PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

# ALGORITMOS DE PROCESAMIENTO DE IMAGEN APLICADOS A LA DETECCIÓN DE FIGURAS GEOMETRICAS Y SUS PROPIEDADES ESPACIALES

**SAN-HO YOON BUSTAMANTE**

Profesor Guía: **Claudio Cubillos Figueroa**

Profesor Co-Referente: **Rodrigo Alfaro Arancibia**

Carrera: **Ingeniería Ejecución en Informática**

MARZO 2014

# Resumen

El procesamiento de imágenes, o también llamado Visión Artificial es la capacidad de analizar imágenes del mundo real adquiriendo información relevante de estas, para poder resolver tareas en algún problema específico del escenario de donde se están obteniendo.

Muchos sistemas de simulación, experimentación y de prueba-error basados en movimiento se componen de herramientas convencionales para determinar variables de posicionamiento y orientación como sensores de tacto, infrarrojos o de sondeo para realizar sus tareas. Por lo que se abordará tal idea y se planteara una nueva forma de determinar la localización de elementos dentro de un escenario a través de imágenes digitales y figuras geométricas, para mejorar el desempeño en sus labores.



*Palabras-claves: Procesamiento de imágenes digitales, Sentidos, Filtros.*



# Abstract

The image processing, or also known as Artificial Intelligence is the ability to analyze images from the real world acquiring useful information from them, to solve task from some other specific problem from the scenario where there are obtained.

Many simulator, experimentation and Trial-Error motion-based systems are composed from conventional tools to determine positioning and orientation variables like touch, infrared, or probing sensors to accomplish their tasks. So this idea will be addressed and will be posed a new way to determine the elements locations within a specific scenario through digital images and geometric figures, to improve the performance on his work.



*Keywords: Digital Image Processing, Senses, Filters.*

# Índice

# Glosario de términos

Clustering: Algoritmo de agrupamiento.

Co-ocurrencia: Presentación de diversos elementos lingüísticos en una misma proporción (por ejemplo palabras).

Fotograma: Secuencia de imágenes captadas por cámaras de video.

Transductor Óptico: Dispositivo capaz de transformar la luz en corriente eléctrica.

# Lista de abreviaturas o siglas

| | |
|---|---|
| CCD | : Charge-coupled device. |
| CMOS | : Complementary metal oxide semiconductor. |
| HSV | : Hue, Saturation, Value. |
| IDE | : Integrated Development Enviroment. |
| RGB | : Red, Green and Blue. |
| USB | : Universal Serial Bus. |
| VGA | : Video Graphics Array. |

# Lista de figuras

# Lista de tablas

# 1 Descripción del proyecto

## 1.1 Introducción

Los sentidos son esenciales para existir y sobrevivir en este planeta. Son la forma que tienen los seres vivos de interactuar con el mundo que les rodea o también de poder determinar su propio estado interno, de tal forma de percibir y entender lo que sucede, ayudándolos a tener consciencia de las necesidades o riesgos asociados a lo que sienten.

Los seres humanos tienen a emular o recrear situación de la vida cotidiana, simular situaciones o eventos en particular que tienen relación con los seres vivos y que generalmente sirven para resolver ciertos problemas de la humanidad complejos de entender, para ayudar a dar un mejor estilo de vida para todos. Para lograr aquello, gracias al ingenio se recurre a la invención de tecnologías, que pretender ser sentidos o sensores que sirvan de conexión entre un objeto y el mundo. Cada vez son más los objetos tecnológicos que poseen alguna especie de sensor que facilite alguna tarea, como automóviles con sensores de retroceso, celulares con sensores de temperatura y movimiento, etc.

Uno de los sentidos más complejos y útiles de replicar, tanto para mejorar como solucionar problemas cotidianos es la visión. Existen un sinfín de uso o formas de utilizar esta tecnología que de hecho es manipulada en lo cotidiano por la mayoría de las personas, como cámaras de vigilancia, scanner, detección de rostros, etc., y por supuesto también es utilizado en aquellos sistemas simulados donde es importante ver y analizar lo que ocurre en el espacio físico.

Es importante recalcar que la visión de una escena en particular no posee mayor relevancia si no es tratada según su contexto, por lo tanto con solo mirar u observar las imágenes no es suficiente. Así nace el concepto de procesamiento de imágenes, que consiste en analizar una imagen, alterándola si es necesario en una forma ideal para obtener datos e información requerida para la tarea adecuada. La ventaja de este procesamiento es que no requiere de una persona para obtener un resultado a la problemática, sino que un programa o aplicación se encarga de procesar, analizar y desplegar los resultados a persona, facilitando su labor ya que obtendrá el resultado de un algoritmo.

El presente proyecto pretende realizar una aplicación o prototipo de programa que utilice el procesamiento de imágenes para ser aplicado en alguno de los sistemas ya mencionado anteriormente, y como finalidad darle un sentido adicional, complementando los sensores que ya poseen, y resolver de mejor forma sus metas. Esto mediante la detección de figuras geométricas dentro del plano captado en video desde una perspectiva superior, determinando su posición dentro del cuadro y su dirección.

## 1.2 Definición de los objetivos

### 1.2.1 Objetivo general

- Implementar algoritmos de procesamiento de imagen para determinar ubicación y dirección de figuras geométricas.

### 1.2.2 Objetivos específicos

- Investigar el procesamiento de imágenes enfocado a la ubicación espacial.
- Indagar en proyectos de procesamiento de imagen enfocado en detección de objetos y movimiento.
- Seleccionar una librería de desarrollo para el procesamiento de imagen que abarque el problema.
- Investigar sobre funciones dentro de la librería y seleccionar los adecuados.
- Implementar la mejor solución al problema de localización y dirección de figuras.
- Implementar la aplicación de procesamiento de imagen para el problema de detección y ubicación.

## 1.3 Plan de Trabajo

Para el correcto desarrollo del proyecto se deben tener claro las actividades que la componen, y las metas que estas buscan. Por eso es la importancia de un plan de trabajo para poder planificar y cumplir de forma ordenada las partes requeridas y que el proyecto realizado tenga éxito.

Y por supuesto, uno de los problemas de no seguir o no trabajar con un plan de trabajo implica que los recursos del proyecto se vean perjudicados, teniendo como consecuencia no solamente la pérdida o gasto de recursos adicionales, sino en el peor de los casos el fracaso de este.

El presente proyecto se rige según la normativa de la escuela. Consta de un proyecto general divido en proyecto 1 y proyecto 2. A su vez cada parte del proyecto se divide en una parte de avance y otra final. La división de tiempo en general consta de dos semestres, uno para cada proyecto, y las fechas son: proyecto 1 comienza en agosto de 2012 y termina en diciembre 2012, mientras que proyecto 2 comienza en marzo y termina en junio. Por motivos ajenos al proyecto, los tiempos fueron cambiados ligeramente, por lo que proyecto 2 termina en el mes de agosto, alargándose su duración, pero no varía mucho la carga de trabajo real.

Por lo tanto, es adecuado separar el plan general en 4 partes, dos por cada proyecto, así se logra sincronizar con las entregas respectivas.

A continuación se presentara el plan de trabajo para el presente proyecto, tomando en consideración que es un plan teórico, para el cual hay aspectos todavía no resueltos o variaciones con respecto a la realidad.

Tabla 1 Tabla Gantt Proyecto 1

| Id. | Nombre de tarea | Comienzo | Fin | Duración |
|---|---|---|---|---|
| 1 | Averiguación a grandes rasgos del proyecto anterior. | 06-08-2012 | 17-08-2012 | 10d |
| 2 | Investigación y elección de tecnologías para el proyecto. | 07-08-2012 | 21-09-2012 | 34d |
| 3 | Familiarización con la herramienta de programación. | 27-08-2012 | 21-09-2012 | 20d |
| 4 | Selección de algoritmos de procesamiento de imagen. | 27-08-2012 | 21-09-2012 | 20d |
| 5 | Redacción del informe proyecto 1 avance | 14-08-2012 | 21-09-2012 | 29d |
| 6 | Preparación de la presentación de avance de proyecto | 24-09-2012 | 05-10-2012 | 10d |
| 7 | Mejoras al informe en base a la presentación. | 01-10-2012 | 12-10-2012 | 10d |
| 8 | Tutoriales de uso de la Herramienta de Desarrollo | 29-10-2012 | 28-11-2012 | 23d |
| 9 | Desarrollo de prototipos de la aplicación. | 05-11-2012 | 29-11-2012 | 19d |
| 10 | Pruebas básicas al prototipo | 05-11-2012 | 23-11-2012 | 15d |
| 11 | Complementar informe de avance | 01-10-2012 | 23-11-2012 | 40d |
| 12 | Preparar presentación final proyecto 1 | 26-11-2012 | 07-12-2012 | 10d |

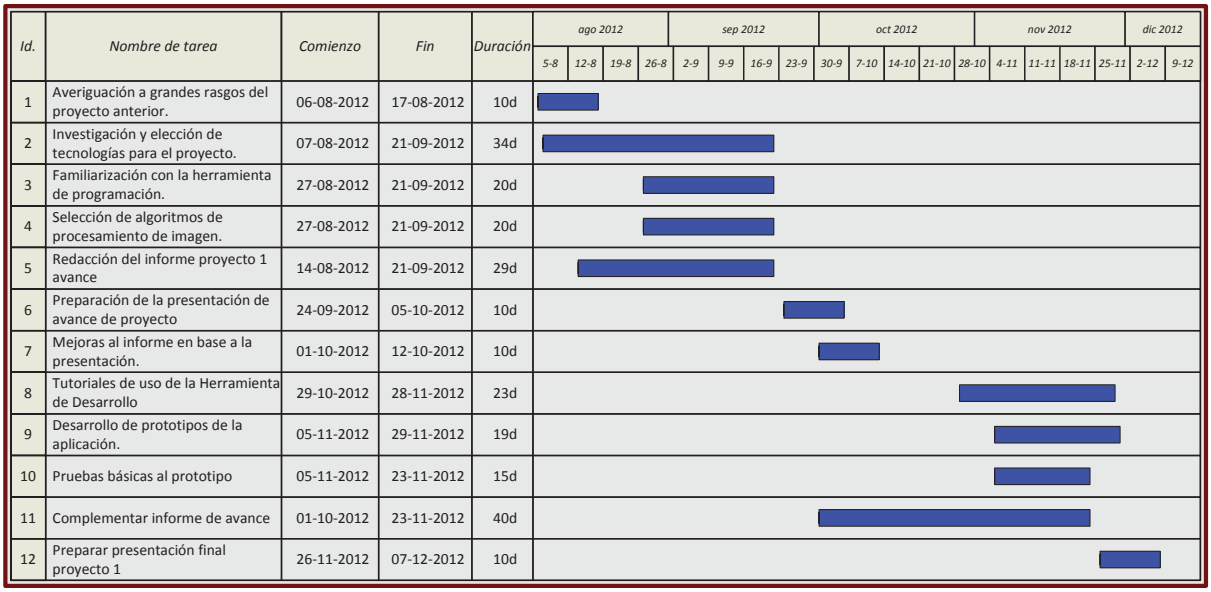
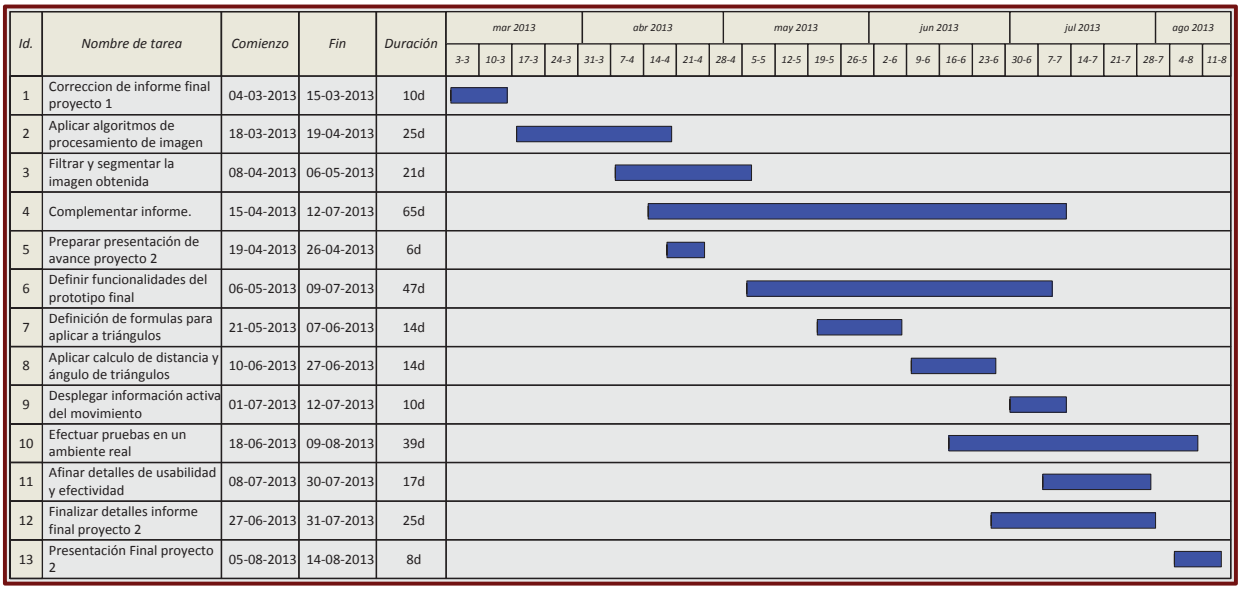



