

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/309204287>

Objects Detection using haarscascade algorithms and OpenCV

Thesis · July 2016

CITATIONS

0

READS

1,440

1 author:



[Carlos M. Padilla](#)

Universidad Politécnica de Madrid

1 PUBLICATION 0 CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:



Combining wireless sensor network technologies and low cost unmanned aerial vehicles for sea turtles monitoring in the Guanahacabibes Peninsula [View project](#)



DETECCIÓN DE OBJETOS EN IMÁGENES DIGITALES

INTELIGENCIA ARTIFICIAL



Por Carlos M. Padilla

TUTORES: DR. C. ABDEL RODRIGUEZ | MSC. MARCOS LEIVA

Camagüey | julio 2016

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de este trabajo y autorizo al Departamento de Ingeniería Informática de la Universidad de Camagüey para que haga el uso que con el estime pertinente. Siempre y cuando respete los derechos morales del autor y haga debida citación de la fuente.

Para que así conste firmo la presente a los _____ días del mes de _____ del 2016.

Carlos M. Padilla

Autor

Dr. C. Abdel Rodríguez

Tutor

MSc. Marcos Leiva

Tutor

*A Dios, ante todo, por Todo. Y a ti Laura,
por lo difícil que me fue trabajar pensándote.*

Agradecimientos:

A Dios por saberme cuidado por El en todo momento, a mi madre por soportar la desorganización de mis instalaciones y a los abuelos por apoyarme sin demora en mis faltas, a los tutores por su paciencia y consejos y a los amigos todos por sus sugerencias, preocupaciones y ánimos.

RESUMEN

El Parque Nacional de la Península de Guanahacabibes es un área crítica para el desove de las tortugas marinas en la región del Caribe. Esta especie que viene a depositar sus huevos está en peligro de extinción y uno de los factores que lo provoca es la acción del hombre. En Cuba se han desarrollado acciones para asegurar que las nuevas vidas se conserven gracias a la creación de grupos de activistas, en su mayoría voluntarios. Pero estos grupos no cuentan con suficientes efectivos en la actualidad, el costo de mantenerlos en activo es elevado y su presencia continua resulta de impacto negativo para dicho hábitat protegido.

El proyecto *“Combining wireless sensor network technologies and low cost unmanned aerial vehicles for sea turtles monitoring in the Guanahacabibes Peninsula”* se enfoca en el diseño, prueba y desarrollo de un sistema de monitoreo automatizado para ayudar a salvaguardar y observar los nidos de las poblaciones de tortugas marinas. Todo esto aprovechando las bondades de recientes tecnologías en el campo de los Vehículos Aéreos no Tripulados (UAV por sus siglas del inglés), la Visión por Computador (CV por sus siglas del inglés) y los relativamente bajos costos de su implementación integral en un sistema completo de reconocimiento de objetos en imágenes digitales. Los resultados obtenidos por dicho sistema proporcionan a los especialistas del área la información necesaria para desarrollar su labor, ganando en tiempo, exactitud y reduciendo tanto el impacto ambiental como los costes de mantenimiento. Este trabajo de grado constituye la primera etapa de desarrollo del sistema final requerido, contribuyendo así con la selección, implementación y validación de algoritmos capaces de llevar a cabo en su conjunto, las citadas tareas con eficiencia y eficacia.

Palabras claves: Vehículos aéreos no tripulados, Visión por computador, Algoritmo de detección de objetos.

ABSTRACT

Guanahacabibes Peninsula National Park is a critical area for the marine turtles of the Caribe's region spawn. This species is in extinction danger and one of the factors that produce that is the human action. Action have been developed in Cuba to assure that the new lives are conserved thanks to the creation of activists' groups, in the majority volunteers. But these groups don't have sufficient quantity at the moment, the cost of maintaining them active is high and their continuous presence results in negative impact for the protected habitat.

“Combining wireless sensor network technologies and low cost unmanned aerial vehicles for sea turtles monitoring in the Guanahacabibes Peninsula” project is focused in the design, prove and development of an automatic monitoring system to help to safeguard and to observe the nest of populations marine turtles. All this taking advantage of the results of recent technologies in the field of the Unmanned Aircraft Vehicle (UAV), Computer Vision (CV) and the relatively low cost of their integral implementation in a complete object recognition system in digital images. The results obtained by that system provides necessary information to develop the area's specialists' works, gaining in time, exactitude, and reducing the environmental impact and the maintenance costs. This grade work constitutes the first stage of development of the required final system, contributing this way with the selection, implementation and validation of algorithms able to achieve in their group, the mentioned tasks with efficiency and effectiveness.

Key Words: Unmanned Aircraft Vehicle, Computer Vision, object detection algorithms.

