Ricardo Pavez, ex alumnos de Ingeniería Civil Informática de la Universidad del Bio-Bío, y ser controlado por un Coordinador de manera sincronizada para ser una de las cuatro estaciones del actual sistema SMF o sistema CIM.

3.3.2 Problemas y desafíos.

Capturar un objeto que se encuentra dentro de un campo de visión delimitado por la cámara y moverlo hacia un lugar distinto de donde fue capturado mediante un brazo robot implica distintos desafíos. Uno de estos desafíos es la obtención de la ubicación espacial dentro del campo de visión, además debe ser capaz de interpretar de una manera correcta la información obtenida por la cámara, para así obtener la ubicación de un objeto de interés, además, de evaluar el objeto de acuerdo a la similitud con otro objeto de referencia. Por otra parte, debe poder interactuar con el robot para que se mueva y de una u otra forma darle un cierto poder de razonamiento para definir como dirigirse hacia el objetivo.

Para cumplir esta funcionalidad se propone que el control del robot se realice a distancia por medio de comunicación inalámbrica Bluetooth, y utilizar la cámara de un Smartphone.

Desde el punto de vista de software, otro desafío es implementar una interfaz de comunicación entre el teléfono móvil y el puerto serial del robot, también desarrollar un "chat" de comunicación con otro dispositivo, para recibir órdenes y capturar una fotografía, que luego será procesada y evaluada en el propio Smartphone, y de acuerdo al resultado obtenido ejecutar un determinado movimiento del robot.

En proyectos anteriores se ha conseguido desarrollar el control y coordinación, del sistema del laboratorio CIM, en otros lenguajes de programación como C++ y C#. El principal desafío enfrentado es desarrollar la aplicación para controlar una celda flexible de inspección y control de calidad, que utilice comunicación Bluetooth desde un Sistema Operativo Android, con la finalidad de ejecutar la totalidad de los procesos de forma remota.

El problema va a ser segmentado en diferentes partes para poder atacar los distintos puntos del problema.

3.4 Ambiente de Ingeniería de Software.

3.4.1 Modelo de proceso a implementar.

El modelo de proceso a utilizar en el desarrollo de la aplicación corresponde al modelo de desarrollo evolutivo exploratorio y el modelo iterativo incremental.

Los criterios de evaluación utilizados para su elección fueron los siguientes:

- En la primera entrevista con el jefe del laboratorio se obtuvo una breve explicación del tema del proyecto a desarrollar y de los requisitos generales de este. Además, se dio a conocer la descripción del laboratorio y de los procesos del sistema de manufactura asistida por computador (CIM).

- Se definió la forma de comunicación y la plataforma tecnológica, que utilizo una aplicación para comunicarse entre dos dispositivos Android, y entre un dispositivo y un brazo robot a través de Bluetooth.

- El desarrollo evolutivo exploratorio se enfocó a explorar con el usuario los requisitos hasta llegar a un sistema final, comenzando con la especificación de los requerimientos, así logrando un mejor entendimiento del sistema.

- El cambio en los requerimientos del sistema, en lo que corresponde al proceso de visión y control del robot, llevó a utilizar el desarrollo evolutivo y el modelo iterativo incremental con el fin de mostrar al usuario una visión y una funcionalidad del proyecto. Además, el usuario puede dar una opinión del sistema en desarrollo, permitiendo que se realicen cambios en los requisitos, y en cada entrega corregir los posibles errores de la aplicación en desarrollo.

3.4.2 Técnicas y notaciones.

3.4.2.1 Técnicas.

- Programación Orientada a Objetos (POO).
  Es un paradigma de programación que basa la estructura de los programas en torno a los objetos abstractos o de la vida real.
  Los lenguajes de programación orientada a objetos ofrecen medios y herramientas para describir los objetos manipulados por un programa. Más que describir un objeto individualmente, estos lenguajes proveen una construcción (Clase) que describen un conjunto de objetos que cumplen con las mismas propiedades.

- Modelo Vista Controlador (MVC).
  Es un patrón de desarrollo que divide una aplicación en tres niveles distintos, uno que representa la interfaz gráfica (Vista), otro que representa el tratamiento de datos e información (Modelo) y otro que se encarga de toda la lógica y control de los otros niveles que se tiene que llevar a cabo por la aplicación (Controlador). Esto se hace para permitir una mayor portabilidad de una aplicación, e incluso facilitar su mantenimiento.

Cabe destacar que la aplicación se desarrolló en lenguaje de programación Java, debido a que fue elegido por el entorno de desarrollo oficial de Android (Android Studio) para el desarrollo de aplicaciones móviles, por lo que la aplicación es de carácter nativa.

Java es un lenguaje de programación orientado a objetos permitiendo emular el sistema físico de forma virtual a través de este lenguaje.
La librería de visión por computador utilizada en el desarrollo de la aplicación está orientada a objetos, así como la estructura de la aplicación (APP) por lo que este paradigma de programación fue utilizado para el desarrollo de la aplicación.

Como la mayoría de las aplicaciones desarrolladas en el entorno de desarrollo de Android se hizo uso del patrón Modelo Vista Controlador, dividiendo la aplicación en tres niveles distintos, uno que representa a la interfaz gráfica (Vista), otro que corresponde al tratamiento de datos y almacenamiento de imágenes (Modelo), y el otro que se encargara de ejecutar toda la lógica de la aplicación (Controlador).

3.4.2.2 Notaciones y definiciones.

- IEEE : Instituto de Ingeniería Eléctrica y Electrónica.
- IDE : Entorno de desarrollo integrado.
- MER : Diagrama modelo entidad-relación, es una herramienta para el modelado de datos que permite representar las entidades relevantes de un sistema de información, así como sus interrelaciones y propiedades.
- MR : El modelo relacional utilizado para la gestión de una base de datos, es un modelo de datos basado en la lógica de predicados y en la teoría de conjuntos.
- UML : Lenguaje Unificado de Modelado. Es un lenguaje grafico para visualizar, especificar, construir y documentar un sistema.
- BPMN : Notación de modelado de procesos de negocio
- DFD : Diagrama de flujo de datos.
- CU : Casos de Uso
- LA : Layout.
- PR : Prueba.
- RF : Requerimiento Funcional.
- IS : Interfaz o dato de salida.
- DE : Interfaz o dato de entrada.
- IU : Interfaz de usuario.
- GUI : Interfaz gráfica de usuario.

3.4.3 Estándares de documentación.

Para mantener una lectura y comprensión técnica del documento que respalda el desarrollo de este sistema, se usó un estándar de acuerdo a la necesidad del proyecto. Es por esto que se utilizó una adaptación basada en IEEE Std 830-1998 (Institute for Electrical and Electronics Engineers.), la cual proveerá de las herramientas para la comprensión y correcta interpretación del desarrollo. Cada punto de este documento será revisado según este estándar y se deberá considerar en todo momento de explicación y redacción.

3.4.4 Software de apoyo utilizado en el desarrollo de la aplicación.

- Android Studio, versión 2.1: Android Studio es el Entorno Integrado de Desarrollo (IDE) oficial, para desarrollar aplicaciones para el sistema operativo Android de forma gratuita y que está basado en IntelliJ IDEA, un IDE que también nos ofrece un buen entorno de desarrollo Android (Android Developers, 2016).

  Android Studio ofrece las siguientes características y funcionalidades:
  - Soporte para construcción basada en Gradle.
  - Consola de desarrollador: consejos de optimización, ayuda para la traducción, estadísticas de uso.
  - Renderización en tiempo real.
  - Plantillas para crear diseños comunes de Android y otros componentes.
  - Soporte para programar aplicaciones para dispositivos como teléfonos móviles, Android Wear, y Android TV.

- Genymotion 2.9.0: Emulador de Android de licencia gratuita para uso personal que puede ser agregado como plugin a Android Studio para la ejecución de las aplicaciones desarrolladas en este IDE (Genymobile, 2017).

- OpenCV 3.2.0 para Android: (OpenCV Team, 2017) OpenCV es una abreviación de la librería Open Source Computer Vision. Es la librería más utilizada de visión por computadora y es desarrollada originalmente por Intel. OpenCV es multiplataforma, existiendo versiones para Linux, Mac OSX, Windows y Android. Ha sido escrito de forma nativa en el lenguaje de programación C/C++, pero posee wrappers (adaptaciones) para Python, Java, y cualquier lenguaje JVM, que están diseñadas para transformar el código en bytecodes de Java, como Scala y Clojure. A raíz de que la mayoría de Android sus aplicaciones se desarrollan en C/C++/Java, OpenCV se ha portado como un SDK que los desarrolladores pueden utilizar para implementar las aplicaciones de visión por computador en la plataforma de Android.

  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 estéreo y visión robótica. Ejemplos de sistemas desarrollados con OpenCV son los sistemas de seguridad con detección de movimiento, sistemas de reconocimiento facial, control de calidad, y sistemas de control de procesos donde se requiere el reconocimiento de objetos. La publicación de la librería OpenCV se da bajo licencia BSD, que permite ser usada libremente para propósitos educativos, comerciales y de investigación.

  Se eligió esta librería de visión artificial porque es de licencia libre, existe variada documentación sobre el uso de esta librería, y además permite reutilizar las funciones desarrolladas e implementadas en este módulo.

- yEd: Version 3.16: Herramienta CASE de licencia gratuita utilizada para el modelamiento de datos, y procesos.
- Astah Professional: Herramienta CASE de licencia gratuita para uso académico, utilizada para el modelado de datos y procesos.

3.5 Conceptos, siglas y abreviaciones.

- Automatización:
  Es la tecnología que sustituye al operador humano referida al uso de sistemas mecánicos, hidráulicos, eléctricos, electromagnéticos y computarizados, para operar y controlar un proceso o tarea afín.

  "Automatización es la tecnología referida al uso de sistemas mecánicos, eléctricos y computarizados para operar y controlar la producción" (Groover, 2010).

Existen tres tipos de automatización:

- Automatización Rígida, se caracteriza por realizar procesos de producción de gran volumen conllevando una gran velocidad en el desarrollo de la producción, siguiendo siempre un esquema predefinido, por lo cual resulta más sencillo de implementar. Debido a esta rigidez se ve dificultado el querer cambiar el esquema predefinido, por ejemplo, querer añadir una nueva máquina a la producción de un producto X.
  Ejemplos: Líneas de mecanizado, máquinas de ensamblaje automático.

- Automatización Programable, esto viene siendo el aspecto contrario al anteriormente mencionado, siendo los sistemas que se encuentran dotados de equipos con una gran posibilidad de ser configurados, siendo capaces de tomar decisiones basados en ciertas pautas o valores introducidos por un operador.
  Ejemplos: Robots industriales, Control numérico, PLCs, relés programables.

- Automatización Flexible, es una extensión de la programación programada diferenciándose en su capacidad para cambiar parte de los programas o disposiciones físicas, sin perder tiempo de producción (Automatización, 2017).

El tipo de automatización utilizada principalmente en el laboratorio es la Automatización Flexible y en menor medida la Automatización Programable ya que todos los componentes y estaciones del sistema están programadas por el operador en el momento de construir un producto. Los programas definidos para cada estación del sistema de manufactura son programados previamente por un operador antes de construir un producto de acuerdo a las decisiones tomadas por el operador. El cambio de los parámetros del proceso de manufactura no involucra una pérdida de tiempo de producción siempre y cuando las configuraciones y parámetros estén almacenados en la memoria de los computadores que componen el sistema.

- Visión Artificial.

Es el área de la inteligencia artificial que intenta imitar el proceso de visión humano.
Es el conjunto de técnicas que permiten la obtención de información por métodos ópticos, tanto de manera automática como asistida.
Este conjunto heterogéneo de técnicas incluye iluminación, captación de imagen, digitalización de la misma, pre procesamiento o acondicionamiento, procesamiento y comunicación de los resultados.
En algunos procesamientos se emplean algoritmos asimilables a Inteligencia Artificial (redes neuronales, sistemas basados en reglas, clasificadores, etc.), pero son minoritarios frente a procesamientos específicos que van desde la medida de distancia en pixeles o el contar de los mismos hasta tratamientos en el dominio de la frecuencia.

- Visión por computador.

Es un campo de la Inteligencia Artificial enfocado a que los computadores puedan extraer información a partir de una imagen para ofrecer soluciones a problemas del mundo real (Mery, 2004) .El objetivo es utilizar los computadores para inferir el ambiente que se está examinando.

La visión por computador se puede definir desde varios puntos:

- Procesamiento digital de imágenes: Proceso mediante el cual se toma una imagen y se produce una versión modificada de esa imagen (véase en capitulo procesamiento de imagen).

- Análisis de imágenes: Proceso mediante el cual a partir de una imagen se obtiene una medición, interpretación o decisión.

  Ejemplos de análisis de imágenes son los sistemas de inspección visual, de control de calidad, detección y seguimiento de objetos, etc.

El análisis consiste en cinco etapas:

- Adquisición de la imagen: se obtiene la imagen adecuada del objeto de estudio. Dependiendo de la aplicación la imagen puede ser una fotografía, radiografía, termografía, etc.
- Pre procesamiento: Con el fin de mejorar la calidad de la imagen obtenida se emplean ciertos filtros digitales que eliminan el ruido de la imagen o bien aumentan el contraste.
- Segmentación: Se identifica el objeto en objeto de interés en la imagen obtenida.
- Medición (extracción de características): Se realiza una medición de ciertos atributos de interés del objeto en estudio.
- Interpretación (Clasificación): De acuerdo a los valores obtenidos en las mediciones se lleva a cabo una interpretación del objeto.

o Reconocimiento de patrones: Asignación de objetos a diferentes clases a partir de mediciones de objetos, está estrechamente relacionado con el análisis de imágenes.

Es utilizado en la clasificación de objetos o patrones que cumplen con ciertas características similares, por ejemplo, en el caso de sistemas de detección de rostros en algunas cámaras digitales, los rostros cumplen con ciertas características o patrones similares.

o Computación gráfica: Generación computacional de imágenes a partir de modelos.

En la computación gráfica se generan imágenes artificiales a partir de modelos físicos o geométricos. Está enfocado más al lado del diseño gráfico y de realidad virtual.

- Scorbot ER-V Plus:

El Scorbot ER-V Plus es un robot vertical articulado con cinco juntas rotativas, con el agregado de la mordaza por lo que el robot posee seis grados de libertad.
Es un robot fabricado por la empresa israelí Eshed Robotec (1982) Ltda y está diseñado para fines educativos (ver sección 7).

- Celda flexible de inspección:

Se refiere a la estación de trabajo WS3, correspondiente la estación de inspección y control de calidad del sistema de manufactura CIMUBB, esta se compone de un brazo flexible robotizado Scorbot, el controlador del robot, una cámara web y un computador. Esta celda se encuentra ubicada frente a la cinta transportadora del sistema CIM, en el cual transporta los productos mediante pallets, que es controlado mediante una aplicación “Manager” del sistema.

4 ESPECIFICACIÓN DE REQUERIMIENTOS DE SOFTWARE.

4.1 Alcances y limitaciones del producto.

4.1.1 Alcances.

- La aplicación permitirá que desde un teléfono (A) se puedan ejecutar programas y movimientos del brazo robot.
- Se permitirá configurar y guardar la configuración definida por el usuario que se utilizará para segmentar la imagen obtenida por la cámara del teléfono (B), para permitir una correcta comparación y reconocimiento de patrones.
- El campo de visión del teléfono (B) se encontrará justo por debajo del gripper del brazo robot Scorbot ER-V.
- La visión seleccionará automáticamente una región de interés en donde se encuentra el objeto para medir su porcentaje de igualdad en relación al objeto de referencia, como también identificar a que patrón guardado en la memoria del dispositivo se asimila más el patrón obtenido.
- El robot se moverá a determinadas posiciones bajo tres criterios que serán: posición de captura de la imagen (por ejemplo, sobre la cinta transportadora), piezas aceptadas y piezas descartadas, además el robot tomará estas piezas después de obtenido el resultado de la comparación y las transportará a la posición correspondiente.
- Permitirá realizar un control de calidad a los productos elaborados en madera en el Sistema de Manufactura Asistido por Computador (CIM), integrando esta aplicación al administrador (Manager) del sistema desarrollado en Android.

4.1.2 Limitaciones.

- La calibración de la cámara no es exacta, debido a los movimientos que puedan ser provocados por agentes externos o por el propio robot.
- La detección del objeto puede estar afectado por la iluminación o sombra del espacio de captura, afectando algunos colores del objeto que podría no detectar correctamente.
- Debido al tamaño de la pantalla del teléfono, el procesamiento de imágenes, la segmentación y la configuración, solo se permitirá realizar desde otro teléfono por medio de comunicación WI-FI.
- La aplicación solo hará comparación de imágenes o de patrones que se ubican dentro de una región cuadrilateral específica y que estará definida por un rango de colores definidos mediante la configuración establecida por el usuario, y se seleccionará automáticamente la región de mayor área obtenida.
- Los movimientos del brazo robot estarán limitados y predefinidos por el usuario de la aplicación, por lo que la serie de movimientos del robot serán finitos, es decir solo ejecutará programas que estén guardados en la memoria del robot y que estén registrados en la base de datos de la aplicación.

4.2 Objetivos del software.

4.2.1 General.

Establecer un sistema de inspección y control de calidad de los productos elaborados en madera en el sistema de manufactura flexible SMF, de acuerdo a sus respectivas figuras y patrones a través de dispositivos móviles, cuyo resultado será el porcentaje de similitud entre dos figuras capturadas y su identificación.

4.2.2 Específicos.

- La aplicación permitirá desde un teléfono (A) enviar instrucciones de captura y segmentación de imágenes a través de comunicación inalámbrica hacia un dispositivo B, mostrando el resultado en pantalla de manera de establecer una región de captura a través del rango de colores y de binarización.

- La aplicación permitirá controlar un brazo robot Scorbot, enviando instrucciones desde un teléfono (A) a un teléfono (B) y éste último enviar instrucciones de programa a través de Bluetooth.

- La aplicación permitirá el registro de los parámetros de control de calidad que permitirá automáticamente establecer la región de captura de la imagen y su respectiva figura, el porcentaje de similitud entre una figura guardada y otra a comparar, y los movimientos del brazo robot Scorbot para capturar, aceptar y descartar los productos que son fabricados experimentalmente en el sistema CIM, al ser solicitados por un administrador (Manager) de producción del CIM.

- La aplicación entregará el resultado de la comparación de figuras a través de un porcentaje de similitud e identificación de la figura, y controlará automáticamente el robot Scorbot según la calidad de ésta, para luego aceptar o descartar un producto.

- Deberá estar habilitada para ser integrada en un futuro a otra aplicación de gestión (Manager) de producción que se desarrollará para dispositivos móviles en Android.

4.3 Descripción global del producto.

4.3.1 Interfaz de usuario.

La vista de la aplicación es comprensible y de fácil manejo para el usuario, la interfaz de navegación está diseñada de acuerdo a las buenas prácticas de desarrollo en Android.

Las funcionalidades de la aplicación pueden ser ejecutadas por medio de la interfaz de usuario, y por mensajes o comandos a través de una entrada de texto desde un dispositivo (A) al (B).

La aplicación se ejecutará en dos dispositivos móviles (A) y (B), donde el usuario podrá manipular y controlar la cámara del dispositivo (B) desde el dispositivo (A), como también configurar los parámetros para el control de calidad y el control del robot, de manera de que el usuario tenga una mínima interacción con el dispositivo (B) que se encontrará ubicado sobre el cabezal del Scorbot.

4.3.2 Interfaz de Hardware.

En términos de hardware el sistema utiliza como mínimo dos Smartphones uno como administrador o controlador y el otro como receptor y quien será encargado de ejecutar los procesamientos de imágenes y comunicarse directamente con el Scorbot.

El sistema interactúa con un circuito conversor Bluetooth-Serial construido por personal del laboratorio, este dispositivo está controlado por un microcontrolador Arduino Mega 2560, que permite intercambiar mensajes entre un dispositivo Android el Scorbot a través de comunicación Bluetooth y serial RS232.

Scorbot ER-V, brazo robot vertical articulado que será utilizado para ejecutar las instrucciones emitidas por el dispositivo móvil, de acuerdo al resultado obtenido en la etapa de la comparación de patrones de imagen y control de calidad.

4.3.3 Interfaz de Software.

El sistema utilizará:

- Android 4.4.2 o superior, Sistema operativo para dispositivos móviles de código abierto donde operará la aplicación a desarrollar en este proyecto.

- OpenCV Manager versión 3.0, Aplicación Android de licencia gratuita disponible en Google Play, contiene las librerías de OpenCV para las versiones 2.4.x, y 3.0.x. OpenCV Manager se utilizará para la ejecución de las aplicaciones desarrolladas en Android en que incluyan procesamiento de imágenes utilizando la librería OpenCV.

- SQLite, Gestor de bases de datos de licencia gratuita usada en Android, la aplicación utiliza este gestor para almacenar los parámetros definidos por el usuario para el control de calidad.

La aplicación interactúa con si misma en uno de sus dos modos en cada Smartphone, uno en Modo Administrador(A) y el otro en Modo de Cámara o servidor(B).

4.3.4 Interfaz de comunicación.

Para la comunicación entre dos dispositivos Android, la aplicación utiliza el protocolo de comunicación TCP de arquitectura cliente-servidor utilizando sockets de Java, la comunicación será por vía red WIFI. El medio de comunicación entre un dispositivo móvil y el controlador de la celda flexible Scorbot será a través de comunicación Bluetooth-Serial mediante un conversor Bluetooth-Serial compuesta por un microcontrolador Arduino, un módulo bluetooth HC-05 y un adaptador serial rs232 (sección 7.2.1).

La aplicación posee dos subprogramas, uno de configuración (dispositivo A) y el otro de ejecución de la cámara y control del robot (dispositivo B), por lo tanto, en el dispositivo (A) se ejecutará como un cliente y en el dispositivo (B) como un servidor.
El dispositivo (B) se comunicará con el controlador del brazo robot a través de comunicación Bluetooth y comunicación serial.

El Controlador forma parte de la celda flexible Scorbot y es el aparato encargado del control de los movimientos del brazo robot, del almacenamiento y ejecución de los programas y vectores de posición, además de recibir la información de los codificadores de las juntas rotativas robot.

La interfaz de comunicación entre todos los dispositivos y periféricos se encuentra representada en la Figura 5

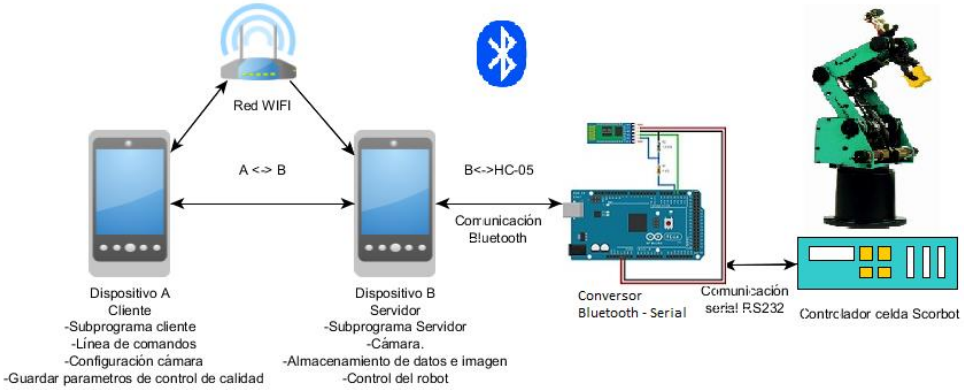


Figura 5.Interfaz de comunicación del sistema.

4.4 Requerimientos específicos.

4.4.1 Requerimientos funcionales de la aplicación.

Id	Descripción
RF1: Ingreso y control de la aplicación.	
RF1.01	La aplicación deberá tener dos modos de ingreso a éste. estos serán administrador-cliente y servidor-cámara, que se ejecutarán de manera independiente.
RF1.02	El usuario deberá permitir ingresar como cliente en la aplicación del dispositivo (A) y controlar la cámara del dispositivo (B), ingresando un nombre de usuario, la dirección IP del dispositivo y el puerto habilitado del dispositivo (B).
RF1.03	La aplicación deberá permitir controlar el brazo robot Scorbot por comunicación mediante Bluetooth, por lo que la aplicación deberá permitir intercambiar mensajes entre el brazo-robot implementando una interfaz de comunicación Bluetooth-Serial con el Scorbot.
RF1.04	La aplicación deberá controlar el robot de acuerdo al resultado de la comparación de dos imágenes mediante parámetros de control del robot y de procesamiento de imagen establecidos. Referencias: RF2.08, RF3.11
RF2: Procesamiento de imagen.	
RF2.05	Ser capaz de obtener objetos y patrones de imagen mediante la manipulación parcial de tratamientos de la imagen.
RF2.06	Mostrar en pantalla la imagen resultante de cada etapa realizada en el procesamiento de imagen, controlada por otro dispositivo. Referencia: RF1.02
RF2.07	Guardar imágenes como figura o modelo de referencia en la memoria del dispositivo, de acuerdo la región de interés (ROI) seleccionada mediante el tratamiento de la imagen, además registrar los datos del patrón(figura) encontrado dentro de la región de interés, para su posterior análisis. Referencia: RF2.05
RF2.08	Calcular porcentaje de coincidencia entre una imagen guardada en memoria del dispositivo y la imagen capturada por el usuario.
RF2.09	Guardar los parámetros utilizados en el procesamiento de imagen y comparación, como rango de color, umbral y el modo de captura de la región de interés (ROI).
RF3: Inspección de objetos	
RF3.10	Establecer y guardar los programas que serán utilizados por el brazo-robot para ejecutar el control de calidad. Se deberá registrar un programa para las siguientes funciones: - capturar imagen producto, - guardar producto, y – descartar producto.
RF3.11	Mostrar en pantalla del dispositivo (A), modo cliente o mánager, el resultado del porcentaje de coincidencia y el nombre de la figura identificada obtenida del proceso de control de calidad mediante visión por computador. Referencias: RF2.08

Tabla 1. Requerimientos funcionales

4.4.2 Interfaces externas de entrada.

Id.	Nombre ítem.	Detalle de datos contenidos en ítem.
DE01	Datos de conexión al socket servidor	IP, Puerto, usuario
DE02	Modo selección de región de interés ROI y modo vista previa de procesamiento	Configuración de segmentación, vista (modo de visión imagen)
DE03	Datos modo ROI, (1) dimensionar imagen mediante puntos en coordenadas.	Coordenadas $p(x, y)$ de los cuatro puntos vértices del cuadrilátero del ROI.
DE04	Datos modo ROI (2),	H_MINIMO, Nivel de matiz mínimo H_MAXIMO, Nivel de matiz máximo S_MINIMO, Nivel de saturación mínimo

	Establecer región rectangular sobre una máscara por rango valores del modelo de representación de color	S_MAXIMO, Nivel de saturación máximo V_MINIMO, Nivel de brillo mínimo V_MAXIMO, Nivel de brillo máximo, representado en el modelo HSV de representación del color.
DE05	Umbral binario detección figura dentro del ROI en escala de grises.	THRESH_MINIMO, THRESH_MAXIMO. Umbral mínimo y máximo de una imagen en escala de grises para segmentar y destacar e identificar un patrón o figura de interés.
DE06	Datos de imagen de referencia	Nombre de la imagen o patrón de referencia y la imagen registrada en la memoria del dispositivo.
DE07	Datos de registro de configuración ROI.	Nombre de la configuración se segmentación del ROI más los datos contenidos en: DE03, DE04 y DE05.
DE08	Datos de la secuencia de programas del robot utilizados para la etapa del control de calidad.	Programas guardados en la memoria del robot que serán ejecutados para mover la celda a la posición de captura, de aprobación y de descarte de un determinado objeto inspeccionado, más el porcentaje mínimo de coincidencias entre el patrón del objeto a evaluar para su aceptación.

Tabla 2. Interfaces externas de entrada.

4.4.3 Interfaces externas de salida.

Identificador.	Nombre ítem	Datos contenidos en el ítem	Medio de salida
IS01	Resultado comparación figuras (patrón)	Porcentaje de coincidencia de pixeles, Distancia euclidiana e identificación del patrón más aproximado mediante algún algoritmo de reconocimiento	Mensaje en pantalla del dispositivo.
IS02	Movimiento del brazo-robot	Nombre del programa del brazo robot, que el robot puede ejecutar como resultado de los datos de IS01 u orden definida para la posición de captura del brazo robot.	Mensaje desde el dispositivo Android hacia el Controlador Scorbot ER-V y la ejecución del programa recibido por el robot
IS03	Respuesta de termino de ejecución del programa del robot	Mensaje de respuesta del robot que envía el robot hacia el dispositivo cuando ejecución del programa solicitado ha terminado, por ejemplo:" OK"	Mensaje desde el controlador del robot hacia el dispositivo Android. Movimiento del robot.

Tabla 3. Interfaces externas de salida.

4.4.4 Atributos del producto.

- FUNCIONALIDAD
  El sistema debe cumplir con todos los requerimientos funcionales obtenidos a partir del usuario, basados en la especificación de requerimientos realizada anteriormente en la sección 4.4.1 más atrás

- SEGURIDAD
  Debido a que la aplicación será utilizada de manera local dentro del laboratorio CIM y con una finalidad educativa, la aplicación no poseerá algún método de control para entrar a la aplicación y/o acceder a los datos de ésta.

- EFICIENCIA - TIEMPO DE EJECUCIÓN / RESPUESTA
  Debido a que la cantidad de recursos de un dispositivo móvil Android son mucho menores que de un computador, el sistema hará uso de los algoritmos de la librería OpenCV más eficientes en el procesamiento de imágenes. La aplicación debería procesar imagen obtenida en el dispositivo (cámara) de manera inmediata desde la orden de comparación hasta la entrega del resultado en un tiempo no superior a tres segundos.

- USABILIDAD - OPERATIVIDAD
  El sistema contará con una interfaz amigable de acuerdo a las buenas prácticas de desarrollo en Android. La interfaz gráfica es de fácil comprensión para los usuarios y su funcionalidad está adaptada de acuerdo a las operaciones realizadas actualmente en el laboratorio CIMUBB en el área de inspección de calidad.

- PORTABILIDAD
  El sistema se podrá instalar en cualquier dispositivo móvil con S.O Android desde la versión 4.4.2. Debido a que el gestor de base de datos SQLite se ejecuta como parte de la aplicación no necesita de un servidor externo para guardar los parámetros utilizados para el proceso de inspección y control de calidad. Todo esto proporcionará libertad para manipular el sistema en diversos ambientes.

- MANTENIBILIDAD
  La aplicación será de fácil mantenimiento debido al uso de librería de procesamiento de imagen OpenCV para Android reduciendo la cantidad de líneas de código utilizado en el procesamiento facilitando la edición y actualización del código. La aplicación se encuentra modularizada en actividades, y su base de datos funcionará como parte de la aplicación haciendo la aplicación escalable en el tiempo.

5 ESTUDIO DE FACTIBILIDAD DEL PROYECTO.

5.1 Factibilidad técnica.

Software requerido para el desarrollo del proyecto.

Software	Tipo/descripción	Licencia	Observación
Microsoft® Windows® 7/8/10 (32 o 64 bits)	Sistema operativo.	Microsoft OEM.	Adquirido por el desarrollador
Android Studio 2.1.1	Entorno de desarrollo IDE de Android.	Freeware.	
Genymotion 2.9.0	Emulador de Android para Windows.	Genymobile, Freeware para uso personal.	
Sublime Text 3.	Editor de código.	Freeware.	
ATS Terminal	Interface para el ACL. Emulador de terminal para el acceso al ACL del Scorbot desde una PC.	Eshed Robotec	Adquirido por el laboratorio
yEd Graph Editor.	Modelado.	Freeware.	
OpenCV para Android 3.2.0.	Librería de procesamiento de imágenes de Intel.	Intel BSD Freeware	
Astah Professional.	Modelado	Freeware para fines académicos.	Adquirido por el desarrollador

Tabla 4. Recursos de software necesarios para desarrollo del proyecto.

Hardware mínimo requerido necesario para el desarrollo de la aplicación.

En base a los recursos de hardware mínimo necesario para poder desarrollar el software en forma óptima.

- Memoria RAM DDR de 3 GB para PC.
- Procesador Intel CORE I3 1.6 GHz.
- 4 GB de espacio mínimo disponible en el disco duro.
- Monitor de 14” con resolución mínima de 1280 x 800 pixeles.
- Mouse.
- Teclado.

El equipo del laboratorio cuenta con los recursos de hardware necesario para el desarrollo del proyecto.

Software mínimo necesario para el correcto uso y funcionamiento del sistema.

Software	Tipo/descripción	Licencia
Android versión 4.4.2 o posterior	Sistema Operativo para dispositivos móviles.	Gratuita
OpenCV Manager 3.0.0	Aplicación Android.	Gratuita

Tabla 5. Software mínimo necesario para el funcionamiento del sistema.

Hardware mínimo requerido para el correcto funcionamiento del sistema.

- 2 Smartphone con S.O Android 4.4.2 o superior, con resolución de pantalla 320 x 240 pixeles o superior, con conectividad WIFI y Bluetooth.
- 1 microcontrolador Arduino Uno.
- 1 módulo Bluetooth bidireccional HC-05.
- 1 celda Robot flexible Scorbot ER-V (Eshed Robotec).

El laboratorio actualmente cuenta con el robot Scorbot y su controlador que forma parte del sistema de manufactura CIM.

Conocimientos requeridos por el equipo desarrollador.

- Programación en lenguaje Java.
- Programación en Android Studio.
- Lenguaje ACL (Lenguaje de programación del Scorbot).
- Procesamiento de imágenes con OpenCV en C++, Python y Java.

El equipo desarrollador posee los conocimientos de Java y de programación en Android, además el equipo se encontrará en constante capacitación para adquirir los conocimientos de programación del robot y el procesamiento de imágenes.

5.2 Factibilidad operativa.

La implementación y cambio de plataforma del sistema a Android provocará los siguientes impactos:

Efectos negativos:

- La organización tendrá que invertir dinero en la adquisición de dispositivos requeridos para el funcionamiento del sistema.
- A pesar de que la aplicación es intuitiva y de fácil manejo para los usuarios, se requerirá una sesión de capacitación para el correcto uso del software.
- Se requerirá integrar la aplicación con las demás estaciones del sistema CIM.

Efectos positivos:

- Actualización de la plataforma tecnológica desde sistemas de escritorio a dispositivos móviles convirtiendo el sistema de gestión de producción y el control de calidad por visión del sistema CIM en un sistema portable y escalable.
- Facilitará el estudio y comprensión del proceso de comparación de patrones de imagen por los usuarios del sistema.
- El personal del laboratorio está familiarizado con los avances tecnológicos.
- Al equipo del laboratorio y a los usuarios del sistema les será fácil aprender a utilizar e integrar la aplicación ya que ellos poseen conocimientos sobre el dominio del sistema y están familiarizados con su terminología.
- Ahorro y optimización del espacio de trabajo dentro del laboratorio.
- El sistema podrá ser ejecutado adecuadamente ya que en el recinto donde opera el sistema posee una red WI-FI, en que se siempre se encuentra activa.

5.3 Factibilidad económica.

Recursos tecnológicos.

Gastos en recursos tecnológicos de hardware para la implementación del sistema.

Aparato	Cantidad	Costo unitario	Costo total	Observación
Smartphone	2	$39.990	$79.980	Costo asumido por el Laboratorio CIM.
Microcontrolador Arduino Mega 2560 R3	1	$49.990	$49.990	Costo asumido por el Laboratorio CIM.
Módulo Bluetooth HC-05	1	$12.990	$12.990	Costo asumido por el Laboratorio CIM.
Armado circuito electrónico conversor Bluetooth-Serial para el HC-05, Arduino y RS 232	2	$25.000	$50.000	Costo asumido por el Laboratorio CIM.
Notebook, con procesador Intel Core I3 1.6GHz o superior y memoria RAM 3GB o superior.	1	$349.000	$349.000	Adquirido por el desarrollador.
Celda flexible Scorbot ER-V	1	Adquirido por el laboratorio	$2.000.000	Valor del robot amortizado, adquirido por el Laboratorio CIM.

Tabla 6. Recursos de hardware mínimos necesario para el desarrollo y prueba del sistema.

Gastos en recursos tecnológicos de software para la implementación del sistema.

Software	Licencia	Costo
Microsoft® Windows® 7/8/10 (32 o 64 bits)	Microsoft OEM.	Adquirido por el desarrollador
Android 4.4 KitKat o superior	Google.	Freeware.
Android Studio 2.1.1	Google.	Freeware
Genymotion 2.9.0	Genymobile.	Freeware, para uso personal
HyperTerminal.	Freeware.	Adquirido por el laboratorio
ACL Device Driver, ATS Terminal.	Eshed Robotec.	Adquirido por el laboratorio.
OpenCV para Android 3.2.0.	Intel. BSD.	Freeware

Tabla 7. Recursos de software mínimos necesario para el desarrollo y prueba del sistema.

Otros gastos.

Descripción	Costo unitario	Cantidad	Costo total	Observación
Recursos humanos	Valor hora			
Desarrollador Android	$ 4.450	1.080	$ 4.806.000	El proyecto se desarrolló en un total de 1080 horas. El costo no será traspasado al Laboratorio CIM y será asumido por el desarrollador. Valor estimativo.
Gastos imprevistos	N/A	N/A	$ 150.000	Estos gastos serán asumidos por el Laboratorio CIM

Tabla 8. Otros costos del proyecto.

Resumen costo total del proyecto.

Descripción	Costo
Recursos de hardware.	$ 2.496.960
Recursos de software.	$ 0
Otros gastos.	$ 4.956.000
Total.	$ 7.452.960

Tabla 9. Costo total del proyecto.

Costos asumidos en el proyecto.

Costo asumido por el usuario del proyecto (Laboratorio CIM)	$ 2.646.960
Costo asumido por el desarrollador del proyecto	$ 4.806.000

Conclusión de la factibilidad económica.

El costo de la implementación tecnológica del proyecto será asumido en su totalidad por el laboratorio, El costo en mano de obra será asumido por el equipo desarrollador y no será traspasado al Laboratorio CIM.
La aplicación desarrollada en este proyecto será utilizada para fines educativos y experimentales, el sistema solo será de prototipo para el control de calidad a través de visión de patrones en una imagen, por lo tanto, este proyecto no retornará beneficios económicos tangibles.
De acuerdo a los gastos de implementación estos serán asumidos una vez salvo de que se requiera una actualización en las funcionalidades del sistema.

5.4 Conclusión del estudio de factibilidad.

Con respecto al estudio de factibilidad operativa se concluyó que la organización tendrá los siguientes beneficios:

Beneficios tangibles:

- Reducción en tiempo y espacio físico en el proceso de inspección, y control en el proceso de manufactura del CIM.
- Reducción en los tiempos de organización en los grupos de trabajo que trabajarán en el sistema CIM en las etapas del proceso de manufactura y control.
- La aplicación se puede ejecutar en cualquier dispositivo con Android desde la versión 4.4.2 por lo que el reemplazo de los dispositivos será de bajo costo.

Beneficios intangibles:

- Mejorar y optimizar el uso de los recursos tecnológicos y móviles actualmente disponibles.
- Facilitar el trabajo de los usuarios en los procesos de manufactura CIM.
- Facilitar la configuración y la ejecución del proceso de visión por computador y el control del robot de manera inalámbrica.
- Sentido de innovación en la incorporación de tecnologías móviles y portátiles a los procesos de manufactura y control por visión ejecutados en el sistema de manufactura del laboratorio CIMUBB.

Se presenta una gran oportunidad para la organización, considerando las múltiples ventajas que aportan una aplicación móvil en un sistema de manufactura, el laboratorio podrá optimizar una gran parte de sus procesos, aprovechando los beneficios que se pueden extraer de la tecnología en la actualidad principalmente de los dispositivos móviles y de conexión inalámbrica.

Con respecto a la factibilidad técnica, se determinó que el equipo desarrollador dispone de los recursos y conocimientos mínimos necesarios para el desarrollo de la aplicación. Además, el equipo humano del laboratorio posee los equipos y conocimientos necesarios sobre el proceso de visión por computador para la implementación y uso del sistema.

Con respecto al estudio de factibilidad económica del proyecto, se determinó que el proyecto es tanto factible económicamente para el usuario como para el equipo de desarrollo ya que el costo de su implementación es mínimo, al ser un trabajo de título para el Laboratorio CIM de la universidad el costo de desarrollo será asumido por el desarrollador.

Con estos tres estudios de factibilidad se concluye que el proyecto es factible de implementar.