Tabla 2 Tabla Gantt Proyecto 2

| Id. | Nombre de tarea | Comienzo | Fin | Duración |
|---|---|---|---|---|
| 1 | Correccion de informe final proyecto 1 | 04-03-2013 | 15-03-2013 | 10d |
| 2 | Aplicar algoritmos de procesamiento de imagen | 18-03-2013 | 19-04-2013 | 25d |
| 3 | Filtrar y segmentar la imagen obtenida | 08-04-2013 | 06-05-2013 | 21d |
| 4 | Complementar informe. | 15-04-2013 | 12-07-2013 | 65d |
| 5 | Preparar presentación de avance proyecto 2 | 19-04-2013 | 26-04-2013 | 6d |
| 6 | Definir funcionalidades del prototipo final | 06-05-2013 | 09-07-2013 | 47d |
| 7 | Definición de formulas para aplicar a triángulos | 21-05-2013 | 07-06-2013 | 14d |
| 8 | Aplicar calculo de distancia y ángulo de triángulos | 10-06-2013 | 27-06-2013 | 14d |
| 9 | Desplegar información activa del movimiento | 01-07-2013 | 12-07-2013 | 10d |
| 10 | Efectuar pruebas en un ambiente real | 18-06-2013 | 09-08-2013 | 39d |
| 11 | Afinar detalles de usabilidad y efectividad | 08-07-2013 | 30-07-2013 | 17d |
| 12 | Finalizar detalles informe final proyecto 2 | 27-06-2013 | 31-07-2013 | 25d |
| 13 | Presentación Final proyecto 2 | 05-08-2013 | 14-08-2013 | 8d |



## 1.4 Metodología de Desarrollo

La metodología de desarrollo de software que se utilizará en el presente trabajo corresponde al basado en prototipos, el motivo por el cual fue escogida esta metodología se debe a la naturaleza del proyecto y a las características de las entregas de éste.

El proyecto consta de 4 entregas en total, divididas entre proyecto 1 y 2 que a su vez poseen entrega de avance y de final. Esto de inmediato genera la necesidad de entregas parciales de trabajo que no pueden ser satisfechas por ejemplo por el modelo de cascada, y también se requiere un desarrollo paulatino de la aplicación debido a las novedades y desafíos que presenta desarrollar en un nuevo ámbito desconocido hasta el momento, utilizando trabajo anterior como ayuda para llegar al producto. Por otro lado el modelo basado en prototipos presenta una buena dinámica frente a los cambios, lo cual es muy útil en este problema, ya que se va desarrollando y va tomando nuevos rumbos según las necesidades nuevas o cambiantes.

### 1.4.1 Desarrollo basado en prototipos

El uso de prototipos se centra en la idea de ayudar a comprender los requisitos que plantea el usuario sobre todo si este no tiene una idea acabada de lo que se desea. Además puede utilizarse cuando el ingeniero en software tiene dudas acerca de la viabilidad de la solución pensada.

El Modelo de prototipos pertenece a los modelos de desarrollo evolutivo, el prototipo debe ser construido en poco tiempo, usando los programas adecuados y no se debe utilizar mucho dinero pues a partir de que este sea aprobado se pueda iniciar el verdadero desarrollo del software.

El diseño rápido se centra en una representación de aquellos aspectos del software que serán visibles para el cliente o el usuario final. Este diseño conduce a la construcción de un prototipo, el cual es evaluado por el cliente para una retroalimentación; gracias a ésta se refinan los requisitos del software que se desarrollará. La interacción ocurre cuando el prototipo se ajusta para satisfacer las necesidades del cliente. Esto permite que al mismo tiempo el desarrollador entienda mejor lo que se debe hacer y el cliente vea resultados a corto plazo.

La construcción de prototipos se puede utilizar como un modelo del proceso independiente, se emplea más comúnmente como una técnica susceptible de implementarse dentro del contexto de cualquiera de los modelos del proceso expuestos. Sin importar la forma en que éste se aplique, el paradigma de construcción de prototipos ayuda al desarrollador de software y al cliente a entender de mejor manera cuál será el resultado de la construcción cuando los requisitos estén satisfechos. De esta manera, este ciclo de vida en particular, involucra al cliente más profundamente para adquirir el producto [1].

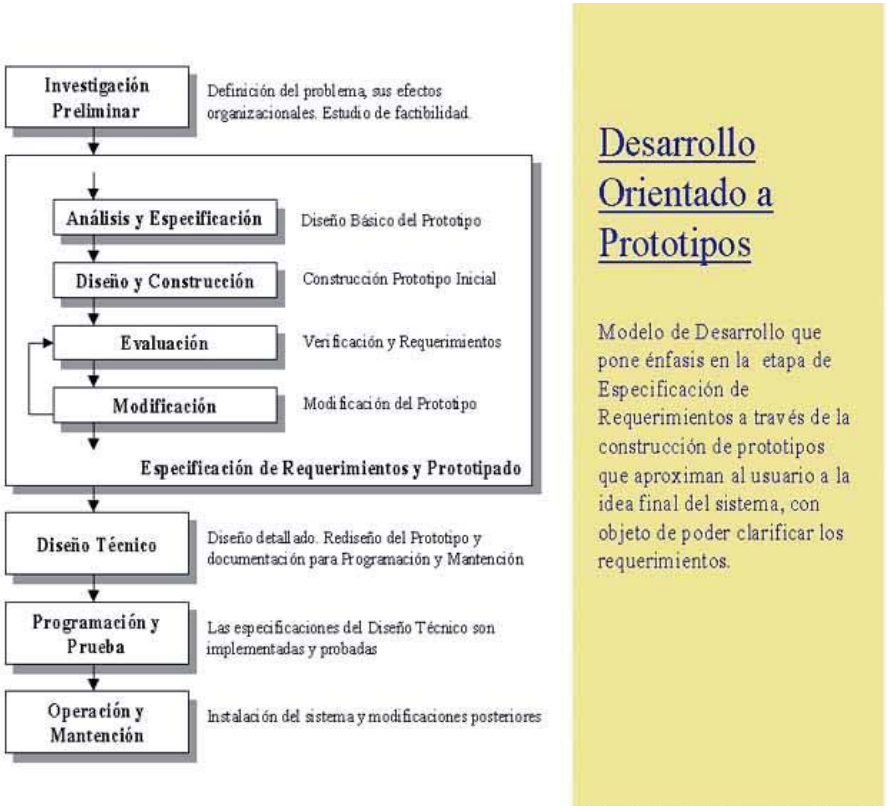



Figura 1.1 Representación gráfica del desarrollo basado en prototipos [2].

Las ventajas que ofrece un desarrollo basado en prototipos son las siguientes:

- Es útil cuando los requerimientos son cambiantes.
- Se utiliza cuando no se conoce bien la aplicación.
- Cuando el usuario no se quiere comprometer con los requerimientos.
- Cuando se quiere probar una arquitectura o tecnología.

Y las desventajas que presenta son:

- No se conoce cuando se tendrá un producto aceptable.
- No se sabe cuántas iteraciones serán necesarias.
- Da una falsa ilusión al usuario sobre la velocidad del desarrollo.
- Se puede generar un producto aun cuando no este con los estándares.

## 1.5 Estudio de Factibilidad

El presente proyecto debe cumplir con ciertos requisitos técnicos necesarios para el correcto desarrollo de la aplicación, lo cual puede ser determinante al momento de ver la viabilidad del proyecto completo. En el caso particular del presente, en gran parte los requisitos se enfocan en el correcto funcionamiento del entorno de desarrollo utilizado.

### 1.5.1 Factibilidad técnica

La factibilidad técnica es aquella necesidad, disponibilidad y capacidad del hardware y software suficientes para el soporte del desarrollo de la aplicación.

#### 1.5.1.1 Hardware

La herramienta de desarrollo utilizada, para su correcto uso requiere un computador de prestaciones normales para la captación, procesamiento e interpretación de una señal de video continua desde la cámara digital y que trabaje bajo el sistema operativo Windows o Linux. Los requerimientos específicos mínimos de hardware del computador son los siguientes [4]:

- Procesador de 1.6 GHz o superior (32 o 64 bits).
- Memoria RAM 1 GB.
- Almacenamiento mínimo disponible 10 GB (NTFS).
- Disco duro de 5400 RPM.
- Tarjeta de video capaz de utilizar DirectX 9.x.
- Resolución mínima de ejecución de 1024 x 768.
- Periféricos (mouse, teclado, pantalla).
- Entrada USB 2.0.

En el actual desarrollo de la aplicación se utilizará un computador personal propiedad propia, con los requerimientos suficientes para ejecutar y alojar el entorno de desarrollo. En cuanto a captura del video digital, se requiere de una cámara digital con interfaz USB 2.0 que posea una óptica especial que capture la escena de forma completa o de gran angular.

#### 1.5.1.2 Software

En cuanto a las necesidades de software, la más evidente es la cual colaborara con el desarrollo del prototipo, que en el presente caso será con entorno de desarrollo integrado. Bajo los elementos usados en el presente proyecto, y mayormente al uso del lenguaje de programación C++, se pueden utilizar 2 tipos de IDE: el más obvio seria el entorno de desarrollo Visual Studio. Este cumple con buenos estándares de desarrollo y herramienta unificada cómoda de utilizar. Y la otra opción disponible es SharpDevelop, muy parecido a Visual Studio pero de código abierto, cumple con su mayoría de las funciones. Esta

herramienta servirá tanto para programar como compilar la aplicación. Frente a situaciones inesperadas dentro del desarrollo del primer prototipo, se escogió que sería una mejor alternativa desarrollar la aplicación en el Sistema Operativo Linux, distribución Ubuntu, por la facilidad y simplificación al cargar o utilizar la librería utilizada para el procesamiento de imagen. Y dentro de los IDEs presentes en Ubuntu, se escogerá por Eclipse C++, en consecuencia a su compatibilidad con el lenguaje utilizado y su interfaz más amigable que otras opciones disponibles para ese sistema.

Junto con el entorno de desarrollo necesitado, también surge la utilización de una librería de algoritmos de programación que manipule y trabaje con imágenes o en caso presente con videos. Y para esto está disponible una librería de visión artificial compatible con C++ y es OpenCV, es de código abierto y posee gran soporte y múltiples funciones, muy completo para el trabajo en esta área.

### 1.5.2 Factibilidad operativa

La factibilidad operativa son aquellas condiciones que permiten que la aplicación sea operativa y funcione de acuerdo a lo estimado, estas condiciones permitirán el correcto análisis de los patrones obtenidos a través de las imágenes.

- La escena capturada por la cámara digital debe poseer buena iluminación, de ser necesario un módulo de iluminación propio dependiendo de la cámara utilizada.
- La cámara deberá estar situada de forma de abarcar la gran parte del área donde se recreara el sistema, de forma perpendicular al punto medio.
- La cámara no debe tener movimiento, o no puede ser afectada por situaciones que ocurran en la escena, siempre debe mantenerse estática.
- La cámara deberá poseer un objetivo gran angular, que facilita la captación de la situación en toda su dimensión.
- El suelo no debe ser irregular o no debe poseer muchas líneas, lo cual podría confundir al algoritmo.

### 1.5.3 Factibilidad económica

La factibilidad económica es aquella condición para ver si es posible el desarrollo de la aplicación con respecto a los gastos que esto implica. El hardware necesario para poder trabajar en la creación del prototipo está totalmente adquirido por parte del desarrollador, por lo tanto no se incurren en gastos de compra de tales materiales, tanto la cámara como computador están disponibles.

En cuanto al software necesario para el trabajo, un IDE en Ubuntu, que en este caso corresponde a Eclipse y su naturaleza es de código abierto, y por el lado de la librería OpenCV al ser de código abierto tampoco incluye gastos de compra de software. En definitiva tampoco se incurrirá en gastos de compras de software por parte del desarrollador.

Solo se incluye un gasto adicional posible, y es el la impresión o fotocopia de los documentos requeridos para el informe del proyecto, el cual se puede costear por el desarrollador y no es un impedimento para su realización.

### 1.5.4 Factibilidad legal

Es importante, para el desarrollo del proyecto, y a pesar de las otras factibilidades que todo se encuentre dentro del marco legal de la republica de chile. Por lo tanto cabe destacar que todas las herramientas utilizadas en el proyecto, tanto de desarrollo como auxiliares, cuentan con licencias autorizadas o en la mayoría de los casos de software libre (Ubuntu, Eclipse) y el caso de las librerías es correlativo al ser también de naturaleza de código abierto. Por ende no se traspasan derechos de autor ni propiedad intelectual.

Es correcto afirmar entonces, que se cuenta con todas las medidas de precaución legales y que se cumple con las normas vigentes y estándares requeridos para su desarrollo.