Índice

Introducción.....	1
Capítulo 1: “Fundamentación Teórica”	5
1.1 Caracterización de las imágenes digitales.....	5
1.1.1 ¿A que llamamos imagen?.....	5
1.1.2 ¿A que llamamos imagen digital?	6
1.1.3 Muestreo y Cuantificación.....	6
1.1.4 ¿Cómo representar la imagen digital?	7
1.1.5 Formatos de imágenes digitales.....	8
1.2 Caracterización del procesamiento digital de imágenes (PDI).....	9
1.2.1 Definición del procesamiento digital de imágenes.....	10
1.2.2 Etapas del PDI	10
1.2.3 Técnicas de realce y mejora de imagen	12
1.2.4 Campo de acción y aplicaciones.....	14
1.3 Visión por computadora y detección de objetos.....	16
1.3.1 Algoritmos para la detección de objetos.....	17
1.3.2 Selección del algoritmo.	23
1.4 Herramientas utilizadas	23
1.4.1 Lenguaje de programación Java	23
1.4.2 Librería de visión por computador: OpenCV	24
1.4.3 Adaptación para java de la librería OpenCV: JavaCV	25
1.4.4 Plataforma de Hardware Libre: Arduino.	25
1.5 Conclusiones parciales	25
Capítulo 2: Descripción del sistema.	26
2.1 Descripción de algoritmos y técnicas que se proponen	26
2.1.1 El algoritmo Viola y Jones	26
2.1.2 Boosting.....	30
2.1.3 Cascada.....	32
2.1.4 Principales ventajas y desventajas de Viola & Jones	34

2.2	Cálculo de la complejidad temporal y espacial de los algoritmos	34
2.3	Propuesta del sistema	35
2.4	Especificación de los requisitos de software	35
2.4.1	Requisitos Funcionales	36
2.4.2	Requisitos no funcionales	36
2.5	Diagrama de clases del sistema.....	37
2.6	Captura de las imágenes.....	38
2.7	Conclusiones parciales	38
Capítulo 3: Experimentación y prueba.		39
3.1	Experimentación.	39
3.1.1	Confección de la base de imágenes.	39
3.1.2	Entrenamiento.....	42
3.1.3	Resultados del algoritmo.	44
3.2	Interfaz del Software.	45
3.3	Ambiente de prueba.....	47
3.4	Conclusiones parciales.	48
Conclusiones.....		49
Trabajo Futuro:		50
Bibliografía:		51

Introducción

Los seres humanos y la mayoría de las especies del mundo animal somos capaces de reconocer diferentes objetos en imágenes con poco esfuerzo, no importa cuánto estos puedan variar en cuanto a tamaño o escala e incluso ángulos de vista. No obstante, esta tarea no es poco desafío para los sistemas de visión computarizadas. El mundo de la visión artificial o visión por computador es un derivado de la inteligencia artificial cuyo propósito es entender una escena o las características de una imagen. El reconocimiento de objetos es la tarea para encontrar e identificar objetos en una imagen o secuencia de video. Todo esto encuentra un millar de aplicaciones en la actualidad en campos como la previsión meteorológica, medicina, sistemas de seguridad, búsqueda y detección de patrones en general, entre otros.

En mayoría, para las capturas aéreas, los Vehículos Aéreos no Tripulados (UAV por sus siglas del inglés) son la mejor opción no pilotada a la hora de las asignaciones donde por problemas de difícil acceso, suciedad ambiental o alta peligrosidad para las vidas humanas no es seguro o eficiente enviar unidades humanas. Los UAV han sido usados militarmente tanto como plataformas para armas, entrenamiento y reconocimiento de objetivos de interés. Más recientemente se ha intensificado su uso comercial, aficionado y casero con nuevos y variados modelos siendo disímiles sus aplicaciones y teniendo como límite la imaginación del usuario. Aunque cabe destacar que al presente emergen nuevas leyes en algunos países, liderados por Estados Unidos, para intentar regular su uso pues sus bondades no son usadas para el bien únicamente.

Los UAV poseen varios métodos de control de vuelo automatizado conocido como “piloto automático” que ayudan a su estabilización para obtener mejor calidad en las imágenes que capturan. Los métodos de control han variado durante los años de acuerdo con los avances tecnológicos que van sucediendo. A la fecha, pueden ser equipados en su mayoría por una Unidad de Medición Inercial (IMU) que consta principalmente de equipos electro-mecánicos de acelerómetro y giroscopio de tres dimensiones o ejes cada uno para la determinación de la altitud y el Sistema de Posición Global (GPS) para su ubicación y navegación.

Combinando sensores adicionales como el de presión para determinar altitud y velocidad del aire o el de ultrasonido para determinar distancias a objetivos o terreno se obtiene una lectura del entorno bastante precisa como para poder llevar a buen término sus metas.

Las cámaras ópticas, que pueden ir desde las infrarrojas (IR), térmicas o especializadas para visión nocturna hasta potentes lentes de alta resolución y calidad con rápido enfoque y disparo más un competente nivel de compresión para transmisión en tiempo real. Las cámaras pueden poseer su propio IMU para mantener su orientación independiente al resto del UAV.

La mayoría de los UAV cuentan con controles inalámbricos para operadores humanos en tierra y así poder tener acceso a un control manual del vuelo. Usualmente se hace uso de una conexión continua para proveer de información en tiempo real, siendo de las más conocidas la transmisión de imágenes y videos fácilmente procesados por equipos con mejor capacidad de cómputo en tierra.