6 PROCESAMIENTO DIGITAL DE IMÁGENES.

6.1 Introducción.

La visión ha sido el sentido más avanzado que los humanos poseen, siendo las imágenes las que juegan un papel importante en la percepción que se tiene de nuestro entorno.
La visión es limitada ya que solo es posible ver la banda visible del espectro electromagnético (luz visible para el ojo humano) a diferencia de las máquinas que pueden percibir el espectro completo desde los rayos gamma hasta las ondas de radio, es por esto la importancia que hay al tratar imagen por medio de las computadoras, cuya exactitud es superior a nuestros alcances.

A diferencia de los estudios de la visión humana, el análisis y procesamiento de imágenes digitales nace en el momento en el que se disponen recursos tecnológicos para captar y manipular grandes cantidades de información espacial en forma de matrices numéricas. Esta distinción sitúa al procesamiento y análisis de imágenes digitales como una tecnología asociada a las Ciencias de la Computación y por tanto cabe pensar de ella como una proyección del término Visión Artificial dentro del ámbito de la Inteligencia Artificial.

En este capítulo se dará a conocer los procesos que se aplicarán para una imagen para obtener información de ella, que será útil para obtener posteriormente las coordenadas de los objetos en pixeles dentro de la imagen. Para desarrollar este proceso implicará tener un entendimiento en el tema del procesamiento digital de imágenes, lo que será abordado posteriormente. Además, se indicarán los procedimientos y algoritmos utilizados, el uso de las librerías y el lenguaje de programación que se será utilizado para realizar este módulo.

6.2 Conceptos relacionados.

6.2.1 Imagen digital.

Una imagen digital es una imagen captada del mundo real por algún dispositivo electrónico y que es convertida en un fichero de bits para ser interpretado por una computadora.

Consiste en una colección ordenada de valores que son representados en una matriz de dos dimensiones, por lo que una imagen es matemáticamente una función bidimensional $F(x, y)$, donde x e y son coordenadas espaciales en un plano, y F que en cualquier par de coordenadas es la intensidad o nivel de gris en esa coordenada, al momento en que x e y, los valores de F son todos valores finitos y discretos, se puede decir que la imagen es una imagen digital (Gonzalez & Woods, 2002).

Cada imagen se compone de una cantidad finita de elementos, cada uno con un lugar y un color específico representado en un valor, cada uno de estos elementos se llama pixel.

Un pixel correspondería a un elemento de la matriz que representa una imagen, esto quiere decir que, si una imagen posee una resolución 300 x 200, cantidad de pixeles contada en una fila es de 300 y por columna es de 200 por lo que la resolución total de la imagen es de 60.000 pixeles. Mientras mayor sea la resolución, la imagen será de mayor tamaño y de mayor nitidez.

Obtención de una imagen digital.

En el caso de las cámaras digitales una imagen se obtiene mediante el uso de matriz de sensores de imagen. Muchos dispositivos sensores electromagnéticos como las cámaras y algunos de ultrasonido con frecuencia son dispuestos en un formato de matriz.

Actualmente existen dos sensores de imagen cámaras digitales e industriales: El sensor CCD (Dispositivo de carga acoplada), y el CMOS (Semiconductor complementario de óxido metálico) en el cual se encargan de convertir la energía de la luz en (fotones) en señales eléctricas (electrones).

Actualmente el sensor CMOS está reemplazando rápidamente al CCD debido a su velocidad, su resolución en número de pixeles, consumo de energía eléctrica, y la calidad de imagen con respecto al CCD.

Un sensor de imagen sólo es capaz de entregar valores en escala de gris. Para obtener la información del color, como se muestran en la Figura 6 y en la Figura 7, en cada píxel del sensor está cubierto por un filtro de color permitiendo el pixel entregar el valor de gris por el color de su filtro de color (colores primarios). Para obtener información a todo color, se utilizan filtros de color de diferentes colores (por ejemplo, rojos, verdes, azul). Se les asigna a los píxeles del sensor de un patrón regular (por ejemplo, formando el patrón de Bayer) que píxeles vecinos están cubiertos por los filtros de color de diferentes colores primarios. Por ejemplo, los vecinos más cercanos de un píxel "verde" son píxeles "rojos" y "azules". Los valores de gris de los colores primarios pueden ser interpolados para cada píxel de los valores de gris de los píxeles vecinos. Como alternativa al uso de filtros de color de diferentes colores en un solo sensor, se pueden utilizar sensores diferentes donde cada uno está cubierto por un filtro de color de un solo color (Basler AG, 2017).

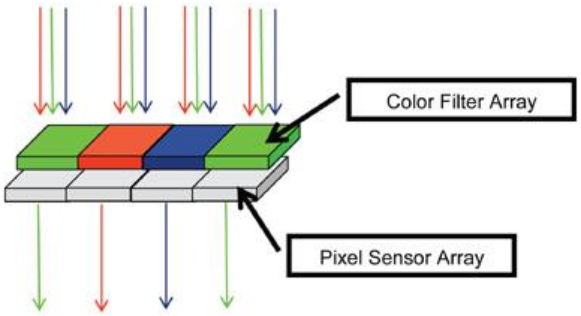


Figura 6. Matriz de filtros permite pasar los colores R, G, y B hacia la matriz de sensor de pixeles. (Vide Imaging Design Ware, 2016)

Figura 7. Canalización de proceso para un típico Sensor de imagen CMOS, desde la detección de imagen hasta la proyección en video y su salida en pantalla. (Vide Imaging Design Ware, 2016)

6.2.2 Procesamiento digital de imágenes.

Es la técnica que se encarga del procesado de las imágenes por medio de una computadora digital. La mayor parte de las técnicas empleadas en el procesamiento digital de imágenes actúan tratando la imagen como una señal de dos dimensiones y después aplicando técnicas de procesamiento de señales de 1 dimensión.

Dentro de las operaciones más comunes de procesamiento están las transformaciones geométricas, correcciones de color, alineación, segmentación, interpolación y reconocimiento de patrones y características en una imagen.

Gracias al desarrollo de la tecnología se ha logrado que las aplicaciones de procesamiento digital de imágenes se extiendan a la industria manufacturera automatizada, a la medicina, a la industria del

Retail, al transporte y control del tránsito, como también a las aplicaciones de seguridad, de logística y agricultura. (Basler AG, 2017).

Los procesos que involucran el tratamiento digital de imagen y la visión por computador se pueden clasificar de acuerdo a la complejidad y minuciosidad que lleva asociada su implementación en tres niveles de procesamiento:

- Procesos de bajo nivel: Comprende las etapas de captación y pre procesamiento, algunas de las principales funciones a realizar son el Filtrado (reducción de ruido en la imagen), Restauración (perfeccionar una imagen degradada por diversas causas), Realce (aumento del contraste de los niveles de gris), Extracción de contornos (reduce los datos que contiene la imagen conservando la información de mayor interés), etc.

  Son caracterizados porque sus entradas son imágenes y sus salidas también son imágenes con algún resultado obtenido en el procesamiento.

- Procesos de nivel medio: Comprende las etapas de Segmentación, Descripción y Reconocimiento. Estos procesos extraen, caracterizan y etiquetan componentes de la imagen extraída en el proceso de bajo nivel. Una de las funciones es el análisis de escenas, reconocimiento de formas, bordes y objetos individuales. Se caracterizan porque sus entradas son imágenes y sus salidas son atributos extraídos de las imágenes.

- Procesos de alto nivel: Comprende la etapa de interpretación, Se refiere a los procesamientos que tratan de emular la cognición humana (a lo que se refiere a la inteligencia artificial) asociadas a la vista. Una de las funciones que se realizan es la de reconocimiento de objetos o clasificación, por comparación de las características extraídas de la imagen con las de los modelos previamente aprendidos en forma artificial.

6.3 Partes fundamentales en el procesamiento digital de imágenes.

En este apartado se muestran los pasos o partes fundamentales del procesamiento digital de imágenes, aunque después de adquirir la imagen, en un procesamiento no necesariamente se siguen todos los pasos. La secuencia de pasos será de acuerdo a los objetivos y resultados deseados a obtener a través del procesamiento de una imagen digital. (Gonzalez & Woods, 2002).

1. **Adquisición de la imagen.** Es el proceso de obtener la imagen. Puede incluir decisiones como qué sistema de iluminación es más conveniente, el mejor rango de longitud de onda para la aplicación, etc. Generalmente incluye pequeños pasos de pre procesamiento, como el corregir características no lineales, distorsiones geométricas, escalar.

2. **Mejora de la imagen.** La idea de este paso es obtener detalles de la imagen que no sean percatados perfectamente a simple vista, o simplemente recalcar ciertas características que son de nuestro interés. Logrando de esta forma que la imagen procesada se vea mejor.

3. **Restauración de la imagen.** Produce una mejora de la apariencia de la imagen, a diferencia de la mejora de la imagen, subjetiva, la restauración es objetiva, en el sentido en que las técnicas de restauración tienden a ser modelos probabilísticos o matemáticos de degradación de la imagen, esto quiere decir, que una imagen que no se viera nítida producto de ruido externo, aplicando la restauración a la imagen obtenemos dicha imagen como es originalmente sin el ruido causado por algún agente externo.

4. **Procesamiento del color.** Procesamientos especiales para el color para un modelo de color. Los modelos de color más utilizados son el modelo RGB, CMYK y el HSL/HSV y escala de grises:

- Modelo HSV:
  Es un modelo similar al modelo HSL a diferencia que este representa los colores a través de su matiz o tonalidad, saturación y valor o brillo (Hue, Saturation, Value).
  Es una transformación no lineal del modelo RGB en coordenadas cilíndricas de manera que cada color viene definido por las siguientes dimensiones:

  - Tinte o matiz: Ángulo que representa el matiz, normalmente definido entre 0 y 360 grados.
  - Saturación: Nivel saturación del color, dado entre 0 y 1, 0 representa sin saturación alguna (blanco), hasta 1 que sería el matiz en toda su intensidad. Es común también darlo en percentiles 0%-100%.
  - Valor o Brillo: Nivel del brillo entre 0 y 1. 0 es negro; 1, blanco. Al igual que la saturación puede darse en porcientos entre 0% y 100%. De esta forma el 50% indica el nivel medio o normal del brillo del color.

  A pesar que el modelo HSL y HSV son transformaciones en coordenadas cilíndricas del modelo RGB, la diferencia que en el modelo de color HSV está representado por un cono simple.

- Modelo Escala de grises: El modelo de escala de grises utiliza distintos tonos de gris en una imagen. En una imagen de 8 bits puede haber 256 tonos de gris según el valor de intensidad del color blanco o del brillo. Cada pixel en una imagen tiene un valor de brillo entre 0 (negro) y 255 (blanco).
  La conversión de una imagen a color RGB de 32 bits a escala de grises se basa en el cálculo del valor de intensidad del color azul, verde y rojo en cada pixel.

- Modelo Monocromático (Mapa de bits): Este modelo está compuesto por dos valores de color (blanco y negro) para representar los pixeles de una imagen. Las imágenes en modo Mapa de Bits o binarias se denominan imágenes de 1 bit porque solo tienen profundidad de 1 bits y solo pueden representar valores de 0 y 1 para cada pixel. Para convertir una imagen al modelo de color binario, se utiliza técnicas de umbralización.

5. **Ondeletas (Wavelets).** Son la base para la representación de imágenes en diversos grados de resolución. En particular se utiliza para la comprensión de datos de imagen y para la representación piramidal, en que imágenes se subdividen sucesivamente en regiones más pequeñas.

6. **Compresión.** Reduce el almacenamiento requerido para guardar una imagen, o el ancho de banda para transmitirla, un ejemplo de esto son las imágenes que se guardan en formato (.jpg), este es una extensión de imagen que ocupa un algoritmo de compresión para poder dejar la imagen en un peso menor al de la extensión original comparándola por ejemplo con una extensión (.bmp) que utiliza más memoria y conserva muchos detalles de la imagen.

7. **Procesamiento morfológico.** Corresponde a la extracción de componentes de la imagen útiles en la representación y descripción de formas.

8. **Segmentación.** Divide una imagen en objetos o partes constituyentes, la segmentación autónoma es una de las tareas más difíciles en el procesamiento digital de imágenes.

9. **Representación y descripción.** Usualmente sigue después del proceso de segmentación como una salida, una vez obtenido el resultado del proceso anterior se toman decisiones tales como si la forma obtenida debe ser tratada como una frontera o una región, además de extraerle atributos que resultan información de interés.

10. **Reconocimiento de objetos.** Es el proceso que asigna una etiqueta o clasificación (por ejemplo "vehículo") a un objeto captado en la imagen basada en sus descriptores. Se desarrollan métodos para el reconocimiento de objetos individuales.

Los pasos incluidos en el desarrollo del módulo de procesamiento de imágenes de la aplicación desarrollada para este proyecto fueron:

- Adquisición.
- Mejora de la imagen (Reducir el ruido en la imagen).
- Procesamiento del color (Cambio de espacio de colores y binarización).
- Compresión (Reducir el tamaño y cambio de tipo de archivo de imagen).
- Procesamiento morfológico (Erosión, Dilatación de una imagen).
- Segmentación (Obtener un área o elemento de interés).
- Representación y descripción (Obtener formas, descriptores, etc.).
- Reconocimientos de objetos (en menor medida identificar formas y figuras).

La Figura 8 presenta un esquema de un ejemplo de procesamiento digital de imagen para la evaluación de un objeto de acuerdo a su intensidad de cada componente del color en el espacio RGB y la medición en área y perímetro del objeto obteniendo estos factores descriptores para su interpretación.

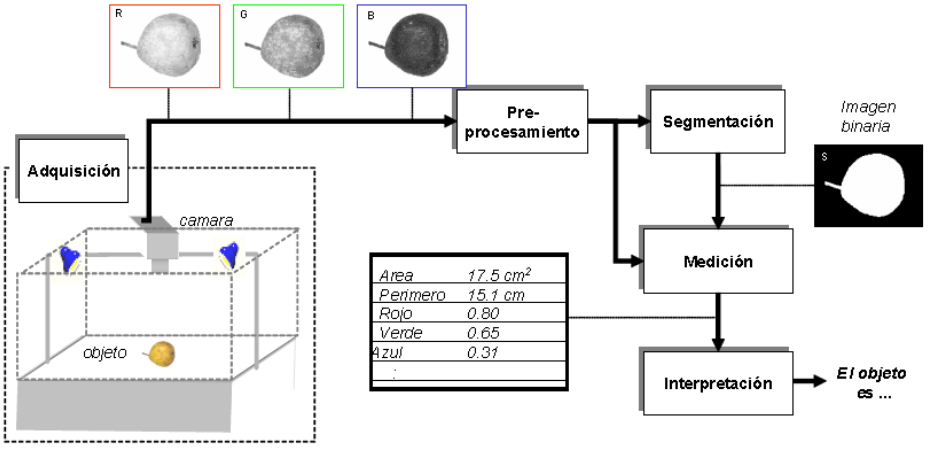


Figura 8. Esquema de un ejemplo de procesamiento de imagen. (Mery, 2004).

6.4 OpenCV, librería de Visión Artificial para Android.

6.4.1 Descripción y estructura de OpenCV versión 3.2.

OpenCV para Android está programada en Java, pero también posee funciones escritas en código nativo C/C++ con el fin de optimizar el uso de los recursos y ser más rápida la ejecución de la aplicación que utilice la librería.

Las funciones nativas en su mayoría corresponden a las utilizadas para detectar las características de una imagen, el reconocimiento de objetos, y funciones matemáticas. Para ejecutar el código nativo en una actividad de Android escrita en Java se necesita el uso de una interfaz JNI (Java Native Interface) quien se encarga de ejecutar el código nativo de C/C++ en la máquina virtual de Java.

OpenCV almacena una imagen digital como un objeto personalizado denominado **Mat**. Este objeto guarda la información como (filas, columnas, datos), y así sucesivamente almacenar los datos y recrear la imagen cuando sea necesario para su posterior procesamiento.

Una imagen de color contiene más datos que una imagen en blanco y negro en lo que es una misma imagen, esto es porque una imagen de color en el modelo RGB se utilizan 3 canales y una imagen en escala de grises solo 1 canal, en la Figura 9 se muestra el almacenamiento de una imagen en forma de matriz de una imagen en escala de gris, y en la Figura 10 una imagen en colores de tres canales BRG o RGB.

	Columna 0	Columna 1	Columna ...	Columna m
Fila 0	0,0	0,1	0,...	0,m
Fila 1	1,0	1,1	1,...	1,m
Fila ...	...,0	...,1	...,...	...,m
Fila n	n,0	n,1	n,...	n,m

Figura 9. Almacenamiento de imagen (objeto Mat) en escala de grises, 1 canal.

	Columna 0			Columna 1			Columna ...			Columna m		
Fila 0	0,0	0,0	0,0	0,1	0,1	0,1	0,...	0,...	0,...	0,m	0,m	0,m
Fila 1	1,0	1,0	1,0	1,1	1,1	1,1	1,...	1,...	1,...	1,m	1,m	1,m
Fila ...	...,0	...,0	...,0	...,1	...,1	...,1	...,...	...,...	...,...	...,m	...,m	...,m
Fila n	n,0	n,0	n,0	n,1	n,1	n,1	n,...	n,...	n,...	n,m	n,m	n,m

Figura 10. Almacenamiento de imagen (objeto Mat) en el modelo de color RGB o BGR, tres canales.

En el modelo RGB o bien BGR, cada canal está representado por un nivel de intensidad de algún color rojo, verde o azul, que va desde 0 a 255, siendo 0 la nula presencia del color y 255 la saturación máxima del color. Como resultado de la combinación de los tres canales da como resultado un color específico dentro del espectro visible.
OpenCV tiene una estructura modular, lo que significa que el paquete de OpenCV incluye varias bibliotecas estáticas y compartidas. Estos son los siguientes módulos:

- Core: Módulo compacto que define estructuras básicas, incluyendo el denso array multi-dimensional Mat y funciones básicas utilizadas por el resto de los módulos.
- Imgproc: Módulo de procesamiento de imágenes que incluye filtros lineares y no lineares de imágenes, transformaciones geométricas (redimensionamiento, deformación afín y de perspectiva, mapeo genérico basado en tablas) conversión de espacios de colores, histogramas y más. Este módulo es el más utilizado para la etapa del procesamiento de imágenes de este proyecto.
- Video: Un módulo de video análisis que incluye estimación de movimiento, sustracción de fondo, y algoritmos de rastreo de objetos.
- Calib3d: Algoritmos de geometría multivísta básica, calibración simple y estéreo de cámara, estimación de posición de objetos, algoritmos de correspondencia estéreo, y reconstrucción de elementos en 3D.
- Features2d: Detectores de características resultantes, descriptores y emparejadores de descriptores.
- Objdetect: Detección de objetos e instancias de las clases predefinidas (por ejemplo, caras, ojos, tazas, gente, coches y más).
- Highgui: Interfaz de fácil uso para la captura de vídeo, imágenes y video codecs, a al igual que simples capacidades de UI.
- Gpu: Algoritmos de distintos módulos de OpenCV acelerados mediante GPU. Esta aplicación se basa principalmente en el uso de las funciones de segmentación y matching de la biblioteca, para ello recurrimos a funciones de varios de los módulos principales: Core, Imgproc, Highgui y Feature2D.
- Mat: Contenedor básico de imágenes, corresponde al objeto más importante dentro de la librería OpenCV para Java y Android. Está representada por una matriz multidimensional según la variante del objeto (Figura 10, y 11).

6.4.2 Beneficios de OpenCV.

OpenCV es una librería multiplataforma con una gran cantidad de documentación y soporte. OpenCV es de código abierto (Open Source) por lo que es posible utilizarla libremente sin pagar una licencia para poder utilizarlo.

Como OpenCV es la librería de procesamiento de imágenes y visión artificial más utilizada, cuenta con innumerables problemas que ya se han solucionado, facilitando y complementando así el trabajo para este

proyecto. Transformándose de esta manera en una ventaja competitiva dentro de las herramientas principales para el desarrollo óptimo de una solución al problema propuesto.

6.5 Pipeline utilizado en este proyecto en el procesamiento de imagen y obtención de resultados con OpenCV.

En esta sección se señalarán las acciones del pipeline utilizado para el correcto tratamiento de la imagen, obtenida por medio del punto de visión del sistema, que corresponderá a una cámara de un dispositivo móvil Android, la que está fijamente montada en un soporte sobre el cabezal del robot en un punto determinado.

La imagen obtenida se podrá analizar para así obtener los objetos que la componen y seleccionar posteriormente el que sea de nuestro interés, con el que se trabajará más adelante y se analizará para obtener los resultados necesarios para cumplir con los objetivos de este proyecto.

A continuación, serán explicado dos métodos se segmentación de imagen utilizados para definir la región de interés en donde se encuentra un objeto y sus respectivos algoritmos (secciones 6.5.1 y 6.5.2) además el método de corrección de perspectiva de la región de interés (Sección 6.6) y finalmente se describirán los métodos utilizados para la etapa de comparación (Sección 6.7.1) y reconocimiento de figuras en imágenes binarias (Sección 6.7.2).

6.5.1 Dimensionar región de interés mediante selección por coordenadas (Cropping).

El objeto contenedor de las imágenes en OpenCV (Mat), está representado como una matriz bidimensional, y cada pixel está representado por un elemento dentro de la matriz. Para cada pixel se almacena su posición y los datos contenidos en ese pixel. El tamaño de la matriz será igual al del tamaño de la imagen.

En los dispositivos Android, la librería OpenCV determina automáticamente el tamaño de la imagen capturada por la cámara del dispositivo según el tamaño y densidad de la pantalla del dispositivo, por lo que si el dispositivo posee una pantalla con resolución de 1280 x 720 pixeles entonces el tamaño de la imagen capturada y será de 960 x 540 y si el dispositivo tiene una resolución de pantalla de 480 x 320 el tamaño de la imagen será de 320 x 240 y así sucesivamente.
Con el fin de establecer la región de interés en imágenes de cualquier tamaño se definirá el espacio escalado de la imagen estableciendo una escala de 0 a 1000.

Los parámetros utilizados para la definición de la región de interés serán las coordenadas $p(x, y)$ en cuatro puntos P1, P2, P3, y P4
En el siguiente ejemplo de demuestra una imagen de tamaño 960 x 540, con su región de interés definida. Los parámetros definidos en esta región son:

$P_1(x_1, y_1)$, $P_2(x_2, y_2)$, $P_3(x_3, y_3)$,y $P_4(x_4, y_4)$.

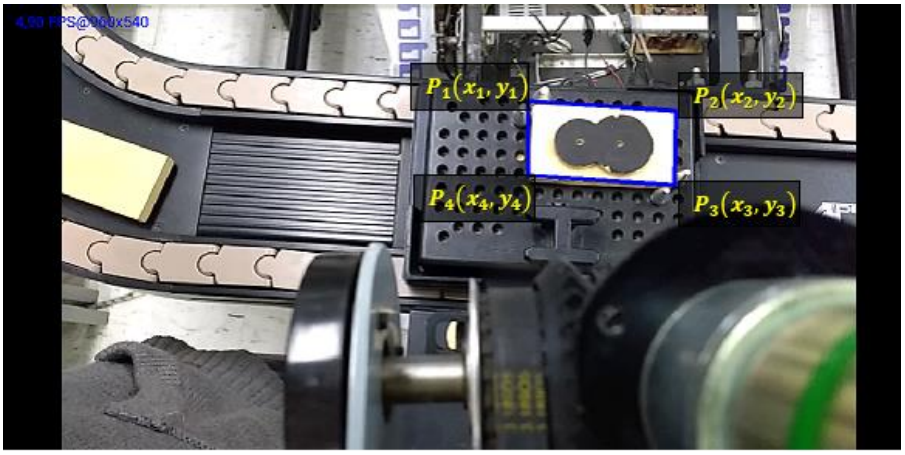


Figura 11. Dimensionar región de interés ROI mediante coordenadas.

Con los puntos ya establecidos se procede a cortar la región marcada y almacenarla en una nueva imagen transformándola mediante transformaciones proyectivas de corrección (Sección 6.6).

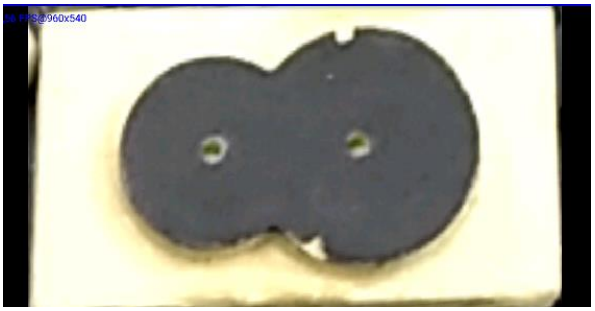


Figura 12. Región ROI cortada y redimensionada mediante transformación de perspectiva.

6.5.2 Determinar la región de interés de un objeto mediante filtro en espacio de colores HSV.

El proceso a utilizar para detectar la región de interés a través de manchas será similar al algoritmo utilizado para el seguimiento de objetos por textura o color MeanShift (Dawson-Howe, 2014). Este método es probabilístico y selecciona un espacio de colores contenidos, en una región seleccionada, y almacenada en un histograma, para su posterior procesamiento, retroproyección y seguimiento de un objeto, de acuerdo a su color y textura.

En el proceso utilizado para el desarrollo de la aplicación, a diferencia del método MeanShift, la imagen será segmentada para proyectar una máscara binaria de retroproyección que contenga la selección del rango de colores del espacio HSV seleccionado por el usuario, luego determinar el contorno o mancha de mayor tamaño para luego determinar su contorno y finalmente determinar la región de interés de acuerdo a la ubicación y color del objeto de interés definido por el usuario.

A continuación de describen las acciones implementadas para la determinación de la región de interés mediante proceso de segmentación mediante el espacio de colores HSV.

1. Inicialmente se capturó una imagen mediante la cámara del dispositivo y esta se almacena en un objeto Mat utilizando la librería OpenCV.

2. Luego se transformó la imagen desde el modelo de color RGB a HSV, utilizando el módulo "Imgproc" de la librería OpenCV como en la Figura 13.

```
// Función de conversión de imagen a espacio de color HSV.
Imgproc.cvtColor(mRGB, mHSV, Imgproc.COLOR_RGB2HSV);
```

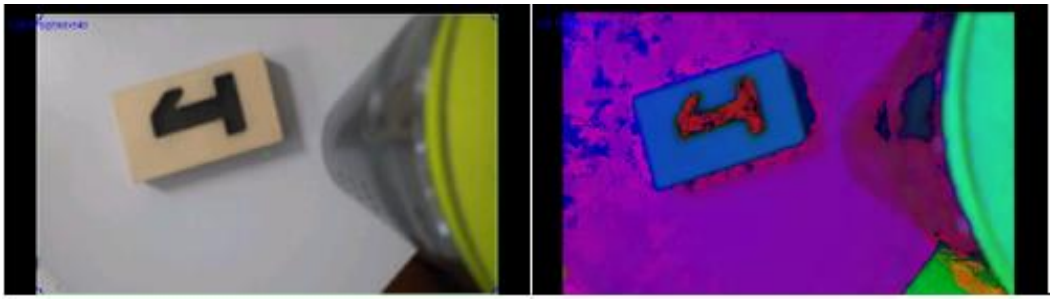


Figura 13. Resultado de conversión de imagen de RGB a HSV.

3. Finalmente se proyectó una máscara binaria de la imagen HSV mediante la selección de un rango mínimo y máximo dentro del espacio de colores en el modelo HSV

En los programas de edición de color y/o imagen, el rango de matiz puede ser de 0 a 100 o de 0° a 360°, para la saturación y el brillo es de 0% a 100%. El rango establecido para el espacio de colores HSV en la librería OpenCV es el siguiente:

- Matiz: de 0 a 179,
- Saturación: de 0 a 255, y
- Brillo: de 0 a 255.

Utilizando la librería OpenCV y con la imagen ya convertida a HSV, el siguiente paso fue definir los parámetros de rango mínimo y máximo para cada variable contenida en una imagen HSV. Estas variables fueron almacenadas en un array de tres valores para cada rango, en la versión Android de OpenCV es almacenado en un objeto llamado "Scalar". La función OpenCV para definir una máscara binaria es la función "inRange" del módulo "Core" y es programada de la siguiente manera:

```
Core.inRange(mHSV, new Scalar (H_MIN, S_MIN, V_MIN), new Scalar(H_MAX, S_MAX, V_MAX), mMask);
```

Como lo indica la Figura 14, en la máscara se identifican los contornos o manchas contenidas mediante la filtración y segmentación a través del rango de valores mínimos y máximos del matiz, de la saturación, y el brillo o valor del espacio de colores HSV.

Figura 14. Máscara binaria mediante conversión de una imagen RGB al espacio de colores HSV y definición de máscara de la imagen mediante segmentación del color valores de Matiz, Saturación y Valor.

Los parámetros definidos en este proceso de segmentación son:

Parámetro/variable	Matiz (0-179)	Saturación/Valor (0-255)	Escala modelo HSV
Matiz mínimo	12	-	24,1°
Saturación mínima	-	83	32%
Valor brillo mínimo	-	151	59%
Matiz máximo	19	-	38,2°
Saturación máxima	-	255	100%
Valor brillo máximo	-	228	89%

Tabla 10. Parámetros de segmentación mediante rango de colores en espacio de color HSV a escala predeterminada de OpenCV.

Utilizando las herramientas de diseño gráfico disponibles como Gimp, Adobe Photoshop o Inkscape es posible definir más fácilmente el rango de colores para identificar algún objeto de un color determinado.
En este ejemplo se utilizó herramienta Gimp y en la siguiente figura se demuestra los rangos de espacio de color seleccionado:

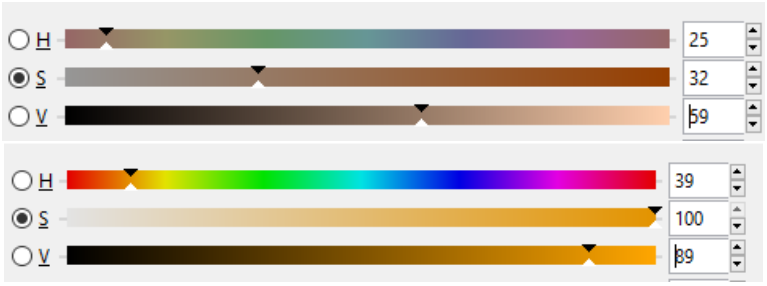


Figura 15. Definición del rango de color en el espacio de colores HSV utilizando la herramienta de diseño Gimp.

6.5.3 Umbralización de imagen en escala de grises.

La umbralización, es uno de los más importantes métodos de segmentación. El objetivo es convertir una imagen en escala de grises a una nueva con sólo dos niveles: 0 y 1, de manera que los objetos queden separados del fondo.

Una forma de extraer el objeto del fondo es seleccionar un umbral de intensidad $\boldsymbol{T}$ que separe los dos conjuntos. De este modo, cualquier punto $p(\boldsymbol{x},\boldsymbol{y})$ para el que $f(\boldsymbol{x},\boldsymbol{y}) > \boldsymbol{T}$, pertenecerá al objeto, en caso contrario, pertenecerá al fondo. Si los valores de gris del objeto y del resto de la imagen difieren claramente, entonces el histograma mostrará una distribución bimodal, con dos máximos distintos, separados por una región vacía. Con lo cual se logró una separación perfecta entre el objeto y el fondo.

si (viejo_píxel>umbral)
nuevo_píxel = MAX
sino
nuevo_píxel = MIN

, donde MIN es 0 y MAX es 1 o bien 255.

OpenCV dispone de cinco tipos de umbralización global simples (Image Thresholding , 2016) que se muestra en la Figura 16:

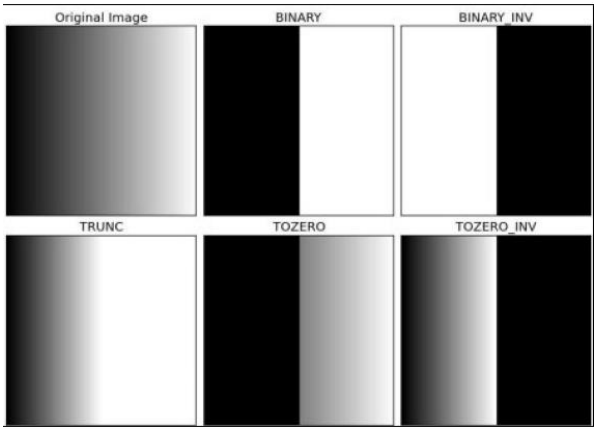


Figura 16. Métodos de umbralizacion binaria en escala de grises implementadas en OpenCV.

La umbralización general puede no ser buena en todas las condiciones donde la imagen tiene las condiciones de iluminación diferentes en diferentes áreas. En ese caso, se puede recurrir a la umbralización adaptativa. En esto, el algoritmo calcula el umbral para pequeñas regiones de la imagen. Así se obtiene diferentes umbrales para diferentes regiones de la misma imagen obteniendo mejores resultados para imágenes con diferente iluminación.

En la función de umbralización adaptativa se emplean tres argumentos de entrada y un argumento de salida:

1) Método adaptativo:
   - ADAPTIVE_THRESH_MEAN_C: el valor de umbral es la media de los píxeles vecinos
   - ADAPTIVE_THRESH_GAUSSIAN_C: el valor de umbral es la suma ponderada de los valores de los píxeles vecinos, donde los valores son núcleos gaussianos.

2) Tamaño del bloque en pixeles.

3) C: esto es una constante que tiene que ser restada de la media ponderada calculada para cada pixel.

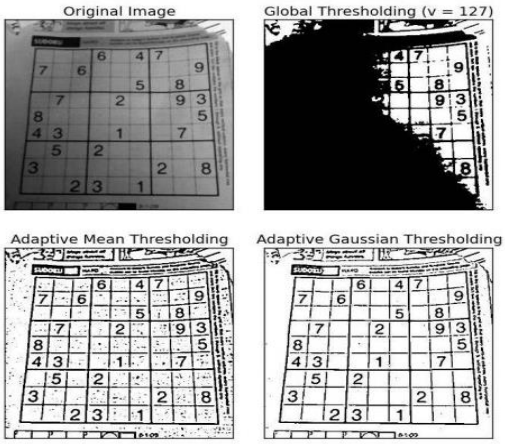


Figura 17. Imagen en escala de gris, implementado los métodos de umbralización global con un umbral de 127, y umbralización adaptativa de núcleo de valores medios y gaussiano.

Para detectar figuras y patrones dentro de la región de interés, primero se convierte la imagen contenida en el ROI en una imagen en escala de grises, y luego se procede a la umbralización utilizando el método adaptativo deseado. En el caso de la aplicación desarrollada se utilizó el método adaptativo de núcleo de valores gaussiano.

En la Figura 18c está demostrado el resultado de la umbralización de la región de interés de la Figura 18b.

Figura 18. Umbralización de imagen contenida en la región de interés con valor umbral inverso de 100.

Por qué el método de umbralización adaptativa por núcleos gaussianos.

Cuando se aplica un núcleo de valores medios se obtienen resultados artificiosos en el ámbito del filtrado o del suavizado de una imagen a diferencia del núcleo gaussiano que es una matriz ponderada de valores donde se obtienen resultados más reales, como el ejemplo mostrado en la Figura 19:

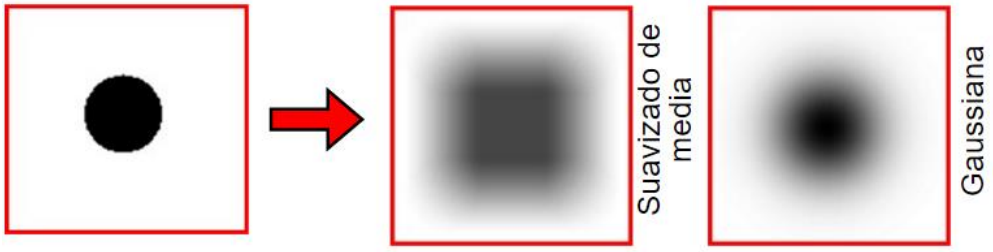


Figura 19. Diferencia del suavizado de una imagen a través de un núcleo de valores medios y un núcleo de valores gaussiano (Mateos, 2017).

Mientras la de media se calcula en una región cuadrada, la gaussiana se aplica en una región redonda.

Matriz de valores medios donde el valor del pixel central corresponde al valor promedio de sus pixeles vecinos:

$$\frac{1}{5*5} * \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Figura 20. Matriz de valores medios.

Matriz de valores gaussiano ponderados según la distancia de los pixeles vecinos al pixel central.

$$g_5 = \frac{1}{246} * \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Figura 21. Matriz de valores gaussiano.

Normalmente el método gaussiano se aplica en dos dimensiones y los pesos de la máscara dependen de la distancia al pixel central.

Esto se demuestra a través de la campana de Gauss en la Figura 22:

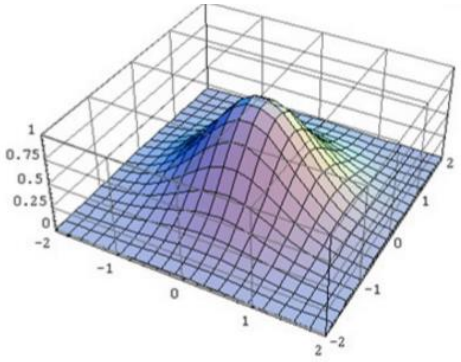


Figura 22. Campana tridimensional de Gauss, (Mateos, 2017).

6.5.4 Implementación del algoritmo de detección de bordes de Canny.

En OpenCV se implementa las cuatro etapas (sección 14.4) del algoritmo en una sola función:

Imgproc.Canny(Imagen_mascara , imagen_canny , minVal , maxval)

El primer argumento es la imagen de entrada y el segundo argumento es la imagen de salida con la detección de bordes. El tercer y cuarto argumento corresponde los valores de umbral mínimo y máximo (minVal y maxVal) respectivamente.
En la Figura 23 se presenta la implementación del algoritmo Canny en la identificación de bordes de una imagen binarizada.

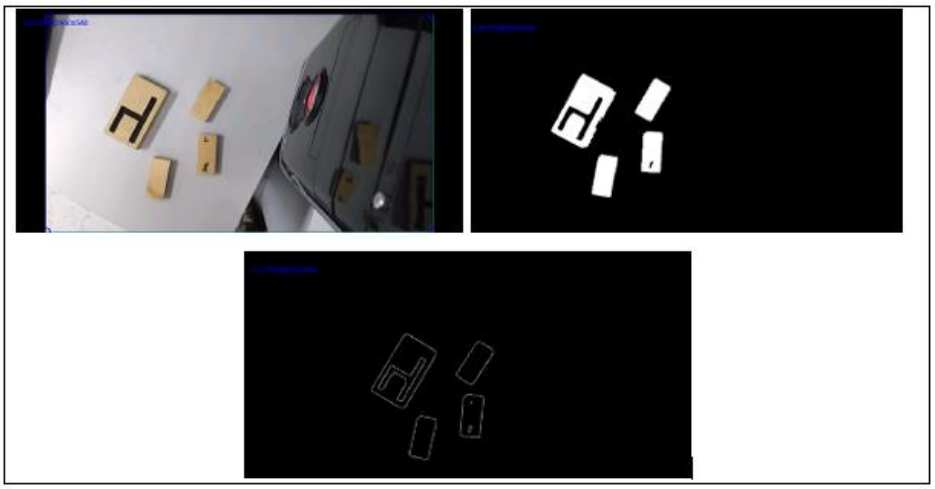


Figura 23. Detección de bordes de una imagen binaria mediante algoritmo de Canny.

6.5.5 Operaciones morfológicas.

La morfología matemática está basada en operaciones en la teoría de conjuntos (Morfología, 2016). Las operaciones morfológicas simplifican imágenes y conservan las principales características de forma de los objetos. Un sistema de operadores de este tipo y su composición, permite que las formas subyacentes sean identificadas y reconstruidas de forma óptima a partir de sus formas distorsionadas y ruidosas. La morfología matemática se puede usar, entre otros, con los siguientes objetivos:

- Pre-procesamiento de imágenes (supresión de ruidos, simplificación de formas).
- Destacar la estructura de los objetos (extraer el esqueleto, detección de objetos, envolvente convexa, ampliación, reducción, ...)
- Descripción de objetos (área, perímetro, ...)

En OpenCV, primeramente antes de aplicar alguna definición morfológica se debe establecer un kernel, que es una matriz de tamaño fijo de coeficientes numéricos junto con un punto de anclaje en esa matriz, que está típicamente ubicado en el centro. El tamaño del kernel es definido por el número de filas y de columnas de la matriz.