## 1.6 Análisis de riesgo

El presente proyecto presenta una planificación definida a grandes rasgos y en el caso de que esta resulte positiva, pero esta no contempla lo que acontece en la práctica ni los contratiempos que pueda tener, por lo que se debe realizar un nuevo análisis de los posibles caminos alternativos que esta pueda presentar en casos no tan favorables. Es por esto que el análisis de riesgo es un tópico importante para detectar situaciones indeseables en el proyecto, que como consecuencia puede tener la demora del proyecto o su fracaso inminente.

A continuación se mostrara una tabla con los riesgos que el proyecto pueda presentar en su desarrollo, como medidas de impacto hay cuatro categorías (bajo, medio, alto, critico), la probabilidad corresponde a un porcentaje, y el ID corresponde a un indicador del riesgo para facilitar su reconocimiento:

Tabla 3 Análisis de riesgos.

| Riesgo | Consecuencia | Probabilidad | Impacto | ID |
|---|---|---|---|---|
| Cálculo erróneo o mal estimación de los tiempos establecidos. | Atraso de entregas o hitos del proyecto. | 60% | Medio | 1 |
| Falla o mal uso de las herramientas de desarrollo. | Atraso de entregas o hitos del proyecto, cambio de funcionalidades u objetivos. | 30% | Alto | 2 |
| Problemas o situaciones personales indeseables del desarrollador. | Atraso de entregas o hitos del proyecto. | 20% | Bajo | 3 |

| | | | | |
|---|---|---|---|---|
| Subestimación de los límites del proyecto o sobrepaso de las facultades del desarrollador. | Atraso de entregas o hitos del proyecto. | 30% | Medio | 4 |
| Mal planteamiento de los requerimientos del proyecto | Proyecto no cumple con las expectativas. | 30% | Medio | 5 |
| Falta de disponibilidad de herramientas tanto de software como de hardware. | Atraso de entregas o hitos del proyecto, cambio de funcionalidades u objetivos. | 20% | Critico | 6 |
| Problemas de carácter o mala comunicación con el profesor guía. | Cambio de funcionalidades u objetivos, desmotivación. | 10% | Critico | 7 |
| Poco dominio de las herramientas o la falta de capacidad para solucionar problemas. | Cambio de funcionalidades u objetivos, desmotivación, atraso de entregas o hitos del proyecto. | 40% | Alto | 8 |

Una vez identificados los riesgos y sus características, es necesario crear un plan de contingencia o de mitigación, que prevengan las consecuencias de la mejor forma posible.

El plan de mitigación lo que busca es la reducción de la vulnerabilidad, es decir la atenuación de los daños potenciales sobre el proyecto realizado por un evento fortuito, mientras que el plan de contingencia ve la posibilidad de que ocurra un problema o hecho de forma imprevista, a continuación se verá los posibles problemas de nuestro proyecto y sus respectivos planes:

Tabla 4 Plan de mitigación y de contingencia.

| **Identificador del riesgo** | **Plan de mitigación** | **Plan de contingencia** |
|---|---|---|
| 1 | Monitorear y controlar de forma completa y periódica la organización de las tareas y la planificación. | Rediseñar o planificar nuevamente las fechas de los hitos o actividades del proyecto. |
| 2 | Averiguar y familiarizarse con las herramientas utilizadas. | Adecuar las actividades a la herramienta, buscar nuevas opciones. |
| 3 | Realizar planificación con un colchón para atrasos. | Re planificación, cuidando dejar una ventana disponible. |
| 4 | Verificar los requerimientos con respecto a las aptitudes, analizar plan de trabajo realista. | Reorganizar los tiempos o actividades, priorizar tareas críticas, definir nuevos requerimientos. |
| 5 | Conversaciones constantes con profesor jefe y fijar límites e hitos a conseguir. | Nuevas conversaciones, conversar falencias y corregir los errores o definir nuevos requerimientos. |
| 6 | Buscar y analizar documentación del | Cambiar herramientas por opciones |

| | | |
|---|---|---|
| | software y hardware, buscar otros proyectos que utilicen problemáticas parecidas y ver otras herramientas. | más confiables o sencillas, aplicar experiencias de otros proyectos. |
| 7 | Tener buena comunicación y relación con el profesor guía, ser empático y responsable. | Empezar de cero las interacciones, cambio de estrategias y modalidades de comunicación, cambio de profesor guía y tema de proyecto. |
| 8 | Averiguar y familiarizarse con las herramientas y su documentación, buscar alternativas más simples, revisar proyectos similares y recoger experiencias. | Cambio de funcionalidades, hitos o requerimientos, acotar solución o problema, redefinir el uso de los recursos, cambio de tema o camino tomado. |

# 2 Marco teórico

Dentro de las ideas y conceptos teóricos que serán utilizados dentro del proyecto necesarios para el reconocimiento de los patrones o dibujos que se situaran en la escena se utilizara la visión artificial, que conlleva al uso de filtros, manipulación de imagen, mejoramientos y transformaciones de la imagen para su correcto análisis.

## 2.1 Visión computacional

Se puede definir la visión artificial o visión computacional como un campo de la inteligencia artificial que, mediante la utilización de las técnicas adecuadas, permite la obtención, procesamiento y análisis de cualquier tipo de información especial obtenida a través de imágenes digitales.

La visión artificial es una disciplina que la componen procesos que son capaces de otorgar la capacidad de mirar o de observar a una máquina, y se podría decir que la visión artificial o comprensión de imágenes corresponde a la estimación automática de la estructura y propiedades de un lugar en tres dimensiones dinámico, a partir de imágenes obtenidas del mundo en dos dimensiones. Las estructuras y propiedades de ese lugar que se quieren deducir en visión artificial incluyen no solo sus propiedades geométricas, sino también sus propiedades materiales. Ejemplos de propiedades de los materiales son su color, iluminación, textura y composición. Si el mundo se modifica en el proceso de formación de la imagen, se necesitara inferir también la naturaleza del cambio, e incluso predecir el futuro [4].

La visión, tanto para un hombre como para un computador, consta principalmente de dos fases: captar una imagen e interpretarla. A pesar de la complejidad que presenta el ojo humano, la fase de captación de imágenes hace mucho tiempo que está resuelta. El ojo del

computador es la cámara de video, y su retina un sensor que es sensible a la intensidad luminosa. Así que en la visión artificial lo que resta es interpretar las imágenes, distinguir los objetos de la escena, extraer información de ellos y resolver aspectos más particulares según las necesidades que se deseen satisfacer [4].

Dentro de la fase de captación de las imágenes se encuentra un proceso llamado mejoramiento de la imagen, donde se obtiene un resultado más acorde que el original para el objetivo por el cual la imagen necesita ser analizada. Dependiendo del objetivo o finalidad del análisis de imágenes digitales, hay una variedad de técnicas de mejoras que pueden ser utilizadas para conseguir resultados, por ejemplo para mejorar una imagen casera de una familia y de sus rostros no será muy aplicable a imágenes satelitales que necesitan otro tratamiento del punto de vista de filtros [5].

### 2.1.1 Fases de un sistema de visión artificial

Se tiene conocimiento de que el ser humano y seres vivos en general poseen la capacidad de capturar la luz a través de los ojos, y esta información es transferida a través del nervio óptico al cerebro donde es procesada. Existen razones para estipular que dentro del proceso de visión, un primer paso consistiría en identificar y detectar elementos más simples para poder descomponer la imagen (como segmentos y arcos). Luego el cerebro como intérprete procesa la escena y realiza un acto o acción en su consecuencia. Por lo tanto la visión artificial, en una imitación del funcionamiento de este comportamiento trata de definir fases para el análisis de imágenes, que tradicionalmente de divide en cuatro fases principales [6]:

- La primera fase, corresponde básicamente a una etapa sensorial, que consiste en la *captura* o adquisición de la imagen del mundo real en imágenes digitales, a través de algún tipo de sensor.
- La segunda etapa consiste en el tratamiento digital de las imágenes obtenidas, que tiene como objeto facilitar las etapas posteriores. Esta etapa, llamada *procesamiento previo*, es donde, mediante filtros y transformaciones geométricas, entre otros, se eliminan partes indeseables de la imagen o se realzan partes interesantes de la misma.
- La siguiente fase se le conoce como *segmentación*, y consiste en aislar una zona o elementos que son relevantes en una escena para poder ser comprendida.
- Por último se llega a la etapa de *reconocimiento* o *clasificación,* en ella se obtiene la distinción de los objetos segmentados de la escena, gracias a los análisis de ciertas características que fueron establecidas previamente para diferenciarlos.

Estas etapas o fases del sistema no siempre siguen su orden o secuencialidad, sino que en ocasiones se deben retroalimentar hacia atrás. De esta forma, es normal volver a etapas anteriores si falla alguna posterior, por ejemplo encontrarse en la etapa de reconocimiento y tener que volver a la de segmentación si falla, o incluso a la de captura, esta retroalimentación puede ser visualizada en la figura [6]:

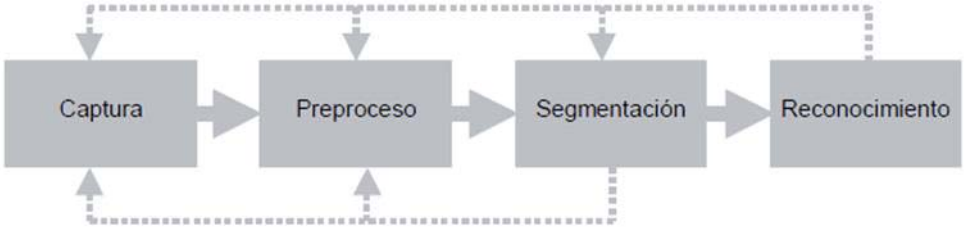



Figura **2.1** Representación gráfica del desarrollo basado en prototipos **[**6].

### 2.1.2 Componentes de un sistema de procesamiento de imagen

Los dos pilares del sistema físico de visión artificial fundamentalmente son: el sistema de formación de las imágenes y el sistema de procesamiento de éstas. El primer apartado está constituido por el subsistema de iluminación, de captación de la imagen y de adquisición de la señal en el computador. Una vez introducida la señal en el computador, ésta es procesada mediante los algoritmos para transformarla en información de alto nivel. La cual puede ser utilizada para su representación visual, para actuar en el planificador de un robot o ser fuente de datos para un autómata programable. En definitiva, múltiples periféricos pueden ser receptores de esta información y vincularse con el sistema de procesamiento de las imágenes, un ejemplo se encuentra en la figura 3 [7].

Cada uno de los subsistemas que participan del sistema de visión artificial posee las siguientes características:

- Subsistema de iluminación: conjunto de artefactos que producen radiación electromagnética para que incidan sobre los objetos a visualizar. Se puede citar algunos elementos como lámparas, pantallas fotográficas, filtros de luz, láseres, etc.
- Subsistema de captación: son los transductores que convierten la radiación reflejada luminosa en señales eléctricas. Fundamentalmente se habla de las cámaras CCD, no sólo en el espectro visible, sino que van desde la radiación gamma hasta la radiofrecuencia o microondas, dando paso a sensores de ultrasonidos, sonar, radar, telescopía, etc.
- Subsistema de adquisición: la señal eléctrica procedente de las cámaras forman la señal de vídeo. Hay una tendencia creciente a que su naturaleza sea de tipo digital, pero todavía existen muchas señales de vídeo de carácter analógico (CCIR, PAL, RS170, NTSC,…). Para ser tratadas hay que muestrearlas y cuantificarlas. Ambas tareas son realizadas por las tarjetas de adquisición. También se las llama frame grabbers. Se depositan en el bus de expansión del computador. Hay para buses desde PCI hasta VMP. Recientemente, también se están empleando las tecnologías de USB o FireWire [14].
- Subsistema de procesamiento: Suele ser un computador o un cluster de computadores, dependiendo de las necesidades de los algoritmos de Visión Artificial. Parten de una representación digital de las imágenes y procesan esta información hasta alcanzar otro tipo de información de más alto nivel. La transformación dependerá de la algoritmia.

- Subsistemas de periféricos: conjunto de elementos receptores de la información de alto nivel. Puede ser un monitor de altas prestaciones gráficas, un automatismo, una impresora sacando las características, etc. [7].

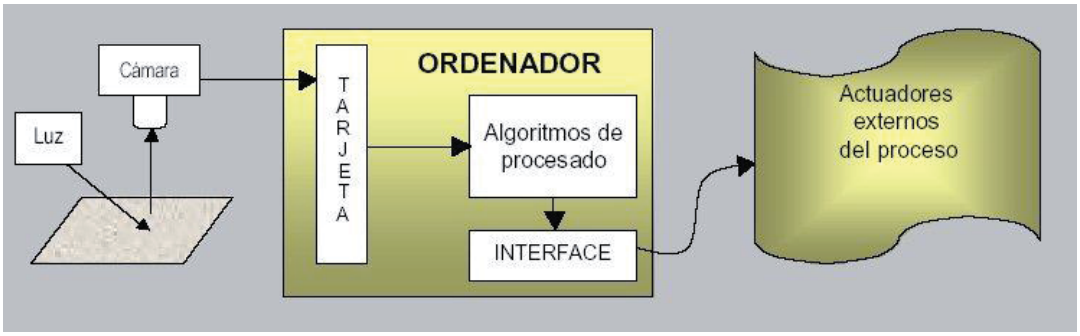



Figura **2.2** Representación gráfica de un sistema de procesamiento de imagen **[**7].