Todo esto logra que los sistemas UAV sean la opción a considerar para dar solución a problemas locales como es el caso que desde los últimos días de mayo hasta los primeros de septiembre cientos de tortugas marinas arriban a aislados nidos en las playas del archipiélago cubano cada año. Uno de los mayores reservorios de estos nidos está ubicado en la Península de Guanahacabibes, el punto más occidental de Cuba. Voluntarios locales ayudan a monitorear la población cada año. Pero esta área es una de las que menos densidad de habitantes humanos presenta en el país, así que se reclutan también voluntarios extra que son en su mayoría estudiantes universitarios que pasan parte de sus días libres ayudando a preservar la preciada especie de tortuga marina de su extinción, (ver anexo 1).

Durante la noche, voluntarios supervisan los nidos cerca del agua y la línea costera para detectar el arribo de hembras adultas. Una vez que se instalan en sus nidos, lo cual puede tomar muchas horas hasta que se sientan cómodas, son clasificadas y sus características morfo métricas son anotadas junto con la hora de su arribo. Cuando las tortugas dejan la playa los nidos son marcados. Durante el día, los especialistas realizan estudios en los nidos, recolectan y documentan datos relevantes. Los huevos empiezan a romperse a los 55 días de haber sido depositados. Esta actividad de monitorización es el componente fundamental en el estudio de las tortugas marinas. Además, es una de las más efectivas vías para preservar la anidación y población de nuevas tortugas de los animales salvajes y humanos irresponsables.

Las universidades locales, especialistas medioambientales e instituciones gubernamentales están trabajando conjuntamente en este programa desde 1998. Sin embargo, los costos logísticos de mantenimiento de grandes grupos de voluntarios durante meses cada año combinado con los efectos colaterales de la presencia humana en estas áreas de reserva natural

ha inspirado la iniciativa de automatizar parte de este proceso de monitorización. Recientemente, algunos estudios de investigación arrojan resultados que revelan el amplio uso de WSNs para el monitoreo en ambientes marinas (Xu, Shen, & Wang, 2014). También, el uso de UAV para inspeccionar la fauna marina y específicamente las tortugas marinas tienen lograda la atención de comunidades de investigadores (Hodgson, Kelly, & Peel, 2013) (Hardin & Jensen, 2011) (Jones IV, Pearlstine, & Percival, 2006) (Watts, 2010). Tomando inspiración de esto, el proyecto *“Combining wireless sensor network technologies and low cost unmanned aerial vehicles for sea turtles monitoring in the Guanahacabibes Peninsula”* se ha propuesto proveer de tecnologías WSNs integradas con UAV modificados para monitorizar las tortugas marinas en dicha península. La meta del final del proyecto es desarrollar un sistema automático de reconocimiento, de bajo costo y poco impacto ambiental negativo que apoye las tareas de monitoreo. Este propósito es el resultado de sesiones de tormentas de ideas entre equipos de universidades locales con especialistas del Programa de Conservación de Tortugas Marinas en la Península de Guanahacabibes y el Ministerio de Ciencia, Tecnología y Medio Ambiente de Cuba (CITMA).

Entendemos como problema el que no se cuente con los medios necesarios para mantener un adecuado seguimiento de las tortugas en su ambiente natural. Sostenemos como hipótesis que: Los algoritmos de reconocimiento de objetos en imágenes digitales deben ser capaces de detectar tortugas desde perspectivas aéreas facilitando la construcción de un sistema automático de monitorización.

Todo lo anterior conduce a la selección de dichos algoritmos de detección. La validación de los algoritmos por simulación viene dada por razones de cuidado a los medios empleados (UAV) ya que probar directamente algoritmos en fase de desarrollo en estos dispositivos conlleva a riesgos y costos innecesarios de campo y logística, incluido el tiempo, que son totalmente evitables desde los laboratorios mediante el uso de simuladores de entornos reales en ambientes controlados. Además, un software dedicado ayudaría en la práctica a la prevención de errores lógicos que son más difíciles de percibir en campo abierto donde muchas otras variables influyen. De allí, surge el objetivo específico de este proyecto de tesis: seleccionar un algoritmo de detección que satisfaga las necesidades del proyecto.

Las tareas de investigación han sido resumidas en:

- Estudio del estado del arte de los algoritmos de detección de objetos.
- Selección de un algoritmo por medio de experimentación.

- Generación de una base de imágenes, que comprenda cambios de iluminación, entono, etc.
- Estudio del estado del arte en métodos de simulación asociados al problema de investigación.
- Desarrollo de una herramienta de simulación que integre el algoritmo de detección seleccionado.

Para concebir el resultado deseado se hace uso de herramientas tales como OpenCV (desarrollado originalmente por Intel desde 1999) (OpenCV, 2016) y su derivada JavaCV, las cuales son bibliotecas públicas de visión artificial con licencia de software libre BSD (Berkeley Software Distribution) que permiten que sean usadas indistintamente para propósitos comerciales y de investigaciones libres o no en el campo del tratamiento de imágenes. Las bibliotecas pueden correr sobre Linux, MacOS X y Windows.

El presente trabajo de tesis está estructurado en tres capítulos. En el primero de ellos, dedicado a la fundamentación teórica, se introducirá acerca del mundo de las imágenes, especialmente de las imágenes digitales, sus principales características, así como su procesamiento, etapas que lo componen y aplicaciones en nuestro mundo contemporáneo. También será imprescindible el acercamiento a la visión por computador y cómo esta potente herramienta ayuda en tareas cada vez más demandantes de la detección de objetos. De manera que se analizarán tres de los principales algoritmos que mejores resultados están logrando en este campo para poder elegir cuál de ellos se adapta mejor a las exigencias de la presente investigación. Culminará el capítulo haciendo mención breve de las herramientas principales de las que se ha servido este proyecto para su desarrollo.