0	1	0
1	1	1
0	1	0

Para realizar un filtro u operación morfológica, para un kernel de tamaño 3x3, el kernel debería ser:

$$K = \frac{1}{3 * 3}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Para definir el kernel en la aplicación se utilizó la función getStructuringElement del módulo de la librería OpenCV Imgproc:

```
Mat kernel = Imgproc.getStructuringElement(Imgproc.MORPH_ERODE, new org.opencv.core.Size(5, 5), new Point(1, 1));
```

Luego de establecer el Kernel creado se procedió a implementar las operaciones morfológicas mencionadas a continuación:

Apertura.

La apertura corresponde al proceso de erosión seguida por una dilatación. Es útil para eliminar el ruido o manchas blancas no deseadas. El Kernel es el mismo en la erosión y dilatación.

Mancha la definiremos como conjunto de pixeles vecinos con características similares en una imagen binaria.

Figura 24. Ejemplo de aplicación de operación morfológica de apertura.

Clausura.

Es el proceso inverso a apertura, dilatación seguida por una erosión. Es útil para cerrar pequeños agujeros dentro de los objetos en primer plano, o pequeñas manchas negras en el objeto. Al igual que en el proceso de apertura el valor del Kernel es el mismo en la erosión y dilatación.

Figura 25. Ejemplo de aplicación de operación morfológica de clausura.

6.5.6 Detección de contornos.

Un contorno representa el borde externo de una figura geométrica, en una imagen para este proceso representa un conjunto de pixeles localizados en el borde externo de una mancha, o figura.

La imagen a analizar debe ser binaria, es decir de valores 1 y 0 ó 0 y 255, de un canal y de 8 bits por pixel, en el caso de que la imagen corresponda a una imagen a color se debe utilizar el algoritmo de detección de bordes de Canny.

La función del módulo *"imgproc"* de la librería OpenCV *"findContours()"* computa contornos desde una imagen binaria ya sea creadas mediante las funciones de detección de bordes de Canny, o de umbralizacion.

La salida de la función *findContours()* corresponde a una lista que almacena en cada una de sus posiciones un vector que contiene una cantidad finita de puntos que corresponde a cada pixel que se sea vecino entre si y que tengan su mismo valor binario, además cada punto tiene relación con un pixel vecino anterior y un pixel vecino siguiente formando una curva ,en el que el ultimo pixel, su pixel vecino siguiente corresponde al pixel inicial de la curva. En OpenCV para Java esta salida es almacenada en un objeto llamado "MatOfPoint", cuyo objeto permite realizar todos los análisis y modificaciones correspondientes a cada uno de los contornos, además permitiendo encontrar patrones de imágenes y figuras geométricas que sirven para el desarrollo del proyecto.

Los parámetros utilizados en la función *"findContours"* son:
- Imagen binaria de entrada (Mat).
- Lista de contornos vacía (MatOfPoint).
- Matriz de jerarquía de los contornos, establece el orden de los contornos y corresponde a una matriz auxiliar (Objeto Mat).
- Valor índice del método de recuperación de los contornos (ver Sección 14.6).
- Valor índice del método de aproximación de contornos (ver Sección 14.6).

La jerarquía básicamente indica la forma y el orden del arreglo general de contornos de la imagen binaria.

En la implementación de los algoritmos para la definición de la región de interés ROI y para la detección de algún contorno para definirlo como patrón o figura, se utilizó el modo de recuperación de contornos externos ya que son los contornos de interés para el proyecto, y el método de aproximación simple dejando solo los puntos que definen la forma del contorno, en el caso del contorno del ROI.

Una vez obtenidos los contornos podemos mostrarlos usando la función OpenCV *"drawContours"* esta se encargará de unir con líneas los puntos obtenidos por la función anterior, podemos indicar un grosor de línea en pixeles o indicar –1 para rellenar toda la figura.

```
//Encontrar contornos en la imagen mMask y almacenarlo en el vector countours.
Imgproc.findContours(mMask, contours, hierarchy, Imgproc.RETR_EXTERNAL,
Imgproc.CHAIN_APPROX_SIMPLE);

//Dibujar y proyectar los contornos detectados de la imagen mMask en la imagen a color mRgba.
for(int i = 0 ; i< contours.size() ; i++){
   Imgproc.drawContours(mRgba, contours, i, color, 1);
}
```

Donde "*mRgba*" es la imagen de salida, "*contours*" es el vector de contornos, "*i"* es el índice o posición del contorno en el vector, "*color*" es un objeto Scalar que define el color mediante los canales RGB, en el caso de la Figura 26 es (0, 255,255) correspondiente al color Cyan , y *1* es el grosor de la línea del contorno en pixeles.

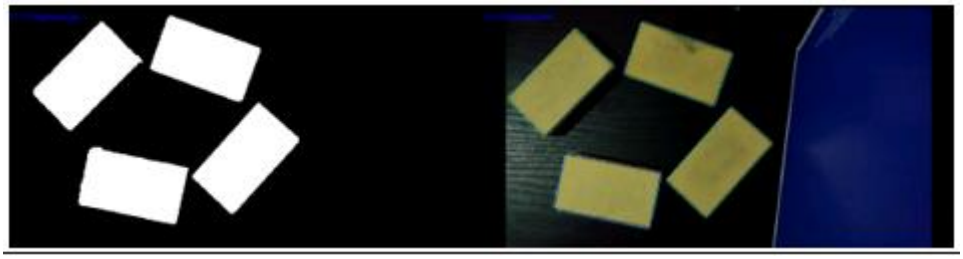


Figura 26. Identificación y proyección de contornos de una imagen binaria en una imagen a color.

6.5.7 Capturar región de interés a través del contorno de mayor tamaño.

El modo utilizado para definir el ROI está descrito en la siguiente secuencia:

1. Identificar la mancha o contorno de mayor tamaño dentro una imagen binaria. Esto se logra calculando los perímetros de cada mancha o figura encontrada en la imagen. En OpenCV está a función del módulo de procesamiento de imagen "Imgproc.arcLenght":

```
//Calcular área del objeto contorno "MatOfPoint" contorno_a_medir.
double area = Imgproc.arcLenght(contorno_a_medir);
```

2. Aproximación de contornos a un polígono. El objetivo es aproximar un contorno a una figura con menos número de vértices dependiendo la precisión que se especifique. Para entender esto, supongamos que se está tratando de encontrar un cuadrado en una imagen, pero debido a algunos problemas en la imagen, no obtuvo un cuadrado perfecto, sino una "mala forma" (como se muestra en la primera imagen de la Figura 27).

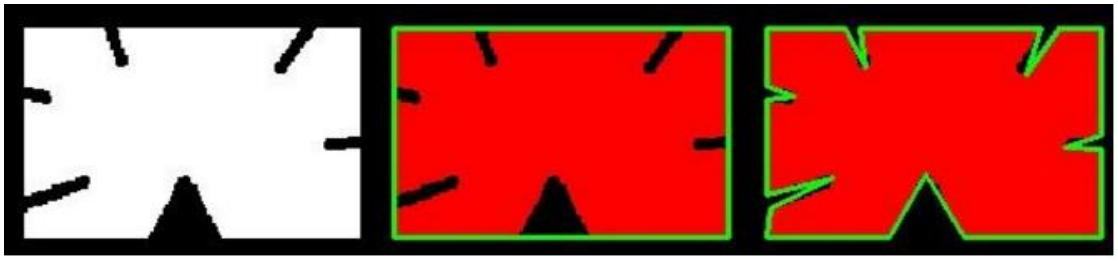


Figura 27. Aproximación de contorno a una figura.

Ahora puede utilizar esta función para aproximar la forma.

```
Imgproc.approxPolyDP(new_mat, approxCurve_temp, epsilon, true);
```

Para asignar el primer parámetro se debe crear una matriz de puntos de valores flotantes "MatOfPoint2f" que almacenará y convertirá el contorno a un objeto "MatOfPoint2f".

```
//Crea un objeto MatOfPoint2f (Matriz de puntos con valores flotantes)
MatOfPoint2f new_mat = new MatOfPoint2f();
//Convierte el contorno actual desde un objeto MatOfPoint a MatOfPoint2f
temp_contour.convertTo(new_mat, CvType.CV_32F);
```

Para el segundo parámetro se debe crear un objeto "MatOfPoint2f" que almacenará el contorno aproximado de acuerdo al porcentaje del contorno aproximado, asignado por el tercer parámetro "*épsilon*", que es la

distancia máxima entre el contorno y el contorno aproximado. Es un parámetro de precisión. Una sabia selección de épsilon es necesaria para obtener la salida correcta. El parámetro "épsilon" es un parámetro numérico decimal, y en OpenCV en lo calcula como:

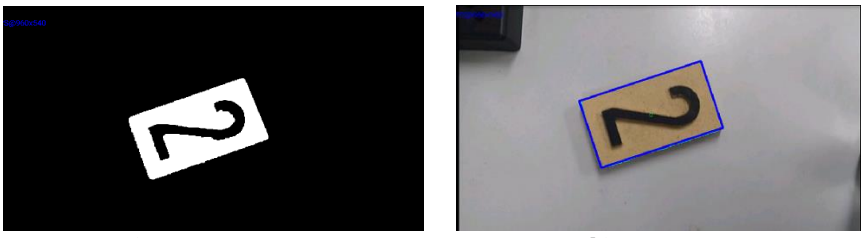
```
double epsilon = porcentaje * Imgproc.arcLength(new_mat, true);
```

donde "*porcentaje"* corresponde al porcentaje del perímetro total del contorno a aproximar dividido por 100. Como lo indicado en la Figura 27 , en la segunda imagen el contorno esta aproximado al 10%, en la tercera imagen esta aproximado al 1% del perímetro total.

3. Encontrar el número de vértices del contorno aproximado. El objetivo es encontrar un contorno aproximado de mayor tamaño de cuatro vértices para el caso de definir un ROI.

4. Definir el rectángulo de menor área que define la región de interés. Definir un rectángulo que delimite el contorno seleccionado con el área mínima considerando la rotación. La función utilizada es *Imgproc.minAreaRect* cuyo argumento corresponde a un contorno, devuelve un objeto de tipo rectángulo que en OpenCV para Android es "*RotatedRect*" y contiene los detalles de centro $p(x, y)$, (ancho, altura), ángulo de rotación y coordenadas de los vértices. Para dibujar el rectángulo se necesita definir las cuatro esquinas del rectángulo que se pueden obtener con la función "Imgproc.points()".

En la Figura 28, en la imagen de la derecha 28b se observa que se encuentra dibujado el rectángulo de color azul, calculado del contorno de la figura destacada en la imagen de la izquierda 28a con sus cuatro vértices ya definidos.



a) b)

Figura 28. Definición de la región de interés del objeto mediante un color específico.

6.6 Corrección de perspectiva de una región de interés.

Teniendo las coordenadas de los vértices del rectángulo es posible capturar la subimagen contenida dentro del rectángulo, además, girar, corregir y redimensionar la región de interés, utilizando las transformaciones geométricas proyectivas de corrección.

En el proceso de corrección, ya capturados los cuatro vértices y sus coordenadas se procede a calcular la transformación de perspectiva asignando cuatro puntos de destino. Los puntos de destino serán:

$P_{1(0,0)}$,$P_{2(0,N\_COLUMNAS)}$,$P_{3(N\_FILAS,N\_COLUMNAS)}$, $P_{4(N\_FILAS,0)}$,

donde N_FILAS y $N\_COLUMNAS$, serán las dimensiones de la imagen de destino en pixeles.

En transformaciones geométricas y afines, la principal transformación realizada es la de similitud o transformación de escala o redimensionado de un conjunto de pixeles contenidos dentro de una región de interés, además de la transformación afín en el caso que sea necesario. En el caso de este proyecto gracias a la librería OpenCV están implementadas por lo que el fin de su explicación fue dar a entender los conceptos de las transformaciones afines.

Para identificar una región de interés dentro de la imagen y hacer la transformación y corrección de perspectiva primero se detectará la región de interés dentro de una imagen, y luego de efectuará transformación perspectiva o correctiva a dicha región.

Para tener una transformación o corrección de perspectiva de una imagen en este caso de una región de interés se debe primero obtener la región de interés, este procedimiento se puede efectuar de acuerdo a lo investigado de tres maneras:

1) Modo rígido: Detecta un rectángulo perpendicular dentro de la imagen, mediante identificación de contornos, se captura las coordenadas de los cuatro vértices, y finalmente se redimensiona y almacena la imagen contenida en esa región rectangular (Sección 6.5.7).

2) Modo flexible: Detecta un conjunto de líneas rectas dentro de la imagen mediante transformadas de Hough, luego se calcula las coordenadas de los puntos de intersección entre esas líneas y se estima una región rectangular no perpendicular, si se identifican cuatro intersecciones, finalmente se redimensiona y almacena la imagen contenida en esa región rectangular.

3) Modo manual: El usuario asigna las coordenadas de los cuatro vértices, se corrige la imagen contenida entre esos vértices, se redimensiona y se almacena en una nueva imagen (Sección 6.5.1).

En el desarrollo de la aplicación fueron implementados los modos rígido y manual debido a que resulta más eficiente en el ámbito del gasto de recursos de hardware del dispositivo a diferencia del modo flexible ya que el algoritmo de detección de líneas por transformadas de Hough a pesar de ser podría ser eficiente en la entrega de resultados como la región de interés resulta ineficiente en el ámbito del uso de recursos de hardware en el dispositivo móvil.

Aplicando transformación de perspectiva de corrección a una imagen.

Teniendo las coordenadas de los puntos del cuadrilátero que definen a la región de interés y de los puntos que serán de destino, se procede a calcular la transformación de perspectiva utilizando la función GetPerspectiveTransform del módulo Imgproc, esta función calcula la transformada en una matriz de 3 x 3 de la siguiente manera.

$$\begin{bmatrix} t_i x_i' \\ t_i y_i' \\ t_i \end{bmatrix} = mappedmatrix * \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

Donde $dst(i) = (x_i', y_i')$, $src(i) = (x_i, y_i)$, $i = 0,1,2,3$.
Parametros:
Src: Las coordenadas de los vértices del cuadrilátero definido como ROI.
Dst: Las coordenadas de los vértices del cuadrilátero definido para la imagen de destino.
Mapped_matrix: Corresponde a la matriz de 3x3 que almacena la transformación perspectiva.

Con la transformación perspectiva se procede a la implementación de la transformación de utilizando la función warpPerspective de OpenCV. Esta función utiliza la matriz de transformación de 3x3 obtenida previamente se la siguiente manera:

$$dst(x,y) = src\left(\frac{M_{11}x + M_{12}y + M_{13}}{M_{31}x + M_{32}y + M_{33}}, \frac{M_{21}x + M_{22}y + M_{23}}{M_{31}x + M_{32}y + M_{33}}\right)$$

Donde los parámetros son:
Src: Imagen de entrada.
Dst: Imagen de salida de mismo tamaño y tipo de la imagen de entrada.
M: Matriz de transformación.

La función implementada en Android utilizado OpenCV es:

```
Mat M = Imgproc.getPerspectiveTransform(srcPoints, dstPoints); //Matriz de transformación
Imgproc.warpPerspective(src, dst, M, dst.size()); //Aplica la transformación
```

En la Figura 29a se observa el rectángulo azul dibujado alrededor del objeto y en la imagen 29b se presenta la transformación de perspectiva de la imagen contenida dentro del rectángulo o región de interés de 29a.

29a)  29b) 

Figura 29. Transformación de corrección de perspectiva de la región de interés.

6.7 Descripción y comparación de figuras y contornos.

6.7.1 Calcular el porcentaje de aciertos entre dos figuras mediante operaciones lógicas de bit a bit.

En dos imágenes de m*n pixeles, la función disponible de OpenCV "bitwise" calcula los valores en una tercera imagen con la operación OR exclusiva (XOR). La Figura 30 muestra la operación realizada y cuya imagen resultante (Resultado) obtenida por la operación se utiliza para calcular el porcentaje de coincidencias entre dos imágenes.

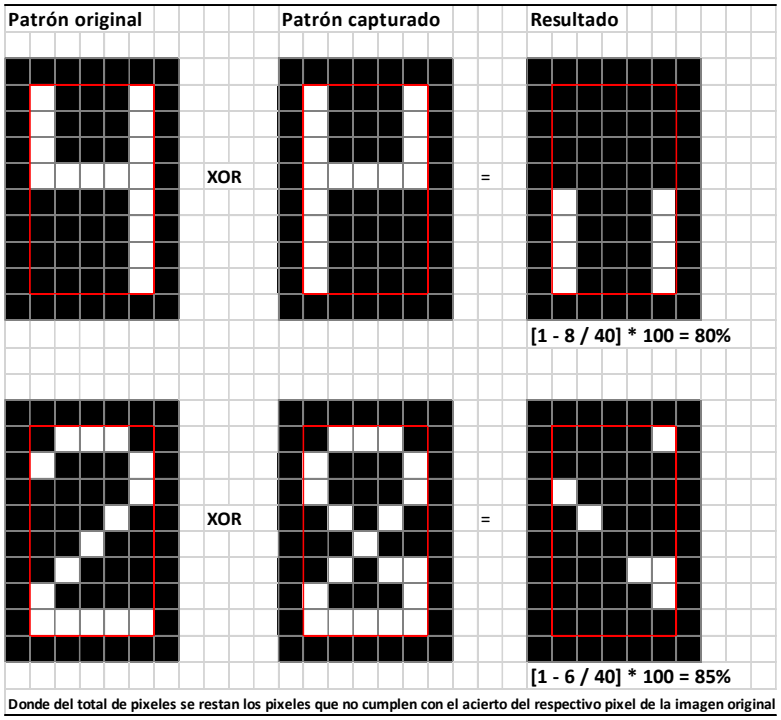


Figura 30. Comparación de dos imágenes binarias mediante la operación lógica OR Exclusiva.

El porcentaje de aciertos entre las imágenes fue calculado como:

$$Pixeles\ de\ coincidencias\ =\ total\_pixeles(Resultado)\ -\ countNonZero(Resultado)$$
$$\%\ aciertos\ =\ ((Pixeles\ de\ coincidencia\ /\ total\_pixeles(Resultado))\ *\ 100$$

El código utilizado para implementar la comparación de dos imágenes en la aplicación fue el siguiente:

```
//Operación bitwise XOR , para encontrar pixeles que se diferencien entre dos imágenes binarias
Core.bitwise_xor(patronOriginal, patronCapturado, resultadoComparacion);

//Calcular la cantidad de pixeles de color blanco en la imagen resultante
int distantPixels = Core.countNonZero(resultadoComparacion);//pixeles con valor 1, del resultado

//Calcula el porcentaje de aciertos restando del número de pixeles del tamaño de la imagen, los pixeles con valor 1
float porcentajeAciertos = 100 - (((float) distantPixels / (float) pixelsPatronOriginal) * (float) 100);
```

6.7.2 Comparación e identificación de figuras y objetos mediante descriptores.

En este proyecto se optó por utilizar los Momentos Invariantes de Hu [1962], debido a su robusto soporte matemático (ver Sección 14.7). Específicamente por que provienen de los momentos estadísticos que describen a cada objeto en 2D unívocamente con un valor en esos descriptores.
En el reconocimiento y análisis de patrones se hará uso de los momentos de Hu debido a su rapidez y robustez en el reconocimiento de patrones además por el eficiente uso de los recursos de hardware del dispositivo para su cálculo.

Primero se hace búsqueda de una mancha principal que describirá una figura o patrón de color blanco sobre el negro, en la región ROI que se encuentra binarizada. De esta mancha serán extraídos los 7 momentos que describirán la figura.

Al identificar un patrón ya sobre una región ROI procesada, se guardará la imagen contenida dentro del ROI junto al nombre de la imagen asignado por el usuario, la ruta de la imagen ROI, mas sus 7 momentos de Hu de la figura principal encontrada y contenida dentro del ROI. La imagen se guardará en el directorio asignado por la aplicación en la memoria del dispositivo, y los datos del directorio y los momentos de Hu de la imagen se almacenarán en una base de datos SQLite para ser utilizados si es necesario.

En el proceso de comparación de patrones en el momento en que la cámara del dispositivo apunta sobre una figura o patrón, el usuario comparará ese patrón encontrado con una figura ya guardada en la memoria del dispositivo, a raíz de esto la aplicación calculará el porcentaje de coincidencias mediante la función de correlación de bit a bit, y también calculará los momentos del patrón proyectado en la cámara del dispositivo. Para identificar el patrón se buscará su patrón similar comparando la diferencia entre los momentos del patrón capturado por la cámara y de cada uno de los patrones almacenado en la memoria del dispositivo.

Uno de los algoritmos utilizados para la comparación y reconocimiento de patrones y contornos mediante los momentos invariantes de Hu corresponde al método de comparación *matchShapes( ),* del módulo *Imgproc* de la librería OpenCV. Naturalmente con los momentos de Hu se puede comparar dos objetos y determinar si estos son similares. Para hacer este proceso algo más fácil, la función de OpenCV matchShapes*()* simplemente proporciona dos objetos o contornos y sus momentos calculados y comparados de acuerdo con un criterio seleccionado.

Los objetos a comparar pueden ser contornos o manchas como también imágenes en escala de grises. La función *matchShapes* (Kaehler & Bradski, 2016) proporciona tres métodos de comparación descritos en la Tabla 11:

Método (valor)	Fórmula y valor de retorno de comparación
CONTOURS_MATCH_I1	$\nabla_1 = \sum_{i=1}^{7} \left\lvert \frac{1}{\eta_i^A} - \frac{1}{\eta_i^B} \right\rvert$
CONTOURS_MATCH_I2	$\nabla_2 = \sum_{i=1}^{7} \lvert \eta_i^A - \eta_i^B \rvert$
CONTOURS_MATCH_I3 (*)	$\nabla_3 = \sum_{i=1}^{7} \left\lvert \frac{\eta_i^A - \eta_i^B}{\eta_i^A} \right\rvert$

Tabla 11. Métodos de comparación de figuras y contornos de la función matchShapes().

Donde:

$$\eta_i^A = sign(\phi_i^A) * \log \phi_i^A \quad , \quad \eta_i^B = sign(\phi_i^B) * \log \phi_i^B$$

, y ϕ_i^A y ϕ_i^B son los momentos invariantes de Hu del contorno A y B respectivamente.
El método utilizado para la comparación de patrones corresponde al del valor: ***CONTOURS_MATCH_I3***.

Este método fue utilizado para comparar el patrón capturado por la cámara del dispositivo con cada uno de los patrones almacenados en la memoria del dispositivo, con el objetivo de encontrar el patrón que cuyo resultado de la operación *matchShapes( )* sea el de menor valor, en el cuál corresponderá al patrón identificado.

6.8 Diagrama de flujo del procesamiento de imágenes aplicado.

La forma en que el sistema en su primera fase de su desarrollo correspondiente a el procesamiento de imágenes con OpenCV, se describe en el siguiente diagrama de flujo (Figura 31) desde que la imagen es visualizada por la cámara del dispositivo y a la vez segmentada hasta la etapa de entrega de resultados en la comparación entre dos figuras en imágenes binarias y su reconocimiento de ésta.

6.9 Problemas y dificultades ocurridos en el tratamiento de imagen en la aplicación.

El primer problema fue seleccionar la versión adecuada de la librería OpenCV para aplicaciones desarrolladas en Android y entender su estructura debiendo invertir tiempo valioso debido a que existía poca documentación o dicha documentación no era para la versión adecuada de OpenCV, teniendo dificultades en su instalación y el funcionamiento en la cámara del dispositivo.

El desafío fue entender el proceso de visión e identificación de patrones utilizado en el laboratorio, desarrollado en el lenguaje de programación C# en MS Visual Studio utilizando la librería EmguCV, en cual corresponde a una adaptación (wrapper) de la librería OpenCV para aplicaciones de visión desarrollada en MS Visual Studio. La mayor parte de estos procesos fue adaptada a Android utilizando la librería OpenCV para Android y desarrollada en Java.

La documentación y archivos tutoriales de OpenCV para Android fue limitada a diferencia de la documentación existente para lenguajes C++ y Python, por lo que la implementación de algunos algoritmos, como la resolución de los errores de la aplicación fueron obtenidos de tutoriales de OpenCV para C++ y Python y adaptada a Java.

Otros de los problemas presentados fue el tiempo utilizado en las pruebas de cada avance y la compilación de la aplicación, junto con los errores en tiempos de ejecución, y su dificultad en la búsqueda del error y encontrar su causa con el fin de darle solución a éste.

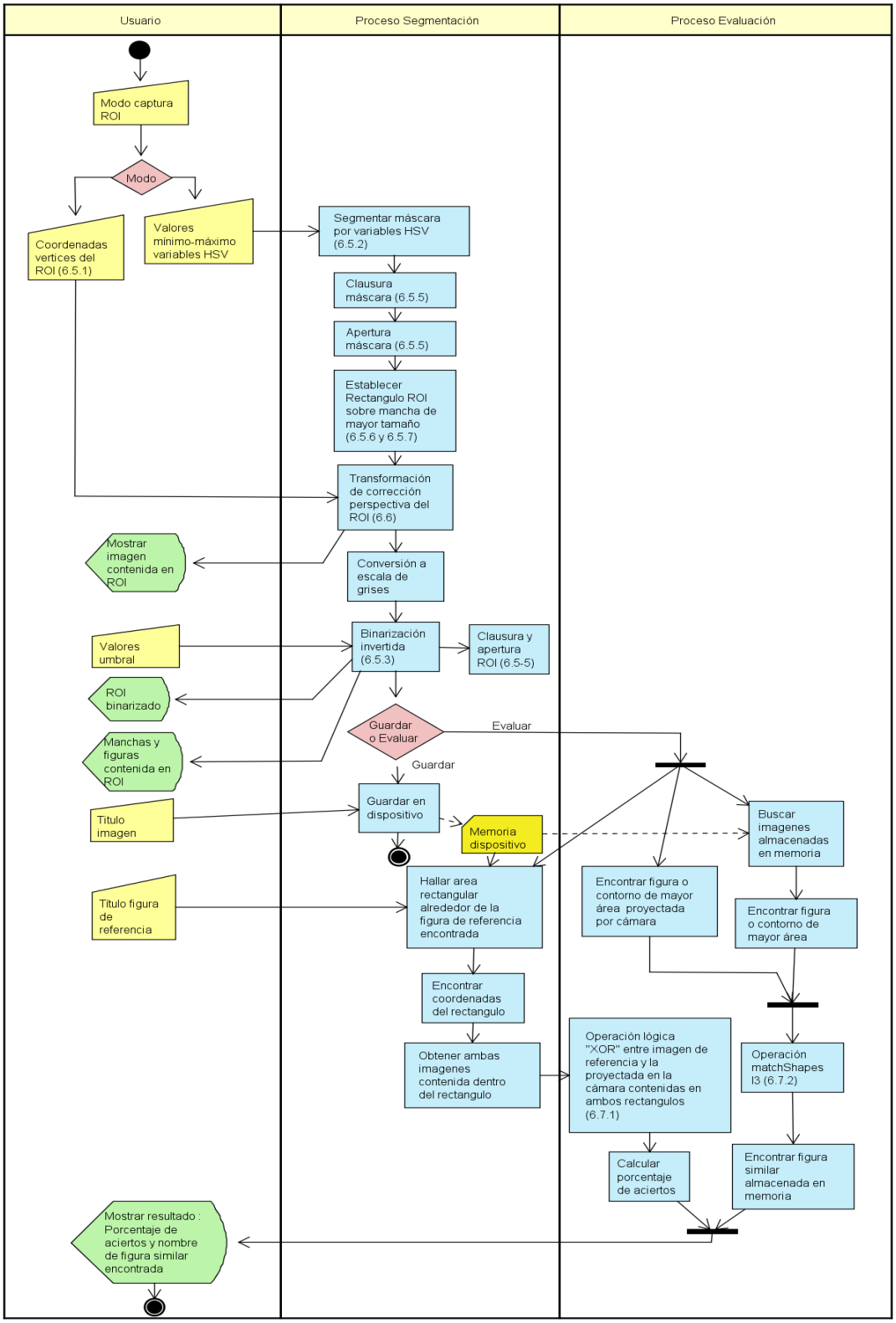


Figura 31. Diagrama de flujo del proceso de tratamiento digital de imágenes utilizado por la aplicación.

7 ROBÓTICA.

7.1 Introducción.

En el proyecto se utilizó la celda flexible Scorbot ER-V Plus del CIMUBB, esta celda flexible está compuesta por un brazo-robot, una botonera de enseñanza y un controlador.

Figura 32. Brazo robot de la celda flexible Scorbot ER-V del Laboratorio CIMUBB.

Usualmente es parte de una célula de trabajo y se puede ver interactuando con otras células por medio de un PC, con la finalidad de realizar tareas integradas de manufactura. El PC se comunica con el controlador del robot, ya sea para enviar instrucciones o recibir comandos de control por medio de alguna interfaz de programación. Con la botonera de enseñanza se practican los primeros movimientos del robot y se envían instrucciones básicas al controlador; éstos constituyen los primeros pasos que deben dar los estudiantes para entender y puedan interactuar de manera segura con el robot.

A lo largo de este capítulo se darán a conocer los propósitos del brazo-robot en el proyecto, como también la forma de comunicación entre un Smartphone y el controlador del robot, los comandos en lenguaje ACL utilizados para su operación, el método usado para el grabado de posiciones de movimiento, finalizando con los programas creados y ejecutados por el robot para cumplir con los objetivos del proyecto.

7.2 Rol que cumple el Scorbot ER-V en este proyecto.

Los puntos de este capítulo anteriormente tratados han sido de mucha importancia para que en primera instancia familiarizarse con la celda flexible y el brazo robot, conocer sus partes fijas y móviles junto a sus juntas rotativas, que es lo primero que se debe hacer antes de comenzarlo a utilizar, que hacer en caso de querer abortar una operación, conocer su sistema de coordenadas, su lenguaje de programación, en qué modo utilizarlo, entre muchos otros aspectos que debemos indagar para poder obtener de él su mayor provecho, para que de ésta manera contribuya al cumplimiento de los objetivos del proyecto.

Entendido todo esto, se sabe que el brazo robótico es una de las partes de mucha importancia en este proyecto, considerando que cada fase de desarrollo de este proyecto aporta su porcentaje para lograr el objetivo final, por ello el rol que cumple en el proyecto es de importancia ya que el brazo robótico será considerado como el principal dispositivo "actuador" dentro del sistema desarrollado en este proyecto, pues después de haber procesado la imagen a través de un dispositivo móvil, el controlador del robot recibirá las instrucciones enviadas por el dispositivo móvil como resultado de la comparación de imágenes, y se encargará de ejecutar estas instrucciones a través del brazo-robot desplazando la pinza hacia una posición de destino definida mediante un programa guardado en lenguaje ACL que almacena un conjunto de instrucciones y posiciones guardadas registradas por el usuario a través de la botonera de enseñanza (Teach Pendant).

Para lograr tomar un objeto seleccionado y moverlo hacia una posición determinada debemos conocer los comandos del lenguaje de programación ACL que se requerirán para el correcto funcionamiento del robot, cuáles serán necesarios para lograr mover el robot, y como guardar posiciones e instrucciones a través de la

botonera de enseñanza o por medio del terminal ATS que serán necesarias para lograr mover el brazo robótico por medio de la aplicación desarrollada. Este movimiento del brazo robótico ocurre como consecuencia a partir de la integración entre la fase correspondiente al análisis de la imagen que define el porcentaje de similitud del patrón de un objeto capturado por la cámara de un dispositivo móvil ubicado sobre el brazo robot con respecto a un objeto de referencia, y el controlador del Scorbot quien recibe el nombre del programa almacenado a ser ejecutado por el brazo robot.

Este análisis dejó como consecuencia el establecer los movimientos que debe realizar el Scorbot, por lo que vamos a tener que identificar y seleccionar el conjunto de pasos a seguir, estableciendo un orden en la ejecución de comandos ACL para que el robot se pueda desplazar.

También se debió ver la forma de comunicar el Scorbot con la aplicación ya que la interacción va a estar a cargo de este sistema a través de una interfaz de comunicación entre el dispositivo móvil y el controlador del robot. En la interacción directa con el brazo robot fue utilizado el terminal ATS y la botonera de aprendizaje en el ámbito de la programación del robot, En el ámbito de las pruebas de la aplicación se interactuó con el robot de forma indirecta solo a través de comandos en lenguaje ACL entre un dispositivo móvil y el controlador del robot.

Para el grabado de posiciones se utilizó la botonera de enseñanza guardando las posiciones del robot deseadas por el usuario en la memoria del controlador. Para crear un programa correspondiente a un conjunto de posiciones y comandos en ACL se utilizaron las aplicaciones HyperTerminal, para Windows y el terminal ATS, para dar mayor facilidad en la edición de cada programa a ejecutar por el robot.

7.3 Interfaz de comunicación del robot.

Como fue necesario interactuar directamente con el Scorbot a través de la aplicación móvil que fue desarrollada, se observó primero que este se conecta al computador directamente al puerto serial con el conector RS-232 (el cual es una interfaz que se designa para el intercambio de datos binarios). La comunicación que se establece con el robot es bit a bit, es decir permite enviar un bit a la vez, al contrario de dispositivos de transmisión más modernos que envían varios bits simultáneamente.

Para este proyecto fue necesario comunicarse con el robot a través de teléfonos móviles de manera inalámbrica. Para tal objetivo se debió implementar una interfaz de comunicación que pueda enviar comandos desde el teléfono hacia el puerto serial del Scorbot de manera inalámbrica a través de Bluetooth. Por lo tanto, fue necesario buscar un lenguaje de programación, un dispositivo Bluetooth, y un microcontrolador, para implementar la interfaz de comunicación entre el teléfono y el puerto serial de la celda flexible y permitir integrar esta comunicación en las demás etapas del proyecto.

Luego de tener presente estos requisitos se optó realizar la conexión del teléfono con el puerto serial del Scorbot mediante un conversor que corresponde a un microcontrolador Arduino Mega, un dispositivo Bluetooth HC-05 y un conector para el puerto serial RS232. La razón de esta elección de implementar esta interfaz de comunicación es su bajo costo económico, además de ya estar disponible para ser utilizado en el Laboratorio al momento de desarrollar este proyecto por lo que para su desarrollo el conversor utilizado lo consideramos como conjunto de caja negra.

Esta interfaz permite recibir comandos y líneas de texto del teléfono y enviarlos en forma de bits hacia el puerto serial del controlador del Scorbot. El HC-05 recibe el texto los envía al microcontrolador Arduino para posteriormente procesar el texto y ser enviado al robot a través del puerto serial. Luego que el Scorbot recibe las ordenes, este ejecuta la acción y envía una respuesta al puerto serial para ser recibida por el microcontrolador Arduino y esta se encarga de enviarla al teléfono mediante el módulo HC-05.

Para el correcto funcionamiento de la comunicación los dispositivos (teléfono y el módulo HC-05) deben estar correctamente sincronizados.

Como el conversor Bluetooth-serial que utilizaremos como interfaz de comunicación entre el dispositivo móvil y el Scorbot ya está disponible en el laboratorio, solo fue necesario crear la comunicación entre el dispositivo y el módulo Bluetooth HC-05 del conversor.

A continuación, en la Figura 33 se muestra el circuito y los dispositivos utilizados de la interfaz de comunicación entre el dispositivo Android y el controlador del Scorbot a través del conversor serial-bluetooth.

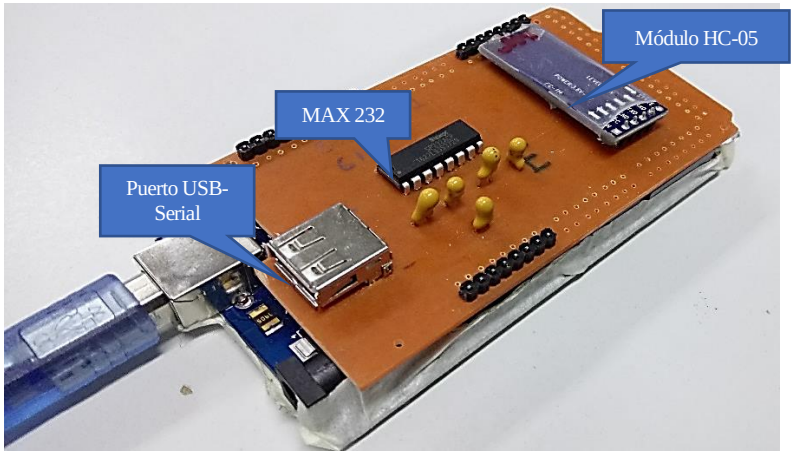


Figura 33. Conversor Bluetooth-Serial para la interfaz de comunicación entre el Smartphone y el Scorbot.

7.4 Programación del Scorbot.

En el ámbito de la programación del robot este proceso fue ejecutado con el apoyo del personal del laboratorio que fue el encargado de establecer el conjunto de posiciones a la que se debería ubicar la pinza del robot para ejecutar el proceso de inspección visual de cada objeto, revisar y llevar el objeto a un lugar determinado según el criterio de evaluación resultante en la etapa del procesamiento de imágenes.

7.4.1 Inicialización del robot.

La posición de los ejes es medida por codificadores (encoders) que registran la variación de movimiento referida a una posición inicial. Para que el robot tenga una buena performance, esta posición deberá ser la misma cada vez que el robot sea utilizado. Para ello el controlador tiene un programa interno llamado "home", el cual deberá ser ejecutado cada vez que se ponga en marcha el robot.
Para ejecutar el "homing" desde la PC escribir:
>home <enter>

Para ejecutar el "homing" desde el teach pendant (TP) oprimir:
run – 0 - enter
Para inicializar los ejes (ya sea del robot o de los periféricos) escribir:
>home <nº de eje>

7.4.2 Grabado de posiciones.

Para el grabado de posiciones se utilizó la botonera de enseñanza y un computador conectado al robot mediante el programa ATS del Scorbot estableciendo dos vectores de posiciones: CIM[ ] y CIMB[ ] que vienen integrados por defecto en la memoria controlador del robot.

Las acciones realizadas para grabar una posición en el vector de posiciones fueron:

- Mover la pinza con la botonera de aprendizaje en coordenadas cartesianas y dejar la pinza en una determinada posición.
- Grabar la posición mediante:
  - Botonera de aprendizaje:

    Record Position + **<N° posición del vector>** + **Enter**.

  - Terminal ATS:
    - En coordenadas Joint:

      HERE CIM[*posición del vector***]** . (Para ejes del grupo A).

      HERE CIMB[*posición del vector***]** . (Para ejes del grupo B)

    - En coordenadas cartesianas:

      HEREC CIM[*posición del vector***]** . (Para ejes del grupo A).

      HEREC CIMB[*posición del vector***]** (Para ejes del grupo B)

Para visualizar las posiciones guardadas en el controlador se utiliza el siguiente comando: **Listpv <Enter>** y así la lista de posiciones ya definidas aparecerán en pantalla.

Para ver las coordenadas correspondientes a una posición guardada use la orden LISTPV de ACL más el número o posición del vector.

Por ejemplo:

LISTPV CIM[10] + <Enter>

Las coordenadas de la posición aparecen en la pantalla de la siguiente manera:

1:17632	**2:10004**	**3:6131**	**4:1030**	**5:-734**
X: -100	**Y: 3837**	**Z: 2159**	**P:-952**	**R:-54**

Se ven dos grupos de cantidades para las posiciones del robot:

1) El primero muestra las coordenadas de ejes, definidas en cuentas de codificador.

2) El segundo muestra las coordenadas cartesianas (XYZ), definidas en décimas de milímetros. Por ejemplo:

   Z: 2159 → Z = 215.9 mm

   P (pitch, o inclinación de la pinza) y R (roll, o giro) están definidas en décimas de grados. Por ejemplo:

   P: -952 → P = -95,2°

En este proyecto como se explicó anteriormente se definieron inicialmente tres posiciones fundamentales: la posición de captura de una imagen, la posición para los objetos evalúalos positivamente y la posición

para los objetos descartados. Las posiciones inicialmente utilizadas en el proyecto se muestran en la Tabla 12:

	Coordenadas de eje (cuentas de codificador (encoder))					Coordenadas cartesianas (mm*10)			Inclinación / giro de la pinza (x 10 grados)	
Posición	1	2	3	4	5	X	Y	Z	P	R
CIM[1]	17998	11523	-8653	-1130	678	-248	3485	1042	-960	-82
CIM[14]	17531	8909	-8267	-1020	797	-43	3144	1890	-962	-40
CIM[15]	17533	11950	-10015	-1023	794	-44	3144	652	-962	-42
CIM[16]	17501	11635	-4815	1357	3175	-42	4221	2071	-962	812
CIM[17]	17500	14197	-5618	1357	3175	-42	4222	956	-962	812
CIM[18]	-8487	7763	-9645	-8404	-6714	-115	-2539	1874	-939	-2710
CIM[301]	-6101	14049	-12554	-9031	-7157	581	-22293	-286	-972	-2902
CIM[311]	-6197	6651	-8879	-9071	-7210	554	-2299	2232	-969	-2918
CIM[321]	-9256	14035	-12742	-8387	-6611	-327	-2323	-290	-954	-2688
CIM[331]	-9257	6570	-8995	-8387	-6611	-327	-2325	2217	-954	-2688

Tabla 12. Conjunto de posiciones y coordenadas del robot utilizadas en la demostración del proyecto.