## 2.2 Procesamiento previo de la imagen digital

Dentro del procesamiento previo, se encuentran todas las metodologías o algoritmos de edición de imágenes digitales, con la finalidad de destacar o realzar elementos de importancia, y ocultar o despreciar elementos de la escena que no tengan un valor para la tarea que quiere conseguir con su procesamiento.

### 2.2.1 Aumento y reducción de contraste

Las modificaciones del histograma se pueden visualizar eficazmente mediante las *funciones de transferencia del histograma.* Estas funciones corresponden a aplicaciones, pues para cada punto del dominio solo tiene un valor imagen. Estas aplicaciones están acotadas entre 0 y 1 tanto en la abscisa, que se hace corresponder con la entrada *Ie* del filtro, como en la ordenada, que se corresponde con la salida *Is* del filtro.

En la figura 4, se presentan tres ejemplos de funciones de transferencia: la función lineal, la función cuadrado y la función raíz cuadrada. La función de transferencia lineal (a) no introduce modificación alguna sobre el histograma, al coincidir exactamente los niveles de intensidad de la entrada y de salida. La función cuadrado produce un oscurecimiento general de la imagen (en la figura (b) se aprecia que el rango entre 0 y 0,5 se hace corresponder con el rango entre 0 y 0,25 que es más oscuro). Por último, la función raíz cuadrada (c) produce un aclarado general de la imagen [6].

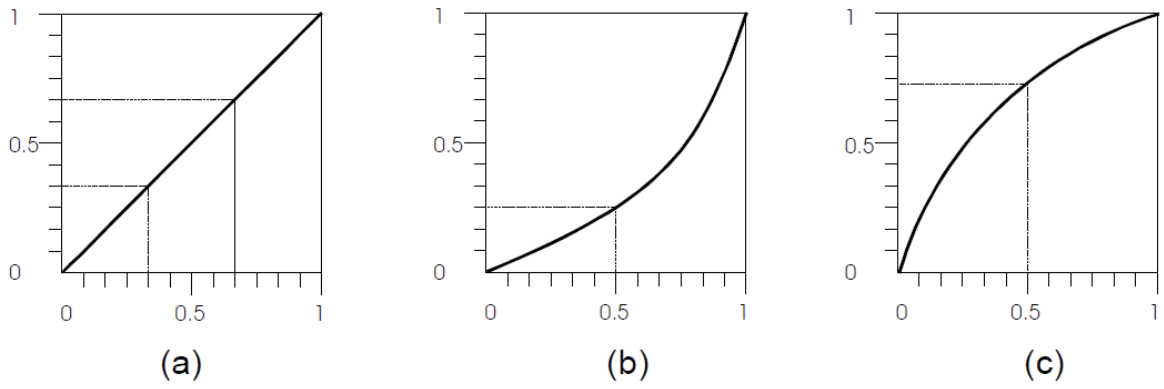



Figura **2.3** De izquierda a derecha las funciones lineal, cuadrada y raíz cuadrada. **[**6].

Se define contraste como la diferencia de intensidad pronunciada. Ahora, en este contexto, se puede hablar de alto contraste en una imagen digital en niveles de gris si sobre el histograma se aprecia masas separadas. En este contexto, un buen indicador del contraste de una imagen podría consistir en el cálculo de la desviación típica de su histograma.

Una función de transferencia que aclare los niveles claros y oscurezca los niveles oscuros, conseguirá sobre el conjunto de la imagen un efecto visual de aumento de contraste. Una función tal se puede obtener componiendo una función de transferencia del histograma que hasta el valor de 0,5 se comporte como la función cuadrado y que en adelante se comporte como la función raíz. En la figura 5 se ha representado esta función de transferencia. La función (b) de la misma figura produce el efecto contrario, esto es una disminución del contraste [6].

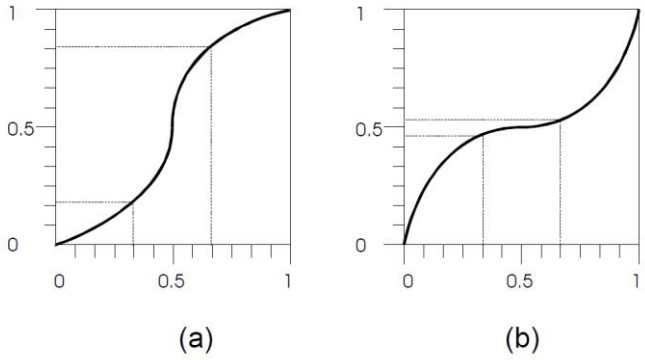



Figura **2.4** Funciones de transferencia para aumento y reducción de contraste. **[**6].

En general, una función de transferencia con una pendiente inferior a la unidad produce un efecto de reducción de contraste. Esto se debe a que concentra los valores de las intensidades de un rango *R* en un rango más pequeño *R'*. Por otro lado una función de transferencia con una pendiente superior a la unidad produce un efecto de aumento de contraste por razones inversas.

Las imágenes de la mujer en la figura 6, en las que respectivamente se ha aumentado y reducido el contraste, han sido obtenidas aplicando estas funciones de transferencia. Así, al aplicar el filtro de aumento de contraste se aprecia en su histograma que se ha saturado los tonos claros y los oscuros, mientras que se ha reducido la densidad en la parte central del histograma. Esto quiere decir que, proporcionalmente, hay muchos más pixeles asociados a los tonos blancos y negros que al resto de tonos.

El filtro de reducción de contraste produce una imagen en la que se aprecia una reducción del rango dinámico del histograma. Esto también significa que la imagen contiene menos información que otra en la que el rango dinámico sea más amplio, ya que se ha homogeneizado zonas de la imagen que antes eran diferentes.

Aunque todas las transformaciones de histograma suelen expresarse matemáticamente con fórmulas que hay que aplicar sobre cada pixel ($x$,$y$), en la práctica, suelen tabularse. Es decir, se usa una tabla para acceder al valor correspondiente a la transformación en vez de realizar el cálculo cada vez. De esta forma, se aceleran las operaciones, más teniendo en cuenta el reducido número de niveles de intensidad involucrados (normalmente 256) [6].

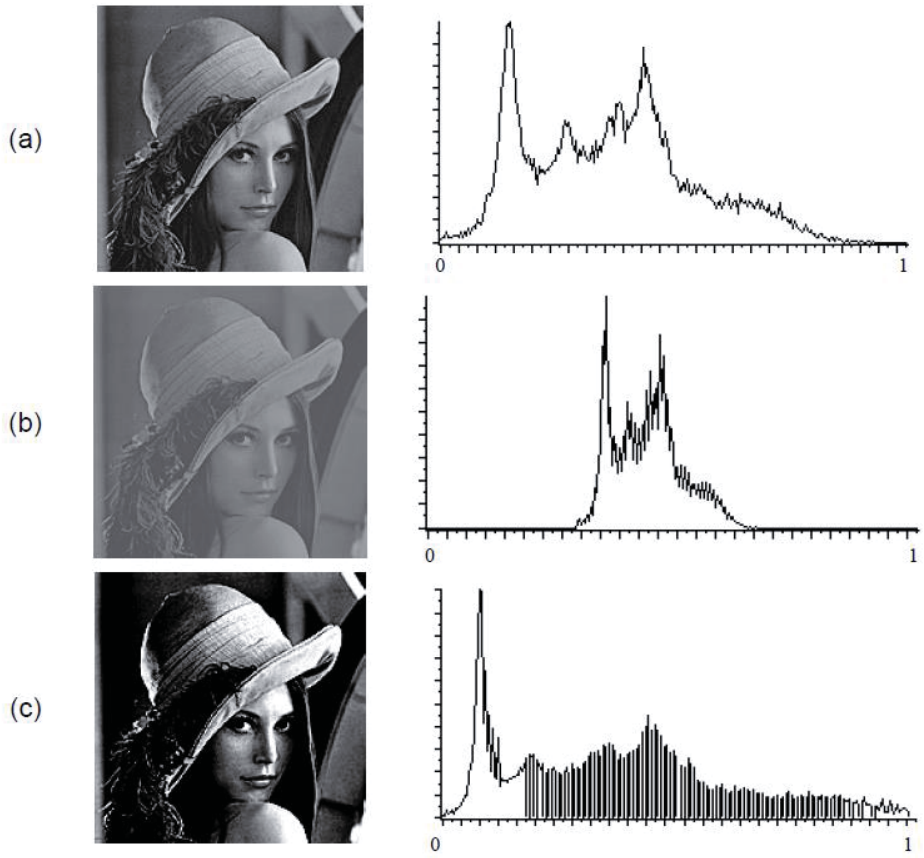



Figura **2.5** Transformaciones del histograma sobre la imagen de la mujer: (a) imagen original con su correspondiente histograma; (b) resultado de una operación de disminución de contraste; (c) aumento de contraste. **[**6].

### 2.2.2 Filtros espaciales

El filtrado espacial es una herramienta utilizada en muchas aplicaciones de procesamiento de imagen. Se utiliza para transformar una imagen a una representación alternativa, que puede ser desde generar una imagen borrosa o incluso para la detección de contorno. El termino *filtro* utilizado en el procesamiento de frecuencias se refiere a la capacidad que tiene de aceptar o rechazar ciertos componentes de una frecuencia.

Por ejemplo, un filtro que traspase bajas frecuencias se llama filtro *pasa-bajos*, el efecto de red producido por aquel filtro en emborronar (suavizar) una imagen.

En el área de procesamiento de imagen, el filtro consiste en una máscara que por lo general consiste en un cuadrado de dimensiones impares (3x3, 5x5, etc.) y alguna operación predefinida que es realizada con los pixeles que están dentro de la máscara. El filtrado genera un nuevo pixel con coordenadas igual a la coordenada del pixel del centro de la máscara y su valor es el resultado de la operación de filtrado.

Dentro de las operaciones predefinidas más difundidas están la de correlación y de convolución espacial. La correlación consiste en el corrimiento del centro de la máscara en cada pixel de la imagen cuyo nuevo pixel será el resultado de la suma de las multiplicaciones de cada pixel dentro de la máscara por el pixel correspondiente de la imagen original.

La convolución es una operación matemática que suma una función *f* consigo misma repetidas veces en todo el dominio de otra función *h*, utilizando en cada suma como valor de escala el valor de *h* en ese punto de su dominio. Su formulación matemática es:

$$f(x) * h(x) = \int_{-\infty}^{\infty} f(x)h(u-x)dx$$

Aunque la operación de convolución es conmutativa, en teoría de filtros, una de las funciones corresponde a la señal y la otra corresponde al filtro que se desea aplicar, denominando a esta última *función impulsional.*

En el caso bidimensional discreto, la operación de convolución se define mediante la siguiente ecuación:

$$I'(x,y) = I(x,y) * h(x,y) = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} I(i,j)\cdot h(x-i, y-j)$$
$$\forall x, y = 0,1,\ldots, N-1$$

Donde *I* es una matriz de tamaño N y *h* es una función periódica de periodo *N.* Así, para el caso bidimensional discreto de una imagen de tamaño *N*x*N*, la transformada inversa (*h*) del

filtro resulta ser una matriz también de tamaño *N*x*N*, que se conoce como *función de filtrado espacial* o *matriz o mascara de convolución.*

En principio la carga computacional de esta operación es elevada si *N* es grande. Sin embargo es posible que la función de transferencia *h* tenga un número de elementos muy inferior a *N*x*N* al estar muchos de sus valores a cero. Utilizando una función de convolución de tamaño menor al de la imagen obtenemos la siguiente expresión:

$$I'(x,y) = \frac{1}{D}\sum_{i=0}^{n}\sum_{j=0}^{n} I(i,j)\cdot h(x-i,y-j) \qquad \forall x,y = 0,1,\ldots,N-1$$

Donde *D* es un factor de escala que se conoce como *factor de división.* El siguiente ejemplo muestra la aplicación de un filtrado espacial con una máscara de convolución de 3x3.

### 2.2.3 Filtros de suavidad

La convolución de la imagen con un filtro de suavizado se utiliza generalmente para la reducción del ruido que pueda generarse durante la captación de la imagen. El filtro de suavizado espacial se basa en el promediado de los pixeles adyacentes al pixel que se evalúa. Quizás el filtro de suavizado más simple que se puede diseñar se corresponde con una matriz de 3x3 con todos los elementos a 1. El resultado de la convolución de cada pixel se deberá dividir por 9 para asegurar el obtener valores dentro del rango de la paleta. En la figura adjunta se puede apreciar el resultado de la aplicación de este filtro.

Las siguientes matrices de convolución definen otros filtros de suavizado:

$$h = \frac{1}{10}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix} \qquad h = \frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$

Otro filtro de suavizado es el *filtro de la mediana.* Éste se basa en sustituir el valor de un pixel por el de la mediana del conjunto formado por el mismo y sus ocho vecinos.

El *filtro del bicho raro* es otro ejemplo de filtro suavizado. Consiste en comparar la intensidad de un pixel con la de sus 8 vecinos. Si la diferencia es superior a cierto umbral *U* (que debe elegirse previamente), se sustituye tal pixel por el valor promedio de los pixeles vecinos, en otro caso se mantiene su valor de intensidad.