El capítulo segundo aborda en la descripción del sistema que se propone, desde los detalles de los algoritmos y técnicas hasta el cálculo de la complejidad que poseen dichos algoritmos en cuanto a los recursos de cómputo que requieren, haciendo notar que en las aplicaciones que se despliegan en tiempo real, es vital un buen desempeño.

Una vez en el tercer capítulo se ahonda en la descripción del proceso de confección de la base de imágenes, la cual resulta indispensable para el resultado final a alcanzar. Se detallan los pasos para realizar el entrenamiento y obtención del clasificador con dichas imágenes, así como los resultados y efectividad obtenidos con este. También se ilustra la interfaz del software propuesto y se explica un sencillo ejercicio de demostración que ayuda a probar la utilidad del proyecto.

Capítulo 1: “Fundamentación Teórica”

En este primer capítulo resulta necesario abordar el tema de la formación de la imagen desde sus elementos teóricos medulares para lograr así una comprensión más íntegra del problema que nos ocupa y cómo llegar eventualmente a la propuesta de su solución efectiva. Se tratará el procesamiento digital de dichas imágenes ahondando en sus distintas etapas técnicas, para lograr una mayor calidad y las diversas aplicaciones útiles que de su estudio se desprenden.

Llegados a este punto no será difícil centrarnos en el campo de la Visión por Computador (CV por sus siglas del inglés) en el mundo actual y dedicarnos a descubrir y evaluar algunos de los distintos algoritmos de detección de objetos pensados para dar solución a problemas como el que nos ocupa. Por último, resaltar discretamente a las principales herramientas de desarrollo de las que nos hemos servido.

1.1 Caracterización de las imágenes digitales

Un buen ejercicio para entender es imaginarse una rejilla rectangular donde a cada intersección corresponderá un punto de color. Todos estos colores (distintos o no) vistos en conjunto conforman la imagen. Mientras mayor sea el universo de colores que puede ser adoptado por un solo punto de color y mayor sea la cantidad de estos puntos en un mismo espacio limitado, mayor calidad de imagen se aprecia.

1.1.1 ¿A que llamamos imagen?

De manera matemática y según (Mejía, 2005) en sus apuntes de procesamiento digital de imágenes el cual se apoya en el clásico de (Gonzalez & Woods, 2002), se entiende a la imagen como a una función binomial $f(x, y)$, asumimos al par (x, y) como coordenadas espaciales en un plano (se están describiendo imágenes de dos dimensiones) y la función f con cualquier par de valores de coordenadas nos brindaría entonces el nivel específico de intensidad de la luz o nivel de gris para imágenes monocromáticas, en dicha coordenada. (Aristizábal R. & Ramírez M., 2006) plantean que una imagen también puede ser conceptualizada como señal binomial.

Las características más importantes de las imágenes, que son las que mayormente la definen, son tratadas en (Domenech M. 2009):

- **Profundidad en el color:** Se viene a referir a los distintos niveles de valor de en los colores (o solamente gris para las imágenes en blanco y negro)

empleados para el dibujo de la imagen. Bastante manejada es la profundidad de 8 bit pues esta equivale a contar con 256 niveles distintos de color, lo cual brinda una aceptable calidad.

- **Resolución:** Cuando se refiere a la cantidad de unidades de distancia que esta imagen en cuestión tiene, de manera que mejor definida estará cuando mayor sea el número de puntos por centímetro. La resolución se suele definir por su altura y anchura.

1.1.2 ¿A que llamamos imagen digital?

Cuando se adentra en definir a la imagen en su manera digital (Domenech M., 2009) la propone como la misma función $f(x, y)$ pero que esta vez se encuentra discretizada en las dos coordenadas espaciales (x, y) y también en su nivel de color. Como norma, la ausencia de color o negro equivale al nivel de brillo mínimo, el máximo constituido por todas las longitudes de onda del espectro visible se le atribuye al blanco. El resto son valores intermedios que varían sus niveles de gris entre los extremos del blanco y el negro.

En (Aristizábal R. & Ramírez M., 2006) defienden que, si el dominio de las coordenadas y su rango correspondiente en la función f son continuos, entonces la imagen en cuestión es por consecuente continua o análoga; si en cambio estos mismos dominios y rango son discretos pues se está, de hecho, en presencia de una imagen digital.

(González & Woods, 2002) afirman que una matriz que usa sus filas y columnas para simular los distintos elementos (llamados también *pels* o *pixels*) puede ser utilizada para representar a una imagen digital.

Uniendo y resumiendo los anteriores planteamientos se puede concluir que la imagen digital responde a una función $f(x, y)$ discretizada en todas sus coordenadas espaciales y que puede ser representada sencillamente como una conveniente matriz.