	Coordenadas de ejes (Grupo B)	
Posición	7	8
CIMB[1]	4901	0

La posición de captura será sobre el pallet ubicado en la cinta transportadora en la Estación 1 (**CIM[14]**).
La posición establecida para objetos aceptados será sobre el Rack 1 de la Estación 1 (**CIM[301]**).
La posición establecida para objetos descartados será sobre el Rack 2 de la Estación 1 (**CIM[321]**).

7.4.3 Creación y edición de programas en ACL.

A continuación, se definieron los tres programas mínimos necesarios en ACL para la ejecución de la etapa de inspección visual y el control de calidad.

Programa 1: CAM, Posición de captura de imagen del robot.

```
OPEN
MOVED CIM[14]
SPEED 10
MOVELD CIM[15]
PRINTLN “OK1”
PRINTLN
DELAY 200
MOVELD CIM[14]
```

Programa 2: GUARD, Aceptar y guardar al Rack 1.

```
OPEN
MOVED CIM[16]
SPEED 10
MOVELD COM[17]
CLOSE
MOVELD CIM[16]
PRINTLN “OBJETO_TOMADO”
PRINTLN
MOVED CIM[18]
PRINTLN “OKG”
```

Programa 3: DESC, Descartar al Rack 2.

```
OPEN
MOVED CIM[16]
SPEED 10
MOVELD COM[17]
CLOSE
MOVELD CIM[16]
PRINTLN “OBJETO_TOMADO”
PRINTLN
MOVED CIM[18]
PRINTLN “OKD”
PRINTLN
MOVED CIM[331]
MOVED CIM[321]
OPEN
MOVED CIM[331]
MOVE 0
```

PRINTLN *MOVED CIM[311]* *MOVED CIM[301]* *OPEN* *MOVE 0*	

Comandos ACL utilizados.

Comandos de control de ejes y posiciones grabadas	
OPEN	Abrir pinza
CLOSE	Cerrar pinza
MOVE *pos*	El robot se mueve a la posición grabada en *pos*.
MOVEL *pos*	La pinza del robot se mueve en línea recta a la posición *pos*.
MOVED *pos*	El robot se mueve a la posición grabada en *pos* y hasta que no llega hasta dicha posición no continúa con el programa.
MOVELD *pos*	La pinza del robot se mueve desde la posición actual hasta la posición *pos* en línea recta, siempre que esto sea posible y no continúa con el programa mientras no se haya completado el movimiento.
HERE *pos*	Graba en la posición *pos* definida anteriormente las coordenadas joint.
HEREC *pos*	Graba en la posición *pos* definida anteriormente, las coordenadas cartesianas.
SPEED *vel*	Establece la velocidad para los ejes del Grupo A. La velocidad se especifica en porcentajes. La velocidad máxima es 100 y la mínima es 1, la velocidad por defecto es 50, para efectos del proyecto fijamos la velocidad en 40%
SPEED *vel*	Establece la velocidad para los ejes del Grupo B. La velocidad se especifica en porcentajes. La velocidad máxima es 100 y la mínima es 1, la velocidad por defecto es 50, para efectos del proyecto fijamos la velocidad en 40%
Comandos de manipulación de programas	
EDIT *prog*	Activa el modo *edit* y llama al programa *prog* y permite modificar su contenido, sino existe el programa, es creado.
REMOVE prog	Borra el programa *prog* y libera la memoria ocupada por éste.
EMPTY prog	Borra las líneas del programa *prog* manteniéndolo existente y valido.
EXIT	Sale del modo *edit.*
Comandos de control de programas	
RUN *prog*	Ejecuta el programa *prog*.
A *prog*	Aborta la ejecución del programa *prog.*
STOP	Aborta la ejecución de todos los programas que estén corriendo.
DELAY *var*	Suspende la ejecución del programa durante el tiempo (en centésimas de segundo) especificado en *var*.
PRINTLN "*text*"	Envía un texto "*text*" más un salto de línea a través del puerto serial. Si no hay texto solo envía el salto de línea.
Comandos de reporte	
DIR	Proporciona la lista de programas guardados en memoria.
LIST *prog*	Muestra el listado de instrucciones que componen el programa *prog.*
LISTP	Muestra una lista de todas las posiciones definidas.
LISTPV *pos*	Muestra en pantalla las coordinadas correspondientes a la posición *pos.*

Tabla 13. Comandos ACL utilizados para el control del robot en el proyecto.

En esta etapa se debieron determinar cuáles fueron los comandos del Lenguaje ACL que fueron utilizados para la programación y operación del robot, recordando que estos comandos deben cumplir con el objetivo de poder mover el brazo robot, por lo que estas secuencias de comandos deben estar programadas tanto en la aplicación desarrollada como en los programas que se definirán en la memoria del robot, por ello es importante definirlos.

Estas fueron las operaciones básicas realizadas en el ámbito de la programación y el control del Scorbot necesarias para cumplir con los objetivos del sistema y del proyecto.

Se espera que se halla comprendido básicamente el manejo del brazo robótico a partir de su lenguaje de programación ACL.

8 ANÁLISIS.

El objetivo de este capítulo es describir las características del sistema en el ámbito de los procesos que realiza, el flujo de datos e información entre los procesos, y de cómo el sistema operará desde el punto de vista del usuario.

8.1 Proceso de Negocios Futuros.

El objetivo de esta parte del análisis es describir los procesos de negocios futuros como punto de partida de este análisis mediante diagramas de actividad. A continuación se describen los procesos fundamentales que servirán para el control de calidad de figuras y objetos, en la cual se dividirán en dos partes fundamentales: la primera parte consiste en registrar figuras y configuraciones de segmentación de una región de interés (ROI) proceso descrito en la Sección 8.1.1 ,y la segunda parte consiste en el proceso de ejecución del proceso de inspección de figuras y objetos y el control de calidad mediante visión artificial y el uso celda flexible Scorbot para el traslado del Smartphone y de los objetos procesados, proceso descrito en la Sección 8.1.2.

8.1.1 Proceso: Guardar figura y parámetros de configuración de región de interés ROI.

Este proceso (Figura 34) se relaciona con el registro de imágenes y parámetros de segmentación de una imagen capturada por la cámara de un Smartphone, comenzando por la comunicación entre dos Smartphone continuando por la captura de imagen y la búsqueda de una región de interés por medio de segmentación, binarización y corrección de perspectiva. Al haber cumplido estos pasos, se podrá guardar una imagen binaria con una figura determinada y sus datos o guardar la configuración de segmentación para su posterior uso en algún proceso indicado en la Sección 8.1.2.

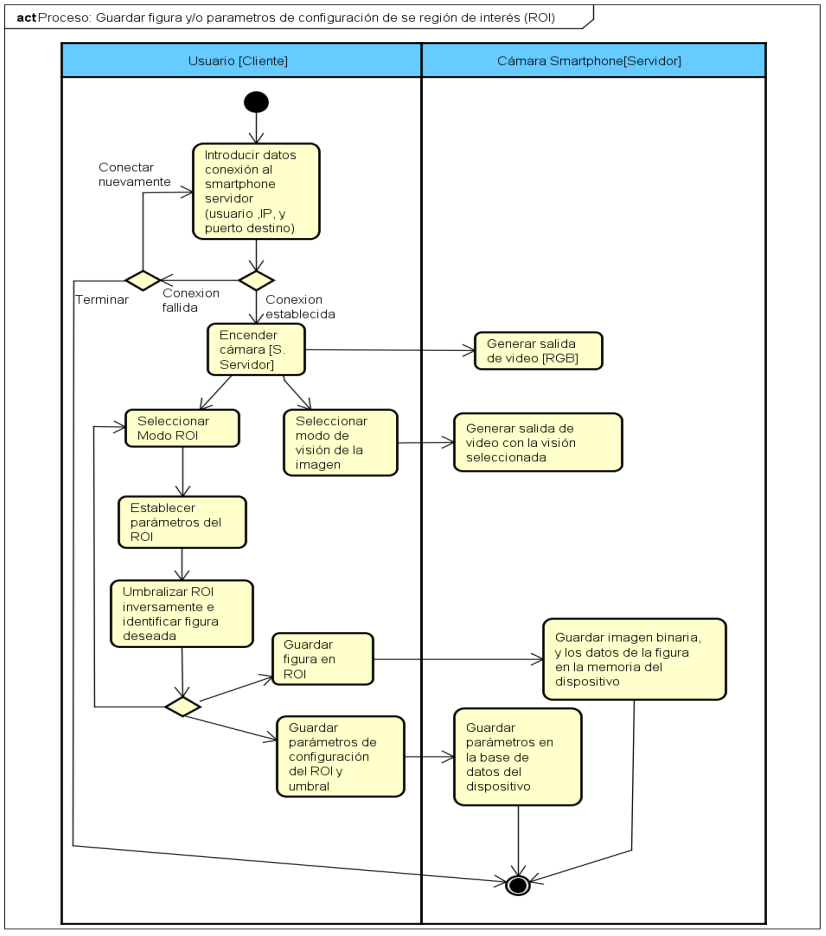


Figura 34. Proceso: Guardar figura y parámetros de configuración de región de interés ROI.

8.1.2 Proceso: Inspeccionar calidad de figuras y objetos mediante el control de una celda flexible.

El modelo explicado en la Figura 35 define el proceso de control de la celda de inspección y comparación de figuras, solicitando una orden de inspección de un objeto hacia el Smartphone cámara indicando el nombre de la figura a comparar, el conjunto de programas a ejecutar por el brazo robot y opcionalmente los parámetros de segmentación registrado en el proceso de la Sección 8.1.1. Obtenida esa orden el Smartphone envía la orden de ejecución de programa registrado en el robot a moverse a una determinada posición de captura donde se ubica el objeto de interés, luego se captura la región de interés del objeto, se umbraliza inversamente para luego comparar la imagen binaria capturada con la imagen de referencia señalada en la orden de inspección, calculando el porcentaje de coincidencias, y encontrar una figura similar a la figura capturada de manera invariante, para finalmente mover el robot para recoger el objeto y moverlo a una posición según sea el resultado de similitud entre los objetos evaluados, finalmente se envía el resultado de la comparación e inspección al Smartphone que solicita la orden de comparación.

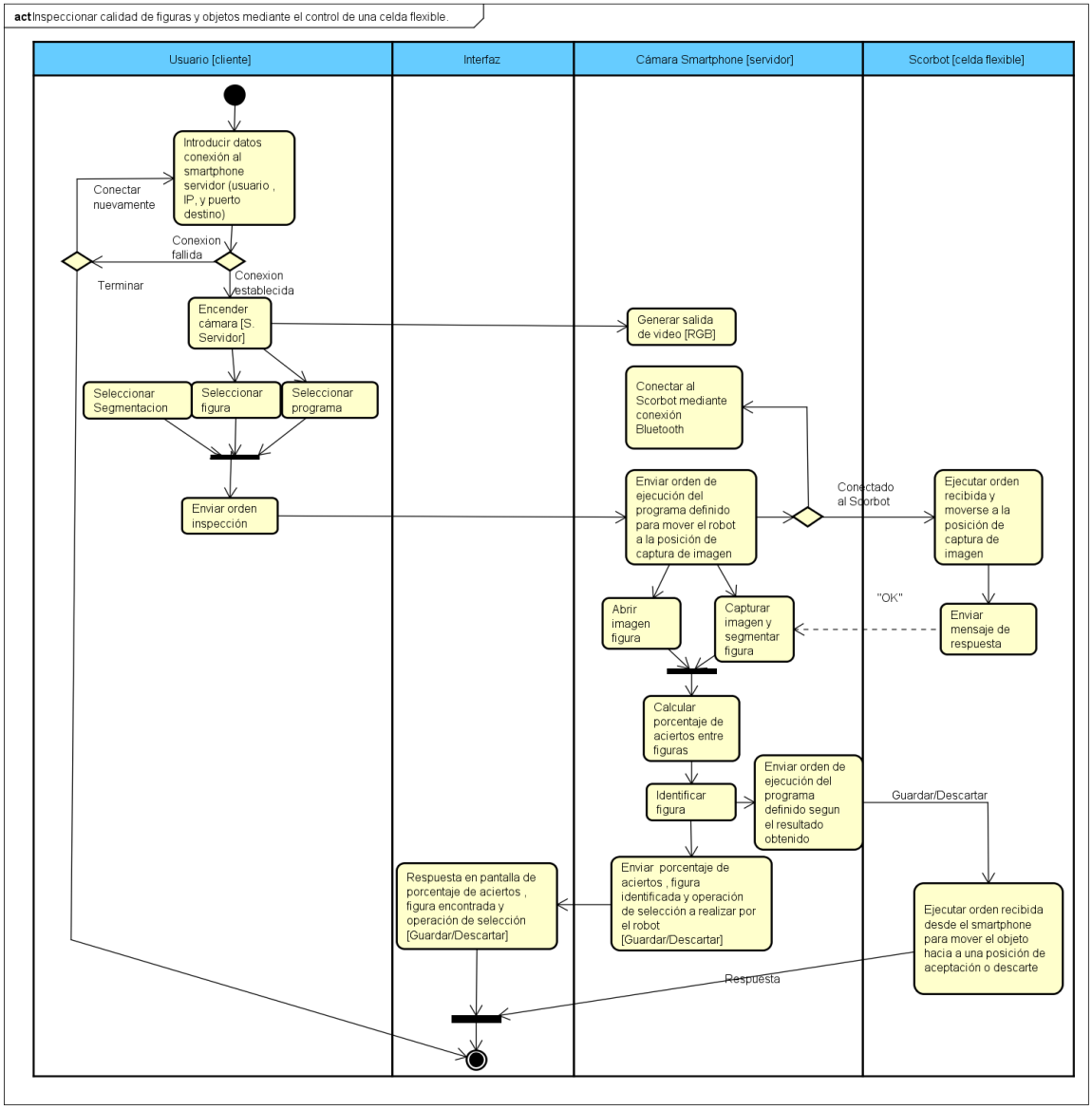


Figura 35. Proceso: Inspeccionar calidad de figuras y objetos mediante el control de una celda flexible.
Nota: Un programa se define como una secuencia de movimientos del robot según su operación, y el porcentaje de aciertos mínimo de aprobación de un objeto.

8.2 Diagrama de Flujo de Datos (DFD).

Los Diagramas de Flujo de Datos (DFD) buscan modelar un sistema desde el punto de vista de la información, en el cual se estudia cómo se usan los datos que corresponden a las entradas, en salidas que vendría a ser la información, por lo tanto, el propósito para un Diagrama de Flujo de Datos es mostrar para un sistema o subsistema lo siguiente:

- Cuáles son los límites del sistema
- De dónde vienen los datos
- A dónde van los datos cuando dejan el sistema
- Dónde se almacenan los datos
- Qué procesos transforman los datos y las interacciones entre los procesos y los depósitos de datos.

Definición de elementos:

- Entidades: Representan a las fuentes o destino de los datos.
- Proceso: Representa a la función donde se transforman los datos.
- Flujo de información: Representa al movimiento de los datos, se compone de datos elementales.
- Almacén de datos: Repositorio de los datos procesados u utilizados por los procesos del sistema.

La notación DFD utilizada en este proyecto es la de Yourdan/Demarco, con esta breve presentación de los diagramas se especificarán por tres niveles: DFD nivel de contexto, DFD nivel superior y DFD nivel de detalle.

La Figura 36 muestra un breve ejemplo de los elementos y su interacción con los demás componentes del diagrama de flujo de datos DFD.

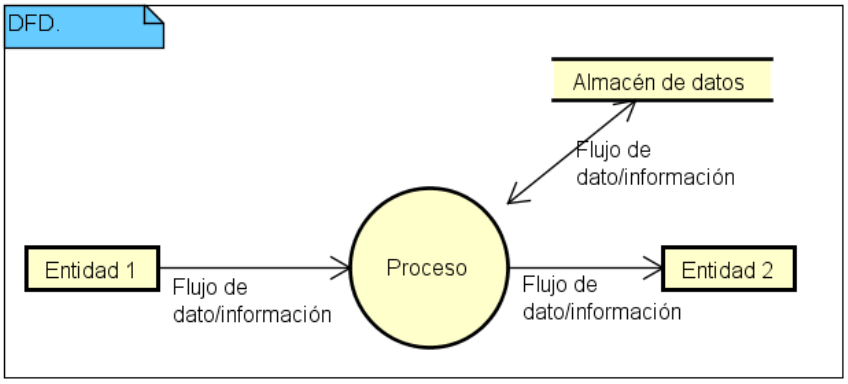


Figura 36. Descripción de componentes del diagrama DFD.

8.2.1 DFD Nivel de contexto del sistema.

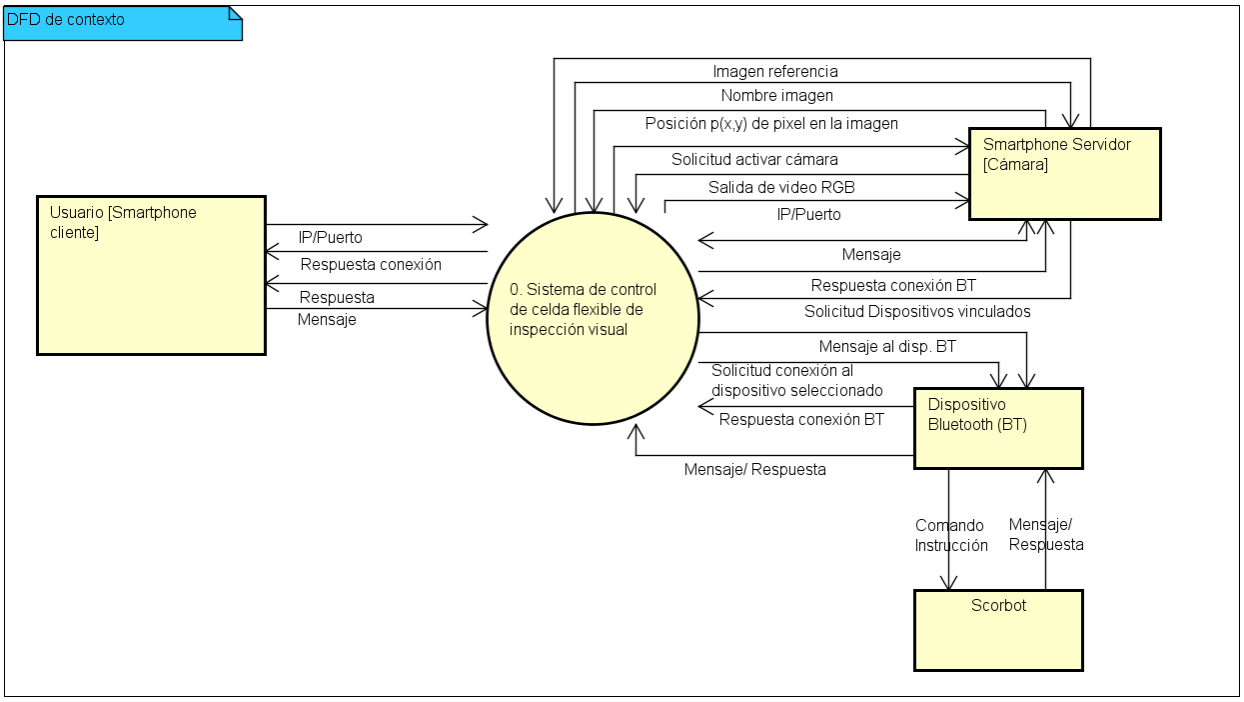


Figura 37. DFD Nivel de contexto del sistema.

8.2.2 DFD Nivel superior del sistema.

El diagrama de flujo de datos de nivel superior (Figura 38) separa el proceso principal del sistema en cinco funciones conectadas entre sí, dividiendo las etapas del proceso principal para definir el traspaso de datos entre estos procesos.

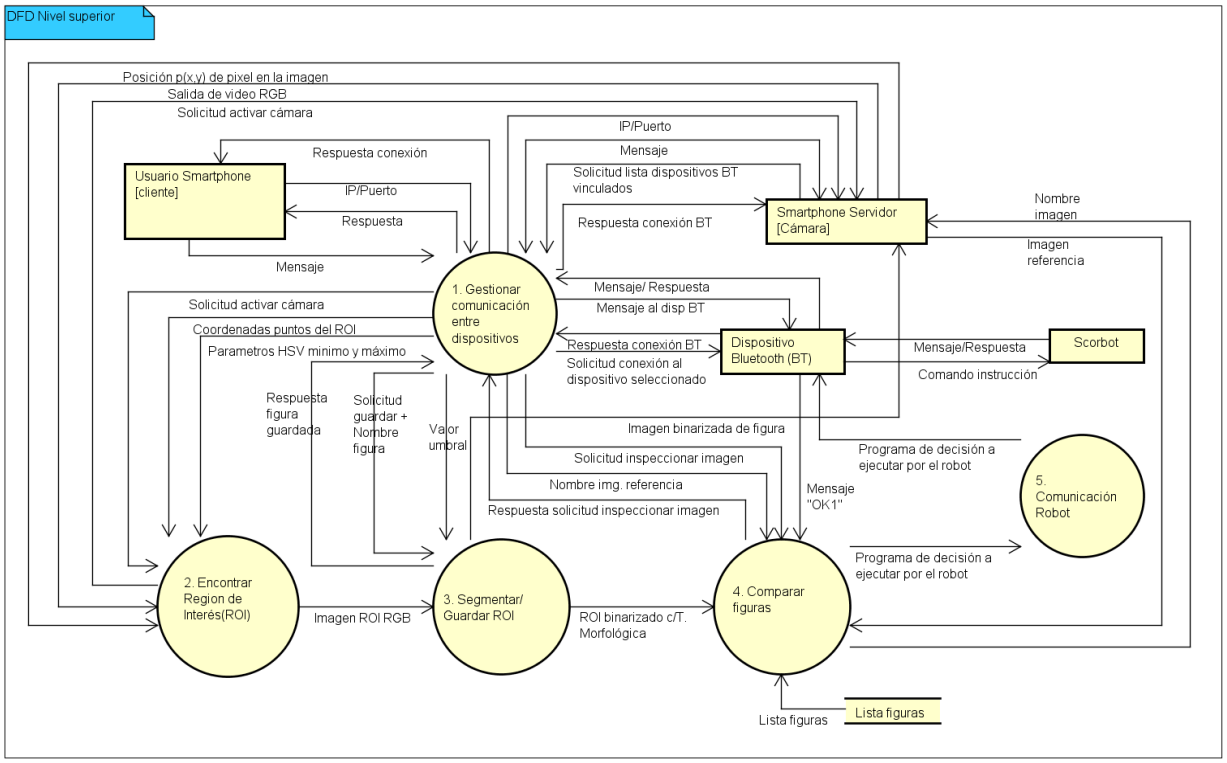


Figura 38. DFD de nivel superior del sistema.

Los cinco procesos de nivel superior son los siguientes:

1) Gestionar comunicación entre dispositivos: El objetivo es establecer la comunicación entre dos Smartphones y el robot mediante comunicación Bluetooth-Serial.

2) Encontrar región de interés: El objetivo es buscar la región dentro del campo de imagen de la cámara en donde se encuentra un determinado objeto, además de umbralizar esta región. La principal salida es una imagen (*Mat*).

3) Segmentar/Guardar ROI: El objetivo es guardar la imagen binarizada contenida la región de interés y sus datos, y opcionalmente permitir guardar la configuración de segmentación con la que se encontró el ROI. Su principal entrada es un objeto de imagen (*Mat*), el nombre de la imagen y/o las variables de segmentación. Su salida es un registro de segmentación en la base de datos y/o una imagen binaria.

4) Comparar figuras: Es uno de los procesos principales dentro del control del robot en el que su objetivo es comparar dos figuras binarias y calcular su resultado de similitud y encontrar la figura más semejante de manera invariante. Sus principales salidas son la comunicación de los resultados de comparación y las ordenes de movimiento a ejecutar por el robot.

5) Comunicación robot: Su función es enviar y recibir información desde y hacia el robot, ejecutando comandos enviados desde un Smartphone y enviando la respuesta de la ejecución a éste.

Estos procesos serán descritos mediante los DFD de nivel de detalle en la Sección 8.2.3.

8.2.3 DFD Nivel de detalle del sistema.

8.2.3.1 DFD Proceso 1: Gestionar comunicación entre dispositivos.

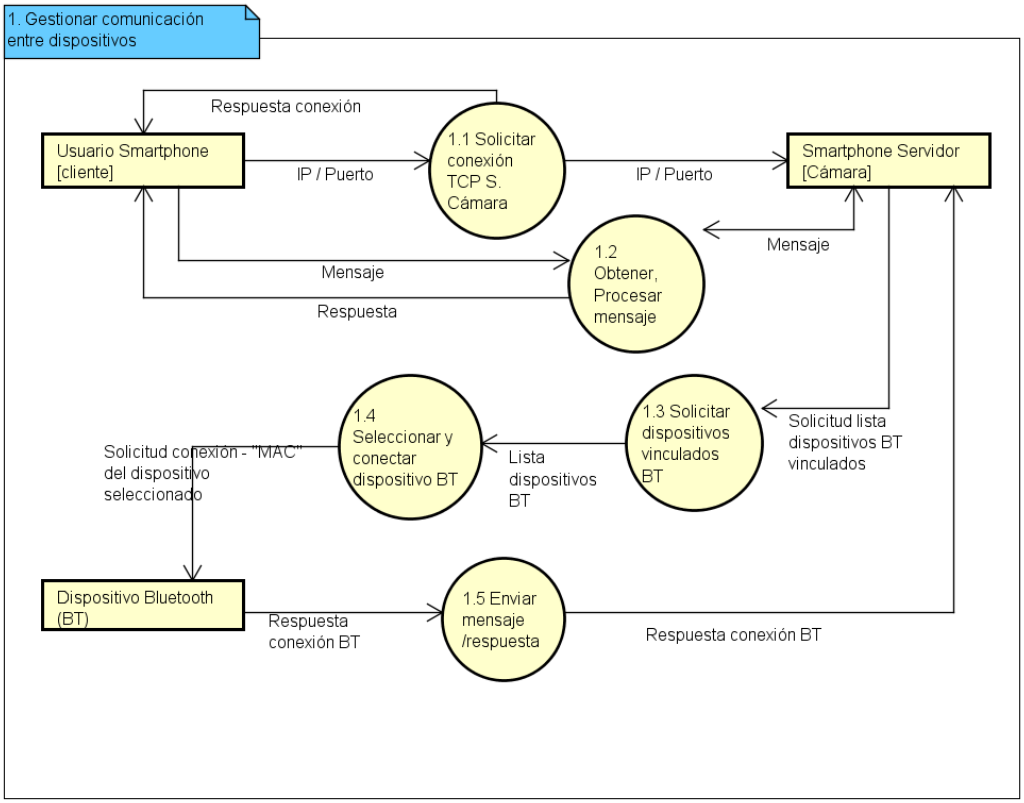


Figura 39. DFD Proceso 1: Gestionar comunicación entre dispositivos.

8.2.3.2 DFD Proceso 2: Encontrar región de interés.

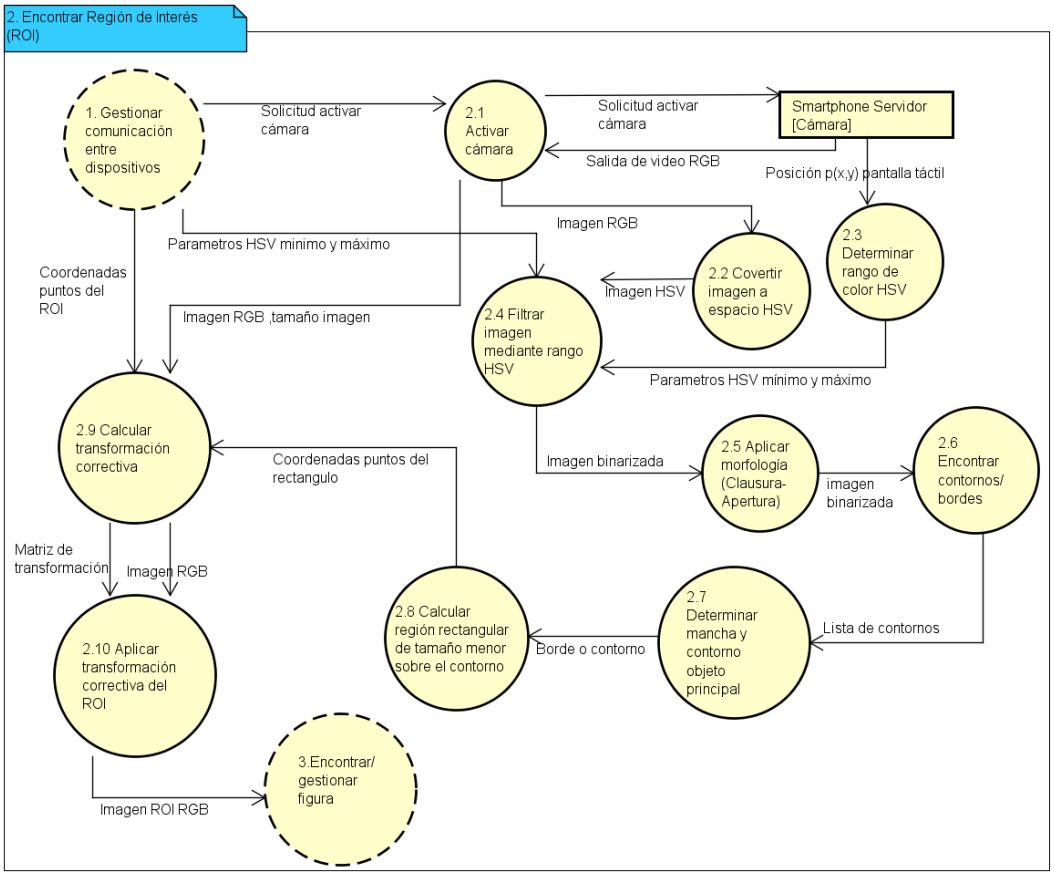


Figura 40. DFD Proceso 2: Encontrar región de interés.

8.2.3.3 DFD Proceso 3: Segmentar / Guardar región de interés.

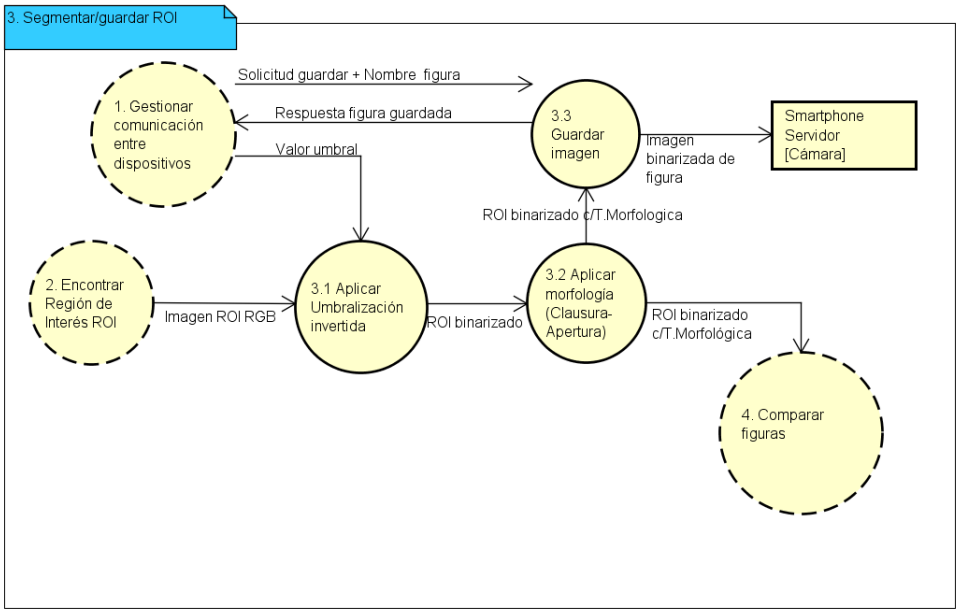


Figura 41. DFD Proceso 3: Segmentar / Guardar región de interés.

8.2.3.4 DFD Proceso 4: Comparar figuras.

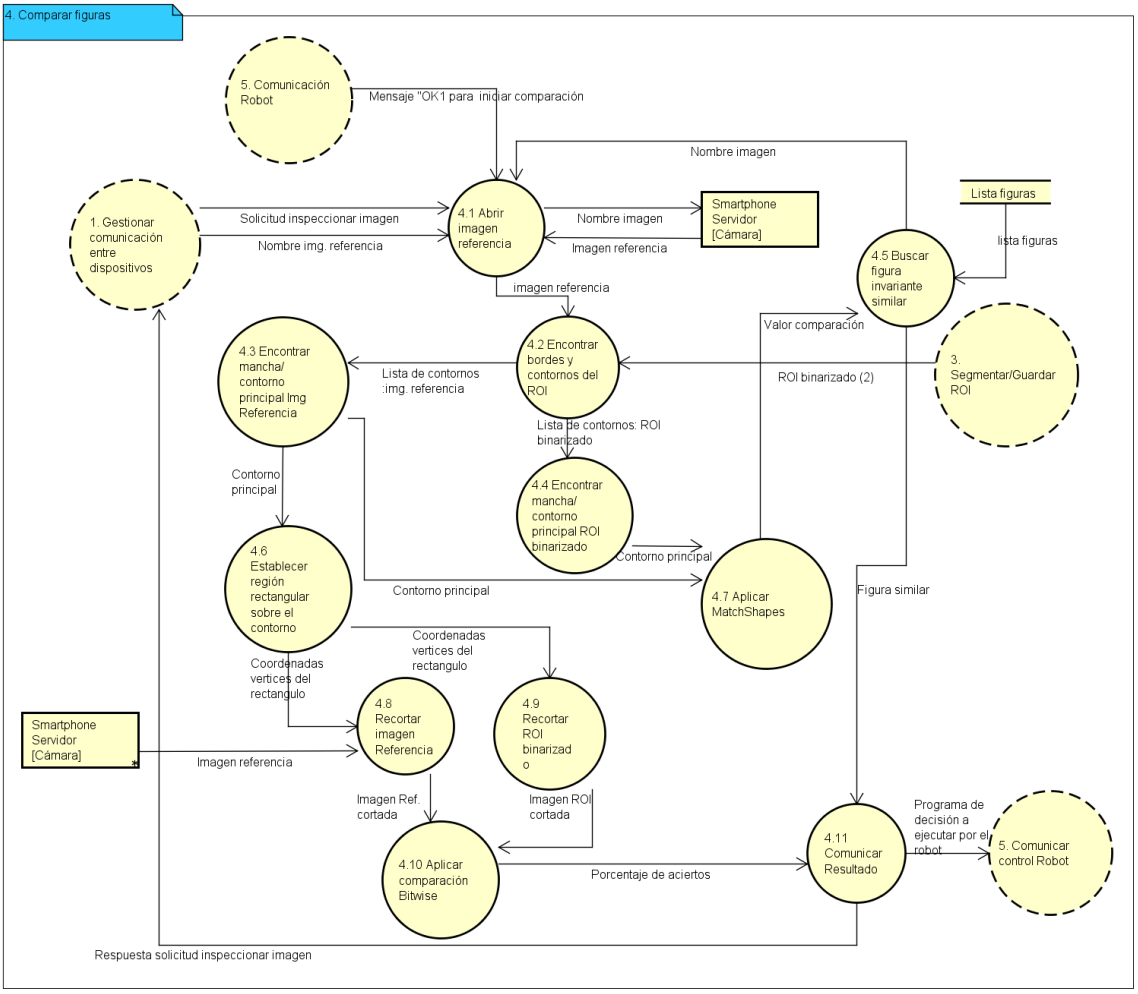


Figura 42. DFD Proceso 4: Comparar figuras.

8.2.3.5 Comunicación robot.

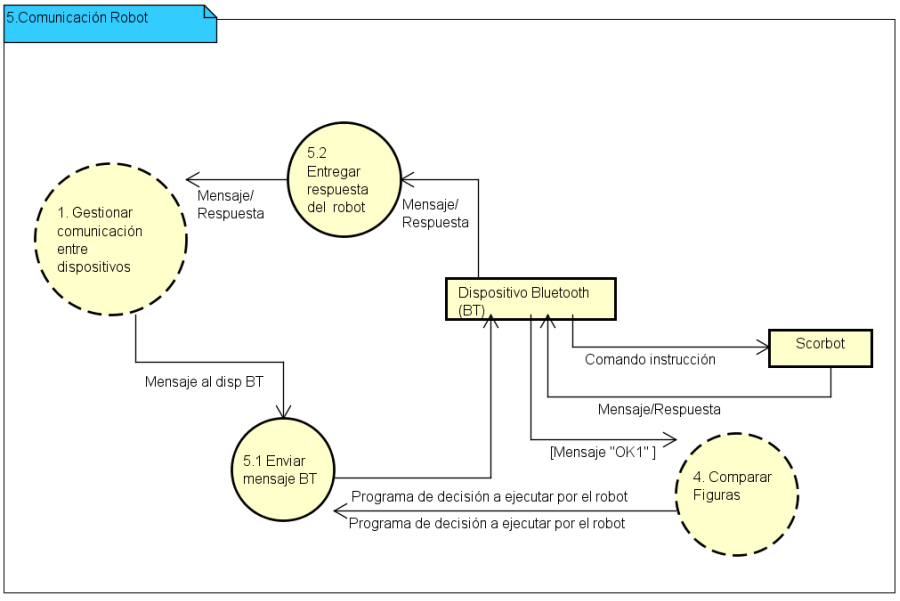


Figura 43. DFD Proceso 5: Comunicación robot.

8.3 Casos de Uso (CU) del sistema.

Los Casos de Uso (CU) son descripciones narrativas que describen las características y el comportamiento del sistema desde el punto de vista del usuario. Ayudan a tener un mayor conocimiento sobre los requerimientos del sistema, a definir los límites y las relaciones entre el sistema y el entorno. El formato utilizado para describir los casos de uso consta de los siguientes elementos (Larman, 2003).

- Caso de Uso: Nombre del Caso de Uso.
- Actores: Agentes externos al sistema o aplicación.
- Objetivo: Objetivo el Caso de Uso.
- Precondiciones: Acciones necesarias para ejecución del CU.
- Postcondiciones: Acciones que ocurren luego de la ejecución del CU.
- Flujo principal: Descripción en detalle de la ejecución normal del CU.
- Flujo alternativo: Descripción de otras formas de ejecución del CU.
- Otros: Requisitos especiales, variaciones de datos, factores tecnológicos, etc.

8.3.1 Actores.

- Usuario cliente (operador): Es el encargado de operar la aplicación desde el módulo control de la aplicación desde el punto de vista de la gestión de las conexiones entre los dispositivos, del control de la cámara del Smartphone (servidor) y de gestionar el proceso de inspección de objetos y control del robot Scorbot.

- Scorbot (robot): Corresponde al brazo robot, encargado de ejecutar las instrucciones enviadas por el usuario cliente o por la aplicación de acuerdo a los resultados obtenidos en la ejecución de cada proceso del sistema. Una de las funciones es mover el Smartphone (servidor) a la posición de captura de una imagen de un objeto específico ubicado en el PLC del sistema de manufactura, y la otra función es de trasladar objetos entre distintos puntos. Este actor lo consideramos como ficticio ya que no es un usuario que interactúa directamente con el sistema, sino como un dispositivo actuador externo, por lo que los casos de uso involucrados directamente con el robot no fueron especificados, pero si señaladas para una mejor descripción del sistema.