Debe observarse que tanto el filtro de la mediana, como el filtro del bicho raro son filtros no lineales, y por lo tanto no se pueden obtener mediante una operación de convolución.

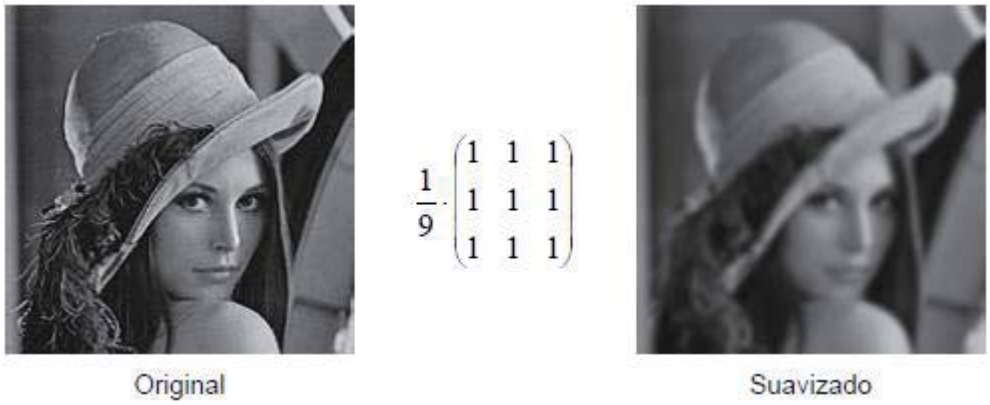



Figura **2.6** Aplicación de un filtrado espacial de suavizado. **[**6]

### 2.2.4 Filtros de obtención de contornos

El cálculo de la derivada direccional de una función permite conocer cómo se producen los cambios en una dirección determinada. Tales cambios suelen corresponder a los contornos de los objetos presentes en las imágenes.

Partiendo que el operador gradiente se define como:

$$\nabla(I(x,y)) = \frac{\partial I}{\partial x}\vec{u}_x + \frac{\partial I}{\partial y}\vec{u}_y$$

Se definen los filtrados de convolución *G:*

$$G_x = \frac{\partial I}{\partial x} = I(x,y) * h_1(x,y)$$

$$G_y = \frac{\partial I}{\partial y} = I(x,y) * h_2(x,y)$$

Obteniendo *h1* y *h2* mediante una aproximación a la derivada con la resta. Es decir, si se consideran los pixeles de la siguiente figura:

| | | |
|---|---|---|
| $z_1$ | $z_2$ | $z_3$ |
| $z_4$ | $z_5$ | $z_6$ |
| $z_7$ | $z_8$ | $z_9$ |

Las derivadas serían:

$$\frac{\partial I}{\partial x} = z_5 - z_6 \quad \text{y} \quad \frac{\partial I}{\partial y} = z_5 - z_8$$

Sustituyendo en *Gx* y *Gy* se deduce que las matrices de convolución *h1* y *h2* serán:

$$h_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{pmatrix} \qquad h_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \end{pmatrix}$$

Por ejemplo la matriz *h1* proporciona un filtrado en el que un cambio de brillo entre dos pixeles adyacentes en horizontal produce un valor distinto de cero. En particular los cambios de claro a oscuro se marcan con un valor positivo y los cambios de oscuro a claro con un valor negativo. Por otro lado, cuando dos pixeles adyacentes tienen el mismo valor la convolución con *h1* en ese punto devuelve cero.

En el caso de que no se desee considerar la dirección del vector gradiente, sino solo su módulo, se puede utilizar:

$$\left|I_G(x,y)\right| = \sqrt{G_x^2(x,y) + G_y^2(x,y)}$$

Con el fin de reducir la carga computacional la expresión anterior puede sustituirse por esta otra que produce un resultado similar:

$$\left|I_G(x,y)\right| = \frac{1}{2}\left(\left|G_x(x,y)\right| + \left|G_y(x,y)\right|\right)$$

Una aproximación mejor al gradiente está dada por las expresiones:

$$\frac{\partial I}{\partial x} = (z_1 + 2z_4 + z_7) - (z_3 + 2z_6 + z_9)$$

$$\frac{\partial I}{\partial y} = (z_1 + 2z_2 + z_3) - (z_7 + 2z_8 + z_9)$$

Dando lugar a las matrices *h1* y *h2* a las que también se les debe añadir un factor de división y uno de suma para evitar que se salga de rango el resultado.

$$h_1 = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} \qquad h_2 = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

Estas matrices se conocen como *ventanas de Sobel*, que fue quien las propuso. Mediante ellas se calcula el gradiente en las direcciones horizontal y vertical. En la figura 8 se ve cómo el resultado de aplicar *h1* sobre la imagen produce una imagen en la que aparecen los contornos horizontales de la figura de la imagen original. Ese resultado se obtiene utilizando un factor de división de 4 y presentando el valor absoluto de la convolución, utilizando niveles de gris en escala desde 0 como blanco hasta 255 como negro.

Una alternativa muy común al uso de valor absoluto para evitar los valores fuera de rango consiste en el uso de un factor de suma que se aplica tras la convolución y la división. Por ejemplo, en el caso del filtro de Sobel un factor de división de 8 y un factor de suma de 128 evitarían los valores fuera de rango.

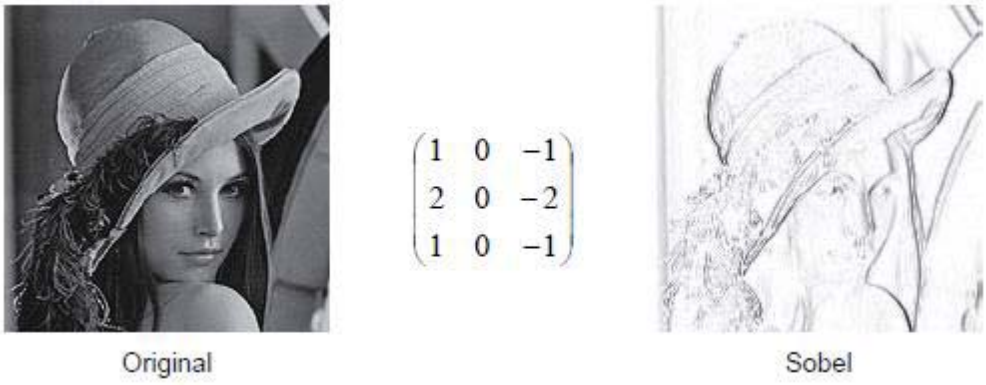



Figura **2.7** Filtrado de Sobel en la dirección *x* en valor absoluto. **[**6]

## 2.2.5 Operaciones morfológicas

Clásicamente la morfología ha sido una parte de la biología que estudia la forma de los animales y de las plantas. De la misma forma, la *morfología matemática* consiste en un conjunto de técnicas matemáticas que permiten tratar problemas que involucran formas en una imagen. La morfología matemática tiene su origen en la teoría de conjuntos. Para ella también las imágenes binarias son conjuntos de puntos 2D, que representan los puntos activos de una imagen, y las imágenes en niveles de gris son conjuntos de puntos 3D, donde la tercera componente corresponde al nivel de intensidad. En este apartado se tratará detalladamente la morfología sobre imágenes bitonales, para luego presentar los operadores básicos sobre imágenes en niveles de gris.

**Definiciones Básicas:**

Sea *A* en conjunto (con las operaciones habituales entre conjuntos) de Z (con las aplicaciones habituales entre vectores). Cualquier punto *a* de *A* se representa mediante un par (*a1*,*a2*). A continuación se definen las siguientes operaciones sobre *A*:

*Traslación* de $A$ por $X = (x_1, x_2)$, como:

$$(A)_x = \{c / c = a + x, \forall\ a \in A\}$$

*Reflexión* de $A$ como:

$$\hat{A} = \{x / x = -a, \forall\ a \in A\}$$

*Complementario* de $A$ como:

$$A^c = \{x / x \notin A\}$$

También se define la operación *diferencia* entre dos conjuntos $A$ y $B$ como:

$$A - B = \{x / x \in A \quad \text{y} \quad x \notin B\}$$

#### 2.2.5.1 Filtros morfológicos

La morfología puede ser utilizada como una herramienta para crear filtros morfológicos y mejorar una imagen o algún segmento de ella, a continuación se podrá ver unos ejemplos de filtros morfológicos útiles:

**Eliminación de ruido:**

Este filtro elimina los objetos de una imagen que tienen un tamaño menor que un elemento estructurante $B$ determinado.

$$\text{Limpiar}(A) = (A \circ B) \bullet B$$

En el proceso de apertura elimina los objetos menores que $B$, luego intenta recuperar la misma forma que antes con el proceso de cierre (aunque los procesos de apertura y cierre no son estrictamente inversos).

**Extracción de contornos**

Este filtro obtiene los contornos de una figura $A$ restándole su interior.

$$\text{Contornos}(A) = A - (A \ominus B)$$

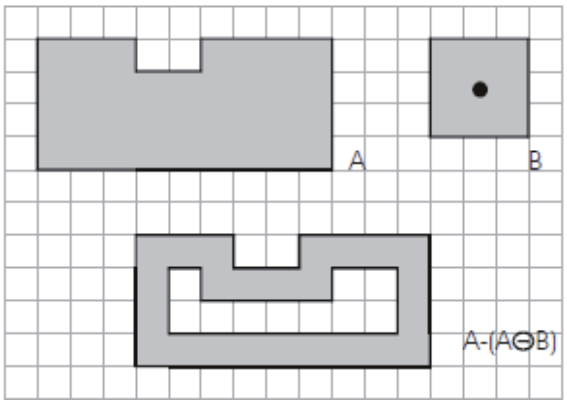



Figura **2.8** Ejemplo de extracción morfológica de contornos. **[**6]

**Relleno de agujeros**

Este filtro precisa de un proceso iterativo que concluye cuando no se producen más cambios sobre la imagen. Se parte de *X0* igual a un punto de agujero que se desea rellenar. Luego se aplica de manera iterativa:

$$X_k = (X_{k-1} \oplus B) \cap A^c$$

**Adelgazamiento**

Esta operación adelgaza los elementos de una imagen hasta que se producen a un esqueleto interior a la misma. Para poder realizar esta operación es preciso definir la siguiente operación:

$$A \otimes B = A - (A * B)$$

Este paso, utilizando la operación de coincidencia estructural, elimina un punto de *A*. Eligiendo cuidadosamente un conjunto de elementos *B*, que solo deben erosionar los bordes de *A*, y repitiendo esta operación sucesivamente se obtiene el resultado deseado.

$$A \nabla \{B\} = (\ldots((((A \otimes B1) \otimes B2) \otimes B3) \otimes B4) \ldots \otimes Bn)$$

La siguiente figura ilustra el proceso de adelgazamiento de un objeto mediante esta operativa.

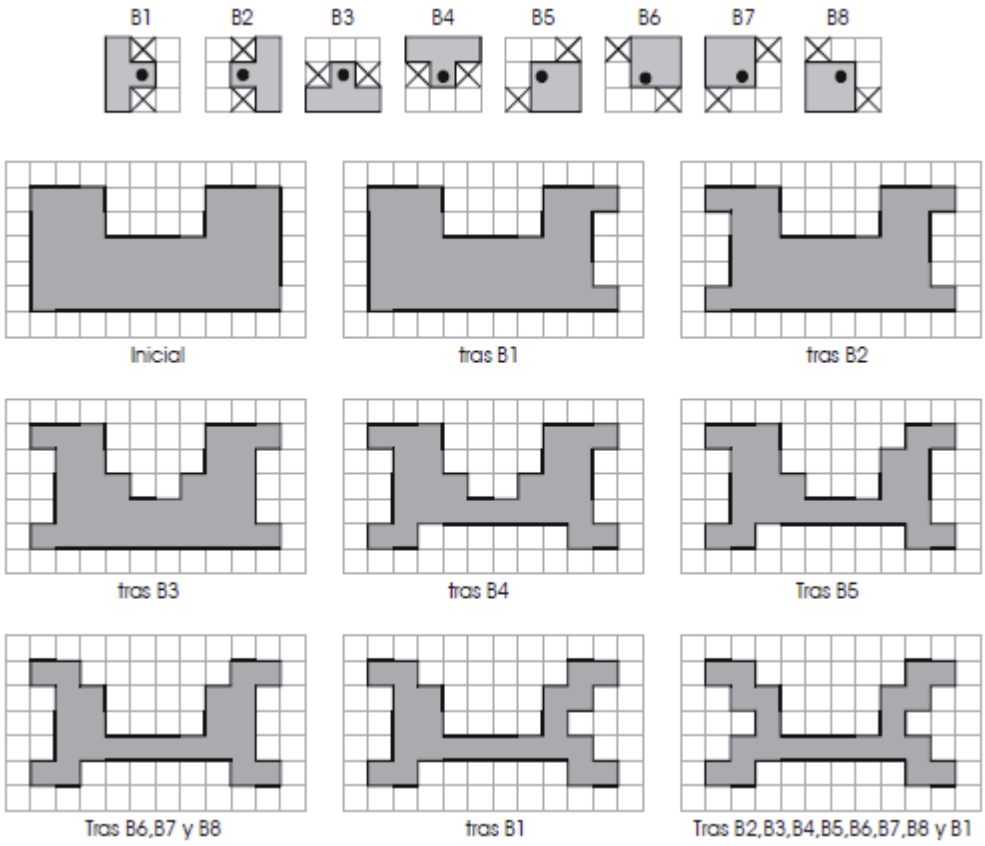



Figura **2.9** Ejemplo de aplicación de algoritmo de adelgazamiento. Las máscaras definidas van erosionando por el borde la figura original, en iteraciones sucesivas. **[**6]

### 2.2.6 Espacio de color

El propósito de un modelo de color (también denominado *espacio de color*) es para facilitar la especificación de los colores en algún estándar, de alguna forma aceptada generalmente. En su esencia, un modelo de color es una especificación de un sistema de coordenadas y un subespacio dentro de ese sistema donde cada color es representado por un punto específico.