1.1.3 Muestreo y Cuantificación

Para lograr una imagen digital es mandatorio convertir una señal continua a digital (Rodríguez, 2004). Dicho proceso involucra otros subprocesos para cuantificación y muestreo. Como ya se ha leído, una señal analógica es también continua con respecto

de sus coordenadas y de su amplitud. Para digitalizarla se requiere realizar la apropiada discretización tanto en las coordenadas como en la amplitud. Dicha digitalización cuando se refiere a las coordenadas se conoce como muestreo de la imagen, y cuantificación de los niveles de color cuando se trata de la amplitud.

Estos procesos son vitales para hacer persistentes a la función $f(x, y)$ en la memoria del ordenador. A la práctica, una imagen se puede entender como el grupo de celdas o píxeles organizados correspondientemente en el arreglo matricial y bidimensional $M \times N$.

La transformación sufrida por estas dos dimensiones de la señal analógica o imagen continua es el muestreo, generando a su vez la noción de píxel. Por otro lado, la cuantificación es la conversión que experimenta la amplitud de los niveles de color de la señal analógica, la cual es homóloga al valor tomado por los píxeles (i, j) . Si se cuenta con 256 niveles de color que toman valores desde 0 hasta 255, dicho 0 representará la mínima intensidad o negro y el 255 el blanco o máximo nivel de intensidad. M y N son ambas potencias de 2.

(Aristizábal R. & Ramírez M., 2006) enseña que las causas de pérdida de información cuando se ejecuta la captura a una imagen digital son: el propio píxel, discreto por definición y el espectro limitado de valores que pueden tomar la intensidad luminosa en cada píxel. De hecho, actualmente estamos más relacionados y nos resulta familiar e incluso cotidiano el término de resolución de una imagen, que en parte está asociado a la cantidad de píxeles (Ej. 800×600 ; 1600×900 ; 8661×5774) y en otra a los niveles de color que puedan tomar (Ej. 8, 16, 32, 64 bits) de profundidad.

1.1.4 ¿Cómo representar la imagen digital?

Los números obtenidos en una matriz, producto del muestreo y la cuantificación pueden ser tanto enteros como reales. Dicho esto, la imagen digital se representa sencillamente gracias a una matriz convencional $N \times M$:

$$f(i, j) = \begin{bmatrix} f(0,0) & f(0,1) & f(0,2) & \dots & f(0,M) \\ f(1,0) & f(1,1) & f(1,2) & \dots & f(1,M) \\ \cdot & \cdot & \cdot & & \\ \cdot & \cdot & \cdot & & \\ f(N,0) & f(N,1) & f(N,2) & \dots & f(N,M) \end{bmatrix}$$

Así (Rodríguez M., 2004) plantea que a cada coeficiente de dicha matriz se le conoce como elemento de la imagen o píxeles. Obviamente los valores que pueden tomar las variables M y N deben estar estrictamente dentro del rango positivo.

1.1.5 Formatos de imágenes digitales

No estaría de más mencionar brevemente los variados formatos digitales que existen en la actualidad para las imágenes y su uso diario y creciente. Para la realización de esta tesis varios de ellos son usados en distintas etapas y funcionalidades del software final. Por ejemplo, en el entrenamiento del algoritmo de detección de objetos que discutiremos adelante (capítulo 2, epígrafe 2.3), así como en el posterior análisis de imágenes independientes.

Algunos de ellos son:

❖ **JPEG: Joint Photographic Experts Group.**

Método de compresión más adecuado para fotografías e imágenes de tonos continuos similares que contiene muchos colores. Este formato analiza las imágenes y elimina la información que no es apreciable.

❖ **PNG: Portable Network Graphics.**

Es un formato de almacenamiento sin pérdida y comprime la imagen de manera totalmente reversible siendo la imagen que se recupera idéntica a la original.

❖ **GIF: Graphics Interchange Format.**

Es un formato sin pérdida para imágenes de hasta 256 colores, si la imagen en cuestión excede los 256 colores se adapta reduciendo sus colores lo que resulta en una pérdida de calidad.

❖ **BMP: Windows bitmap (mapa de bits).**

Es un formato de imagen de mapa de bits sin compresión propiedad de Microsoft. Se debe resaltar que por su fácil implementación es muy demandada su utilización para los distintos sistemas de visión por computadora. (Rodríguez M. 2004).

1.2 Caracterización del procesamiento digital de imágenes (PDI)

Pareciera que nos encontramos frente a un término muy actual, muy de estos tiempos. Por eso sorprende el que justo después de la desastrosa Primera Guerra Mundial (1920) se empleaban códigos telegráficos para su transmisión a través del cable trasatlántico. En aquella temprana ocasión solo eran posibles 5 niveles de gris para componer la imagen. No fue entonces hasta mediados del siglo pasado, y en plena Guerra Fría que la fotografía aérea y espacial ligada por supuesto a operaciones militares tuvo una gran relevancia para el desarrollo del PDI según relata (Gonzales & Woods, 2002).

El avance apresurado del hardware junto a la enorme importancia que posee el procesamiento de las imágenes en el mundo actual han propiciado el imparable desarrollo y mejora de esta técnica. Solo bastaría mencionar el papel que juegan las imágenes digitales en las redes sociales, foros virtuales, entre otros, en todo el mundo que rodea la internet. Piense usted que esto hace posible el que instantáneamente luego de cargar una nueva fotografía en Facebook, este tenga la capacidad de reconocer su rostro en ella, o el de su amigo, y etiquetarlos.