8.3.2 Diagrama de Casos de Uso.

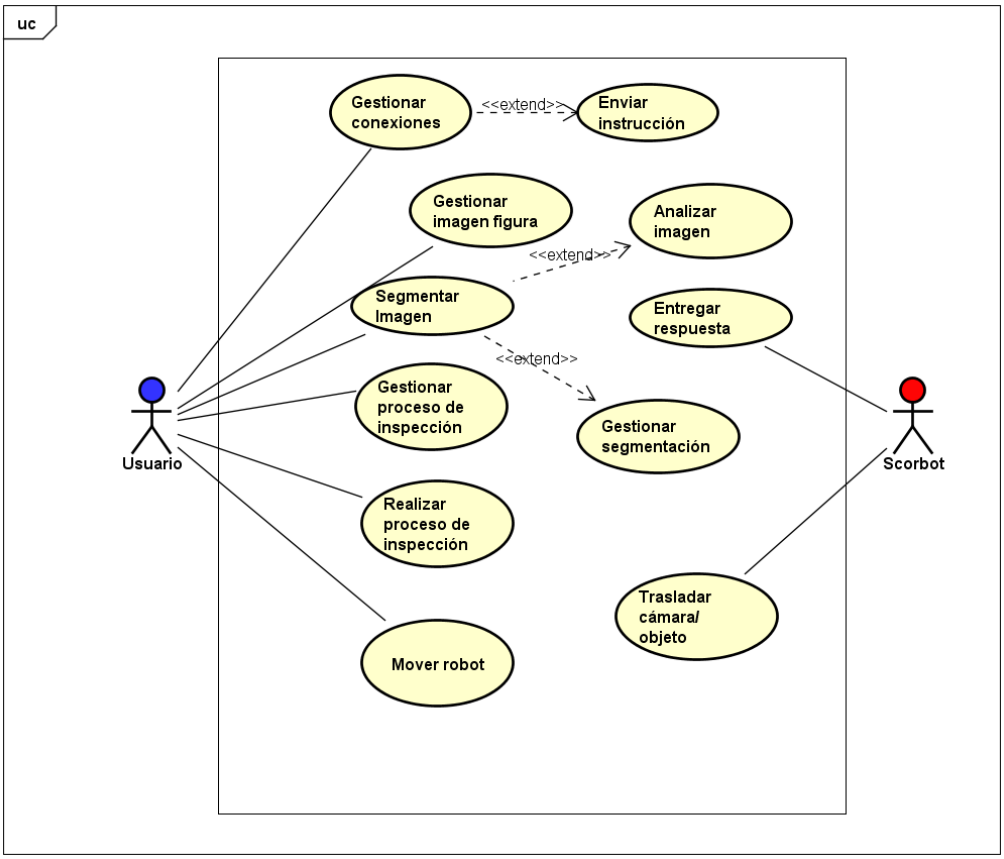


Figura 44. Diagrama de casos de uso.

8.3.3 Especificación de los Casos de Uso.

En este apartado de especificarán los Casos de Uso más relevantes para el usuario, para describir como el principal actor (Usuario) actúa con el sistema para cumplir con los objetivos del proceso de inspección.

8.3.3.1 Caso de Uso: Gestionar conexiones entre los dispositivos.

ID	CU01
Nombre	Gestionar conexiones entre los dispositivos.
Actores	• Usuario
Objetivo	Establecer la conexión entre el Smartphone Cliente (modo administrador) y el Smartphone Servidor (modo cámara), y éste al robot.
Referencia	Requerimientos funcionales: RF1.02, RF1.03. (Tabla 1)
Precondiciones	Para la conexión entre ambos Smartphone se debe cumplir que: • Ambos deben estar conectados a la misma red WIFI. • Un Smartphone debe tener la aplicación en modo Administrador y el otro en modo Cámara. Para la conexión entre el entre el Smartphone Servidor (modo cámara) y el robot se debe cumplir que: • El Smartphone debe tener habilitado el Bluetooth. • El conversor Bluetooth-Serial debe estar encendido y conectado al robot mediante el conector serie. • El Smartphone debe estar vinculado al dispositivo Bluetooth del conversor Bluetooth-Serial.
Postcondiciones	Conexión entre ambos Smartphone y el robot establecidos. Existe comunicación indirecta entre el Smartphone Cliente y el robot.

Flujo Principal:

Acción del actor	Respuesta del sistema
1.- En el módulo (Administrador) el usuario ingresa un nombre de usuario, la dirección IP del Smartphone, y el puerto habilitado para la aplicación en modo cámara.	2.- Se recibe un mensaje de respuesta de que el Smartphone cliente está conectado al Smartphone servidor. El estado de conexión de la aplicación cliente cambia a "conectado al "IP:Puerto"
3.- En el módulo Smartphone (Cámara) el usuario selecciona la opción Bluetooth del menú.	4.- Muestra una lista con los dispositivos Bluetooth vinculados con el nombre del dispositivo y su dirección física (MAC).
5.- Selecciona uno de los dispositivos Bluetooth vinculados al Smartphone.	6.- El estado de conexión Bluetooth cambia de estado "Conectado al <Nombre del dispositivo Bluetooth>

Flujo Alternativo:

Acción del actor	Respuesta del sistema
1.- En el módulo (Administrador) el usuario ingresa un nombre de usuario, la dirección IP del Smartphone, y el puerto habilitado para la aplicación en modo cámara.	2.-No se recibe el mensaje de respuesta alguno de parte del Smartphone servidor. Aparecerá un mensaje de "Error de conexión" y el estado de conexión de la aplicación cliente cambia a "Sin conexión"
5.- Selecciona uno de los dispositivos Bluetooth vinculados al Smartphone.	6.-No se puede conectar al dispositivo Bluetooth, por razones como Bluetooth no habilitado o ya estar conectado a otro dispositivo. El estado de conexión será "No conectado"

8.3.3.2 Caso de Uso: Segmentar imagen de cámara.

ID	CU02
Nombre	Segmentar imagen de cámara
Actores	• Usuario
Objetivo	Analizar la imagen obtenida por la cámara del Smartphone, Buscar una región de interés en que correspondiente al objeto a analizar, y su figura principal contenida en ese objeto.
Referencia	Requerimientos funcionales: RF2.05, RF2.06, RF2.09. (Tabla 1)
Precondiciones	Los Smartphone deben estar habilitados y conectados, uno en modo administrador y el otro en modo cámara.
Postcondiciones	El Smartphone muestra en pantalla la señal de video del proceso de segmentación llevado en curso. La pantalla del dispositivo mostrará la imagen desde las siguientes perspectivas: • Imagen BGR-RGB con la región de interés demarcada. • Imagen BGR-RGB contenida en la región de interés. • Imagen general binarizada mediante captura de color en espacio HSV. • Mascara binarizada contenida en la región de interés.

Flujo Principal:

Acción del actor	Respuesta del sistema
1.-Activa la cámara del Smartphone conectado.	2.-Señal de video en RGB del Smartphone (cámara) conectado.
3.-Selecciona el modo de segmentación, ya sea por toque en pantalla, por definición de puntos o segmentación manual por color	4.-Establece la región de interés inicial de acuerdo al modo seleccionado, y bajo los parámetros guardados en las preferencias de la aplicación.
5.-Referente al modo el usuario ejecuta una de estas acciones: • Selecciona las coordenadas de los cuatro puntos de la región de interés deseada. • Selecciona los parámetros mínimos y máximos del espacio de colores HSV. • Selecciona un punto específico de la imagen sobre la pantalla del Smartphone.	6.- Establece la región de interés ROI deseada por el usuario.
7- El usuario selecciona el modo de proyección al modo mascara binarizada del ROI	8.- Proyecta la imagen binaria contenida en el ROI.
9- Selecciona el valor de umbralización invertida del ROI para identificar la figura principal contenida en el ROI.	10.- Proyecta la figura principal contenida en el ROI.

Flujo Alternativo:

Acción del actor	Respuesta del sistema
1.-Activa la cámara del Smartphone conectado.	2.-No se activa la cámara el dispositivo, no existe conexión.
9- Selecciona el valor de umbralización invertida del ROI para identificar la figura principal contenida en el ROI.	10.-No existe mancha o figura contenida en el área de la región de interés, por lo que no se puede proyectar.
11.-El usuario guarda la configuración de segmentación actual ingresando un nombre para esta.	12.-El sistema registra la configuración se segmentación con los datos de segmentación en la base de datos de la aplicación en el Smartphone en modo cámara.

8.3.3.3 Caso de Uso: Gestionar imágenes-figuras.

ID	CU03
Nombre	Gestionar imágenes-figuras.
Actores	• Usuario
Objetivo	• Guardar imagen binarizada contenida en una región de interés en la memoria del dispositivo, correspondiente a un objeto y una figura, guardar sus datos descriptores de la figura como los momentos de Hu. • Eliminar las imágenes guardadas erróneas o no deseadas.
Referencia	Requerimientos funcionales: RF1.02, RF2.06, y RF2.07. (Tabla 1)
Precondiciones	Los Smartphone deben estar habilitados y conectados, uno en modo administrador y el otro en modo cámara. La cámara del Smartphone (cámara) debe estar activada en el caso de guardar una nueva imagen. Para un correcto registro de una figura u objeto deseado, El ROI debe estar identificado y bien segmentado.
Postcondiciones	Imagen binarizada de la región de interés con su figura principal identificada guardada en la memoria del dispositivo, más sus datos descriptores (Momentos invariantes de Hu).

Flujo Principal:	
Acción del actor	Respuesta del sistema
Proceso “Guardar”	
1.- Ingresa al módulo de segmentación y pulsa el botón “Guardar Imagen ROI”	2.- Habilita una entrada de texto para ingresar el nombre de la figura que se desea guardar.
3.- Ingresa un nombre para la imagen-figura que se desea guardar en el Smartphone, y pulsa el botón “Guardar”	4.- Se muestra un mensaje en la ventana de mensajes recibidos que indica que la figura ha sido guardada, y los siete descriptores (M .Hu) de la figura principal. Se deshabilita la entrada de texto para el nombre de la figura.
Proceso “Eliminar”	
5.- En el editor de texto de envío de mensajes el usuario escribe “delete_img ”+<nombre de la figura a eliminar> y pulsa el botón “Enviar”	6.- Se muestra un mensaje en la ventana de texto que indica que la figura ha sido eliminada.
Flujo Alternativo:	
Acción del actor	Respuesta del sistema
3.- Ingresa un nombre para la imagen-figura que se desea guardar en el Smartphone, y pulsa el botón “Guardar”	4.- Se muestra un mensaje en la ventana de mensajes recibidos que indica que la figura ya existe. Esto dice que ya existe una figura guardada con ese nombre. Se deshabilita la entrada de texto para el nombre de la figura.
5.- En el editor de texto de envío de mensajes el usuario escribe “delete_img ”+<nombre de la figura a eliminar> y pulsa el botón “Enviar”.	6.- En caso de que la figura no exista no habrá respuesta de parte del dispositivo.

8.3.3.4 Caso de Uso: Analizar imagen-figura.

ID	CU04
Nombre	Analizar imagen-figura
Actores	• Usuario
Objetivo	Calcular el porcentaje de aciertos entre la figura proyectada en la región de interés binarizada en la pantalla del Smartphone y una imagen binarizada almacenada en la memoria de éste. Encontrar alguna figura similar con respecto a la figura proyectada en el Smartphone, y de todas las figuras almacenadas en la memoria de este a través de descriptores invariantes.
Referencia	Requerimientos funcionales: RF2.08, RF3.11. (Tabla 1)
Precondiciones	• Debe existir imágenes-figuras almacenadas en la memoria del dispositivo. • Los Smartphone deben estar habilitados y conectados, uno en modo administrador y el otro en modo cámara. • La cámara del Smartphone (cámara) debe estar activada.
Postcondiciones	Se ha obtenido el resultado del porcentaje de aciertos o de coincidencia un objeto a comparar y otro proyectado en la cámara del Smartphone. Se ha encontrado el nombre de la figura más similar con respecto a la figura proyectada por la cámara del Smartphone.

Flujo Principal:

Acción del actor	Respuesta del sistema
1.-Actualiza la lista de figuras almacenadas en el Smartphone Servidor (cámara)	2.-Entrega la lista de figuras almacenadas en el dispositivo y las proyecta en el Smartphone cliente mediante la lista de figuras. Actualiza la lista de figuras.
3.-Selecciona una figura disponible en la lista de figuras disponibles y pulsa el botón para comparar.	4.-Entrega en pantalla el resultado de la comparación, indicando el nombre de la figura comparada, el porcentaje de aciertos y el nombre de la figura encontrada más similar a la proyectada.

Flujo Alternativo:

Acción del actor	Respuesta del sistema
	4.- Error, no existe la imagen en la memoria del dispositivo

8.3.3.5 Caso de Uso: Gestionar secuencia (proceso) de inspección.

ID	CU05
Nombre	Gestionar secuencia (proceso) de inspección.
Actores	• Usuario
Objetivo	Establecer los parámetros de la secuencia de acciones que debe realizar el robot, para la correcta ejecución del proceso de comparación y selección de objetos de acuerdo al porcentaje de coincidencia referente a un objeto modelo. También se definir el porcentaje de comparación deseado.
Referencia	Requerimientos funcionales: RF1.04, RF3.10. (Tabla 1)
Precondiciones	• Los Smartphone deben estar habilitados y conectados, uno en modo administrador y el otro en modo cámara.
Postcondiciones	Secuencia de inspección guardada en la base de datos de la aplicación del Smartphone Servidor.

Acción del actor	Respuesta del sistema
1.- En el Modo Administrador el usuario selecciona la opción "Configurar nueva secuencia de operación"	2.- Muestra el formulario para crear una nueva secuencia, en que permitirá escribir: • Nombre de la secuencia. • Posición de captura. • Posición para objetos aceptados. • Posición para objetos descartados. • Porcentaje mínimo de aceptación.
3.- Introduce los datos pedidos por el formulario y presiona "Guardar Secuencia"	4.- Se muestra un mensaje en la ventana de mensajes recibidos que indica que la secuencia ha sido guardada exitosamente.
Flujo Alternativo:	
Acción del actor	Respuesta del sistema
	4.- Se muestra un mensaje en la ventana de mensajes recibidos que indica que la secuencia ya existe. Esto dice que ya existe una secuencia guardada con ese nombre.

8.3.3.6 Caso de Uso: Ejecutar proceso de inspección.

ID	CU06
Nombre	Ejecutar proceso de inspección.
Actores	• Usuario
Objetivo	Realizar la comparación entre un objeto ubicado debajo del robot y un objeto de referencia almacenada en la memoria del Smartphone, y permitir al robot seleccionar y mover el objeto hacia algún contenedor para objetos que sean aceptados o sean rechazados.
Referencia	Requerimientos funcionales: RF1.03, RF1.04, RF10, RF3.11. (Tabla 1)
Precondiciones	• Los Smartphone deben estar habilitados y conectados, uno en modo administrador y el otro en modo cámara. • La cámara del Smartphone (cámara) debe estar activada. • El Smartphone en que opera la cámara debe estar conectado al conversor Bluetooth-Serial, y este al puerto serial del robot.
Postcondiciones	Se cumple el proceso de inspección, selección y traslado de los objetos fabricados en el proceso de manufactura, de manera de que el robot acepte o rechace el objeto.
Flujo Principal:	
Acción del actor	Respuesta del sistema
1.-Actualiza la lista de figuras, secuencias de operación y de configuración de segmentación.	2.- Entrega las listas con las figuras, secuencias y configuraciones solicitadas y las muestra en un cuadro de lista en la aplicación cliente.
3.- Selecciona una figura dentro de la lista de figuras disponibles, el nombre de la secuencia de inspección (operación) y en caso opcional selecciona una configuración de segmentación registrada. Presiona el botón "Inspeccionar"	4.- Se envía la orden de ejecución del programa de robot de posición de captura de la secuencia de operación al robot.
5.- El robot ejecuta el programa y se mueve a la posición de captura, cuando llega al punto de captura envía un mensaje ("OK1").	6.- Entrega el porcentaje de aciertos de la comparación entre la imagen capturada por la cámara y la imagen solicitada. Entrega el nombre de la imagen que más se

	asemeja a la imagen capturada por la cámara de acuerdo al contorno de la figura principal.
	7.- Se envía la orden ejecución del programa de robot de posición para los objetos aceptados o descartados de la secuencia de operación al robot, según el resultado de la comparación.
8.- El robot ejecuta la orden moviendo del objeto hacia la posición asignada según sea el resultado del proceso de comparación de figuras y al terminar la ejecución envía un mensaje de respuesta.	
Flujo Alternativo:	
Acción del actor	Respuesta del sistema
	6.- Error en la comparación de imágenes por: • Imagen no existe en memoria o no se almacenó en el formato binario. • Falla en la conexión entre los dispositivos • Mala configuración de segmentación y de la región de interés.

8.3.3.7 Caso de Uso: Mover robot.

ID	CU07
Nombre	Mover robot
Actores	• Usuario
Objetivo	Enviar un mensaje de instrucción al brazo-robot para que este realice una acción ya sea de ejecutar un determinado movimiento o que envíe como respuesta datos e información almacenada en la memoria del controlador del robot.
Referencia	Requerimientos funcionales: RF1.02, RF1.03. (Tabla 1)
Precondiciones	• Los Smartphone deben estar habilitados y conectados, uno en modo administrador y el otro en modo cámara. • El Smartphone en que opera la cámara debe estar conectado al conversor Bluetooth-Serial, y este al puerto serial del robot.
Postcondiciones	Se ejecuta la acción del robot solicitada por el usuario, ya sea un movimiento o de captura de datos del estado del robot, cuyo mensaje de respuesta del robot es recibida por el usuario.
Flujo Principal:	
Acción del actor	Respuesta del sistema
1.-El usuario se dirige al editor de texto de envío de mensajes y escribe antes de enviar un mensaje “bt” para indicar que el mensaje va dirigido al dispositivo Bluetooth conectado al robot, más la orden que será enviada al robot ej: bt + <instrucción> .	2.- Envía el mensaje al robot sin el prefijo “bt” más un retorno de carro “\r”.En caso de estar conectado al dispositivo Bluetooth envía respuesta al usuario que el mensaje fue enviado.
	3.-El robot recibe el mensaje y ejecuta la orden solicitada por este. En caso de ejecutar programas de movimiento, ejecuta el movimiento y envía respuesta de que el movimiento ha sido realizado con éxito o con falla, en otro caso envía datos del estado del robot.
4.-El usuario recibe el mensaje de respuesta del robot en la pantalla del Smartphone.	

Flujo Alternativo:	
Acción del actor	Respuesta del sistema
	2.- No existe respuesta, por falla en la conexión en algunos de los dispositivos.

8.4 Modelamiento de datos.

En este apartado, se describe al sistema con respecto a la organización de los datos más importantes de la aplicación, aunque el sistema solo posea una pequeña base de datos que funciona de manera local en el dispositivo, estas serán de importancia a la hora de almacenar las configuraciones referentes a los procesos de segmentación de imágenes e inspección de objetos, por lo que se decidió para tal fin integrar una Base de Datos local (SQLite) en vez de guardar esos datos en archivos de texto por razones de seguridad y escalabilidad de la aplicación.
El modelo de datos utilizado para describir al sistema será el modelo de datos conceptual que indicado en la Figura 45:

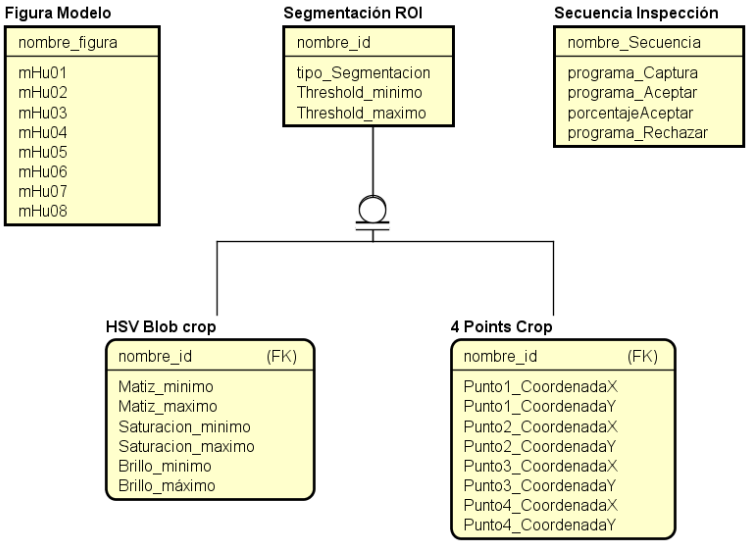


Figura 45. Diagrama modelo conceptual de datos de la aplicación.

La base de datos de la aplicación almacena tres tipos de datos correspondiente a los datos de la aplicación correspondientes a las figuras, a los datos de secuencia de inspección o conjunto de programas del robot a ejecutar durante la operación de control de calidad, y a los datos de cada configuración de segmentación y de búsqueda de la región de interés de un objeto. La tabla de configuración de segmentación ROI hereda en dos tipos de configuración según el modo si es por coordenadas o por detección de manchas.

9 DISEÑO.

9.1 Diseño Arquitectónico.

Mediante el diseño arquitectónico se identifican y estructuran los componentes del sistema y subsistema, permitiendo ver con claridad las relaciones y el control que ejercen las partes del sistema entre sí. Este diseño permite identificar los puntos críticos del sistema que requieren un mayor análisis.

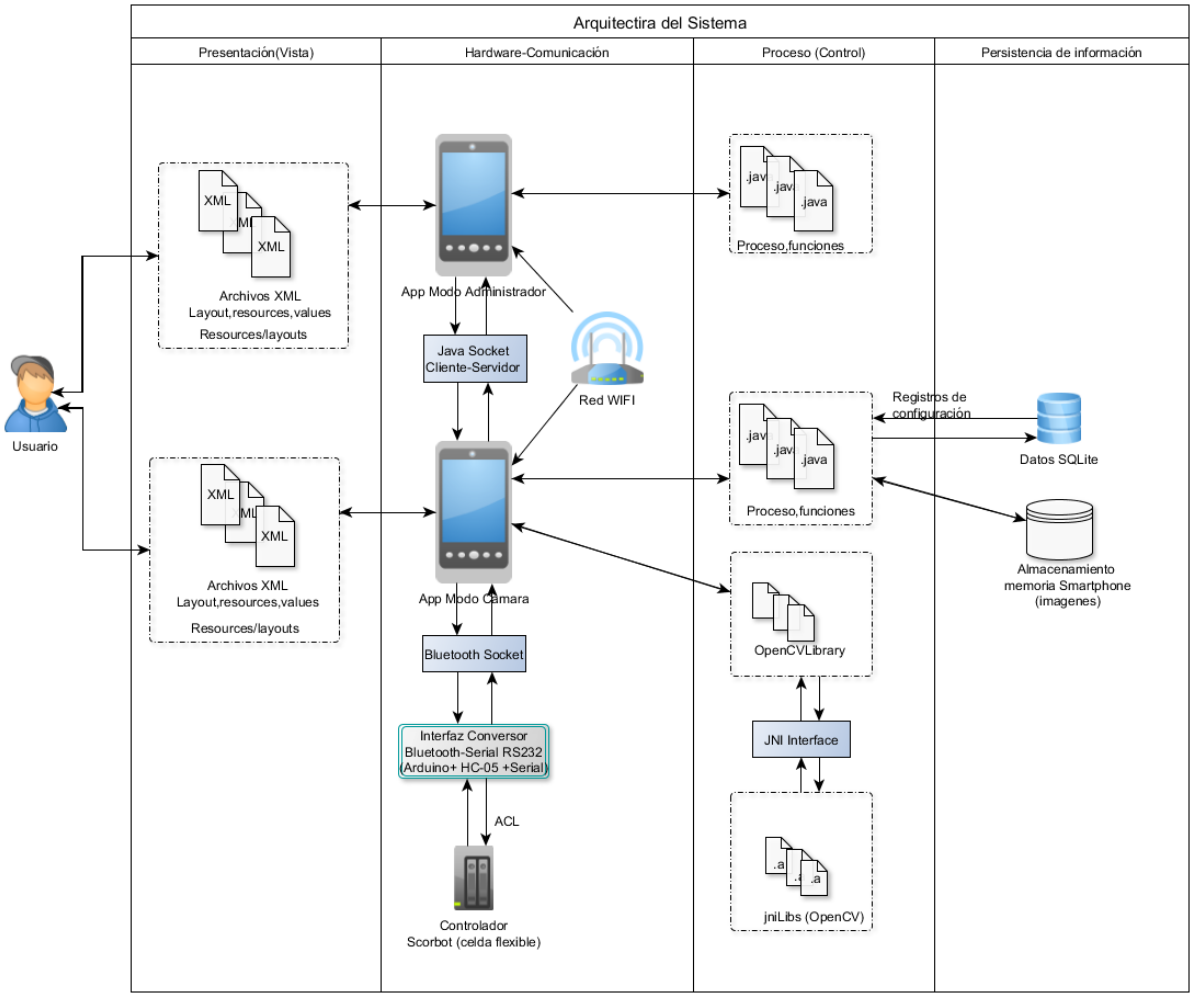


Figura 46. Diseño arquitectónico de la aplicación.

Como se puede ver en la Figura 46, para el funcionamiento del sistema como mínimo es necesario que la aplicación sea ejecutada en dos Smartphone, uno en Modo Administrador y el otro en Modo Cámara. Ambos Smartphone poseen el mismo código fuente y están conectados a la misma red WIFI para poder establecer la comunicación entre ambos. Además, uno de los Smartphone que ejecuta la aplicación en Modo Cámara tiene comunicación directa con el controlador del brazo-robot Scorbot mediante conexión Bluetooth utilizando un dispositivo conversor Bluetooth-Serial.

La aplicación utiliza la librería OpenCV para implementar las funciones de visión artificial y/o procesamiento digital de imágenes. Como toda estructura de una aplicación desarrollada en Android Studio, el diseño de la interfaz gráfica de usuario (presentación) esta codificado en archivos (.xml) y la funcionalidad y control de la aplicación (proceso y control) está programada en lenguaje de programación Java. En el ámbito de persistencia de información, las imágenes se almacenan en la memoria interna del Smartphone y los parámetros de configuración de la aplicación se almacenan en una base de datos SQLite.

9.2 Diseño de arquitectura funcional.

El diagrama de arquitectura de diseño funcional está representado en un árbol de descomposición que tiene por objetivo describir la características de descomposición de los módulos e indicar las principales funciones implementadas en cada proceso que involucra el sistema de manera de dar un mejor entendimiento sobre la organización de los componentes de la aplicación, sus dependencias entre las funciones y su interacción con la principal librería utilizada para el procesamiento y el control de imágenes (OpenCV).

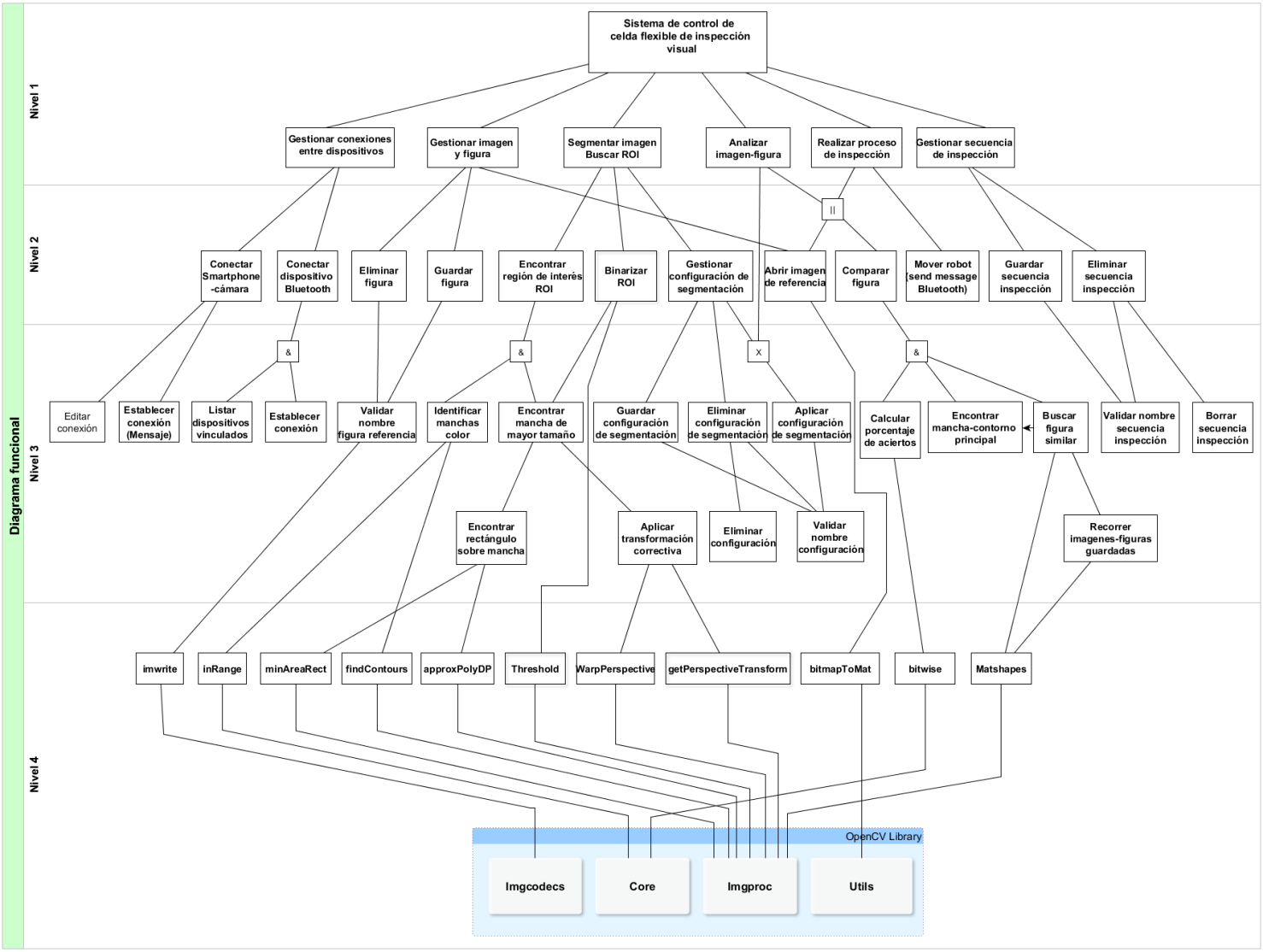


Figura 47. Diagrama de arquitectura funcional.

El diagrama indicado en la Figura 47 indica la descomposición de la funcionalidad de la aplicación y la dependencia entre las funciones y/o librería utilizada por la aplicación. El diseño funcional ha sido separado en cuatro niveles de descomposición que son los siguientes:

- Nivel 1: Nivel de Casos de Uso (CU) del sistema (ver Sección 8.3.3).
- Nivel 2: Indica las funciones principales para cada cumplir con cada función o CU del nivel 1.
- Nivel 3: Indica la división o las variantes de las funciones indicadas en el nivel 2. Para las funciones relacionadas al procesamiento de imágenes las funciones de este nivel invocan a las funciones propias de la librería OpenCV indicadas en el nivel 4.
- Nivel 4: Indica las funciones al nivel de librería y sus respectivos módulos de ésta.

9.3 Diseño de interfaz y navegación.

9.3.1 Introducción.

En este apartado se da a conocer el diseño de la aplicación desde el punto de vista visual, referente al diseño del tema, de la organización y el layout de la interfaz gráfica, de acuerdo los diseños de otras aplicaciones realizadas y desarrollada en Android para el Laboratorio CIM de la universidad. En este sentido el diseño de la IU está dado por los factores ambientales y del entorno.

Además, se describe la estructura de los módulos de la aplicación desde el punto de vista de la interfaz de navegación y jerarquía del menú de acuerdo a las buenas prácticas de desarrollo de Android.

Los principios de diseño de esta aplicación son los siguientes:

- Familiaridad: La interfaz debe usar términos y conceptos familiares al usuario y al dominio de la aplicación.
- Mínima sorpresa: El comportamiento del sistema no debe provocar sorpresa al usuario. El sistema debe comportarse de la forma más predecible posible de acuerdo a las funcionalidades indicadas implícitamente en la IU.
- Uniformidad: La interfaz debe ser uniforme, operaciones comparables deben estar funcionar de la misma manera, por ejemplo, los usos de los colores entre las UI de cada módulo deben ser similares, el layout de los formularios deben ser consistentes entre estos.
- Recuperabilidad: La interfaz debe ayudar al usuario de no cometer errores o evitarlos.
- Diversidad de usuarios: La IU debe estar orientada a todos los tipos de usuarios del sistema, en este caso el tipo de usuario es uno solo y éste posee amplios conocimientos sobre el dominio de la aplicación.

Las formas de interacción de la aplicación con el usuario a través de la IU serán a través de:

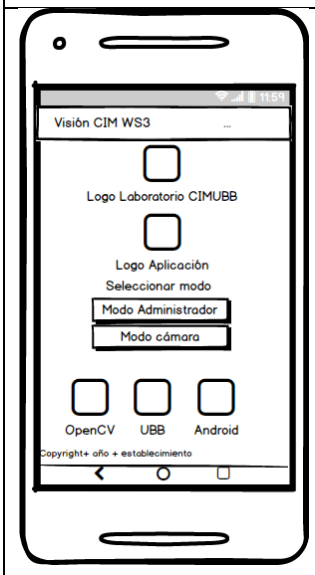
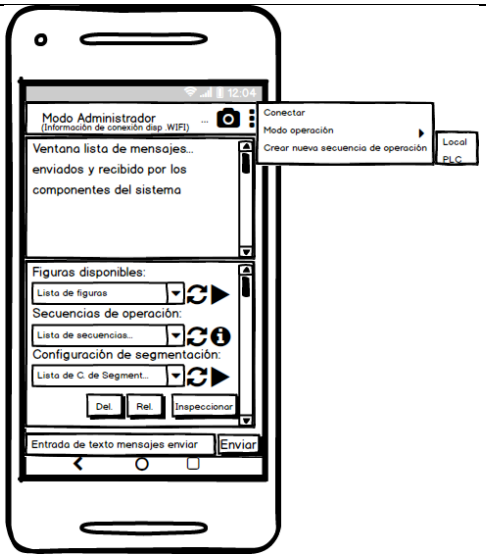
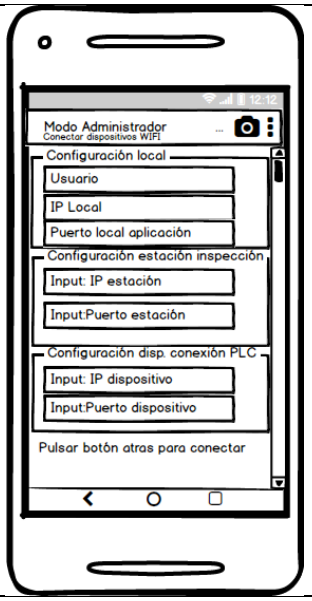
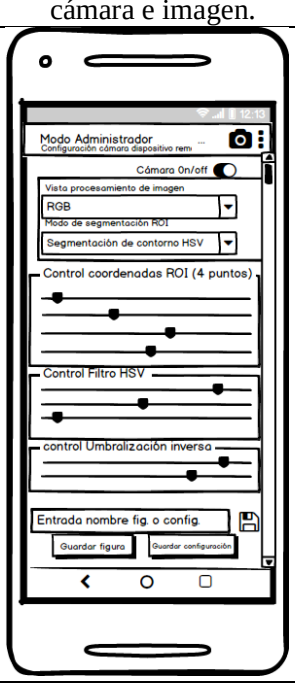
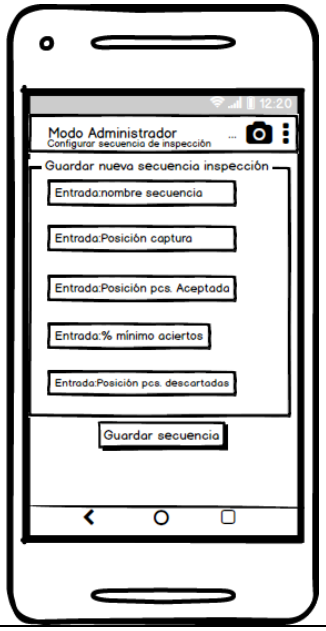
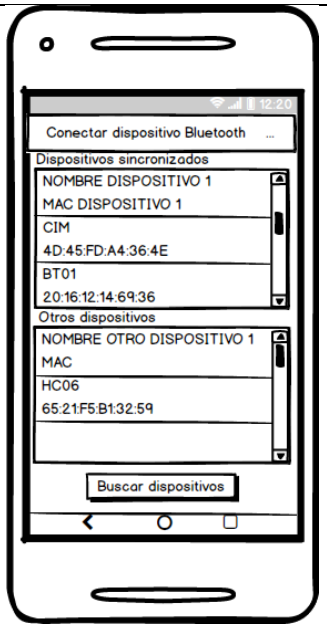
- Manipulación directa: El usuario interactúa directamente con los objetos indicados en la pantalla, como es el caso de botones, pantalla táctil, cuadros de lista, etc.
- Selección de menús: El usuario selecciona un comando de una lista de posibilidades (menú) para ejecutar una determinada acción o dirigirse a un módulo distinto dentro de la aplicación.
- Rellenado de formularios: El usuario rellena campos de un formulario (Similar a cuando se rellena una planilla en papel).
- Lenguajes de comando: Similar al Shell de Linux o de MS-DOS, el usuario escribe comandos y el sistema los ejecuta, en el caso de esta aplicación esta tendrá un editor de texto en la parte inferior de la IU que servirá para enviar texto y comandos tanto al Smartphone (cámara) como al brazo robot en cual estos según el comando el sistema ejecutará una acción y entregará su respuesta, esta será desplegada en una lista de texto de mensajes en parte superior de la IU.

9.3.2 Diseño de prototipo de la interfaz de gráfica de usuario (GUI).

El diseño de los prototipos de la interfaz gráfica de usuario se realizó de acuerdo a las funcionalidades del sistema y los principios de diseño indicados en la Sección 9.3.1.

9.3.2.1 Diseño de layouts de la aplicación.

A continuación, en la Figura 48 da a conocer el diseño de la interfaz gráfica de la aplicación y de sus módulos:

• LA01: Inicio.	• LA02: Operación inspección y línea de comandos.	• LA03: Configuración de conexión Smartphone cliente-servidor.
		
• LA04: Configuración y control remoto de cámara e imagen.	• LA05: Formulario de configuración secuencia de operación del robot.	• LA06: Conectar dispositivo Bluetooth.
		

• LA07: Terminal de comando Smartphone servidor cámara.	• LA08: Visor cámara, proyección de video y procesamiento de imagen.

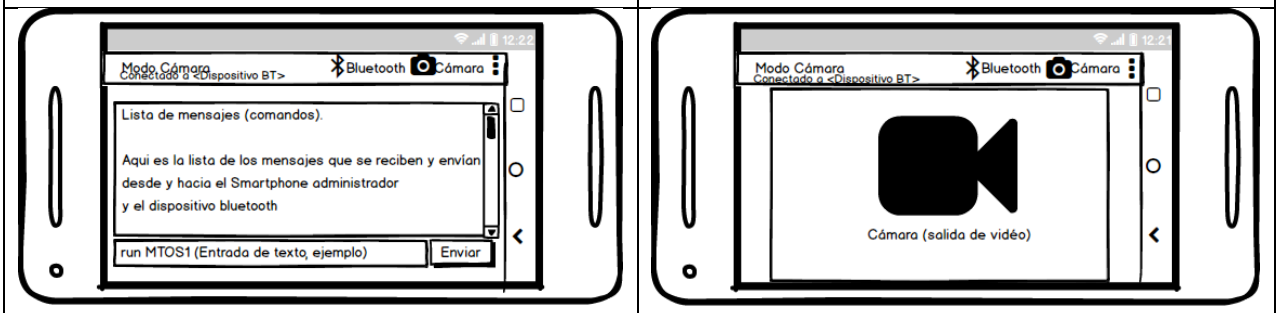


Figura 48. Diseño de interfaz gráfica y layout de la aplicación.

9.3.2.2 Diseño y color de los componentes.

El tema o características de la interfaz gráfica de usuario (GUI) en el aspecto de los colores utilizados para los componentes de la GUI, serán derivados de los matices del color azul.

En la siguiente figura se describen los colores utilizados en el diseño de la mayoría componentes gráficos de la aplicación como ejemplo botones, entradas de texto, sliders, listas, etc.