La mayoría de los modelos de color en uso hoy en día están orientados hacia el hardware (como monitores e impresoras) o hacia las aplicaciones donde la manipulación de color es una meta (como la creación de gráficos de color para la animación). En términos de procesamiento digital de imagen, los modelos orientados al hardware más utilizados en la práctica son el modelo RGB (rojo, azul, verde) para monitores de colores y cámaras de video a color de amplia clase; el modelo CMY (cyan, magenta, amarillo) y el CMYK (cyan, magenta, amarillo, negro) para impresiones a color; y el modelo HSV (tono, saturación y valor), el cual corresponde muy de cerca con la forma que los humanos describen e interpretan el color.

**RGB:**

El modelo de colores RGB representa todos los colores como combinaciones de rojo, verde y azul. Este sistema es el más utilizado por la mayoría de los formatos actuales.

Los colores primarios del modelo RGB son aditivos, es decir, para producir el resultado se suman las contribuciones individuales de cada primario. Como se sabe, los colores básicos son el rojo, verde y azul y la mezcla de estos y su aplicación con mayor o menor intensidad se obtiene el resto de los colores; encontrando en los extremos con el blanco y el negro; es decir un color cuyo valor RGB es (0,0,0) es equivalente al color negro, y el valor (255,255,255) corresponde al blanco, en este punto cabe señalar que los valores de RGB son expresados en una combinación de 3 cifras cuyos valores vacilan entre 0 y 255.

A continuación se presenta un cubo unitario que representa el espacio RGB, donde el origen es la coordenada (0,0,0) que corresponde al color negro y el punto mas alejado con coordenadas (1,1,1) representa el color blanco.

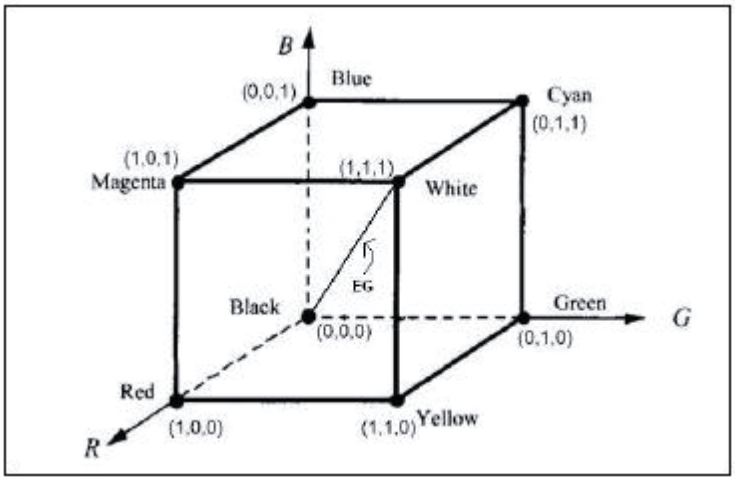



Figura **2.10** Ejemplo de extracción morfológica de contornos. **[**5]

Sin embargo, los seres humanos no perciben los colores como combinaciones de los tres colores primarios, lo que hace que sea poco intuitivo. Para ver una imagen con colores suficientes para ser representativos se habla de profundidad de color.

La profundidad de color se le llama a la cantidad de bits utilizado para representar el color de un solo pixel. A medida que el número de bits aumenta, el número de colores posible se convierte en un gran mapa de colores. Un ejemplo son profundidades de color de 16, 24 y 32 bits.

**HVS:**

El modelo HSV corresponde a identificar los colores según matiz, saturación y valor el cual está basado en la forma del ser humano de describir los colores. En HSV es un sistema de color que varía el grado de propiedades del color para crear nuevos colores. El matiz especifica el color como por ejemplo el rojo, naranja, azul, etc. La saturación (también conocida como cromo o pureza), se refiere a la cantidad de blanco en el matiz. Un color completamente saturado no contiene blanco y aparece puro, un rojo con 50% de saturación resulta un color rosa. Y el valor (conocido también como brillo) es el grado de luminosidad de cierto color (que tanta luz emite). Un tono con alta intensidad es brilloso, y al contrario uno con poca intensidad corresponde a un color más oscuro.

Al sistema HSV le corresponde un sistema de coordenadas cilíndrico, y el subconjunto del espacio donde está definido el modelo es un cono hexagonal o pirámide de seis lados. La parte superior del cono hexagonal corresponde a V=1, que contiene los colores relativamente brillantes. Sin embargo, no todos los colores del plano V=1 se perciben con la misma brillantez.

EL matiz se mide con el ángulo alrededor del eje vertical, con el rojo en los 0º y el verde en 120º, etc. En el modelo HSV los colores complementarios están opuestos en 180º entre sí. El valor de S es una razón que varía de 0 en la línea central (eje V) a 1 en los lados triangulares del cono hexagonal. La saturación se mide con respecto a la gama de colores representada por el modelo, la cual constituye un subconjunto de todo el diagrama de cromaticidad CIE. Por lo tanto, una saturación del 100% en el modelo es menor que el 100% de pureza de excitación.

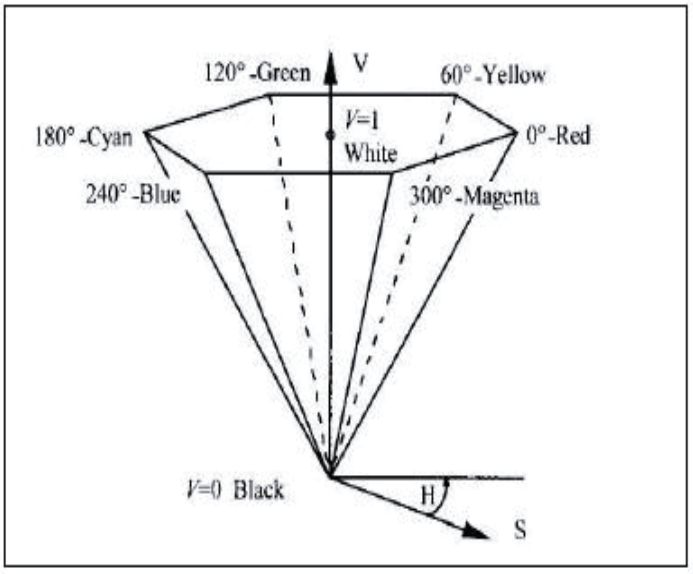



Figura **2.11** Representación del espacio de color HSV. **[**5]

## 2.2.7 Segmentación

Un tópico de mucha importancia en el proyecto es el proceso llamado segmentación. La segmentación es un proceso que consiste en dividir una imagen digital en regiones homogéneas con respecto a una o más características (como por ejemplo el brillo o el color) con el fin de facilitar un posterior análisis o reconocimiento automático. Localizar la cara de una persona dentro de la imagen de una fotografía o encontrar los límites de una palabra dentro de una imagen de un texto, constituyen ejemplos de problemas de segmentación.

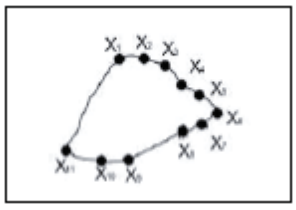
El proceso de segmentación de una imagen depende del problema que se desee resolver. Por ejemplo, sobre una imagen de una página de texto se pueden segmentar las líneas de texto (si el objetivo es localizar la estructura de los párrafos), o las palabras y los caracteres que las forman, o los logotipos y membretes (si se desea calificar el documento), etc. Por ello, dentro de una misma imagen pueden realizarse diferentes segmentaciones.

En general el proceso de la segmentación suele resultar complejo debido, por un lado, a que no se tiene una información adecuada de los objetos a extraer y, por otro, a que en la escena a segmentar aparece normalmente ruido. Es por esto que el uso de conocimiento sobre el tiempo de imagen a segmentar o alguna otra información de alto nivel puede resultar muy útil para conseguir la segmentación de la imagen.

### 2.2.7.1 Contorno

El contorno de un objeto en una imagen digital corresponde al mínimo conjunto de pixeles que separa ese objeto del fondo o *background* de la imagen. Normalmente estos contornos se corresponden con los puntos donde se producen discontinuidades en los valores de pixeles adyacentes (cambios en el matiz o el brillo) o con los puntos donde cambia un patrón que se repite (cambios de textura).

Por ejemplo, cada segmento (X) se especifica mediante el punto inicial y final. La concatenación de estos puntos, con el mismo punto inicial y final describe un contorno.



Normalmente para detectar contornos se utiliza una imagen binaria, los ceros como cero y cualquier valor distinto de cero como unos.

El algoritmo para detectar contornos consiste en un barrido de la imagen hasta encontrar un punto en el objeto, una vez encontrado comienza a recorrer el borde del objeto guardando los puntos que se encuentren, ya sea en un array o en una lista de secuencias. Una vez terminado con el objeto, este y su interior se borra de la imagen para seguir buscando contornos de otros objetos.

### 2.2.7.2 Enfoques de segmentación

Actualmente la segmentación de imágenes continua siendo un área activa de investigación. Existen numerosas técnicas de segmentación, a continuación se mencionaran algunas de las técnicas utilizadas recientemente:

**Segmentación basada en el color:**

El aumento de resolución radiométrica siempre aporta nueva información que puede facilitar el proceso de segmentación. Sin embargo siempre debe tenerse en cuenta que los requisitos computacionales aumentan considerablemente respecto a las técnicas basadas en imágenes en niveles de gris o bitonales.

Ya se ha mencionado que el color se puede representar como la unión de tres planos, cada uno con la información relativa a la intensidad de cada punto respecto de cada una de las componentes de una base de color (por ejemplo de rojo, verde y azul en el modelo RGB). Una técnica común de segmentación en color consiste en separar el proceso en dos fases. En la primera se aplican las técnicas que se ha estudiado para niveles de gris a cada uno de los tres planos RGB. En la segunda se integran los resultados de la primera para producir como resultado la segmentación de la imagen en color.

Una técnica de este tipo es aquella basada en el crecimiento de regiones. En concreto se aplica una variante que contempla el color del algoritmo de "Split-and-merge". El algoritmo queda de la siguiente forma:

**- Algoritmo Split and Merge para segmentación basada en el color-**

**Paso 1.**- Se calculan las características de color usando los valores de las componentes de los planos rojo, verde y azul de una imagen RGB.

**Paso 2.**- La imagen se divide en regiones cuadradas de igual tamaño, usando la estructura de datos de árbol cuaternario o "*quadtree*".

**Paso 3.**- Cuatro cuadrantes situados a un mismo nivel de subdivisión son mezclados si se satisface un cierto criterio de homogeneidad. Un cuadrante se subdivide en otros cuatro si no se satisface una condición de homogeneidad.

A continuación, se enumeran otras dos técnicas descritas en la literatura sobre la segmentación basada en el color:

- Umbralización global de la componente de color usando el espacio de representación HSV.
- Uso de algoritmos de agrupamiento o *clustering*; en particular, la aplicación del algoritmo de las *k-medias* sobre los pixeles de la imagen en color.

Figura **2.12** Ejemplo de segmentación de una imagen en color. **[**6]

### 2.2.7.3 Segmentación basada en la textura

En este enfoque se definen *modelos de texturas* para pensar en una imagen no como en una colección de pixeles, sino para tratarla como una función I(x,y). El propósito del modelo es transformar una ventana de una imagen en una colección de valores que constituyen un *vector de características*. Este vector será un punto de un espacio n-dimensional. La representación corresponderá a la textura si ventanas tomadas de la misma muestra de textura están "cercanas" en el espacio de características, y si ventanas de la imagen con diferentes patrones de texturas quedan "alejadas" en el espacio de características considerado. Los modelos de textura se dividen, a grandes rasgos, en tres categorías: basados en *estructuras piramidales*, que tratan de capturar las frecuencias espaciales a distintos niveles de resolución; basados en *campos aleatorios*, que asumen que los valores de un pixel son seleccionados mediante un proceso estocástico bidimensional; y los basados en métodos estadísticos, que utilizan matrices de *co-ocurrencias* construidas a partir de las imágenes y de estas matrices se extraen una serie de medidas como la media, la varianza, la entropía, la energía y la correlación entre los pixeles.