En el desarrollo de estas técnicas intervienen de manera permanente una considerable cantidad de científicos e instituciones y no solo se reserva la materia de estudio a cursos de post-gradados, sino que ya suman bastantes las carreras pre-grado en todo el mundo que dedican espacio para el aprendizaje y desarrollo de este tema (Rodríguez M., 2004).

En (Gonzalez & Woods, 2002) se define que el PDI, tanto en su entrada como salida, se trata de imágenes digitales. En el análisis, la entrada del sistema es la propia imagen digital y la salida no es más que una descripción simbólica de esta. El objetivo del proceso en general es el reconocer y detallar el contenido de la imagen digital.

1.2.1 Definición del procesamiento digital de imágenes

Gracias a (Rodríguez M., 2004) se encuentra el PDI enunciado como un grupo de procedimientos que se suceden en una escena para su posterior persistencia, transmisión, reconocimiento, análisis e interpretación; sencillamente lo redefine como las distintas técnicas aplicadas a imágenes digitales con los objetivos de realzar su calidad o simplificar la búsqueda de información contenidas en ellas. Seguidamente precisa de manera general a la visión por computadora como una de las disciplinas que trabaja con las teorías, técnicas, y utilidades que tiene relación con el análisis de datos sensoriales en cierto contexto, con la producción de descripciones simbólicas de dichos datos y de proporcionar conclusiones o diagnósticos que tienden a ser de utilidad según estimaciones establecidas anteriormente.

He aquí una definición lo suficientemente abarcadora:

“El Procesamiento Digital de Imágenes es la disciplina científica que se ocupa de realizar transformaciones a la información visual, con el objetivo de facilitar su análisis, almacenamiento, transmisión o empleo en determinadas aplicaciones, empleando medios de computación.” (Gonzalez & Woods, 2002).

1.2.2 Etapas del PDI

La visión por computador (CV por sus siglas del inglés) que definiremos más adelante contiene varias etapas de acuerdo a (Rodríguez M., 2004) donde se encuentran minuciosamente documentadas.

1- Obtención de la imagen digital

Esto se puede llevar a cabo mediante varios periféricos o dispositivos, entre los más usados, una cámara de TV o digital, los propios teléfonos móviles capacitados con una lente, un escáner; u otros menos comunes como conjuntos de LEDs usados para percibir en vez de emitir luz, ultrasonidos, etc. La elección de uno de estos medios está relacionada a la escena y objetivo que se desee. En la mayoría de los casos esta etapa es crucial en el sistema completo de CV dado que las siguientes etapas verán su calidad comprometida y dependiente al éxito de este primer paso.

2- Pre-procesamiento

Esta etapa no siempre es necesaria de efectuar, pero en no pocas aplicaciones prácticas tiene merecida importancia, medularmente se encarga en la mejora de la imagen mediante la eliminación de ruido, mejora del contraste y filtrados distintos.

3- Segmentación

En esta etapa de forma sencilla se describe como la unión de una escena con un grupo de regiones tanto no solapadas como homogéneas refiriéndose a algún criterio que se quiera, dicha unión cubre la totalidad de la imagen. Esta etapa es la principal en todos los sistemas de CV desde las dificultades que conlleva hasta la relevancia de sus conclusiones. El resultado del proceso de segmentación es ahora una imagen etiquetada o simbólica en la que se ha transitado de la imagen digital en bruto a una bastante más simplificada en la que se encuentran bien identificados individualmente los distintos elementos existentes.

4- Descripción y representación

Específicamente se habla de patrones u objetos anteriormente segmentados y es generalmente uno de los primeros pasos que se suceden en gran cantidad de sistemas de análisis automático de imágenes o de CV. Se hayan variadas técnicas de descripción y representación, la meta es conseguir modos de representación que obtengan las diferencias nucleares entre objetos, o entre las distintas clases de objetos, a la misma vez que se conservan lo menos dependientes posibles ante los cambios como la escala, la orientación y traslación.

5- Reconocimiento, análisis e interpretación

Por reconocimiento se entiende al proceder que otorga una determinada marca o etiqueta a un objeto en particular a partir de la información obtenida de sus descriptores.

Si reconocemos cualquier ejemplo de objeto, como un auto podemos etiquetarlo con alguna sintaxis propia del algoritmo, la cual será la identidad irrepetible de dicho objeto. El proceso de interpretar define el significado a un grupo de objetos a los que previamente se han reconocido y que están salvados esos patrones en la base de conocimiento o de datos. La interpretación estudia y determina las relaciones que existen entre todos los objetos para poder llegar a una descripción semántica que abarca toda la imagen, o una parte de ella. Compresión o análisis de la imagen son también términos utilizados para referirse a este proceso. No se debe dejar de tomar en cuenta el hecho de que en la interpretación a la cual estamos haciendo referencia, la representación del conocimiento debe ser lo menos dependiente posible a la aplicación.

El sistema descrito por esta tesis hace uso opcional de la etapa de obtención de la imagen mediante cámaras digitales o puede valerse también de bancos de imágenes previamente capturadas. Cualquiera de estas variantes da lugar al pre-procesamiento, también opcional, al contar con un módulo de filtrados distintos según lo necesitado y dirigidos a eliminar sobretodo el ruido provocado por las distintas intensidades de brillo que tanto afectan a las siguientes etapas de reconocimiento, análisis e interpretación.