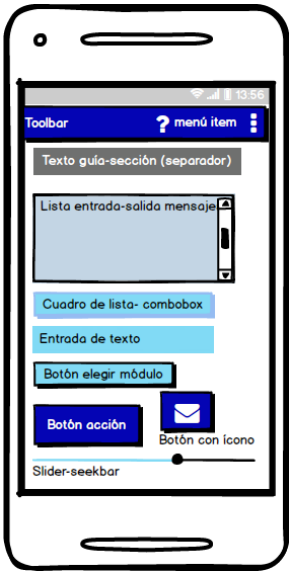


Figura 49. Diseño de componentes de la interfaz gráfica (GUI) de la aplicación.

Esto corresponde prototipo del diseño inicial de la aplicación desde el punto de vista gráfico. La interfaz gráfica real de la aplicación será presentada en el capítulo de implementación, junto con la demostración del producto final (aplicación) desarrollado en este proyecto.

9.3.3 Diseño de la jerarquía del menú.

El diseño de la jerarquía del menú tiene por objetivo representar los anidamientos y agrupaciones de las opciones disponibles del menú para cada nivel o módulo de la aplicación.

Como lo indica la Figura 50 la aplicación se inicia en el menú principal donde existen dos opciones disponibles en el cual se permite al usuario seleccionar solo una opción según el modo de ejecución deseado en el Smartphone ya sea en Modo Administrador (controlador) o en Modo Cámara, cada una con un menú de opciones disponible en que el usuario tiene acceso, cada opción ejecuta una determinada función de la aplicación.

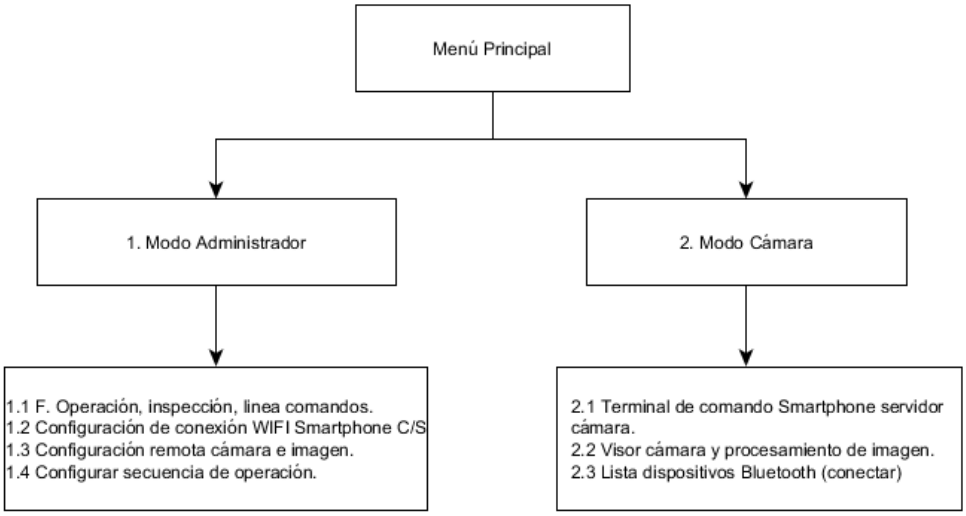


Figura 50. Jerarquía del menú de la aplicación.

9.3.4 Esquema de navegación de la aplicación.

El esquema de navegación tiene por objetivo representar las opciones que tendrá el usuario para navegar o recorrer dentro de las distintas opciones disponibles mientras se está ejecutando la aplicación según el módulo o layout actual desplegado en pantalla.

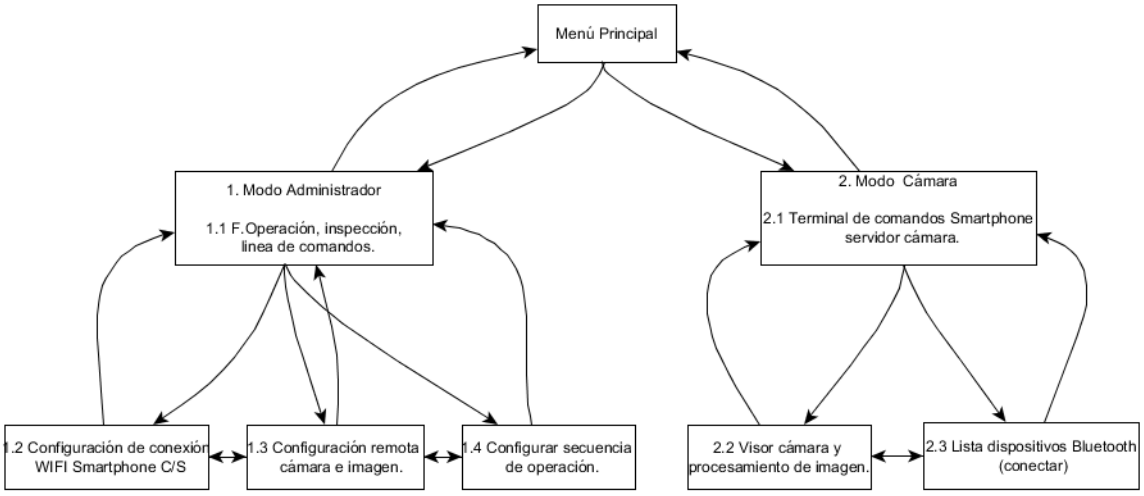


Figura 51. Esquema de navegación de la aplicación.

9.4 Especificación de módulos.

9.4.1.1 M01: Configuración de conexión WIFI a Smartphone C/S.

ID Módulo:	M01		
Descripción:	Conecta el Smartphone en aplicación (Modo Administrador) al Smartphone en aplicación (Modo Cámara) y de manera opcional al Smartphone conectado al PLC.		
Referencias:	Requerimientos: RF1.02 (Tabla 1) Casos de Uso: CU01 (Sección 8.3.3.1) Layout: LA03 (Figura 48)		
Parámetros de entrada:		Parámetros de salida:	
• Conectar Smartphone (modo cámara)			
Nombre:	Tipo de dato:	Nombre:	Tipo de dato:
Nombre usuario (*)	String [1]	Dirección IP local Smartphone	String [1]
IP Smartphone control cámara (*)	String [1]	Puerto habilitado local Smartphone	Integer
Puerto Smartphone c. cámara (*)	Integer	Respuesta de conexión: (Conectado al: IP + Puerto) o (Sin conexión)	String
• Conectar Smartphone control PLC (opcional)			
Nombre:	Tipo de dato:	Nombre:	Tipo de dato:
IP Smartphone sol.Control PLC	String [1]	Respuesta de conexión: (Dispositivo conectado al IP + Puerto)	String
Puerto S. sol. Control PLC	Integer		

9.4.1.2 M02: Configuración y control remoto de cámara e imagen.

ID Módulo:	M02		
Descripción:	Permite controlar la imagen obtenida mediante la señal de video proyectada en la cámara de Smartphone conectado, y otorgarle los parámetros necesarios para encontrar y comparar un objeto. Además, se puede guardar la configuración de segmentación de la imagen y una imagen o figura de referencia.		
Referencias:	Requerimientos: RF05, RF07, y RF09 (Tabla 1) Casos de Uso: CU02, y CU03 (Secciones 8.3.3.2 y 8.3.3.3) Layout: LA04 (Figura 48)		
Parámetros de entrada:		Parámetros de salida:	
• Visualizar y seleccionar modo control imagen			
Nombre:	Tipo de dato:	Nombre:	Tipo de dato:
Activar / desactivar cámara	Boolean(Switch)	Mensaje: on/off (cámara)	String
Vista procesamiento imagen	String + Integer	ID vista seleccionada	Integer
Modo búsqueda del ROI	String	ID modo búsqueda seleccionada	Integer
• Dimensionar ROI por cuatro puntos.			
Nombre:	Tipo de dato:	Nombre:	Tipo de dato:
X1: Punto 1 coordenada x	Integer (Slider)		
Y1: Punto 1 coordenada y	Integer (Slider)		
X2: Punto 2 coordenada x	Integer (Slider)		
Y2: Punto 2 coordenada y	Integer (Slider)		
X3: Punto 3 coordenada x	Integer (Slider)		
Y3: Punto 3 coordenada y	Integer (Slider)		
X4: Punto 4 coordenada x	Integer (Slider)		
Y4: Punto 4 coordenada y	Integer (Slider)		
• Buscar ROI mediante búsqueda de manchas en espacio de color HSV			
Nombre:	Tipo de dato:	Nombre:	Tipo de dato:
H_MIN: Matiz mínimo	Integer (Slider)		
H_MAX: Matiz máximo	Integer (Slider)		
S_MIN: Mínima saturación	Integer (Slider)		
S_MAX: Máxima saturación	Integer (Slider)		
V_MIN: Brillo mínimo	Integer (Slider)		

V_MAX :Brillo máximo	Integer (Slider)		
• Umbralización inversa (binarización) de imagen contenida en el ROI.			
Nombre:	Tipo de dato:	Nombre:	Tipo de dato:
TH_MIN: Valor mínimo en escala de grises.	Integer (Slider)		
TH_MAX: Valor máximo en escala de grises.	Integer (Slider)		
• Guardar figura-imagen / configuración de segmentación.			
Nombre:	Tipo de dato:	Nombre:	Tipo de dato:
Nombre nueva figura	String	Figura <nombre> guardada + Momentos de figura principal	String(mensaje)
Nombre nueva configuración	String	Configuración <nombre> guardada	String(mensaje)

9.4.1.3 M03: Registrar secuencias de operación (inspección).

ID Módulo:	M03		
Descripción:	Envía una orden de registro al Smartphone de los datos de una nueva configuración de secuencia de operación que involucra una serie de programas que se ejecutaran por el robot y el porcentaje de aciertos mínimo de un objeto para ser aceptado. .		
Referencias:	Requerimientos: RF04, RF10 (Tabla 1) Casos de Uso: CU05 (Sección 8.3.3.5) Layout: LA05 (Figura 48)		
Parámetros de entrada:		Parámetros de salida:	
• Registrar nueva secuencia de operación (control inspección) en Smartphone cámara.			
Nombre:	Tipo de dato:	Nombre:	Tipo de dato:
Nombre de secuencia	String[1]	Mensaje respuesta registro	String
Nombre de programa Posición de captura	String[1]		
Nombre de programa Posición de objetos aceptados	String[1]		
Porcentaje de aciertos mínimo	Integer		
Nombre de programa Posición de objetos descartados	String[1]		

9.4.1.4 M04: Operación inspección y línea de comandos.

ID Módulo:	M04		
Descripción:	Despliega lista de mensajes enviados e información recibida desde los Smartphone (B) en aplicación (Modo cámara). Permite enviar mensajes de instrucción al Smartphone (B) en aplicación Modo cámara y al dispositivo Bluetooth-Scorbot. Despliega lista de figuras, secuencias de operación y configuración de segmentación en pantalla del Smartphone(A) en aplicación (Modo Administrador) almacenadas en el Smartphone (B). Envía órdenes de comparación de imágenes y ejecuta inspección en Smartphone (B).		
Referencias:	Requerimientos: RF03, RF04, RF05, RF08, y RF11 (Tabla 1) Casos de Uso: CU03, CU04, CU06, CU07 (Sección 8.3.3) Layout: LA02 (Figura 48)		
Parámetros de entrada:		Parámetros de salida:	
• Comunicación y control Smartphone (B)			
Nombre:	Tipo de dato:	Nombre:	Tipo de dato:
Mensaje a Smartphone (B)	String	Mensaje de respuesta u operación	String
Mensaje a dispositivo Bluetooth	“bt” + String		

• Comparar imagen			
Nombre:	Tipo de dato:	Nombre:	Tipo de dato:
Nombre figura	String (List)	Respuesta comparación (Figura, Porcentaje coincidencias, Figura encontrada, resultado decisión.	String
• Ejecutar operación de inspección			
Nombre:	Tipo de dato:	Nombre:	Tipo de dato:
Nombre secuencia de operación	String (List)		
Nombre configuración de Segmentación (opcional)	String (List)		

9.4.1.5 M05: Visión Cámara (Modo Cámara).

ID Módulo:	M05		
Descripción:	Ejecuta las órdenes recibidas del Smartphone en aplicación Modo Administrador (A). Proyecta la imagen obtenida de la cámara del Smartphone y el resultado del procesamiento de imagen. Ejecuta las todas las operaciones del proceso y control de imagen, de comparación y entrega de resultados de este al Smartphone (A). Envía los mensajes de instrucción al robot.		
Referencias:	Requerimientos: RF02, RF04, RF05, RF06, RF07, RF08 (Tabla 1) Casos de Uso: CU02, CU03, CU04, CU06, CU07 (Sección 8.3.3) Layout: LA07 y LA08 (Figura 48)		
Parámetros de entrada:		Parámetros de salida:	
• Comunicación con Smartphone Administrador y el dispositivo Bluetooth (conectado al Scorbot)			
Nombre:	Tipo de dato:	Nombre:	Tipo de dato:
Mensaje al dispositivo Bluetooth	String	Respuesta envío de mensaje al BT	String
Mensaje recibido del Smartphone (A)	String	Nombre dispositivo Bluetooth conectado.	String
		Mensaje de respuesta al Smartphone (A)	String
		Respuesta del robot	String
• Procesamiento de imagen y comparación.			
Nombre:	Tipo de dato:	Nombre:	Tipo de dato:
		Señal de video cámara	Mat(OpenCV)
		Imagen figura	bitmap
		Porcentaje de aciertos	Double
		Figura encontrada	String
		Decisión comparación	String
		Momentos de Hu	Double[7]

9.4.1.6 M06: Conexión dispositivo Bluetooth-Robot.

ID Módulo:	M06		
Descripción:	Conecta el Smartphone en aplicación (Modo Cámara) al dispositivo Bluetooth conectado al robot Scorbot o PLC..		
Referencias:	Requerimientos: RF03 (Tabla 1) Casos de Uso: CU01 (Sección 8.3.3.1) Layout: LA06 (Figura 48)		
Parámetros de entrada:		Parámetros de salida:	
• Conectar robot mediante dispositivo Bluetooth			
Nombre:	Tipo de dato:	Nombre:	Tipo de dato:
		Lista dispositivos vinculados	String[n][2]
		Lista otros dispositivos	String[n][2]

10 ESPECIFICACIÓN DE PRUEBAS DE SOFTWARE.

10.1 Elementos de Prueba.

Se definen los elementos de prueba de software a los que se será sometida la aplicación. Estos elementos corresponden a interfaz, navegación y funcionalidad. La aplicación será sometida a prueba tanto para la aplicación ejecutada en Modo Administrador como en el Modo Cámara, de manera de comprobar que cada funcionalidad opere sin problemas.
Los elementos de prueba a evaluar en la aplicación son: en el Modo Administrador son:

1. Para la aplicación en Modo Administrador:

1.1 Conexión WIFI (Establecer comunicación Socket TCP).
1.2 Envío de mensajes vía WIFI.
1.3 Control remoto de Smartphone Modo Cámara, segmentación y registro de datos.
1.4 Menú opciones.
1.5 Características gráficas.

2. Para la aplicación en Modo Cámara:

2.1 Conexión WIFI (Aceptar conexión)
2.2 Conexión Bluetooth.
2.3 Envío y recepción de mensajes vía WIFI.
2.4 Envío y recepción de mensajes vía Bluetooth.
2.5 Proyección de imagen capturada.
2.6 Proyección de Región de Interés ROI
2.7 Registro de datos: de configuración y de imagen.
2.8 Evaluación, comparación e identificación de figuras.
2.9 Control del brazo-robot e inspección.
2.10 Características gráficas.

10.2 Especificación de las pruebas.

Característica a probar: Funcionalidad					
Nivel de prueba.	Objetivo de la prueba	Enfoque para la definición de casos de prueba	Técnicas para la definición de casos de prueba	Actividades de prueba	Criterios de cumplimiento
Unidad Integración Sistema Aceptación	La aplicación debe conectar y comunicarse correctamente entre los dispositivos involucrados en el sistema Probar y validar que el sistema hace lo que debe hacer.	Caja negra	Estabilidad y partición equivalente	Conectar dispositivos mediante WIFI y Bluetooth. Comparar el resultado esperado con el resultado obtenido. Medir el tiempo que la conexión tarde en lograrse, y distancia necesaria para mantener la comunicación.	Las funciones han sido ejecutadas y resultados de estas han sido obtenidos correctamente. El tiempo en establecer la conexión no debe ser mas de 5 segundos (WIFI o Bluetooth). La conexión entre los dispositivos debe funcionar en una distancia mínima de tres metros de separación entre estos.
Característica a probar: Desempeño.					
Nivel de prueba.	Objetivo de la prueba	Enfoque para la definición de casos de prueba	Técnicas para la definición de casos de prueba	Actividades de prueba	Criterios de cumplimiento
Unidad Integración Sistema Aceptación	La aplicación debe realizar los procesos del sistema de manera rápida y eficiente con los resultados esperados de las solicitudes entregadas al Smartphone Cámara y al robot.	Caja negra	Valores límites y partición equivalente	Realizar las conexiones necesarias. Comprobar que los datos enviados y la información recibida se reflejen correctamente en el sistema de acuerdo a las peticiones del usuario. Comprobar el correcto funcionamiento del procesamiento de imagen, la evaluación de figuras y control del robot de acuerdo a los resultados de comparación.	No deben existir errores lógicos en el registro de datos, procesamiento de imágenes, de comparación de figuras y de envío de instrucciones al robot. Las respuestas del Smartphone Cámara a las peticiones del Smartphone Administrador no deben ser mayor a 3 segundos. El tiempo utilizado para la comparación y/o identificación de figuras no debe ser mayor a 4 segundos.
Característica a probar: Interfaz.					
Nivel de prueba.	Objetivo de la prueba	Enfoque para la definición de casos de prueba	Técnicas para la definición de casos de prueba	Actividades de prueba	Criterios de cumplimiento
Unidad Integración Sistema Aceptación	Verificar que los componentes gráficos estén ubicados de manera conveniente al usuario y estos funcionen correctamente	Caja negra	Usabilidad	Comprobar que cada elemento gráfico funcione correctamente de acuerdo a la funcionalidad otorgada a ese elemento. Verificar que la ubicación de cada elemento sea la apropiada.	Ningún componente gráfico presenta fallas en su funcionamiento. La interfaz gráfica y sus elementos son comprensibles para el usuario.

Característica a probar: Navegación.					
Nivel de prueba.	Objetivo de la prueba	Enfoque para la definición de casos de prueba	Técnicas para la definición de casos de prueba	Actividades de prueba	Criterios de cumplimiento
Unidad Integración Sistema Aceptación	Verificar que las opciones del menú y de navegación de la aplicación funcionen correctamente y se encuentren disponibles dependiendo del contexto.	Caja negra	Usabilidad	Acceder a los dos modos disponibles de la aplicación. Comprobar que los menús y opciones de navegación cambien según el modo en ejecución de la aplicación.	Los menús cambian de acuerdo al modo de ejecución de la aplicación (Administrador- Cámara). Al navegar con el botón hacia atrás desde un módulo o interfaz actual, la aplicación regresa a un módulo anterior.
Característica a probar: Seguridad.					
Nivel de prueba.	Objetivo de la prueba	Enfoque para la definición de casos de prueba	Técnicas para la definición de casos de prueba	Actividades de prueba	Criterios de cumplimiento
Unidad Integración Sistema	Verificar que las funciones ajenas al modo de ejecución de la aplicación estén bloqueadas, Verificar que el sistema verifique el estado de las conexiones.	Caja negra	Estabilidad.	Cambiar el modo de ejecución de la aplicación. Desconectar aplicación en modo cámara. Desactivar dispositivo Bluetooth.	La aplicación se ejecuta en un solo módulo. La aplicación avisa del estado de conexión WIFI en Modo Administrador y del estado de conexión Bluetooth en Modo cámara.

Tabla 14. Especificación de las pruebas.

10.3 Responsables de las pruebas.

Los principales responsables de las pruebas de la aplicación y a la vez de la funcionalidad del control de la celda flexible de inspección visual es el desarrollador del proyecto y él(los) colaborador(es) presentes en el Laboratorio de Sistemas Automatizados de Producción CIMUBB, quienes tienen que verificar que cada módulo funcione correctamente y así se cumplan con los objetivos del software descritos en la Sección 4.2. Además, el jefe encargado del laboratorio CIMUBB, don Luís Vera, ingeniero de soporte del Sistema de Manufactura Flexible y profesor co-guía de este proyecto, es el encargado de revisar cada incremento entregado del proyecto, y de decidir la aprobación de la aplicación.

10.4 Calendario de pruebas.

Responsables	Pruebas	Descripción	Fecha.
Víctor Monsalves R.	Interfaz y navegación.	Verificar que los componentes gráficos funcionen de acuerdo a su funcionalidad asignada, y que la navegación entre los módulos de la aplicación se ejecute correctamente.	26/05/2017
Víctor Monsalves R. y Benjamín Larenas	Funcionalidad, desempeño y seguridad.	Verificar la conectividad entre dispositivos, la conexión y el control de la celda flexible Scorbot mediante Bluetooth.	16/06/2017
Victor Monsalves R., Luís Vera Quiroga , y parte del personal del laboratorio CIMUBB.	Funcionalidad y desempeño	Evaluar las etapas del procesamiento de imágenes y comparación de figuras y control de celda flexible de acuerdo a resultados de comparaciones.	20/06/2017

Tabla 15. Calendario de pruebas.

10.5 Detalle de las pruebas.

10.5.1 Prueba de conectividad y estabilidad.

10.5.1.1 Conectividad.

Las precondiciones existentes para esta prueba son:

- Red WIFI disponible.
- Todos los Smartphone involucrados en la ejecución del sistema deben estar conectados a la misma red WIFI.
- Los dispositivos que ejecuten la aplicación en Modo Cámara deben tener configurada la dirección IP local y las direcciones físicas (MAC) de los dispositivos Bluetooth (HC-05) de los conversores Bluetooth-Serial que se conectará al robot.
- El robot y/o en caso opcional el PLC, debe(n) estar conectado(s) al dispositivo conversor Bluetooth-Serial y cada uno de estos debe estar encendidos.

La prueba consiste en conectar a través de WIFI, a un Smartphone ejecutando la aplicación en modo Administrador a otro ejecutando la aplicación en modo Cámara y éste se conecte a través de Bluetooth al robot correspondiente en un tiempo no mayor a cinco segundos.

Tiempo medido para establecer conexión WIFI.

ID Prueba	Tiempo estimado	Tiempo promedio medido	Éxito / Fracaso
PR01	5 segundos	~ 2 segundos	Éxito

Tiempo medido para establecer la conexión Bluetooth.

ID Prueba	Tiempo estimado	Tiempo medido promedio	Éxito / Fracaso
PR02	5 segundos	~ 4 segundos	Éxito

Los tiempos registrados corresponden al promedio aproximado después de hacer diez pruebas.

10.5.1.2 Estabilidad.

Para probar la estabilidad de las conexiones entre los dispositivos, el sistema debe haber aprobado la prueba de conectividad.
La prueba consiste en medir el tiempo y distancia de separación límites, en que los elementos del sistema logran mantenerse conectados para el intercambio de mensajes e instrucciones.

Tiempo de conexión medido para la comunicación WIFI.

ID Prueba	Tiempo estimado	Tiempo medido aproximado	Resultado
PR03	15 minutos	< 20 minutos	Tiempo mayor al esperado

Tiempo de conexión medido para la comunicación vía Bluetooth.

ID Prueba	Tiempo estimado	Tiempo medido aproximado	Resultado
PR04	20 minutos	Entre 15 y 25 minutos	Rango de tiempo aceptable.

Distancia en que funciona la comunicación vía WIFI.

ID Prueba	Distancia estimada	Distancia máxima medida	Resultado
PR05	10 metros	< 10 metros, o según cobertura e intensidad de la señal WIFI en ambos Smartphone	Aceptable

Distancia en la que funciona la conexión y comunicación vía Bluetooth.

ID Prueba	Distancia estimada	Distancia máxima medida	Resultado

PR06	5 metros	< 8 metros, sin interferencia	Aceptable

10.5.2 Prueba de funcionalidad y desempeño.

10.5.2.1 Conectar dispositivos por WIFI (Administrador-Cámara).

Referencias:

- Casos de Uso: CU01 (Sección 8.3.3.1)
- Módulo: M01 (Sección 9.4.1.1)

Precondiciones:

- Haber aprobado anteriormente las pruebas de conectividad y estabilidad de la aplicación (Sección 10.5.1).
- El Smartphone a conectar debe tener ejecutada la aplicación en modo cámara.
- En caso de conectar opcionalmente a un segundo Smartphone vinculado al dispositivo Bluetooth-PLC se debe seleccionar en el menú la opción (Modo local + PLC) y este Smartphone debe ejecutar la aplicación en modo cámara.

ID Prueba: PR07						
Caso	Datos Entrada/Valor	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Nombre usuario	Lvq	Ver mensaje de conexión y cambio de estado de conexión a conectado.	Estado conexión: "Conectado a 10.3.6.245: 8080" Mensaje en Pantalla: "Dispositivo conectado a: 10.3.6.245 Puerto: 8080"	Éxito.	Existe un Smartphone con esa dirección IP con la aplicación en modo cámara
	IP Smartphone Cámara	10.3.6.245				
	Puerto Smartphone Cámara	8080				
	IP Smartphone disp. PLC	N/A				
	Puerto Smartphone	N/A				
	Modo conexión.	Modo local				
2	Nombre usuario	Victor	Ver mensaje de conexión y cambio de estado de conexión a conectado. Ver mensaje de conexión dispositivo PLC.	Estado conexión: "Sin conexión" Mensaje: "No se puede conectar, Intente nuevamente".	Éxito.	Se cumple con la salida esperada ya que no existe un Smartphone con esta dirección IP con la aplicación en modo cámara.
	IP Smartphone Cámara	10.3.2.1				
	Puerto Smartphone Cámara	8080				
	IP Smartphone disp. PLC	10.3.2.221				
	Puerto Smartphone	8080				
	Modo conexión.	Modo local				

10.5.2.2 Enviar mensaje desde el modo administrador.

Referencias:

- Casos de Uso: CU01 y CU07 (Secciones 8.3.3.1 y 8.3.3.7).
- Módulos: M04 (Sección 9.4.1.4).

Precondiciones:

- Cumplir exitosamente las pruebas indicadas en la sección 10.5.1 y 10.5.2.1.
- Los dispositivos involucrados deben estar conectados entre sí.

ID Prueba: PR08						
Caso	Datos Entrada	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Mensaje de prueba	"Prueba"	Ver el mensaje en la pantalla del Smartphone emisor y receptor: "Prueba"	Mensaje mostrado en ambas pantallas: "Prueba"	Éxito.	Se envía el mensaje solo al Smartphone conectado.
2	Mensaje de prueba	"bt dir" ("dir" comando ACL del robot que lista los programas guardados en el controlador de este)	Enviar el comando "dir" al robot. Desplegar la lista de programas del robot Scorbot guardados en el controlador de este en la pantalla de mensajes del Smartphone Administrador.	Envía el mensaje "dir" al Scorbot , y este retorna la lista de programas y los visualiza en la pantalla en ambos Smartphones.	Éxito	Al anteponer "bt" antes del mensaje, el contenido del mensaje será enviado al robot conectado.
3	Mensaje de prueba	"bt home" (home comando ACL del robot que inicializa el robot a su posición inicial por defecto)	Enviar el comando "home" al robot. Ver el comando en la pantalla del Smartphone emisor y que el robot se mueva a la posición inicial por defecto.	El comando se muestra en pantalla y el robot mueve a la posición inicial por defecto, y retorna un mensaje "Home Complete" al terminar de ejecutarse.	Éxito.	Al anteponer "bt" antes del mensaje, el contenido del mensaje será enviado al robot conectado.

10.5.2.3 Comparar / evaluar figura-imagen.

Referencias:

- Casos de Uso: CU04 y CU06 (Secciones 8.3.3.4 y 8.3.3.6)
- Módulo: M04 y M05 (Secciones 9.4.1.4 y 9.4.1.5).

Precondiciones:

- Cargar datos de figuras almacenadas en memoria del Smartphone en aplicación modo cámara (Sección 10.5.3.2).
- Cámara del Smartphone en modo cámara activada en la aplicación.
- Haber obtenido la región de interés del objeto a evaluar.
- Haber binarizado la región de interés e identificada la figura principal contenida en esta mediante umbralización inversa.

ID Prueba: PR09						
Caso	Datos Entrada	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Nombre Figura	4M	Porcentaje de aciertos: >= 90% Figura similar identificada: 4M Resultado: Guardar.	Porcentaje de acierto: 93.87% Figura similar identificada: 4M Resultado: Guardar.	Éxito	Un porcentaje de acierto mayor o igual a 90% el resultado es "Guardar"
	Imagen-Figura proyectada en cámara	4M				
2	Nombre Figura	6M	Porcentaje de aciertos: < 90% Figura similar identificada: 4M Resultado: Guardar.	Porcentaje de acierto: 34,12% Figura similar identificada: 6M Resultado: Descartar.	Éxito	Un porcentaje de acierto menor a 90% el resultado es "Descartar"
	Imagen-Figura proyectada en cámara	1M				
3	Nombre Figura	2H	Mensaje en pantalla: "Cámara está inactiva"	Mensaje en pantalla: "Cámara está inactiva"	Éxito	La cámara del Smartphone en modo cámara no está activa.
	Imagen-Figura proyectada en cámara	N/A (Cámara inactiva)				

10.5.2.4 Registrar nueva secuencia de operación (Inspección).

Referencias:
Casos de Uso: CU05 (Sección 8.3.3.5)
Módulo: M03 (Sección 9.4.1.3).

Precondiciones:

- Cumplir exitosamente las pruebas indicadas en la sección 10.5.1 y 10.5.2.1.
- Los dispositivos involucrados deben estar conectados entre sí.
- Entrar al módulo de formulario para nueva secuencia de operación.

ID Prueba: PR10						
Caso	Datos Entrada	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Nombre secuencia	CIM	Mensaje en el panel de mensajes: "Secuencia CIM guardada". Guardar nueva secuencia en la base de datos de la aplicación en el Smartphone cámara.	Mensaje en el panel de mensajes: "Secuencia CIM guardada". Se ha guardado una nueva secuencia en la base de datos de la aplicación en el Smartphone cámara.	Éxito.	Mensaje mostrado en el panel de mensajes del módulo M01.
	Programa Captura	CAM				
	Programa Guardar	GUARD				
	Porcentaje mínimo de Aceptación	90				
	Programa Descartar	DESC				
2	Nombre secuencia	CIM	Mensaje en el panel de mensajes: "Secuencia ya existe"	Mensaje en el panel de mensajes: "Secuencia ya existe"	Éxito.	Mensaje mostrado en el panel de mensajes del módulo M01. No se ha guardado los datos de secuencia en la base de datos.
	Programa Captura	CAM1				
	Programa Guardar	GUARD				
	Porcentaje mínimo de Aceptación	85				
	Programa Descartar	DESC				

10.5.2.5 Inspección y control de robot por comparación de figuras y objetos.

Referencias:

- Casos de Uso: CU06 (Sección 8.3.3.6).
- Módulo: M04 (Sección 9.4.1.4)

Precondiciones:

- Todos los dispositivos involucrados tienen que estar conectados.
- Haber cumplido con las precondiciones de prueba indicadas en la sección 10.5.2.3.
- Debe existir alguna secuencia de inspección (parámetros) en la base de datos del Smartphone cámara (Secciones 10.5.2.4 y 10.5.3.2).
- Opcionalmente tener disponible la lista con las configuraciones de segmentación registradas en el Smartphone cámara.

ID Prueba: PR11						
Caso	Datos Entrada	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Figura Imagen-Figura proyectada en cámara Secuencia operación Configuración segmentación	2H 6H CIM -	Porcentaje de aciertos: < 90% Figura similar identificada: 6H Resultado: Descartar. Programa robot para sacar objeto: DESC	Porcentaje de aciertos: 29.2% Figura similar identificada: 6H Resultado: Descartar. Programa robot para sacar objeto: DESC	Éxito	La figura proyectada n caso de no parecerse a ninguna figura conocida, se identificará a la más similar.
2	Figura Imagen-Figura proyectada en cámara Secuencia operación Configuración segmentación	4M 4M CIM -	Porcentaje de aciertos: > 90% Figura similar identificada: 4M Resultado: Guardar. Programa robot para sacar objeto: GUARD	Porcentaje de aciertos: 94.54% Figura similar identificada: 4M Resultado: Guardar. Programa robot para sacar objeto: GUARD	Éxito	

10.5.2.6 Guardar imagen-figura.

Referencias:

- Caso de Uso: CU03 (Sección 8.3.3.3).
- Módulo: M02 (Sección 9.4.1.2).

Precondiciones:

- Cumplir exitosamente con la prueba de umbralización inversa indicada en la sección 10.5.3.10 caso 1 y todas sus precondiciones correspondientes a los procesos de segmentación.
- Cámara del smartphone en la aplicación modo cámara activa, y el switch "cámara On/Off" debe estar en valor "true" (sección 10.5.3.5 ,caso 1).

ID Prueba: PR12						
Caso	Datos Entrada	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Nombre nueva figura (EditText)	6H	Imagen binaria contenida en el ROI guardada en memoria del dispositivo. Mensaje de figura guardada.	Se guarda la figura principal contenida en el ROI, retorna un mensaje de: "Figura <6H> guardada"	Éxito	
2	Nombre nueva figura (EditText)	8M	Imagen binaria contenida en el ROI guardada en memoria del dispositivo. Mensaje de figura guardada.	Se guarda la figura principal contenida en el ROI, retorna un mensaje de: "Figura <6H> guardada"	Éxito	
3	Nombre nueva figura (EditText)	6H	No se guarda imagen. Mensaje de figura repetida.	No se guardó la imagen contenida en el ROI. Mensaje de retorno: " Figura <6H> ya existe"	Éxito.	Si se quiere reemplazar la figura con una del mismo nombre, entonces esta se debe haber eliminado anteriormente

10.5.2.7 Guardar configuración de segmentación.

Referencias:

- Caso de Uso: CU02 (Sección 8.3.3.2).
- Módulo: M02 (Sección 9.4.1.2).

Precondiciones:

- Ejecutar exitosamente la prueba de umbralización inversa indicada en la sección 10.5.3.10 (caso 1) y todas sus precondiciones correspondientes a los procesos de segmentación.
- Cámara del smartphone en la aplicación modo cámara activa, y el switch "cámara On/Off" debe estar en valor "true" (sección 10.5.3.5 ,caso 1).

ID Prueba: PR13						
Caso	Datos Entrada	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Nombre nueva configuración (EditText)	Madera	Guardar los parámetros de configuración de segmentación con un nombre identificador "Madera"	Configuración guardada en la base de datos con el identificador "Madera"	Éxito	
2	Nombre nueva configuración (EditText)	Madera	No se guardan los parámetros de configuración ya que el nombre identificador "Madera ya existe".	La configuración no se guarda. Mensaje de retorno: "Configuración <Madera> ya existe.	Éxito	Si se quiere reemplazar la configuración con una del mismo nombre, entonces esta se debe haber eliminado anteriormente

10.5.2.8 Conectar Bluetooth.

Referencias:
Casos de Uso: CU01 (Sección 8.3.3.1).
Módulo: M06 (Sección 9.4.1.6).

Precondiciones:

- Ejecutar la aplicación en el modo cámara.
- Bluetooth del dispositivo conversor (HC-05) activado y vinculado al Smartphone.
- Cumplir con las pruebas de conectividad indicadas en la sección 10.5.1.
- Ingresar a la opción bluetooth pulsando el botón en el Toolbar (Sección 10.5.3.13).

ID Prueba: PR14						
Caso	Datos Entrada	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Nombre dispositivo.	CIM	Smartphone conectado al dispositivo Bluetooth CIM	Smartphone conectado al dispositivo Bluetooth CIM	Éxito	Estado de conexión BT: Conectado a CIM.
	MAC dispositivo.	98:D3:31:20:57:4E				
2	Nombre dispositivo.	BT01	Smartphone conectado al dispositivo Bluetooth BT01	Smartphone conectado al dispositivo Bluetooth BT01	Éxito	Estado de conexión BT: Conectado a BT01.
	MAC dispositivo.	00:12:06:01:57:07				

10.5.3 Prueba de interfaz y navegación.

10.5.3.1 Selección de modo ejecución de aplicación.

Precondiciones:

- Entrar a la aplicación y estar en el inicio.
- No haber seleccionado el modo de la aplicación.

ID Prueba: PR15					
Caso	Interfaz a probar	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Botón Modo Administrador.	Iniciar la aplicación en modo Administrador y activar las opciones del menú de este modo	Inicia la aplicación en modo Administrador con las opciones de administrador.	Éxito	
2	Botón Modo Cámara.	Iniciar la aplicación en modo Cámara y activar las opciones del menú de este modo (Cámara y Bluetooth)	Inicia la aplicación en modo Cámara con las opciones de conexión Bluetooth y cámara disponible.	Éxito	

10.5.3.2 Carga de parámetros de datos de proceso de inspección y control del robot.

Referencias:

- Casos de Uso: CU04 y CU06 (Secciones 8.3.3.4 y 8.3.3.6).
- Módulos: M04 (Sección 9.4.1.4).

Precondiciones:

- Aprobar con éxito las pruebas indicadas en la sección 10.5.2.2.
- Deben existir datos registrados en el Smartphone en modo cámara.

ID Prueba: PR16					
Caso	Interfaz a probar	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Botón "cargar figuras".	Cargar lista de figuras disponibles con los nombres de las figuras guardadas en el Smartphone cámara al cuadro de lista de figuras.	Lista de figuras cargadas en el cuadro de lista, desde el Smartphone cámara. Existen figuras disponibles para comparar.	Éxito	
2	Botón "cargar secuencias de operación".	Cargar lista de secuencias de operación disponibles con los nombres de las secuencias guardadas en el Smartphone cámara al cuadro de lista de secuencias de operación.	Lista de secuencias cargadas en el cuadro de lista, desde el Smartphone cámara. Existen Secuencias de programas disponibles para ejecutar inspección y control del robot	Éxito	
3	Botón "cargar configuración de segmentación"	Cargar lista de configuraciones de segmentación disponibles con los nombres de las configuraciones guardadas en el Smartphone conectado en el cuadro de lista de configuración de segmentación.	Lista de configuraciones de segmentación cargadas en el cuadro de lista, desde el Smartphone cámara. Configuraciones predeterminadas disponibles para implementar.	Éxito	

10.5.3.3 Menú lateral superior (Toolbar) aplicación en modo administrador.

Precondiciones:

- Estar en la aplicación en modo administrador.

ID Prueba: PR17					
Caso	Interfaz a probar	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Botón configuración cámara.	Abrir la sección de la aplicación de configuración de cámara remota (Módulo M02)	Se muestra la sección de la aplicación de configuración de cámara remota (Módulo M02)	Éxito	
2	Menú ítem: "Conexión".	Abrir la sección de configuración de conexión a Smartphone por WIFI (Módulo M01)	Se muestra la sección de configuración de conexión a Smartphone por WIFI (Módulo M01)	Éxito	
3	Menú ítem: "Modo local"	Ejecutar la aplicación solo para la comunicación con Smartphone en modo cámara. Se desactivan los botones de control de dispositivo vinculado a PLC.	La aplicación queda disponible para la comunicación con el Smartphone en modo cámara y control del robot. Las opciones de control de PLC quedan desactivadas.	Éxito.	El objetivo del control de PLC solo es opcional en la aplicación, No está involucrado con los requerimientos.
4	Menú ítem: "Modo local + PLC"	Ejecutar la aplicación para comunicarse con el Smartphone en modo cámara y algún Smartphone vinculado al control de PLC. Se activan los botones de control de dispositivo vinculado a PLC	La aplicación queda disponible para la comunicación con el Smartphone en modo cámara y control del robot, y algún Smartphone vinculado al control de PLC. Las opciones de control de PLC quedan activadas.	Éxito.	El objetivo del control de PLC solo es opcional en la aplicación, No está involucrado con los requerimientos.
5	Menú ítem: "Agregar nueva sec. de operación"	Abrir la sección de la aplicación para agregar nueva configuración de operación (Modulo M03).	Se muestra la sección para agregar una nueva configuración de operación (Módulo M03)	Éxito.	

10.5.3.4 Botones de control estación en PLC (opcional).

Referencias:

- Modulo: M04 (Sección 9.4.1.4).

Precondiciones:

- La aplicación en modo administrador debe estar activado el modo "Modo local + PLC" (Sección 10.5.3.3).
- Todos los dispositivos involucrados tienen que estar conectados incluido al Smartphone con la aplicación en modo cámara vinculado al PLC (Sección 10.5.2.1 ,caso 4).