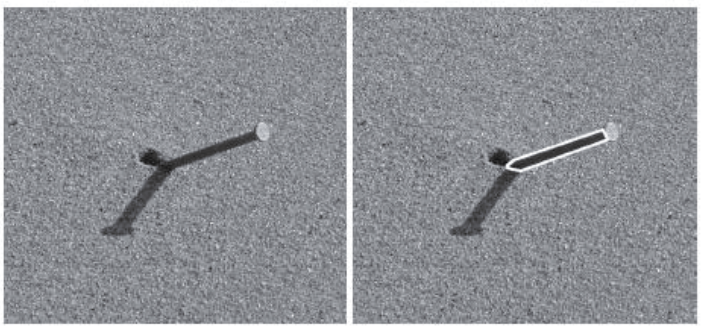


Figura **2.13** Imagen inicial, versus resultado de su segmentación. **[**6]

#### 2.2.7.4 Segmentación basada en el movimiento

El movimiento puede constituir una potente herramienta para la segmentación de objetos animados sobre fondos estáticos. Las técnicas básicas consisten en el estudio de la imagen resultante de la resta de dos imágenes consecutivas de una secuencia animada. Esta técnica se conoce con el nombre de *substracción de fondo*. Los objetos que se desplazan entre estas dos imágenes producen en la imagen resta un conjunto de pixeles con valores distintos a cero. Mientras, los elementos estáticos de la imagen, por no variar, producen cero tras la resta.

Así, partiendo de dos imágenes *I(t)* e *I(t+1)* de dos instantes consecutivos los objetos tras esta segmentación serían los pixeles a uno de la imagen *I(d)*.

$$d_{t,t+1}(x,y) = \begin{cases} 1 & \text{si } \left|I_t(x,y) - I_t(x,y)\right| > U \\ 0 & \text{en otro caso} \end{cases}$$

Siendo *U* un valor umbral que depende de la variación de la iluminación entre los instantes t y t+1.

## 2.3 OpenCV

OpenCV es una librería de código abierto. Esta librería está escrita en C y C++ y corre bajo Linux, Windows y Mac OS X. OpenCV posee un diseño computacionalmente eficiente y está enfocado en las aplicaciones de tiempo real. Si es que la configuración del Hardware así lo permite aprovecha el uso de más de un procesador [8].

Su objetivo es proveer de una infraestructura de simple uso para desarrollar aplicaciones de visión relativamente sofisticada en una cantidad de tiempo limitada. La librería posee alrededor de 500 funciones que abarcan muchas áreas de la visión como imágenes médicas, seguridad, interfaz de usuario, calibración de la cámara, visión estéreo y robótica. Incluye una librería de propósito general Machine Learning (MLL) para clustering y reconocimiento de patrones.

OpenCV se estructura en cinco componentes principales, en la figura a continuación se muestran cuatro de ellas.

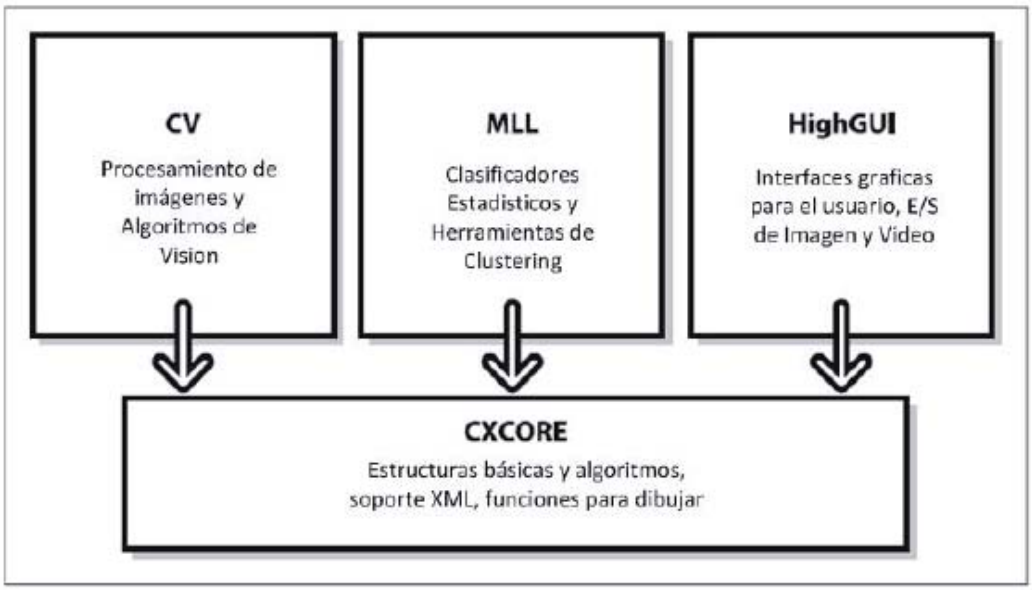



Figura **2.14** Estructura de OpenCV.

# 3 Desarrollo de la aplicación

A continuación, se describirá de forma más detallada, la aplicación que fue desarrollada para la detección de figuras geométricas y sus características, la cual tiene como función determinar la posición y dirección de objetos dentro de la escena, a través de algoritmos de procesamiento de imágenes apoyado por la librería OpenCV.

## 3.1 Formación de la Aplicación

Como se ha visto anteriormente, el presente proyecto pretende dar un nuevo sentido a un escenario de algún tipo de simulación o recreación. Cuando se habla de simulación se entiende por la imitación de cierto sistema del cual se quiere experimentar, que en general se utilizan objetos o agentes que interactúan dentro del ambiente, lo cual sirve para ver el comportamiento por un periodo determinado y lo resultados que se puedan obtener de tal ejercicio. En estos tipos de sistemas es muy común el uso de sensores que son los encargados de determinar ciertas variables importantes, como de posición, temperatura, altitud, humedad, proximidad, etc. y poder obtener lo que se requiere.

Y lo que se trabaja es utilizar un recurso no tan común como los demás, ya que en general puede ser más complejo de manipular. Las imágenes pueden proporcionar un nuevo sentido y nuevas posibilidades dentro de cualquier ambiente que se desea observar, y aún más importante, concluir situaciones y calcular resultados.

La aplicación tiene como finalidad dar una visión global al escenario ficticio, y dar mayores conclusiones sobre la posición y dirección de uno o más objetos en la simulación. A

través de una cámara ubicada sobre la escena, la cual será capturada de forma digital, y manipulada en un computador con algoritmos de procesamiento. Esta capturará no los objetos, sino una forma geométrica de tal forma de que se pueda obtener la posición de tal elemento y su dirección a la cual están orientadas.

La aplicación que estará alojada en el computador principal se encargara de detectar las figuras geométricas determinadas según la necesidad, y determinando su punto medio ubicara su posición dentro del plano, con coordenadas (x,y). Y la dirección será determinada mediante el tipo de figura captada, donde será calculada según la dirección con respecto al eje x. La ejecución del programa será constante, y capturara de forma continua lo que suceda dentro del campo de simulación, y no será necesario almacenar ningún tipo de video dentro de la memoria del computador, al menos que la problemática lo requiera.

Además, dependiendo de la cantidad de elementos que interactúan dentro de la escena, será posible determinar su distancia, ya que puede ser necesario mantener cierta distancia entre objetos, con tal de evitar colisiones o interacciones inesperadas.

## 3.2 Desarrollo del proyecto

El desarrollo de proyecto presente ha variado considerablemente desde su inicio, ya que su desarrollo contempla un cambio dinámico de sus funcionalidades y alcances, determinados por variables como tiempo, dificultad y su uso práctico en la realidad.

Es por esto que en la entrega actual se ha decidido acotar en parte el alcance que tiene, que como principal ausente seria su implementación efectiva en el problema inicial, la cual consistía en la simulación de un juego de futbol con robots LEGO, a través de un sistema multiagente. Por lo que se generalizo el uso de la aplicación, a cualquier ambiente o experimento donde se requiera obtener localización u orientación de algún objeto o agente.

. En concreto se decidió realizar solamente el algoritmo de detección de figuras geométricas a través de la cámara, y que determine su distribución, determinando la distancia mínima entre dos objetos, ángulo de dirección y ubicación espacial, lo cual serviría en una futura implementación de este algoritmo a algún problema futuro, utilizando los parámetros obtenidos del procesamiento de imágenes.

Esta decisión fue resuelta después de determinar que la dificultad para realizar el traspaso de lenguaje, o sea mediante del WRAPPER sería más compleja de lo planeado, y excedería las capacidades y recursos del programador. Simplificando el problema y alcance determina un mayor grado de éxito del proyecto y de finalizar lo planteado.

Mientras tanto, la aplicación de procesamiento de imagen tiene como función capturar cada frame de la cámara, transformarlo, a través de filtros y segmentaciones, a una imagen más fácil de procesar y detectar un triángulo. Posteriormente se le calculara su ángulo de dirección en base al eje X, y la distancia entre dos triángulos en la escena.

No fue necesario aplicar un filtro de corrección de ojo de pez, ya que el algoritmo fue lo bastante eficaz para detectar los triángulos sin mayores problemas.

## 3.3 Alcance futuro

El posible uso futuro de la aplicación es muy amplio, puede ser utilizada en cualquier ambiente o sistema informático de simulación donde se busque obtener variables de objetos en movimiento, su posición y dirección.

Uno de sus posibles usos para la actual aplicación sin duda podría ser el proyecto inicial del presente. El cual consistía en mitigar las colisiones de unos robots lego, que simulaban un juego de futbol simplificado en un campo de futbol miniatura. Este proyecto o sistema, tenía como finalidad recrear el ambiente de un partido de futbol, con tres jugadores y tres roles distintos: arquero, mediocampista y delantero. A través de una figura ubicada en la parte superior de los jugadores y la cámara captando la cancha de forma superior, se puede obtener los datos de la posición de los robots y en relación con sus compañeros, con el fin de mitigar sus colisiones de mejor forma.

Y entre sus usos posibles en forma general, se podría decir que en algún sistema o simulación de cualquier naturaleza, donde sea importante la ubicación de los individuos, su dirección para poder determinar su posible trayectoria o meta, y donde se busque la relación de distancia de dos o más componentes.

## 3.4 Implementación

Ya dicho anteriormente, la solución escogida para el procesamiento de imagen será un triángulo como figura geométrica escogida para ser detectada en la escena. Se seleccionó esta por ser la más adecuada para la detección de la dirección del objeto, siendo también una muy simple. La razón más natural de la elección de la figura fue básicamente por ser la figura geométrica con menor área, pero a la vez la más sencilla de calcular su ángulo de inclinación. Además posee el menor número de vértices, por lo que es más fácil ser detectada por los algoritmos de contornos.

En la escena está definido para que sean detectados dos triángulos como máximo, y estos deben tener un tamaño mínimo establecido. Tienen por característica ser un triángulo isósceles, para hacer más fácil su manejo y calculo, ya que es más sencillo de direccionar, calcular su línea base y su ángulo de inclinación.

A continuación se mencionaran y describirán las funciones principales utilizadas en la programación del prototipo de aplicación. Todas son parte de las librerías de OpenCV, específicamente las librerías <cv.h> y <highgui.h>.

**Funciones:**

- cvFindContours(): Función encargada de buscar todos los contornos presentes en un frame determinado, almacenando en una variable puntero el conjunto de contornos.

- cvAproxPoly(): Función que tiene como tarea determinar los puntos o vértices de los contornos, almacenando en una variable su cantidad.

- cvLine(): Función que sirve para dibujar en alguna pantalla, según coordenadas dadas.

- cvPutText(): Función que permite escribir por pantalla una cadena de caracteres.

- cvSmooth(): Función para aplicar un filtro Gaussiano a un frame.

- cvCvtColor(): Función que otorga propiedades del tipo color a una imagen, como por ejemplo en blanco y negro.

### 3.4.1 Pasos de la aplicación

A continuación se describirá la secuencia de pasos simplificada, utilizada para detectar y localizar los triángulos en la escena capturada:

- Captura de video a través de una cámara conectada al computador. Tarea que se realiza con la función cvCapture(), a la cual se le indica la cámara que se desea manipular.
- Crear imagen idéntica a un frame del video, la cual será manipulada para ser procesada. De forma posterior, el resultado de los algoritmos será fusionada con la imagen inicial.
    - Procesar cada frame capturado, hasta finalizar el programa.
- Aplicar filtro Gaussiano para suavizar la imagen.
- Transformar la imagen a una escala de grises.
- Cambiar el umbral de los colores de la imagen, para facilitar su procesamiento

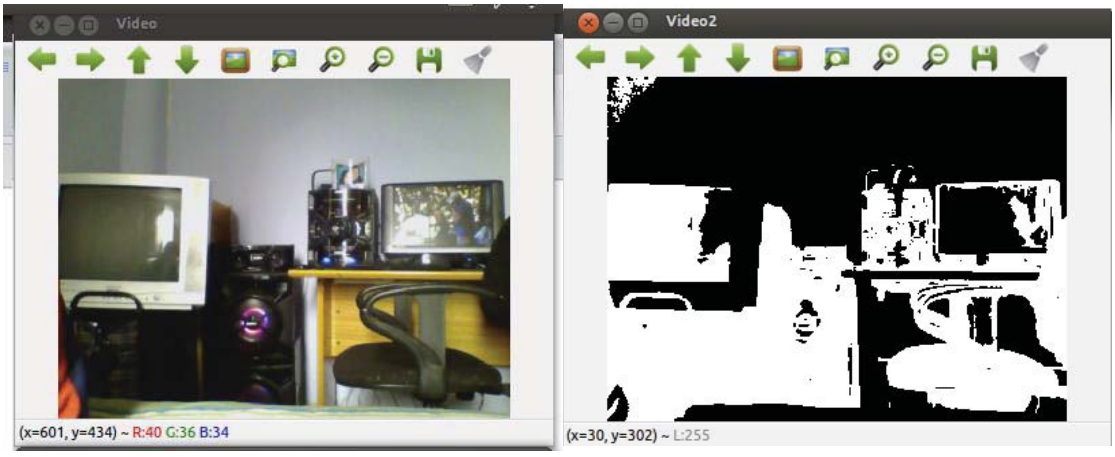



Figura **3.1** Imagen Original y Retocada.