1.2.3 Técnicas de realce y mejora de imagen

La meta de toda técnica de realce en las imágenes es incrementar su calidad para quien la observa según (González A.). Estamos ante un proceso subjetivo, que se realiza habitualmente de maneras interactivas, dinámicas. La elección apropiada de los métodos y parámetros correctos son dependientes de la calidad de materia prima, o sea, de la imagen original y de la aplicación en cuestión. (Gonzalez y Woods, 2002) ilustra con escogidos ejemplos de realce para las imágenes, se trata de realzar el contraste, superar la definición, el procesamiento de filtrado en el

dominio espacial y en el de la frecuencia, existen otros tipos de filtros para la atenuación de ruido. Grande es la cantidad de transformaciones que se pueden aplicar a las imágenes no solo con el objetivo de realzarlas y mejorarlas sino modificarlas hasta quedar irreconocibles de su original. Las pautas para clasificar estas transformaciones son varias, comúnmente subdivididas en cuatro categorías:

➤ ***Transformaciones puntuales:***

Obtenidas de una puntual operación dependiente únicamente del nivel de color de entrada en cierto píxel. Estas operaciones contemplan normalmente el manejo de cada píxel, siendo casos ilustrativos la binarización, umbralización, etc.

➤ ***Transformaciones locales:***

En estos casos los valores en la entrada de varios píxeles adyacentes influyen el resultado del nuevo píxel de salida. Casi todas las operaciones tienen un carácter local, ejemplo de ello lo son el suavizado y el realce de bordes.

➤ ***Transformaciones globales:***

De una imagen en toda su extensión, sus datos de entrada son los que influyen en el resultado que se obtiene en la salida. Estas operaciones son realizadas normalmente en el dominio de la frecuencia como en los ejemplos de la compresión que toma el total de la imagen de entrada y obtiene una comprimida de salida.

➤ ***Transformaciones geométricas:***

Cuando se usan las transformaciones geométricas, el resultado que se obtiene es directamente dependiente de las posiciones distintas de los niveles de color en la imagen entrada. Casos como este son el de la traslación, rotación, rectificación y cambios de escala.

Además de la anteriormente descrita existe otra clasificación para mejorar la calidad de imagen y esta se basa en las aplicaciones que se pueden ejecutar.

➤ **Definición:**

Remarca propiedades de algunos elementos en las imágenes, las vuelve más nítidas.

➤ **Corrección de efectos**

Se esfuerza por lograr la sanación de defectos de la imagen, son los eliminadores de ruido y picos bastante altos de diferencias entre colores.

➤ **Suavizado:**

Contiene técnicas para conseguir una apariencia más suave de la imagen deshaciendo detalles muchas veces en forma de líneas definidas como fronteras en la imagen que separan distintos grupos de niveles de colores, creando una mayor continuidad o saltos más pequeños y progresivos entre ellos.

1.2.4 Campo de acción y aplicaciones

Meditar sobre las aplicaciones diversas en las que el PDI puede ser útil conduce a abordar secuencialmente cada una de las necesidades a las que el hombre se va enfrentando día a día y en diversos campos. Esas aplicaciones se han ido etiquetando con el paso del tiempo que ha tenido su estudio.

➤ **Mejoramiento:**

Mayormente logrado con el aumento de los contrastes, la eliminación del ruido, y los filtrados espaciales y de frecuencia.

➤ **Restauración:**

Producida al estudiar el desenfoque, movimiento, y otros fenómenos comprendidos como degradaciones que se originan con mucha probabilidad en la etapa de captar la escena según (Gonzalez & Woods, 2002)

➤ **Reconstrucción:**

Empleada en campos tan diversos como la geología o la propia medicina para obtener de imágenes tridimensionales luego de procesar diferentes cortes provenientes lo mismo de rayos x, mapas sínicos etc.

➤ **Segmentación:**

Usado para desligar variados tipos de elementos, esta característica mucho tiene que ver con la detección de contornos.

➤ **Compresión de imágenes:**

Se encarga, tal como enuncia, de comprimir las imágenes para optimizar su almacenamiento o transmisión. Esta pauta es un tema mayormente discutido puesto que, en el mundo actual, donde la tendencia de los distintos formatos es mejorar la calidad de los objetos pero que como consecuencia acarrea muchas veces también el incremento del peso de estos.

➤ **Reconocimiento visual de patrones:**

Su reto es encontrar los métodos que sean aptos para reconocer distintas formaciones, estructuras, secuencias.

➤ **Análisis e interpretación:**

Es una operación de elevado nivel de complejidad que estudia todos los problemas relacionados con la interpretación de la imagen según una base de conocimiento o de datos previamente obtenida que puede incluso estar entrenándose continuamente con cada nuevo análisis.