ID Prueba: PR18					
Caso	Interfaz a probar	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Botón DEL (Deliver pallet)	Enviar un comando al PLC estación 1: "@00WD000900015B*" Retornar desde PLC: "@00WD0053" Detener pallet frente a estación 1.	Comando "@00WD000900015B*" Enviado al PLC, con código de retorno: "@00WD0053" El pallet se detiene frente de la estación 1 después de ejecutar una vuelta.	Éxito	
2	Botón REL (Release pallet)	Enviar un mensaje al PLC estación 1: "@00WD004800015E*" Liberar pallet frente a estación 1.	Comando "@00WD004800015E*" Enviado al PLC. El pallet detenido frente a la estación 1 se libera para seguir	Éxito	

			su trayectoria en la cinta transportadora del PLC		

Observación: Esta funcionalidad de la aplicación será de prueba ya que no está indicado en los requerimientos del sistema por lo que su uso será opcional, y funcionará para la Estación 1 del Sistema de Manufactura Flexible SMF.

10.5.3.5 Switch <Activar/Desactivar cámara>.

Referencias:

- Caso de Uso: CU02 (Sección 8.3.3.2).
- Módulo: M02 (Sección 9.4.1.2).

Precondiciones:

- Estar en la sección de configuración remota de cámara de la aplicación (Sección 10.5.3.3 caso 1).
- Los Smartphone tienen que estar conectados uno en modo administrador (cliente) y el otro en modo cámara (servidor).

ID Prueba: PR19						
Caso	Interfaz a probar	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Switch cámara "on/off"	True	Visualizar imagen en cámara del Smartphone con aplicación en modo cámara. Cambiar el estado del switch a "true".	Se activa la cámara del Smartphone vinculado, visualizando la imagen proyectada. El Switch cambia de estado a "true"	Éxito	
2	Switch cámara "on/off"	False	Desactivar la cámara del smartphone con aplicación en modo cámara, visualizando el panel de mensajes. Cambiar el estado del switch a "false". Guardar los datos recientes de configuración de segmentación en preferencias compartidas.	Se desactiva la cámara del smartphone quedando en pantalla el panel de mensajes. El switch cambia de estado a "false" Guarda los datos de configuración de segmentación reciente en preferencias para cuando se vuela a activar la cámara.	Éxito.	

10.5.3.6 Cuadro de lista: <Seleccionar modo vista segmentación>.

Referencias:

- Caso de Uso: CU02 (Sección 8.3.3.2).
- Módulo: M02 (Sección 9.4.1.2).

Precondiciones:

- Ejecutar exitosamente la funcionalidad del caso de prueba indicado en la sección 10.5.3.5 caso 1.

ID Prueba: PR20						
Caso	Interfaz a probar	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Cuadro de lista "Vista imagen"	Vista general RGB (1)	Imagen completa del campo de visión de la cámara en espacio RGB, más detección de contornos.	Imagen completa del campo de visión de la cámara en espacio RGB, más detección de contornos.	Éxito	Ver ejemplo en la segunda imagen de la Figura 14
2	Cuadro de lista "Vista imagen"	ROI Seleccionado RGB (2)	Imagen de la región de interés completa seleccionada en espacio de color RGB.	Imagen de la región de interés completa seleccionada en espacio de color RGB.	Éxito.	Ver ejemplo en la Figura 29b.

3	Cuadro de lista "Vista imagen"	Máscara binaria vista general (3)	Máscara binaria de la imagen completa de acuerdo al rango de color seleccionado.	Se visualiza la máscara de la imagen completa proyectada en pantalla de acuerdo al rango de color seleccionado.	Éxito.	Ver ejemplo en la segunda imagen de la Figura 14
4	Cuadro de lista "Vista imagen"	Máscara binaria del ROI seleccionado (4)	Mascara de la imagen contenida en el ROI de acuerdo al nivel de umbralización inversa.	Se visualiza la máscara de la imagen contenida en el ROI de acuerdo al nivel de umbralización inversa. Se ha identificado figuras en el ROI	Éxito.	Ver ejemplo demostrado en la Figura 18c
5	Cuadro de lista "Vista imagen"	Vista general en espacio HSV (5)	Imagen completa del campo de visión de la cámara en espacio de color HSV	Se visualiza imagen completa del campo de visión de la cámara en espacio de color HSV	Éxito.	Ver ejemplo demostrado en la segunda imagen de la Figura 13.

10.5.3.7 Cuadro de lista: <Seleccionar método de segmentación y selección del ROI>.

Referencias:

- Caso de Uso: CU02 (Sección 8.3.3.2).
- Módulo: M02 (Sección 9.4.1.2).

Precondiciones:

- Ejecutar exitosamente la funcionalidad del caso de prueba indicado en la sección 10.5.3.5 caso 1.

ID Prueba: PR21						
Caso	Interfaz a probar	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Modo de segmentación ROI	Coordenadas 4 puntos (1).	Habilitar la aplicación para buscar el ROI mediante coordenadas. Activa los controles de valores para las coordenadas. Desactiva los controles para los valores HSV (slider).	La aplicación puede buscar el ROI mediante 4 puntos. Los controles de valores para las coordenadas quedan activados, los de espacio de color HSV quedan desactivados.	Éxito	
2	Modo de segmentación ROI	Rango de color en espacio HSV (2)	Habilitar la aplicación para buscar el ROI detección de manchas en espacio de color HSV. Activa los controles de valores para seleccionar el rango en espacio HSV. Desactiva los controles para los valores de coordenadas	La aplicación puede buscar el ROI mediante la detección de manchas en el espacio de color HSV. Activa los controles(sliders) para seleccionar el rango de colores en el espacio HSV, Los controles para los valores de coordenadas quedan desactivados.	Éxito.	
3	Modo de segmentación ROI	Color tocado en pantalla-imagen (3).	Habilitar la aplicación para seleccionar el ROI mediante el espacio de color RGB seleccionando un color en la pantalla del Smartphone cámara.	Se puede seleccionar el ROI mediante la pantalla del Smartphone en modo cámara.	Éxito.	

10.5.3.8 Sliders: <Seleccionar ROI por dimensión en coordenadas>.

Referencias:

- Caso de Uso: CU02 (Sección 8.3.3.2).
- Módulo: M02 (Sección 9.4.1.2).

Precondiciones:

- Ejecutar exitosamente la funcionalidad del caso de prueba indicado en la sección 10.5.3.5 caso 1.
- Ejecutar exitosamente el caso de prueba indicado en la sección 10.5.3.7 caso 1.
- Considerar puntos de referencias como ejemplo de la Figura 11 ,y un tamaño de imagen de 960x540 pixeles.

Considerar: Inicialmente la región de interés corresponde al tamaño total de la imagen contenida en el campo de visión de la cámara, por lo que sus puntos corresponden a las esquinas de la imagen.

ID Prueba: PR22						
Caso	Interfaz a probar	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Slider p1-x (punto 1 en x)	584	Fijar el punto 1 del ROI en el 58.4% del largo de la imagen (coordenada X).	El punto 1 de la región de interés queda en la posición $\boldsymbol{p_1(x,y) = p_1(561,0)}$	Éxito	
2	Slider p1-y (punto 1 en y)	211	Fijar el punto 1 del ROI en el 21.1% del ancho de la imagen (coordenada Y).	El punto 1 de la región de interés queda en la posición $\boldsymbol{p_1(x,y) = p_1(561,114)}$	Éxito.	
3	Slider p2-x (punto 2 en x)	771	Fijar el punto 2 del ROI en el 77.1% del largo de la imagen (coordenada X).	El punto 2 de la región de interés queda en la posición $\boldsymbol{p_2(x,y) = p_2(740,0)}$	Éxito.	
4	Slider p2-y (punto 2 en y)	231	Fijar el punto 2 del ROI en el 23.1% del ancho de la imagen (coordenada Y).	El punto 2 de la región de interés queda en la posición $\boldsymbol{p_2(x,y) = p_2(740,125)}$	Éxito.	
5	Slider p3-x (punto 3 en x)	768	Fijar el punto 3 del ROI en el 76.8% del largo de la imagen (coordenada X).	El punto 3 de la región de interés queda en la posición $\boldsymbol{p_3(x,y) = p_3(737,539)}$	Éxito.	
6	Slider p3-y (punto 3 en y)	402	Fijar el punto 3 del ROI en el 40.2% del ancho de la imagen (coordenada Y).	El punto 3 de la región de interés queda en la posición $\boldsymbol{p_3(x,y) = p_3(737,216)}$	Éxito.	
7	Slider p4-x (punto 4 en x)	583	Fijar el punto 4 del ROI en el 58.3% del largo de la imagen (coordenada X).	El punto 4 de la región de interés queda en la posición $\boldsymbol{p_4(x,y) = p_4(559,539)}$	Éxito.	
8	Slider p4-y (punto 4 en y)	380	Fijar el punto 4 del ROI en el 38% del ancho de la imagen (coordenada Y).	El punto 4 de la región de interés queda en la posición $\boldsymbol{p_4(x,y) = p_4(559,205)}$	Éxito.	

10.5.3.9 Sliders: <Seleccionar ROI por rango de color en espacio HSV>.

Referencias:

- Caso de Uso: CU02 (Sección 8.3.3.2).
- Módulo: M02 (Sección 9.4.1.2).

Precondiciones:

- Ejecutar exitosamente la funcionalidad del caso de prueba indicado en la sección 10.5.3.5 caso 1.
- Ejecutar exitosamente el caso de prueba indicado en la sección 10.5.3.7 caso 2.

En estos casos de prueba se consideran los datos de entrada con los valores indicados en la Tabla 10 en la sección 6.5.2.

ID Prueba: PR23						
Caso	Interfaz a probar	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Slider Matiz mínima.	12	Visualizar las manchas y contornos que cumplan con un valor de matiz mínimo de 24,1° en el espacio de color HSV	Manchas y contornos que cumplen con un valor de matiz mínimo de 24,1° en el espacio de color HSV. (vista máscara imagen)	Éxito	
2	Slider Matiz máxima.	19	Visualizar las manchas y contornos que cumplan con un valor de matiz máximo de 38,2° en el espacio de color HSV	Manchas y contornos que cumplen con un valor de matiz máximo de 38,2° en el espacio de color HSV. (vista máscara imagen)	Éxito.	
3	Slider Saturación mínima.	83	Visualizar las manchas y contornos que cumplan con un valor de saturación mínima de 32% en el espacio de color HSV	Manchas y contornos que cumplen con un valor de saturación mínima de 32% en el espacio de color HSV. (vista máscara imagen)	Éxito.	
4	Slider Saturación máxima.	255	Visualizar las manchas y contornos que cumplan con un valor de saturación máxima de 100% en el espacio de color HSV	Manchas y contornos que cumplen con un valor de saturación máxima de 100% en el espacio de color HSV. (vista máscara imagen)	Éxito.	
5	Slider Brillo mínimo.	151	Visualizar las manchas y contornos que cumplan con un valor de brillo mínimo de 59%° en el espacio de color HSV	Manchas y contornos que cumplen con un valor de brillo mínimo de 59%° en el espacio de color HSV. (vista máscara imagen)	Éxito.	
6	Slider Brillo máximo.	228	Visualizar las manchas y contornos que cumplan con un valor de brillo máximo de 89% en el espacio de color HSV	Manchas y contornos que cumplen con un valor de brillo máximo de 89% en el espacio de color HSV. (vista máscara imagen)	Éxito.	Ver el resultado final de este ejemplo en la segunda imagen de la Figura 14

10.5.3.10 Sliders: <Umbralización inversa>.

Referencias:

- Caso de Uso: CU02 (Sección 8.3.3.2).
- Módulo: M02 (Sección 9.4.1.2).

Precondiciones:

- Ejecutar exitosamente la funcionalidad del caso de prueba indicado en la sección 10.5.3.5 caso 1.
- Ejecutar exitosamente el caso de prueba indicado en la sección 10.5.3.7 en cualquiera de los tres casos.
- Haber definido anteriormente la región de interés.
- Visualizar la imagen que proyecta la máscara binaria de la imagen contenida en la región de interés (sección 10.5.3.6 caso 4).

ID Prueba: PR24						
Caso	Interfaz a probar	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Slider umbral inverso mínimo.	100	Seleccionar los pixeles de las manchas contenidas en la región de interés en escala de grises, con valor de intensidad menor o igual a 155 (255 – 100).	Se visualiza manchas o figuras contenidas en la región de interés en escala de grises, con valor de intensidad menor o igual a 155	Éxito	Ver ejemplo en la Figura 18
2	Slider umbral inverso máximo	255 N/A	N/A	N/A	Fracaso	No aplica en este caso, no afecta a la funcionalidad de umbralización inversa.

10.5.3.11 Guardar imagen-figura y configuración de segmentación.

Referencias:

- Caso de Uso: CU02 y CU03 (Sección 8.3.3.2 y 8.3.3.3).
- Módulo: M02 (Sección 9.4.1.2).

Precondiciones:

- No existen precondiciones aparte de que la aplicación este en la sección de configuración remota de cámara.

ID Prueba: PR25					
Caso	Interfaz a probar	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Botón "Guardar imagen ROI"	Activar la entrada de texto para insertar un nombre a la nueva figura a registrar.	Se activa la entrada de texto para insertar un nombre para registrar una nueva figura, más el botón de confirmación.	Éxito	
2	Botón "Guardar configuración"	Activar la entrada de texto para insertar un nombre a la nueva configuración de segmentación a registrar.	Se activa la entrada de texto para insertar un nombre para registrar una nueva configuración, más el botón de confirmación.	Éxito.	

10.5.3.12 Seleccionar color por tocar sobre la imagen en cámara.

Referencias:

- Casos de Uso: CU02 (Sección 8.3.3.2).
- Modulo: M05 (Sección 9.4.1.5), LA08 (Figura 48).

Precondiciones:

- Ejecutar exitosamente la funcionalidad del caso de prueba indicado en la sección 10.5.3.5 caso 1.
- Ejecutar exitosamente el caso de prueba indicado en la sección 10.5.3.7 caso 3.

ID Prueba: PR26						
Caso	Interfaz a probar	Valor	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	TouchScreen sobre imagen proyectada por la cámara en la aplicación	Coordenadas p(x,y).	Visualizar las manchas y contornos que cumplan con un rango de valores de matiz, saturación y brillo en espacio de color HSV capturados dentro de un radio de 4 pixeles desde p(x,y).	Manchas y contornos que cumplen con un rango de valores de matiz, saturación y brillo en espacio de color HSV capturados dentro de un radio de 4 pixeles desde p(x,y).	Éxito	

10.5.3.13 Conectar dispositivo Bluetooth (Acceder a lista de dispositivos).

Referencias:
Casos de Uso: CU01 (Sección 8.3.3.1).
Módulo: M05 y M06 (Secciones 9.4.1.5 y 9.4.1.6)

Precondiciones:

- Ejecutar la aplicación en modo cámara.

ID Prueba: PR27					
Caso	Interfaz a probar	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Botón "BLUETOOTH" del toolbar	Mostrar lista de los dispositivos Bluetooth vinculados al smartphone.	Muestra la lista de los dispositivos Bluetooth vinculados al smartphone. Por cada dispositivo se da su nombre y la dirección MAC.	Éxito	

10.5.4 Prueba de seguridad.

10.5.4.1 Aviso conexión WIFI perdida entre Smartphones.

Referencias:
Módulo: M04 (Sección 9.4.1.3) en modo administrador.
Precondiciones:

- Estar conectado a un smartphone a través de comunicación WIFI (Sección 10.5.2.1).

ID Prueba: PR28					
Caso	Característica a probar	Salida esperada	Salida obtenida.	Éxito/ Fracaso	Observaciones
1	Alerta de desconexión	Aviso de desconexión. Cambiar el estado de conexión indicado en el toolbar a "Sin conexión"	El estado de conexión WIFI indicado en el toolbar de la aplicación cambia a "Sin Conexión"	Éxito	

10.5.4.2 Aviso conexión Bluetooth perdida.

Referencias:
Modulo: M05 (Sección 9.4.1.5) en modo cámara.

Precondiciones:

- Estar conectado a un dispositivo a través de Bluetooth.
- Intentar conectar a un dispositivo Bluetooth.

ID Prueba: PR29					
Caso	Característica a probar	Salida Esperada	Salida Obtenida	Éxito/ Fracaso	Observaciones
1	Alerta de desconexión (Estado de conexión Bluetooth)	Avisar que se ha perdido la conexión. Cambiar el estado de conexión indicado en el toolbar a "No conectado"	Se avisa en pantalla que se ha perdido la conexión. El estado de conexión cambia a "No Conectado"	Éxito	En este caso se pierde la conexión Bluetooth activa.
2	Alerta de desconexión (Estado de conexión Bluetooth)	Avisar que el dispositivo no se puede conectar. Cambiar el estado de conexión indicado en el toolbar de "Conectando" a "No conectado"	Se avisa en pantalla que el dispositivo no se puede conectar, el estado de conexión cambia de "Conectando" a "No Conectado"	Éxito.	En este caso se intenta conectar a un dispositivo bluetooth sin éxito.

10.6 Conclusiones de las pruebas.

Tras haber realizado y finalizado las pruebas adecuadamente se puede determinar que la aplicación cumple con los requerimientos y objetivos propuestos (Sección 4.2), siempre que sean consideradas las precondiciones necesarias para probar con éxito cada funcionalidad de la aplicación.

Las conexiones se realizan de manera rápida y estable, y el procesamiento de imágenes para la comparación de objetos elaborados en madera por el SMF del laboratorio se ejecuta exitosamente. Además, los comandos de instrucciones enviados al robot son ejecutados sin problemas y la comunicación con este se realiza de manera recíproca.

Por lo tanto, se puede concluir que la aplicación está apta para ser implantada a la estación de inspección del SMF del Laboratorio CIM de tal forma que puedan cumplirse con los requerimientos y objetivos del proyecto.

11 PLAN DE CAPACITACIÓN Y ENTRENAMIENTO.

11.1 Usuarios a capacitar.

La aplicación cuenta con un tipo de usuario de la aplicación, y es la que controla la aplicación ejecutada en los dos smartphones involucrados en el sistema. En algunos casos se requerirá de un supervisor quien se encargará de que el proceso de segmentación y control del robot se ejecute sin inconvenientes.

Se ha designado a la persona encargada del laboratorio de sistemas automatizados de producción, don Luis Vera Quiroga, como la persona a capacitar en el uso de la aplicación de manera que pueda ejecutar el proceso de comparación de figuras y control de la celda flexible de inspección a través de dispositivos móviles. La persona capacitada será el encargado de traspasar sus conocimientos en el uso de la aplicación a quien estime necesario.

11.2 Tipo de capacitación o entrenamiento.

La capacitación se realiza mediante una demostración, explicando paso a paso cada una de las funcionalidades y características de la aplicación, así como también su utilización.

La capacitación se emplazará en el mismo laboratorio de sistemas de manufactura, en la cual las personas y alumnos que trabajan en el laboratorio poseen conocimientos robótica, programación y de visión artificial en diferente grado, por lo que en la capacitación se utilizará un lenguaje de carácter técnico acorde a ello para una mayor comprensión.

El tiempo estimado a utilizar para la capacitación en el uso de la aplicación es de tres horas de tipo pedagógicas (40 minutos cada una).

Los recursos utilizados para la demostración de la ejecución de las aplicaciones serán:

- Aplicación gratuita de transmisión de pantalla a través de conexión WIFI para dispositivos Android (“Screen Stream”, aplicación disponible en Google Play), para visualizar la ejecución de la aplicación en la pantalla de un computador portátil.
- Soporte de sujeción para Smartphone para ser montado en el cabezal del brazo-robot.

11.3 Funcionalidad o aspectos a abordar en la capacitación.

- Introducción y explicación de las funcionalidades de aplicación.
- Componentes utilizados e involucrados en el sistema y su funcionalidad.
- Configuración de las conexiones entre los dispositivos, y explicación de las características de conectividad entre estos (Bluetooth y WIFI).
- Explicación de los comandos utilizados para el control del brazo-robot y el smartphone que ejecuta la aplicación en modo cámara.
- Procesamiento de imágenes, segmentación y búsqueda del objeto de interés a través de la aplicación.
- Registro y eliminación de datos (de imágenes y de configuraciones).
- Orden y ejecución del proceso de comparación de figuras y control del robot de acuerdo a los resultados obtenidos.

11.4 Responsable de la capacitación.

La capacitación inicial estará a cargo del desarrollador de la aplicación Victor Monsalves, luego las siguientes capacitaciones quedarán a cargo del jefe del laboratorio de sistemas de manufactura, don Luís Vera.

12 CONCLUSIONES.

No existe duda de que una de las ciencias que evoluciona de manera más veloz es la informática. La informática se puede encontrar en cada lugar donde podemos estar presentes, como en la electrónica, en los automóviles, en las empresas, en la educación, en la ciencia médica, en los sistemas de seguridad, etc., y por supuesto en la automatización de la industria manufacturera.

La industria manufactura cada día se encuentra en un mundo más competitivo y globalizado, por lo que los sistemas de manufactura necesitan actualizarse constantemente aplicando nuevas tecnologías disponibles en el mercado tanto en el ámbito de las maquinas como las aplicaciones de software para su control. Dentro de los procesos que se ejecutan en los sistemas de manufactura, está el proceso de inspección y control de calidad de los productos manufacturados por este.

El Laboratorio de Sistemas Automatizados de Producción de la Universidad del Bio-Bío (CIMUBB) tiene en sus dependencias un sistema de manufactura flexible SMF que simula todo un sistema de producción automatizado y controlado a través de computadores (CIM). Dado que el personal encargado del laboratorio CIMUBB se encontraba en la necesidad de actualizar y miniaturizar el sistema de control del SMF, se puso como objetivo aplicar tecnologías basadas en dispositivos móviles. Finalmente se decidió controlar el SMF del laboratorio mediante una aplicación desarrollada en Android, encargada de ejecutar los procesos de aprovisionamiento de materia prima y de mecanizado, pero faltaba controlar el proceso de visión e inspección de los productos mecanizados por el SMF a través de esta plataforma, por lo que dada la situación se propuso desarrollar una aplicación en Android que facilitara ejecutar el proceso de visión e inspección de piezas elaboradas en madera.

Para lograr el objetivo anterior se resolvió desarrollar una aplicación Android de visión artificial para la estación de inspección y control de calidad del sistema de manufactura flexible mediante la comparación de figuras (patrones) de los productos elaborados en madera y el control de la celda flexible de inspección de acuerdo a los resultados obtenidos de la comparación de imágenes. Esta aplicación hace uso de conexión remota por medio de conectividad Bluetooth y WIFI, y de visión artificial utilizando la librería OpenCV para facilitar la etapa del procesamiento de imágenes. Para el desarrollo de la aplicación se vio la necesidad de investigar las tecnologías y componentes involucrados en el control de la celda flexible de inspección del SMF, en la visión artificial y en la conectividad inalámbrica, de manera de aplicarla de la mejor manera posible ante el problema que se quiso resolver.

A lo largo de este proyecto se trataron diversos temas como los sistemas de manufactura asistidos por computador, la robótica, la conectividad móvil y la visión artificial. Gracias a la metodología de desarrollo utilizada, la aplicación fue desarrollada de manera evolutiva permitiendo involucrar cada uno de estos temas en cada iteración de manera de corregir las iteraciones anteriores que no pudieran consistir con los resultados de la última iteración o etapa desarrollada de la aplicación, logrando la integración de cada etapa de manera óptima ayudando a cumplir con el principal objetivo del proyecto:

"Desarrollar una aplicación de visión artificial para la estación de inspección y control de calidad de un sistema de manufactura flexible mediante la comparación de figuras o patrones de los productos elaborados en madera y el control de la celda flexible de inspección, integrando los dispositivos móviles a los sistemas de producción y de control de calidad, con el objetivo de modernizar la plataforma tecnológica utilizada el Laboratorio de Sistemas Automatizados de Producción (CIMUBB) de la Universidad del Bio-Bío".

y a la vez el objetivo del producto aplicación:

"Establecer un sistema de inspección y control de calidad de los productos elaborados en madera en el sistema de manufactura flexible SMF, de acuerdo a sus respectivas figuras y patrones a través de dispositivos móviles, cuyo resultado será el porcentaje de similitud entre dos figuras capturadas y su identificación".

El desarrollo de la aplicación es importante para el sistema de manufactura flexible, en especial en su estación de inspección visual ya que, logra dar un enfoque más moderno y vanguardista al laboratorio al incluir tecnología móvil del smartphone actualizando el hardware utilizado en los procesos de manufactura. Esta tecnología está basada en la miniaturización y la utilidad, principalmente en el uso de comunicación inalámbrica a través del uso Bluetooth y de redes WIFI con las que se logra un rápido intercambio de datos y la vez, y la integración de las cámaras digitales.

Tras la implementación del proyecto se ha podido simplificar el trabajo de inspección disminuyendo costos y espacio involucrados en hardware, evitando el uso de computadores, de cámaras web y de cables de red, además, de facilitar el proceso de inspección y control del robot por parte del usuario.

Los objetivos propuestos en este proyecto fueron alcanzados satisfactoriamente debido al esfuerzo plasmado en su desarrollo y las pruebas realizadas en cada etapa por lo que aplicación ha sido instalada en cada uno de los dispositivos móviles pertenecientes al laboratorio, y el código fuente y el archivo de instalación (APK) almacenados en el computador del encargado del laboratorio para futuras actualizaciones. Aun así, se propone como trabajos futuros que logren enriquecer la aplicación con nuevas características como la integración de la aplicación con las demás estaciones del SMF del CIM, y la comunicación y almacenamiento de datos a través de internet. Sin embargo, es importante notar que pueden existir factores externos al sistema que pueden afectar el correcto funcionamiento de la aplicación como, por ejemplo: deficiencias del área de inspección o nula presencia de objetos, no cumplir con las precondiciones de operación, y problemas en la celda flexible que es el principal componente actuador del sistema encargado de ejecutar las instrucciones finales en el movimiento de objetos.

Gracias a la realización de este proyecto de título se permitió insertar en el mundo de la programación de aplicaciones basadas en Android, un área nueva en la cual se permitió profundizar los conocimientos en la programación en el lenguaje Java, aprender a implementar funciones de visión artificial utilizando la librería OpenCV para aplicaciones móviles, e interactuar con dispositivos periféricos como por ejemplo con brazos robot a través de un smartphone , cosas que a lo largo de la carrera no se ha tenido la oportunidad de trabajar.

Con el aprendizaje adquirido a lo largo de este proyecto logró tener en cuenta la importancia de la informática en la automatización de los procesos en el área manufacturera y como el país se atreve a invertir en tecnología de la cual para su implementación se necesita profesionales en el área de la informática.

Todo esto hicieron que este proyecto fuera experiencia enriquecedora, tanto en el área profesional como personal.

13 REFERENCIAS

(11 de 2016). Obtenido de Sistemas CIM y ERP para la Industria: http://148.204.211.134/polilibros/portal/polilibros/P_Terminados/admonInformII-Gaona/POLILIBRO/UMD/UNIDAD%202/SW.htm

Android Developers. (2016). *Android Studio*. Obtenido de https://developer.android.com/studio/index.html

Automatización. (Marzo de 2017). Obtenido de http://www.uhu.es/rafael.sanchez/ingenieriamaquinas/carpetaapuntes.htm/Trabajos%20IM%202009-10/Carlos%20Tutosaus-introduccion_automatizacion.pdf

Basler AG. (2017). *Baslerweb*. Obtenido de Baslerweb: https://www.baslerweb.com/en/service/glossary/

Basler AG. (2017). *Baslerweb Markets*. Obtenido de Baslerweb Markets: https://www.baslerweb.com/en/markets/

CIMUBB. (Diciembre de 2016). *Laboratorio de Sistemas Automatizados de Producción*. Obtenido de http://www.cimubb.ubiobio.cl

Dawson-Howe, K. (2014). MeanShift. En K. Dawson-Howe, *A Practical Introduction to Computer Vision With OpenCV*. Ireland: John Wiley & Sons Ltd.

Eshed Robotec. (2000). *Scorbot ER-V Plus, Manual de Usuario* (N° Cat. 100265 Rev.A ed.).

Genymobile. (2017). *Genymotion*. Obtenido de https://www.genymotion.com

Gonzalez, R. C., & Woods, R. E. (2002). Fundamental Steps in Digital Image Processing. En R. C. Gonzalez, & R. E. Woods, *Digital Image Processing* (págs. 25-27). Prentice Hall.

Image Thresholding . (11 de 2016). Obtenido de http://docs.opencv.org/3.2.0/d7/d4d/tutorial_py_thresholding.html

Institute for Electrical and Electronics Engineers. (1998). *IEEE (830-1998 - IEEE Recommended Practice for Software Requirements.*

Kaehler, A., & Bradski, G. (2016). Matching contours and shapes. En A. Kaehler, & G. Bradski, *Learning OpenCV 3, Computer Vision in C++ with the OpenCV Library* (Primera ed., págs. 569-576). O'Reilly.

Kalpakjian, S., & Schmid, S. R. (2002). Manufactura integrada por computadora. En S. Kalpakjian, & S. R. Schmid, *Manufactura, Ingeniería y Tecnología* (págs. 24-25). Pearson Educacion.

Kalpakjian, S., & Schmid, S. R. (2002). Manufactura Integrada por Computadora. En S. Kalpakjian, & S. R. Schmid, *Manufactura, Ingeniería y Tecnología.* (págs. 24-25). Pearson Education.

Laboratorio CIMUBB. (2017). *Organigrama del laboratorio de Sistemas Automatizados de Producción de la Universidad del Bio-Bío.* Obtenido de http://www.ubiobio.cl/miweb/web2012.php?id_pagina=942

Laboratorio de Sistemas Automatizados de Producción CIMUBB. (2016). Comunicación CIM. *Layout laboratorio CIMUBB.*

Larman. (2003). *Elementos de Casos de Uso.*

Martinez, J. (Noviembre de 2016). *Industria 4.0.* Obtenido de Industria 4.0,Como asegurar el posicionamiento competitivo: http://www.ainia.es/insights/industria-4-0-como-asegurar-el-posicionamiento-competitivo-futuro/

Mateos, G. G. (06 de 2017). *Procesamiento de imágenes. Filtros y transformaciones locales*. Obtenido de http://slideplayer.es/slide/4747946/

Mery, D. (2004). *Visión por Computador*. Santiago.

Morfología. (Septiembre de 2016). Obtenido de Procesamiento de imagenes digitales: http://alojamientos.us.es/gtocoma/pid/tema5-1.pdf

Muhammad, A. (2015). Appling perspective correction. En A. Muhammad, *OpenCV Android Programing By Example* (págs. 152-168). PacktPublishing.

OpenCV Dev Team. (2016). *Imgproc, structural analysis and shape descriptors.* Obtenido de http://docs.opencv.org/2.4/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html#findcontours

OpenCV Team. (2017). *OpenCV Library*.

Sistemas CIM y ERP para la Industria. (11 de 2016). Obtenido de http://148.204.211.134/polilibros/portal/polilibros/P_Terminados/admonInformII-Gaona/POLILIBRO/UMD/UNIDAD%202/SW.htm

Universidad del Bio-Bío. (2017). *Misión y Visión*. Obtenido de http://www.ubiobio.cl/w/#Vision_y_Mision

Universidad del Bio-Bío. (2017). *Estructura Organica de la Universidad del Bio-Bío*. Obtenido de http://ubiobio.cl/miweb/web2012.php?id_pagina=5152

Vide Imaging Design Ware. (12 de 2016). Obtenido de http://videsignwire.com/cmos-image-sensor-processing-with-fpgas/

14 ANEXO 1: PROCESAMIENTO DIGITAL DE IMÁGENES.

14.1 Operaciones morfológicas.

La morfología matemática está basada en operaciones en la teoría de conjuntos (Morfología, 2016).

Las operaciones morfológicas simplifican imágenes y conservan las principales características de forma de los objetos. Un sistema de operadores de este tipo y su composición, permite que las formas subyacentes sean identificadas y reconstruidas de forma óptima a partir de sus formas distorsionadas y ruidosas.

La morfología matemática se puede usar, entre otros, con los siguientes objetivos:

- Pre-procesamiento de imágenes (supresión de ruidos, simplificación de formas).
- Destacar la estructura de los objetos (extraer el esqueleto, detección de objetos, envolvente convexa, ampliación, reducción,...)
- Descripción de objetos (área, perímetro,...)

Operaciones básicas sobre conjuntos:

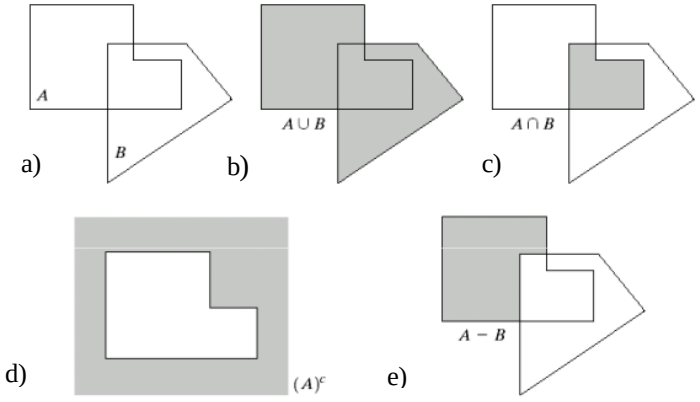


Figura 52. Operaciones básicas sobre la teoría de conjuntos.

Según las operaciones básicas sobre conjuntos la Figura 52 señala:

a) Conjunto de elementos de A y B.
b) La unión de A y B, suma de los elementos que pertenecen a A, B y a ambos.
c) La intersección de los elementos que pertenecen a A y B a la vez.
d) El complemento de A, todos los elementos que no pertenecen a A.
e) La diferencia entre A y B, todos los elementos que pertenecen a A pero no a B.

A continuación, explicaremos sobre las dos operaciones morfológicas utilizadas para mejorar la definición de una mancha o figura en una imagen binarizada ya sea erosionando y dilatación de una imagen, utilizando las operaciones variantes de ellas que son apertura y clausura con el fin de eliminar manchas insignificantes que puedan interferir en la calidad de los procesos de segmentación, detección y comparación de figuras y patrones.

Dilatar:
Dada una imagen A, y un elemento estructural B, (ambas imágenes binarias con fondo blanco), la dilatación de A por B se define como:

$$A \oplus B = \left\{ x \middle| \left(\hat{B}\right)_x \cap A \neq \varnothing \right\}$$

Tengamos en cuenta que, para la intersección sólo consideramos los píxeles negros de A y B. El primer elemento de la dilatación, A, está asociado con la imagen que se está procesando y el segundo recibe el nombre de elemento estructural, la forma que actúa sobre A en la dilatación para producir A⊕ B.

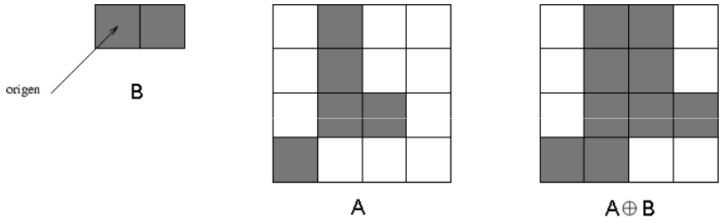


Figura 53. Operación de Dilatación A⊕ B.

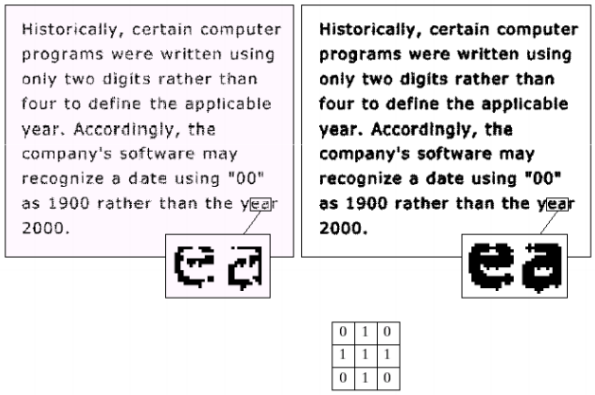


Figura 54. Ejemplo de aplicación de dilatación sobre un texto.

En la Figura 53 y Figura 54 los elementos destacados en la imagen están dilatados o expandidos de acuerdo a la matriz **B** de 3x3, y la imagen de la izquierda representa **A** resultando A⊕ B representado en la imagen de la derecha.

En el proceso de erosión y dilatación es necesario establecer una matriz a la que corresponde al nucleo o Kernel de (n * m) pixeles que será el principal parámetro para definir la distancia espacial de dilatación desde el pixel de origen.

Por ejemplo, si definimos un Kernel de **m*n**, entonces el punto de origen se expandirá a una distancia horizontal de **n** pixeles y a una distancia vertical de **m**.

Erosionar:

Dada una imagen A, y un elemento estructural B, (ambas imágenes binarias con fondo blanco), la erosión de una imagen, A, por un elemento estructural, B, es el conjunto de todos los elementos x para los cuales B trasladado por x está contenido en A:

$$A \ominus B = \{x \mid Bx \subseteq A\}$$

Tengamos en cuenta que, para la condición $Bx \subseteq A$, sólo consideramos los píxeles negros de A y B. La erosión es la operación morfológica dual de la dilatación. La erosión se concibe usualmente como una reducción de la imagen original.

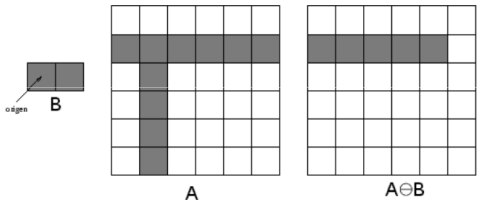


Figura 55. Operación de erosión A Θ B.

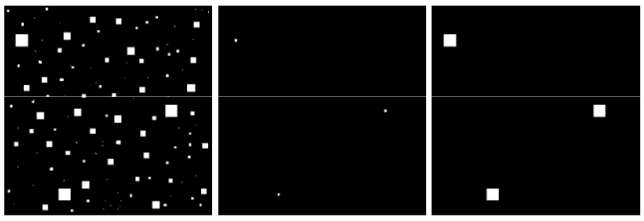


Figura 56. Proceso de erosión y dilatación de una imagen binarizada.

En la Figura 56 en la primera imagen hay cuadrados de tamaño de 1,3,5,7,9,y 15 pixeles por lado , en la segunda imagen la imagen esta erosionada mediante una distancia de pixeles , es decir a un Kernel de tamaño de 13*13, resultando solo cuadrados de tamaño 2*2 y en la tercera imagen a los cuadrados de 2*2 de la segunda imagen son dilatados a una distancia de 13*13, quedando los cuadrados de tamaño 15*15.

Mientras que la dilatación puede representarse como la unión de los trasladados, la erosión puede representarse como la intersección de los trasladados negativos:

$$A \ominus B = \bigcap_{b \in B} A_{-b}$$

La dilatación y la erosión son muy similares en el sentido de que lo que uno hace al objeto el otro lo hace al fondo. Esta relación puede formularse como una relación de dualidad:

Teorema. Dualidad de la erosión y la dilatación.

$$(A \ominus B)^C = A^C \oplus \hat{B} \Rightarrow (A \oplus B)^C = A^C \ominus \hat{B}$$

Propiedades de la erosión.

1.- No es conmutativa: AΘB ≠ BΘA.

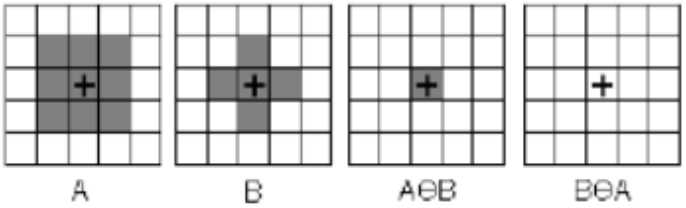


Figura 57. Propiedad de inconmutabilidad de la erosión.

2.- Propiedad Distributiva respecto a la intersección:

$$(A \cap B) \ominus K = (A \ominus K) \cap (B \ominus K)$$

3.- Al igual que la dilatación, la erosión es también creciente:

$$A \subseteq C \longrightarrow A \ominus B \subseteq C \ominus B$$

4.- Además la erosión por un elemento estructural mayor produce un resultado menor:

$$K \subseteq L \Longrightarrow A \ominus L \subseteq A \ominus K$$

5.- Finalmente, con respecto a la descomposición de elementos estructurales, una regla de la cadena para la erosión se verifica cuando el elemento estructural se puede descomponer mediante dilatación.

$$A \ominus (B \oplus C) = (A \ominus B) \ominus C$$

Apertura.