- Aplicar detección.
- Determinar contornos presentes en la imagen.
- Determinar vértices de los contornos
- Si poseen tres vértices significa que son triángulos y por ende serán procesado
- Obtener los puntos de los vértices.

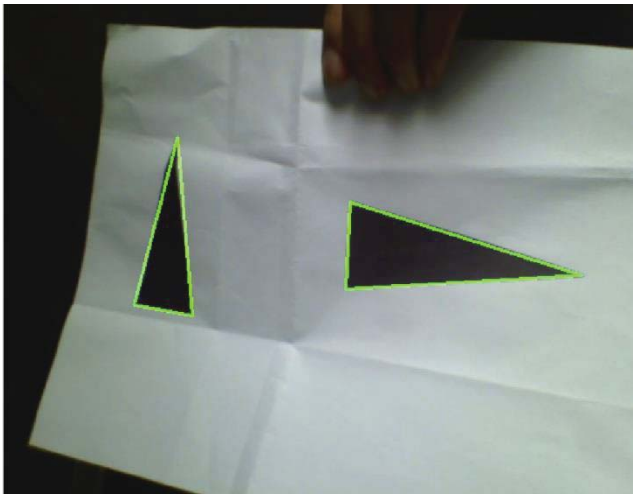


Figura **3.2** Detección de Triángulos en base a sus vértices.

- Obtener puntos medios de los triángulos, en base a sus vértices.
- Dibujar triángulos en la imagen clonada.
- Determinar qué lado es el más pequeño, para localizar su lado base.
- Localizar punto medio del lado, uniéndolo con su vértice opuesto.
- Crear un nuevo triangulo, basándose en la línea creada anteriormente como hipotenusa.

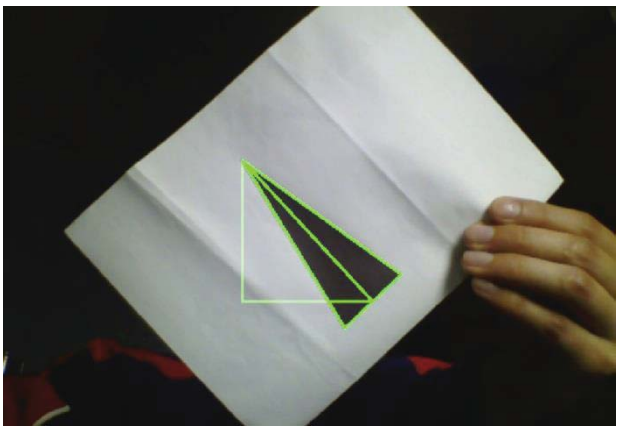


Figura **3.3** Dibujo del nuevo triangulo, del cual se obtiene el ángulo de inclinación.

- Calcular ángulo del nuevo triangulo, que corresponde al ángulo de inclinación del triángulo original.
- Imprimir el ángulo del triángulo en grados.
- Con dos triángulos detectados, calcular la distancia entre sus puntos medios.

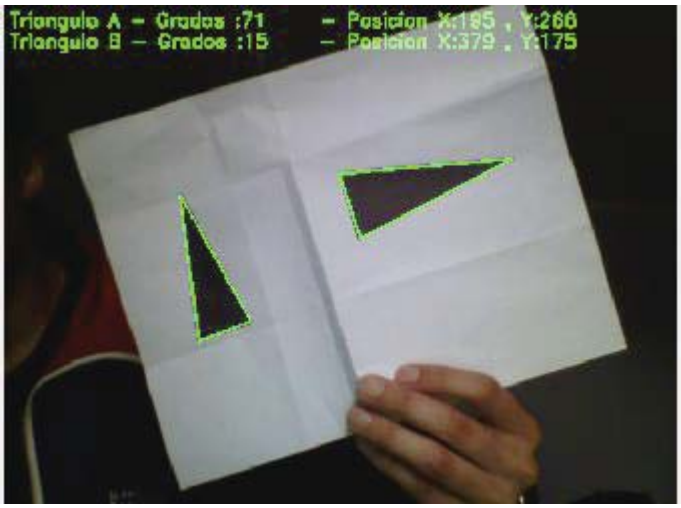



Figura **3.4** Dibujos de los triángulos, y sus respectivos datos espaciales.

- Unir imagen original con la modificada.
- Borrar la imagen modificada para no dejar dibujos anteriores
- Esperar a la cancelación del programa con la tecla Escape.
- Repetir detección.

## 3.5 Plan de pruebas

El plan de prueba consiste en la verificación de las funcionalidades del sistema, chequeando de forma metódica los detalles que puedan presentarse. Estas pruebas se realizan con el fin de poner a prueba el o los módulos y ver cuál es la reacción.

### 3.5.1 Procesador de imágenes

Para probar el procesador de imágenes y su eficacia, es necesario que sea capaz de detectar las figuras que están involucradas en el proyecto. En el caso particular del proyecto actual se requiere que detecte figuras geométricas como el cuadrado, triangulo, y circulo. Bajo ciertas condiciones estas figuras deben ser identificadas para aplicar algoritmos que beneficien a la simulación como lo son su movimiento, interacción con las demás figuras, diferenciación del resto de elementos no relevantes, etc.

Un aspecto importante para probar es la capacidad de detectar contornos en la escena, independiente de su locación o dirección. Ya que al trabajar con un lente especial, este elemento se vuelve crucial al momento de procesar el video captado.

Por lo que esta prueba no es necesario el entorno real donde se aplicara la aplicación de forma práctica, solo basta con ocupar las figuras de colores que se utilizaran y moverlas en un patrón que sea conflictivo y relevante para el caso de estudio, como también emular la luminosidad de la escena y color de fondo.

Otro ítem relevante dentro de lo que es importante probar la presencia de uno o dos triángulos en la escena, y ver cómo se comportan los algoritmos en estas condiciones. Y un detalle importante es que el cálculo de distancias sea solo entre dos elementos, en este caso dos triángulos.

### 3.5.2 Cámara

La cámara es fundamental en el desempeño del algoritmo o del sistema de visión artificial, ya que dependiendo de esta y de sus características puede ser posible alterar variables, umbrales o sistema independiente de luz, para lograr una buena detección.

En perspectiva, fue probado primero la cámara con lente de gran angular, ya que será la opción recomendada, por su gran capacidad de capturar la escena sin estar tan lejos del suelo, la hace más práctica. Pero tiene un inconveniente, la cámara suele deformar la imagen a una situación llamada “Efecto Barril”, el cual le da un efecto de lupa a la imagen. Probando los algoritmos de detección se pudo confirmar que esto no es un problema a una altura adecuada y a un tamaño de los triángulos normal. Las líneas, a pesar de capturarse de forma curva en los bordes de la imagen, estas siguen siendo detectadas, y mejor aún siguen siendo parte del triángulo observado.

En cuanto al color tampoco parece haber un problema, además que la aplicación contempla el uso del filtro blanco y negro, para evitar errores de detección. Para que el color no sea una variable que se entrometa.

### 3.5.3 Figura geométrica

Se ha especificado que la figura a ser utilizada será el triángulo. Esta decisión fue resuelta después de experimentar y manejar la mejor opción para la problemática.

El triángulo tiene la característica de ser la figura geométrica más simple, ya se posee solo 3 lados y tres vértices, quizás sería aún más simple un círculo en cuanto a estos aspectos, pero carece de características que se desean para el cálculo de ángulo. Y también posee la menor área de ocupación del espacio, lo que facilita la distinción y cantidad de triángulos totales en la escena.

Además, se escogió el triángulo isósceles de los tres tipos de triángulos, fue una opción natural ya que será más fácil calcular su ángulo de inclinación por tener un lado distinto a los

otros dos. En este caso, serán dos lados más largos e iguales que el de base, el cual servirá de pivote para el cálculo de su dirección con respecto al eje x.

# 4 Conclusiones

Los sentidos son cruciales para interactuar con el sistema y con lo que rodea a los seres vivos, gracias a estos es posible tener consciencia del entorno, los peligros y ventajas que les otorga y la posibilidad de sacar un provecho, algo positivo, en pro de la sobrevivencia y comunidad. Y los sistemas creados por el ser humano, las simulaciones o experimentaciones no están ajenos a este concepto, muchos de estas invenciones recrean situaciones donde interactúan individuos que necesitan o requieren conocer su entorno, para poder realizar su tarea y ser eficaces en sus labores y llegar a la meta propuesta.

La visión artificial o el procesamiento de imágenes es una gran herramienta informática de algún sistema, muy poderosa y efectiva, que puede ser muy útil según el contexto que se le aplique. Como sentido adicional a los sensores normales es un complemento con buenas capacidades y versatilidad, para todo tipo de aplicación. Donde se busque detectar figuras, contornos, colores, movimiento, ubicación, dirección, trayectoria, este tipo de procesamiento será útil y podrá resolver de buena forma cada una de esas variables.

En si es una herramienta completa, pero no perfecta. Depende mucho de las condiciones donde se aplique, el tipo de escenario, tipo de cámara utilizada, problemática a ser resuelta, cantidad de elementos a ser detectados, etc. si algunas de las especificaciones no ha sido revisada, el algoritmo no será tan eficaz como fue planteado, creando resultados erróneos o falsos positivos. Este tipo de aplicaciones son eficaces cuando son utilizadas bajo condiciones controladas y determinadas.

La aplicación logra varios acontecimientos muy útiles sobre el procesamiento de imágenes, la capacidad de detectar en movimiento una figura geométrica y relacionarla con otra, para poder obtener su posición dentro del plano y determinar su dirección o ángulo de forma individual. A pesar de aquello, lo importante a futuro seria darle un uso práctico a la aplicación, un caso real donde pueda ser utilizada y poder sacar provecho de este sentido adicional.

Como era desde un comienzo, el presente proyecto ha cambiado considerablemente, ya que no se logró conseguir la meta inicial. Por causa de las capacidades del proyectista, falta de motivación y frustración el proyecto no siguió con lo propuesto. Pero con un cambio de actitud y acotando los objetivos y alcances del proyecto, se pudo llegar a una solución más realista se pudo concluir el trabajo, llegando a un resultado más o menos concreto.

Todo el aprendizaje del presente ramo será canalizado hacia el mundo laboral, servirá de experiencia en pro de la solución de problemas reales, estimar de mejor forma las metas, considerar traspiés y mantener un colchón a errores, mejorar la motivación y continuar a pesar de todo, ser constante y consistente durante el trabajo. lo más importante es aprender de los errores y continuar luchando, consiguiendo las metas propuestas.

# 5 Referencias


[1] W. Calero, «Modelo de construccion de prototipos,» 2010. [En línea]. Available: http://ingenieraupoliana.blogspot.com/2010/10/modelo-de-construccion-de-prototipos.html [Último acceso: mayo 2012].

[2] Desarrollo orientado a prototipos.[En línea]. Available: http://html.rincondelvago.com/desarrollo-orientado-a-prototipos.html [Último acceso:mayo 2012]

[3] Windows, «Platform compatibility and system requirements». Available: http://www.microsoft.com/visualstudio/11/en-us/products/compatibility [Último acceso: junio 2012]

[4] Grupo de investigacion EDMANS, «Tecnicas y algoritmos basicos de vision artificial,» 2006. [En línea]. Available: http://silicondevelop.files.wordpress.com/2010/05/tecnicas-y-algoritmos-basicos-de-vision-artificial.pdf [Último acceso: abril 2012]

[5] R. Gonzalez, R. Woods, Digital Image Procesing. 2008, Third Edition.

[6] J. Vélez, A. Moreno, A. Sánchez, J. Sánchez-Mariin, Vision Por Computador, 2003. [En línea] Available: http://www.terra.es/personal/jfvelez/libro2/libro.html [Último acceso: abril 2012]

[7] C. Platero, «Introduccion a la vision artificial,» 2009. [En línea]. Available: http://www.elai.upm.es:8009/spain/Asignaturas/MIP_VisionArtificial/ApuntesVA/cap1IntroVA.pdf [Último acceso: abril 2012].

[8] OpenCV, 2012. [En línea]. Available: http://opencv.willowgarage.com/wiki/ [Último acceso: Septiembre 2012].

[9] W. Sotomonte , J. Gómez, Estrategias de Sistema de Agentes (Simple y Múltiple): Caso de Estudio Fútbol de Robots, Universidad Nacional de Colombia, 2005.

[10] L. Ortega, Sistemas Multiagente y Fútbol de robots: Estado del Arte, Universidad Nacional De Colombia, 2006.

[11] A. Catalan,Sistema Multi-Agente colaborativo para Soccer-Robot , PUCV, 2011