La apertura corresponde al proceso de erosión seguida por una dilatación. Es útil para eliminar el ruido o manchas blancas no deseadas. El Kernel es el mismo en la erosión y dilatación.
Mancha la definiremos como conjunto de pixeles vecinos con características similares en una imagen binaria.

Figura 58. Aplicación de apertura en una imagen binaria.

Clausura.

Es el proceso inverso a apertura, dilatación seguida por una erosión. Es útil para cerrar pequeños agujeros dentro de los objetos en primer plano, o pequeñas manchas negras en el objeto. Al igual que en el proceso de apertura el valor del Kernel es el mismo en la erosión y dilatación.

Figura 59. Aplicación de clausura en una imagen binaria.

Para mejorar la calidad de una imagen binaria de manera de reducir la mayor cantidad de ruido de la imagen sin afectar la estructura de las manchas y patrones se aplicaron los algoritmos de apertura y clausura con una distancia de 5*5 (Kernel de 5 x 5), tanto en la definición del ROI como en la búsqueda de manchas y patrones.

14.2 Transformaciones geométricas y afines.

El objetivo de estas transformaciones es trasladar un conjunto de pixeles contenido en una región de interés desde un punto de origen a un punto de destino, corregir distorsiones dentro de la imagen, editar, etc.

Algunas de las operaciones de estas transformaciones son rotación traslación, escalamiento, y transformaciones matemáticas afín (Muhammad, 2015).

- **Traslación:** Básicamente lo que se hace es mover cada pixel $\boldsymbol{p}(\boldsymbol{x}, \boldsymbol{y})$ en una cantidad $\boldsymbol{t}(\boldsymbol{tx}, \boldsymbol{ty})$.

  Podemos escribir la traslación de un pixel como $\boldsymbol{p'} = \boldsymbol{p} + \boldsymbol{t}$.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- **Rotación:** En la transformación de rotación, cada pixel va seguido con una traslación. Esta transformación es conocida también como transformación Euclidiana bidimensional. Las distancias espaciales entre los pixeles se conservan. Esta rotación se puede representar como: $\boldsymbol{p' = Rp}$, donde R es una matriz de 2x2 que equivale a:

  $$R = \begin{bmatrix} cos\theta & -sen\theta \\ sen\theta & cos\theta \end{bmatrix}$$

  , donde $\boldsymbol{\theta}$ es el ángulo de rotación.
  Donde la siguiente ecuación representa la transformación de rotación para cada pixel $p(x, y)$.

  $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} cos\theta & -sen\theta \\ sen\theta & cos\theta \end{bmatrix} * \begin{bmatrix} x \\ y \end{bmatrix}$$

  Por lo tanto: $x' = xcos(\theta) - ysen(\theta)$, $y' = xsen(\theta + ycos(\theta)$

- **Rotación escalada**: También conocida como transformación de similitud, esta transformación agrega un factor de escala **s** quedando expresada como:
  $\boldsymbol{p' = sRp,}$ esta conserva el ángulo entre las líneas.

  La fórmula que representa a la operación de escalamiento sea de extender o reducir el tamaño un conjunto de pixeles está dado por la siguiente ecuación:

  $$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

  Donde s_x es el factor de escalado en el eje horizontal y s_y en el eje vertical.

- **Transformación afín o paralela**: Las líneas paralelas permanecen paralelas y se expresa como:

  $p' = Ap^{\sim}$, donde $p^{\sim} = [x, y, 1]$, y $A = \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}$

En la Figura 60 se describe las transformaciones mencionadas anteriormente sobre una región y sus respectivas transformaciones de traslación, euclídea, similitud, afín y proyectiva.

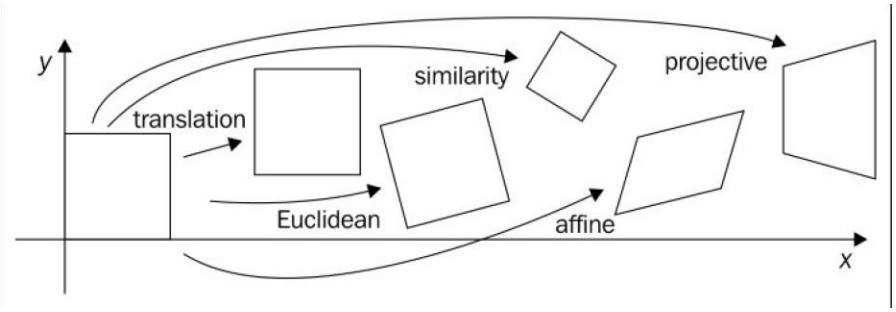


Figura 60. Transformaciones geométricas dentro del plano cartesiano.

14.3 Distancia euclidiana.

Para dos imágenes p, q y z, con coordenadas $p(x, y)$, $q(s, t)$,y $z(v, w)$ respectivamente, D representa la distancia si:

a) $D(p, q) \geq 0$
b) $D(p, q) = 0 \quad \leftrightarrow \quad p = q$
c) $D(p, q) = D(q, p)$

d) $D(p,z) \leq D(p,q) + D(q,z)$.

La distancia euclidiana entre una imagen p y q es definida como:

$$De = [(x-s)^2 + (y-t)^2]^{\frac{1}{2}}$$

Los píxeles que tienen una distancia menor o igual a algún valor r de (x, y) son los puntos contenidos en un área circular de radio r centrado en (x, y).

Un método para comparar y clasificar vectores de patrones es el método de clasificación de menor distancia del cual se efectúa mediante el cálculo de la distancia euclidiana normalizada.

14.4 Algoritmo de detección de bordes de Canny.

Detector de Canny es un popular algoritmo de detección de bordes. Fue desarrollado por John F. Canny. Es un algoritmo multi-etapas las cuales son:

- Reducción del ruido.
  Puesto que la detección del borde es susceptible al ruido en la imagen, el primer paso es quitar el ruido en la imagen con un filtro de Gaussian de un Kernel 5x5.

- Encontrar el gradiente de intensidad de la imagen.
  La imagen suavizada se filtra con un kernel Sobel en dirección horizontal y vertical para obtener la primera derivada en dirección horizontal (Gx) y vertical (Gy). A partir de estas dos imágenes, podemos encontrar gradiente de borde y dirección para cada píxel de la siguiente manera:

$$Gradiente\_Borde\ (G) = \sqrt{G_x^2 + G_y^2}$$

$$Angulo(\theta) = tan^{-1}\left(\frac{G_y}{G_x}\right)$$

  La dirección del gradiente es siempre perpendicular a los bordes. Se redondea a uno de los cuatro ángulos que representan la dirección vertical, horizontal y dos diagonales.

- Supresión no-máxima.
  Después de obtener el gradiente de magnitud y dirección, se realiza una exploración completa de la imagen para eliminar los píxeles no deseados que pueden no constituir el borde. Para ello, en cada píxel, el píxel se comprueba si es un máximo local en su vecindario en la dirección del gradiente, como lo indica en la Figura 61:

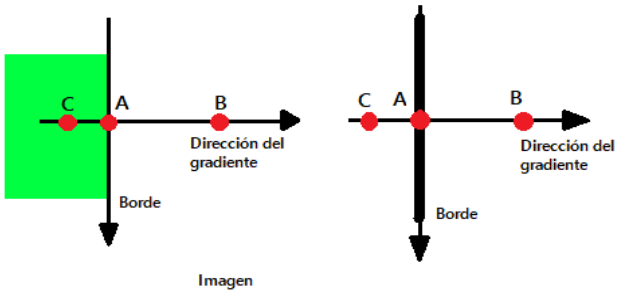


Figura 61. Supresión no máxima.

El punto A está en el borde (en dirección vertical). La dirección del gradiente es normal al borde. Los puntos B y C están en direcciones de gradiente. Así que el punto A se verifica con los puntos B y C para ver si forma un máximo local. Si es así, se considera para la siguiente etapa, de lo contrario, se suprime.

- Umbral de Histéresis.
  En esta etapa se decide cuáles de todos los bordes son realmente bordes y cuáles no. Para ello, se necesitan dos valores de umbral, minVal(umbral mínimo) y maxVal(umbral máximo). Todos los bordes con gradiente de intensidad sobre maxVal serán los bordes seguros y aquellos por debajo de minVal no serán bordes, por lo que se descartan. Aquellos que se encuentran entre estos dos umbrales son bordes clasificados o no-bordes en función de su conectividad. Si están conectados a píxeles "seguros", se considera que forman parte de los bordes. De lo contrario, también se descartan, véase en la Figura 62:

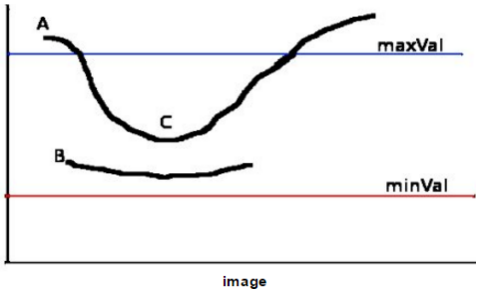


Figura 62. Umbral de histéresis.

El borde A está por encima del maxVal, por lo que se considera como "borde-seguro". Aunque el borde C está por debajo de maxVal, está conectado al borde A, por lo que también se considera como borde válido y obtenemos esa curva completa. Pero el borde B, aunque está por encima de minVal y está en la misma región que el del borde C, no está conectado a ningún "borde-seguro", por lo que se descarta. Por lo tanto, es muy importante que tengamos en consecuencia que seleccionar un umbral mínimo y máximo (minVal y maxVal) para obtener el resultado correcto.

Esta etapa también elimina pequeños ruidos de píxeles en el supuesto de que los bordes son largas.

14.5 Operaciones lógicas sobre imágenes binarias de bit a bit.

En la programación digital de computadores una operación lógica de bit a bit opera en uno o más patrones de bits o números binarios en el nivel de los bits individuales Es una operación rápida y sencilla directamente soportada por el procesador, y es utilizado para manipular valores binarios para realizar cálculos y comparaciones.

En el ámbito del procesamiento de imagen, esta operación es posible demostrarla mediante imágenes mediante comparación de bits.

Por medio del uso de la librería OpenCV es posible proyectar una imagen utilizando máscaras binarias y comparar imágenes en colores, en escala de grises y binarias mediante la función de OpenCV (“Core.bitwise”).

En una imagen digital la cantidad de bits contenidos en cada pixel depende de la cantidad y calidad de los datos almacenados en una imagen. En una imagen a color de tipo RGBA en que sus datos están contenidos en cuatro canales (rojo, verde, azul y Alpha (transparencia)) con valores de intensidad de 0 a 255, lo que cada canal posee 8 bits, ocupando un total de 32 bits por pixel. En una imagen en escala de grises de un solo canal de intensidad de valores 0 a 255 posee 8 bits por pixel.

En las imágenes binarias o de tipo mapa de bits, cada pixel solo puede contener dos valores 0 y 1 o bien 0 y 255.

Operadores:
NOT: La función bit a bit NOT, o complemento, es una operación unaria que realiza la negación lógica en cada bit correspondiente, formando el complemento del valor binario dado. Los bits que son 0 se convierten en 1 y los que son 1 se convierten en 0. Por ejemplo:

$$\sim \mathbf{0111} = \mathbf{1000}$$

AND: La operación lógica de conjunción AND toma dos representaciones binarias de igual longitud y realiza la operación lógica AND en cada par de los bits correspondientes, multiplicándolos. Así, si ambos bits en la posición comparada son 1, el bit en la representación binaria resultante es 1 (1 × 1 = 1); En caso contrario, el resultado es 0 (1 × 0 = 0 y 0 × 0 = 0). Por ejemplo:

$$010\mathbf{1} \wedge 001\mathbf{1} = 0001$$

OR: La operación lógica de disyunción OR toma dos patrones de bits de igual longitud y realiza la operación lógica OR en cada par de bits correspondientes. El resultado en cada posición es 0 si ambos bits son 0, mientras que de lo contrario el resultado es 1. Por ejemplo:

$$0\mathbf{101} \vee 0\mathbf{011} = 0\mathbf{111}$$

XOR: La operación lógica XOR toma dos patrones de bits de igual longitud y realiza la operación OR exclusiva en cada par de bits correspondientes. El resultado en cada posición es 1 si sólo el primer bit es 1 o sólo el segundo bit es 1, pero será 0 si ambos son 0 o ambos son 1. En esto realizamos la comparación de dos bits, siendo 1 si los dos Bits son diferentes, y 0 si son iguales. Por ejemplo:

$$0\mathbf{10}1 \oplus 0011 = 0110$$

En la Figura 63 explica las operaciones lógicas realizadas en una imagen binaria de negación, conjunción, disyunción y disyunción exclusiva.

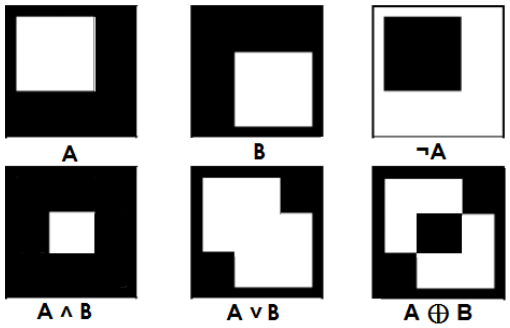


Figura 63. Representación de operaciones lógicas bit a bit sobre imágenes binarias.

14.6 Detección y cálculo de contornos.

Un contorno representa el borde externo de una figura geométrica, en una imagen para este proceso representa un conjunto de pixeles localizados en el borde externo de una mancha, o figura.

La imagen a analizar debe ser binaria, es decir de valores 1 y 0 ó 0 y 255, de un canal y de 8 bits por pixel, en el caso de que la imagen corresponda a una imagen a color se debe utilizar el algoritmo de detección de bordes de Canny.

La función del módulo *"imgproc"* de la librería OpenCV *"findContours()"* computa contornos desde una imagen binaria. Ya sea creadas mediante las funciones de detección de bordes de Canny, y de umbralizacion.

La salida de la función *findContours()* corresponde a una lista que almacena en cada una de sus posiciones un vector que contiene una cantidad finita de puntos que corresponde a cada pixel que se sea vecino entre si y que tengan su mismo valor binario, además cada punto tiene relación con un pixel vecino anterior y un pixel vecino siguiente formando una curva ,en el que el ultimo pixel, su pixel vecino siguiente corresponde al pixel inicial de la curva. En OpenCV para Java esta salida es almacenada en un objeto llamado "MatOfPoint", cuyo objeto permite realizar todos los análisis y modificaciones correspondientes a cada uno de los contornos, además permitiendo encontrar patrones de imágenes y figuras geométricas que sirven para el desarrollo del proyecto, (OpenCV Dev Team, 2016).

Los parámetros utilizados en la función anteriormente mencionada son:

- Imagen binaria de entrada (Mat).
- Lista de contornos vacía (MatOfPoint).
- Matriz de jerarquía de los contornos, establece el orden de los contornos (Objeto Mat).
- Valor índice del método de recuperación de los contornos.
- Valor índice del método de aproximación de contornos.

Los métodos de recuperación de contornos son los siguientes:

- CV_RETR_EXTERNAL recupera sólo los contornos exteriores extremos. Establece la jerarquía [i] [2] = jerarquía [i] [3] = - 1 para todos los contornos.
- CV_RETR_LIST recupera todos los contornos sin establecer ninguna relación jerárquica.
- CV_RETR_CCOMP recupera todos los contornos y los organiza en una jerarquía de dos niveles. En el nivel superior, hay límites externos de los componentes. En el segundo nivel, hay límites de los agujeros. Si hay otro contorno en el interior de un agujero de un componente conectado, se coloca en el nivel superior.
- CV_RETR_TREE recupera todos los contornos y reconstruye una jerarquía completa de contornos anidados

Los métodos de aproximación de contornos son:

- CV_CHAIN_APPROX_NONE almacena absolutamente todos los puntos del contorno. Es decir, cualquiera de los dos puntos posteriores (x1, y1) y (x2, y2) del contorno serán vecinos horizontales, verticales o diagonales, es decir, max (abs (x1-x2), abs (y2-y1) == 1.
- CV_CHAIN_APPROX_SIMPLE comprime segmentos horizontales, verticales y diagonales y deja sólo sus puntos finales. Por ejemplo, un contorno rectangular arriba-derecho se codifica con 4 puntos.
- CV_CHAIN_APPROX_TC89_L1, CV_CHAIN_APPROX_TC89_KCOS aplica uno de los algoritmos de aproximación de la cadena Teh-Chin.

14.7 Momentos invariantes de HU.

Los momentos invariantes de Hu [1962] son utilizados como descriptores de imágenes, figuras y objetos debido a su robusto soporte matemático. Específicamente por que provienen de los momentos estadísticos que describen a cada objeto en 2D unívocamente con un valor en esos descriptores. Los momentos estadísticos se utilizan para describir regiones 2D y son aplicados al análisis de formas, reconocimiento de patrones, análisis de texturas, y en algunas ocasiones los utilizan para el reconocimiento de formas 3D con restricciones. Para el caso de señales continuas de dos dimensiones se describen de la siguiente manera:

$$m_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^p f(x,y)\, dx\, dy$$

Para ***p, q*** = *0,1,2…*

Debido a que **p** y **q** toman todos los valores enteros no negativos, se ha demostrado que dichos índices generan un conjunto infinito de momentos que determinan unívocamente cada función $f(x,y)$,e inversamente dichos valores quedan determinados por $f(x,y)$. Para el caso de imágenes digitales donde la señal es discreta representada por un conjunto de pixeles, la integral es remplazada por una sumatoria:

$$M_{pq} = \sum_{x}^{nx} \sum_{y}^{ny} x^p y^q f(x,y)$$

Donde $\boldsymbol{p}$, $\boldsymbol{q}$ = *0, 1 ,2 ...*
$\boldsymbol{nx}$ = ancho de la imagen.
$\boldsymbol{ny}$ = largo de la imagen.

El momento m_{00} describe el área de $f(x,y)$ en pixeles y la relación con los momentos m_{10} y $m_{01,}$ describen el centroide de $f(x,y)$ como $(\bar{x}\ ,\bar{y})$ donde:

$$\bar{x} = \frac{m10}{m00},\ \bar{y} = \frac{m01}{m00}$$

Debido a que los momentos dependen de la posición de la región de $f(x,y)$ se utilizan los momentos centrales que son invariantes a traslaciones de una región en diferentes posiciones dentro de la imagen. Los momentos centrales se definen como:

$$\mu_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x - \bar{x})^p (y - \bar{y})^q f(x,y)\, dx\, dy$$

En el caso discreto como:

$$\sum_{x=-\infty}^{\infty} \sum_{y=-\infty}^{\infty} (x - \bar{x})^p\ (y - \bar{y})^q\ f(x,y)$$

Con la ayuda de la definición de las coordenadas del centro de gravedad o centroide de una región se definen los momentos que son invariantes a la traslación.

$$\mu_{pq} = \sum_{x,y \in R} (x - \bar{x})^p\ (y - \bar{y})^q$$

Los parámetros p y q denotan el orden del momento.
Los momentos η_{ij} invariantes a la traslación y al escalamiento pueden ser construidos a partir de los momentos centrales dividiéndose a través de un momento central cero escalado adecuadamente:

$$\eta_{ij} = \frac{\mu_{ij}}{\mu_{00}^{\left(1+\frac{i+j}{2}\right)}}$$

Donde $i + j \geq 2$. Nótese que la invariancia de traslación sigue solo usando directamente momentos centrales.

En la comparación de manchas, figuras o patrones, se extraen características derivadas de los momentos centrales e invariantes al escalamiento mencionados anteriormente, estos corresponden a los siete Momentos Invariantes de Hu [Ming-Kuei Hu ,1962] y que son calculados mediante las siguientes ecuaciones:

$$\phi_1 = \eta_{20} + \eta_{02}$$

$$\phi_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}$$

$$\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$

$$\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$

$$\phi_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} - \eta_{03})^2] + 3(\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})$$

$$\phi_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$

$$\phi_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\ - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

Con: $\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}$, con $\gamma = \frac{p+q}{2} + 1$, para p + q = 2,3, …

El parámetro γ es aquella constante que determina la constante del centro de masa del objeto en análisis. Por cada momento se obtiene un valor característico.

Los momentos invariantes de Hu son invariantes a la rotación, traslación y escalamiento, es decir si se comparan dos patrones que tengan la misma forma, pero de tamaño, posición y orientación distinta, los valores de los 7 momentos de Hu serán semejantes.

15 ANEXO 2: CARACTERÍSTICAS Y OPERACIÓN DEL SCORBOT ER-V.

15.1 Descripción.

El brazo robot de la celda Scorbot ER-V Plus es un robot vertical articulado con compuesto de cinco partes fijas y móviles, y cinco juntas rotativas, con el agregado de la mordaza por lo que el robot posee seis grados de libertad (Eshed Robotec, 2000).
Como lo indica la Figura 64 las partes componentes del brazo robot son las siguientes:

- Base.
- Cuerpo (Body).
- Brazo superior (Upper arm).
- Antebrazo (Fore arm).
- Pinza (Gripper).

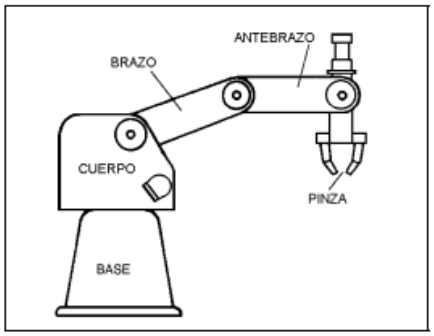


Figura 64. Bielas componentes del brazo robot.

Y las cinco juntas rotativas del robot están indicadas en la Figura 65:

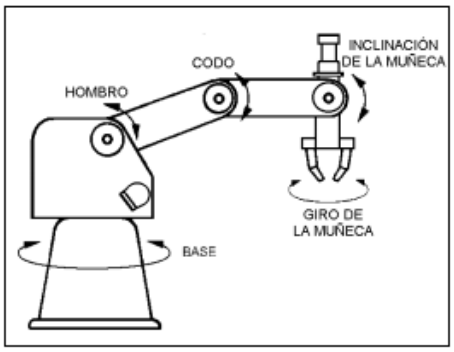


Figura 65. Juntas rotativas del brazo robot.

Los movimientos que las juntas del brazo robot pueden ejecutar están descritas en la siguiente tabla:

Número de eje	Nombre de la junta	Acción
1	Base	Gira el cuerpo (body)
2	Hombro (shoulder)	Sube o baja el brazo superior (upper arm)
3	Codo (elbow)	Sube o baja el antebrazo (fore arm)
4	Inclinación pinza (wrist pitch)	Sube o baja la inclinación de la pinza (gripper)
5	Girar pinza (wrist roll)	Gira la pinza (gripper)

Tabla 16. Tabla explicativa de cada junta del brazo robot y su acción.

Controlador.

El controlador cuenta con 16 entradas (destinadas a recibir señales de dispositivos externos que interactúen con el robot, como sensores, pulsadores, etc.) y 16 salidas (destinadas a transmitir señales hacia dispositivos externos actuadores) de las cuales 4 son a relay y las restantes son de transistor. Los parámetros del controlador incluyen constantes proporcionales, diferenciales e integrales para cada uno de los ejes, protecciones térmicas, de impacto, limites, compensado de centro de herramientas parámetros de la pinza y varios otros cálculos de parámetros. Existen 200 parámetros de control que pueden ser manejados por el usuario.

En el controlador hay un botón de parada de emergencia, este una vez presionado desconecta el sistema y se enciende una luz roja. Para volver al funcionamiento normal se deberá presionar dicho botón nuevamente y reinicializar el sistema.
El controlador incluye el lenguaje ACL que aporta las capacidades de programación avanzada. Permitiendo el control de la trayectoria continua, interpolación lineal y circular, también permite optimización de trayectorias. El efecto final de la trayectoria se puede definir como una formula general del trabajo del robot.

15.2 Especificaciones.

15.2.1 Especificaciones técnicas.

Estructura mecánica	Vertical articulado
Numero de ejes	5 ejes más la servo-pinza
Movimiento de los ejes Eje 1: Rotación Base Eje 2: Rotación Hombro Eje 3: Rotación Codo Eje 4: Rotación Elevar pinza Eje 5: Rotación Girar Pinza	 310° +130° / -35° ±130° ±130° Ilimitado mecánicamente, ±570° eléctricamente
Radio máximo de operación	610 mm
Transmisión	Correas dentadas, engranajes y tornillo de avance
Apertura máxima de la pinza	75mm sin almohadillas goma 65mm con almohadillas de goma
Sincronización (HOME) del robot	Búsqueda del punto 0 de referencia en todos los ejes, usando un micro interruptor.
Retroalimentación	Codificador (encoder) óptico de alta resolución en cada eje
Actuadores	6 motores servo de 12 Volt DC con control de lazo cerrado
Capacidad máxima de carga	1 Kg
Repetitividad de posición	0.5 mm
Peso	11.5 Kg
Velocidad máxima	600 mm por segundo
Temperatura ambiente de funcionamiento	2°C a 40°C

Tabla 17. Especificaciones técnicas del brazo-robot Scorbot ER-V.

15.2.2 Rango de operación.

La longitud de los enlaces y el grado de rotación de las juntas determinan el campo de trabajo de robot. A continuación, se muestran las dimensiones y el alcance del SCORBOT-ER V Plus.
La base del robot está normalmente fijada a una superficie de trabajo estática. Puede, sin embargo, ser unido a una base lineal, lo que resulta en un rango de trabajo ampliado.

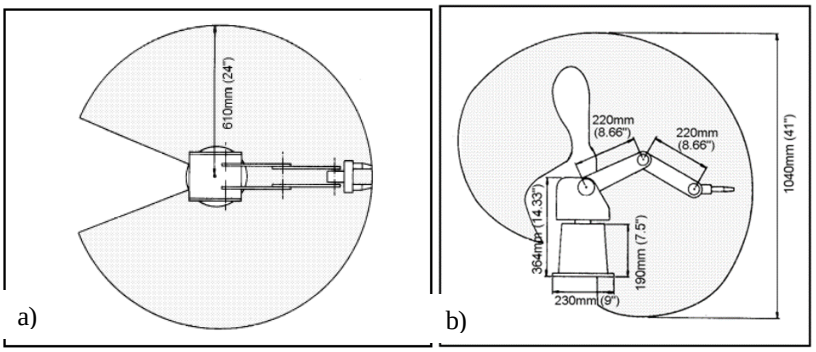


Figura 66. Rango de operación: a) Vista superior, b) Vista lateral.

15.3 Medios de operación.

EL Scorbot ER-V Plus se puede programar y operar de varias maneras. En esta parte se indican el hardware, el software y las funciones de programación.

- ACL (Lenguaje de control avanzado):

  Es un avanzado lenguaje de programación de robots desarrollado por Eshed Robotec y con el cual nos podemos comunicar con el Scorbot ER-V mediante alguna terminal o algún PC por medio de un canal de comunicación serial RS232, proporcionando las siguientes funcionalidades:

  - Control directo de los ejes robóticos.
  - Programación del sistema robótico por el usuario.
  - Control de entradas y salidas.
  - Ejecución de programas de manera simultánea, sincronizada e interactiva; Soporte multi-tasking completo.
  - Fácil gestionamiento de archivos.

- ATS:

  ATS, Advanced Terminal Software, corresponde a la interfaz para el controlador ACL. Este software es un terminal emulador que permite acceder al ACL desde un computador. Entre sus acciones permite la captura de las posiciones del robot, la edición de programas y la descarga de un programa realizado en un procesador de textos. Utiliza el lenguaje de programación ACL y comandos de acción directa.

  Las características incluidas en el ATS son:
  - Forma corta de configurar el controlador.
  - Manejar el robot desde una computadora.
  - Definición de los dispositivos periféricos.
  - Teclas Short-cut para ingresar al comando.
  - Editor de programa.
  - Backup manager.
  - Print manager.

- ScorBase:

  Scorbase es un paquete de software de control robótico que puede ser usado con el controlador. Su estructura basada en menú y las capacidades sin conexión facilitan la programación y operación del robot. Se comunica con ACL, el lenguaje interno del controlador, por medio de comunicación serial RS232.

- Botonera de aprendizaje (Teach Pendant).

  Es un dispositivo opcional, es una terminal de mano, usada para controlar el robot y los periféricos conectados al controlador. Es práctico para el movimiento de ejes, el control de la pinza, grabado de posiciones, direccionamiento de los ejes a posiciones grabadas y ejecutar o abortar programas en el controlador. Provisto de control de aprendizaje en coordenadas (X, Y, Z) y de articulaciones, trabaja tanto en coordenadas de las juntas rotativas (JOINTS) como coordenadas en el plano cartesiano (XYZ), teclas multifuncionales, pantalla LCD de 32 caracteres.

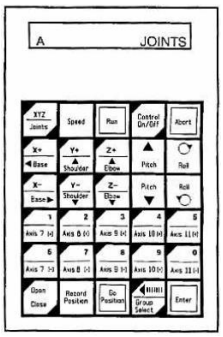


Figura 67. Botonera de aprendizaje del Scorbot.

15.4 Modos de operación.

- Modo Manual:

  El modo manual es accionable cuando el sistema está en modo directo, este modo permite el control directo de los ejes sin la necesidad de usar la botonera de aprendizaje. Para activar el modo manual solo basta presionar las teclas**: <alt> + m.**

Teclas de acción	Eje	Coordenadas cartesianas XYZ
Q-1	Eje 1 (Base)	+X y -X
W-2	Eje 2 (Hombro)	+Y y -Y
E-3	Eje 3 (Codo)	+Z y -Z
R-4	Eje 4 (Inclinación)	Cambia la inclinación, Pinza es estable
T-5	Eje 5 (Giro)	
Y-6	Eje 6 (Pinza)	
U-7	Eje 7	
I-8	Eje 8	

Tabla 18. Teclas de operación modo manual.

- Modo directo:

  Cuando el sistema está en modo directo, el usuario tiene el control directo de los ejes del robot. El controlador va ejecutando los comandos a medida que van siendo ingresados por el operador. Cuando se está en modo directo, el prompt aparece en pantalla de la siguiente manera: >_

15.5 Sistema de coordenadas.

El Scorbot puede ser operado mediante dos sistemas de coordenadas: el sistema de coordenadas Joint y el sistema de coordenadas cartesianas (X, Y, Z).

- Coordenadas Joint o coordenadas de ejes:

  Especifican la posición de los ejes robóticos en unidades de encoders, ya que estos proporcionan un número de señales proporcional al movimiento de los ejes. Cuando los ejes se mueven, los codificadores ópticos generan una serie señales alternas altas y bajas. El número de señales es proporcional a la cantidad de movimiento del eje; el controlador cuenta las señales y determina hasta qué punto un eje se ha movido. Del mismo modo un movimiento o posición del robot puede ser definido como un conteo específico del codificador para cada eje con respecto a una posición inicial, o de otra coordenada de referencia.

- Coordenadas Cartesianas:

  El sistema de coordenadas cartesianas o (X, Y, Z) es un sistema geométrico para especificar la posición del TCP (punto central de la herramienta = punto de la pinza) por la definición de su distancia en unidades lineales, desde el punto de origen (en el centro de la parte inferior de su base) a lo largo de los tres ejes lineales como se muestra en la figura.
  Para completar la definición de la posición, la elevación y el balanceo se especifican en unidades angulares. Cuando el movimiento del robot se ejecuta en el modo cartesiano XYZ, todos o algunos ejes se mueven para mover el TCP a lo largo de las coordenadas X, Y, Z.

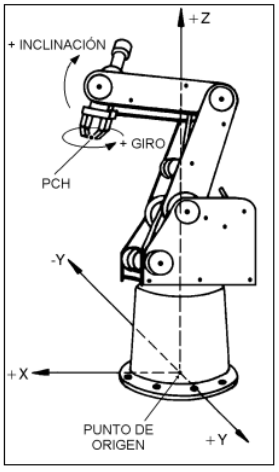


Figura 68. Sistema de coordenadas cartesianas en el robot Scorbot.

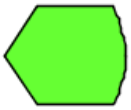
16 ANEXO 3: SIMBOLOGÍA.

16.1 Simbología utilizada en diagrama de flujo.

Inicio

Fin

Utilizado para procesos



Mostrar por pantalla

Condición, iteración

Dato ingresado por el usuario

Medio de almacenamiento, memoria

17 ANEXO 4: DIAGRAMA DE CLASES.

Un diagrama de clases describe gráficamente las especificaciones de las clases de software en una aplicación. Para la creación del diagrama de clases se identifican todas las clases que participan en la solución y se agregan los atributos encontrados, y los métodos que hacen relación a cada clase.

El diagrama de clases está separado mediante dos esquemas representado en la Figura 69 y Figura 70 representando cada una es estas un determinado modo de ejecución (Modo Administrador o controlador y Modo Cámara), donde cuyas clases principales son WSAdmninActivity y WSCamActivity.

17.1 Diagrama de clases: Smartphone en Modo Administrador.

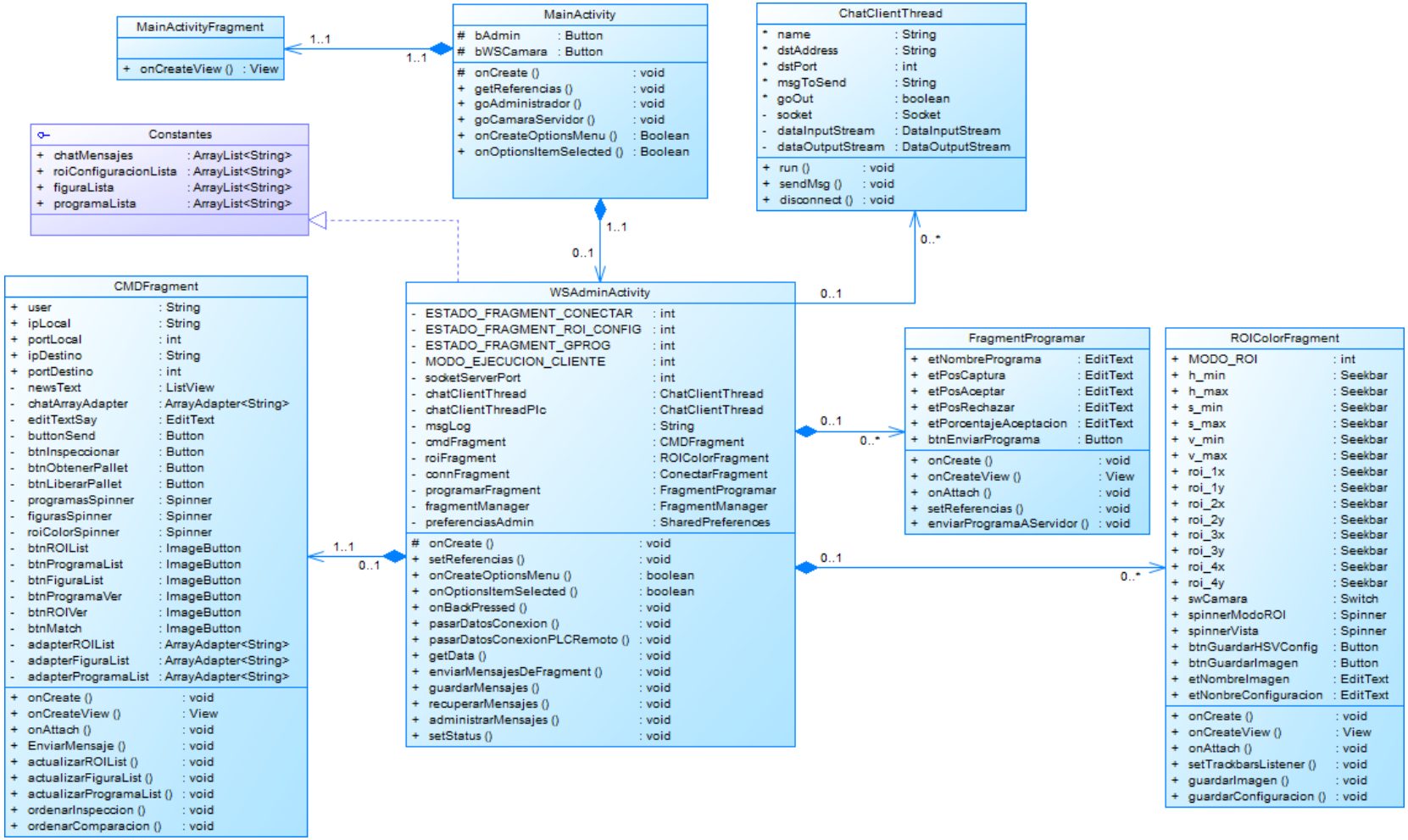


Figura 69. Diagrama de clases aplicación: Modo Smartphone Administrador.

17.2 Diagrama de clases: Smartphone en Modo Cámara.

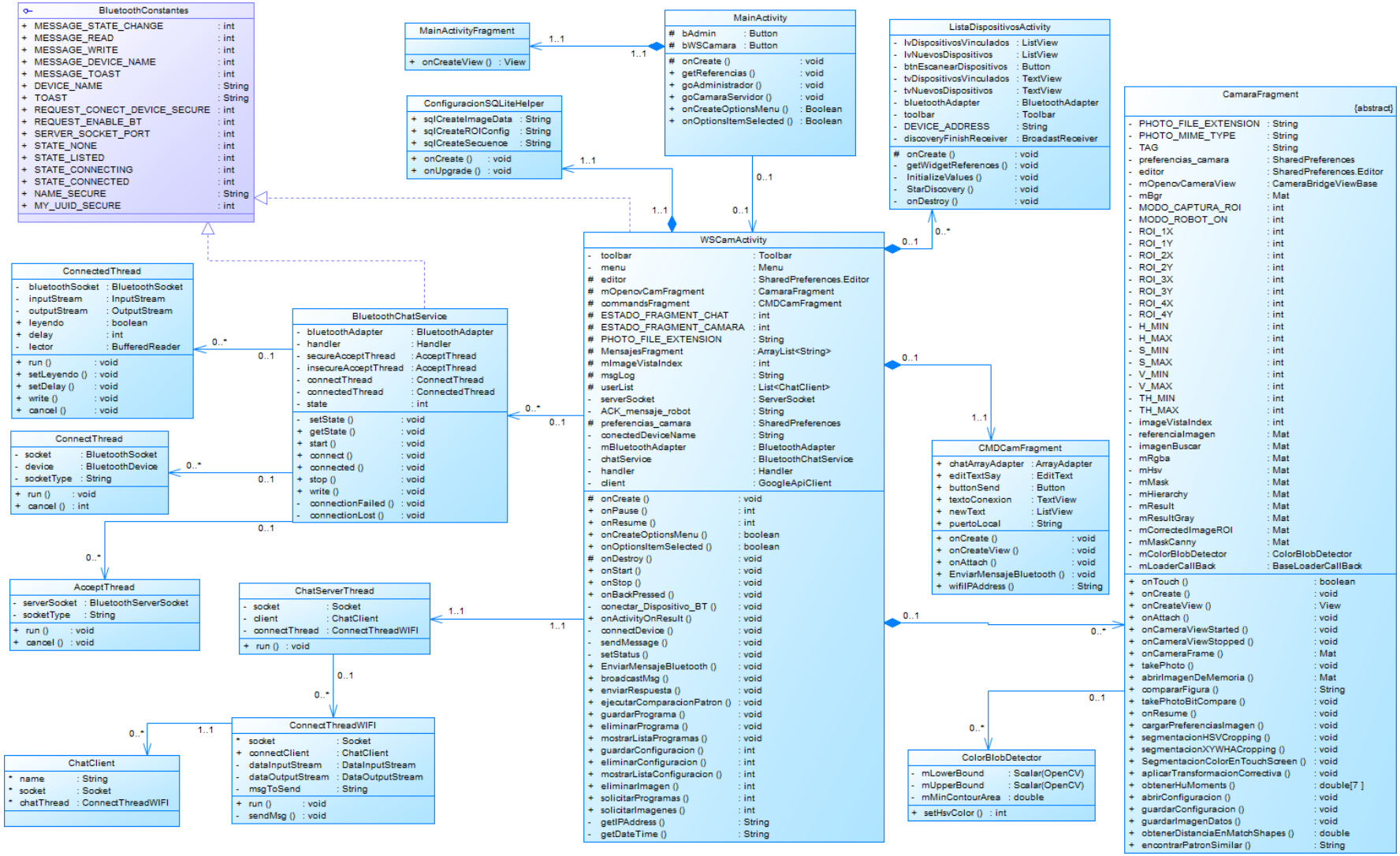


Figura 70. Diagrama de clases aplicación: Modo Smartphone Cámara