En (Gonzalez & Woods, 2002) se citan algunas de las muchas aplicaciones del PDI en distintas bifurcaciones como lo son la medicina, biotecnología, bioinformática, telecomunicaciones, robótica y visión artificial, control de procesos tecnológicos, cartografía y geodesia, geofísica, astronomía, precepción remota, criminalística, aplicaciones militares etc. Se logran salvar vidas todos los días por

cirugías de mínimo acceso o detectando a tiempo cuerpos hostiles. Se evitan pérdidas humanas cuando en vez de un humano es un robot con esta tecnología el que entra en el peligro de una casa en llamas o similares

No se puede dejar de mencionar también el impacto negativo que puede experimentar el desarrollo del PDI puesto que basta simplemente comentar que algunas de las aplicaciones militares para esta tecnología permiten polémicamente a máquinas, vehículos aéreos o no, no tripulados por humanos decidir cuándo pueden apagar vidas humanas.

1.3 Visión por computadora y detección de objetos.

La Visión por Computadora es un campo de la informática que enseña a las computadoras a ver. Es la manera en que las computadoras pueden interpretar información visual. Usualmente la imagen es procesada primeramente en un nivel bajo para mejorar su calidad, removiendo ruido, por ejemplo. Luego, a más alto nivel, la imagen es vuelta a procesar, pero esta vez para detectar patrones, formas e intentar determinar de qué se trata la imagen.

(Mark_S._Nixon) hace también una homología entre el sistema de visión humano (donde el cerebro procesa las imágenes provenientes de los ojos) y el sistema de visión por computador (CV) (que procesa las imágenes adquiridas desde una cámara electrónica).

Si bien es cierto que no podemos esperar que la CV reproduzca exactamente la función del ojo humano, si hay aspectos en que se aprecia ventaja. Por ejemplo, la visión humana puede distinguir distancias relativas entre los objetos bastante bien, pero es pobre cuando de distancias absolutas se trata, lo contrario a CV. La CV es buena estimando diferencias absolutas, pero con una relativamente pobre resolución para diferencias relativas. La gran ventaja de nuestro cerebro es que puede encontrar más fácilmente relaciones entre conceptos de objetos que no son exactamente iguales, sin embargo, esto es toda una pesadilla para la CV puesto que abstraerse es mucho más complicado para la lógica.

Es deber aclarar que las técnicas de CV son usadas para analizar cualquier tipo de datos, no solo las imágenes provenientes de las cámaras.

Rico es el campo de estudio de la CV tanto para ingenieros electrónicos, informáticos y muchos otros. Actualmente existen muchos sistemas de CV que se usan rutinariamente

en la industria: cámaras que inspeccionan partes mecánicas para comprobar su tamaño, la comida es inspeccionada para ver su calidad, y hasta las imágenes usadas en la astronomía son beneficiadas por las técnicas de CV. Estudios forenses y biométricos usan CV incluyendo reconocimiento automático de rostros etc. Estos estudios están secundados por biólogos y psicólogos que continúan el estudio de cómo trabaja el sistema de visión humano y como a través de él podemos ver y reconocer distintos objetos.

De hecho, otro término comúnmente referido es la detección de objetos como la función responsable de descubrir e identificar la existencia de objetos o clases de estos. Puede ser considerado como método del procesamiento de imágenes para identificar objetos en las imágenes digitales.

Una manera de hacer esto es simplemente clasificar los objetos por su color. Pero claro, esta aproximación de código de color no es 100% confiable. Los experimentos muestran que las condiciones lumínicas son extremadamente determinantes. El brillo es un factor rival a vencer en este campo.

1.3.1 Algoritmos para la detección de objetos.

Los sistemas de reconocimiento de objetos buscan objetos del mundo real partiendo de imágenes de dicho mundo, usando modelos de objetos que son conocidos a priori. La implementación de algoritmos para esta tarea en las máquinas es sorprendentemente difícil. El problema del reconocimiento puede ser definido como el problema de etiquetado basado en modelo de objetos ya conocidos. Formalmente, dado una imagen que contiene uno o más objetos de interés (y su fondo o background) y un conjunto de etiquetas correspondientes al conjunto de modelos de objetos conocidos al sistema, se debe asignar correctamente las etiquetas a las regiones o conjunto de estas en la imagen.

A continuación, se repasan tres de los algoritmos más usados y documentados por sistemas de detección de objetos y sus características.

- **Template Matching**

Haciendo referencia al trabajo (Roth, Peter M. and Winter, Martin, 2008), una plantilla (template) es una imagen pequeña o sub-imagen.



Figura 1.1: Ejemplo de template que es objetivo usado para buscar ocurrencias en otras imágenes

El objetivo con este algoritmo es encontrar ocurrencias de esta plantilla en una imagen más grande. A grandes rasgos eso es todo, simplemente encontrar coincidencias de la plantilla en la imagen.

Por ejemplo:

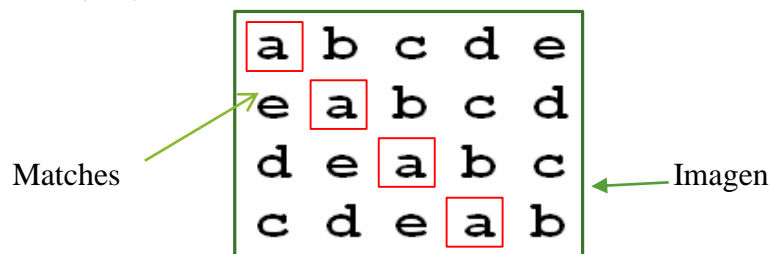


Figura 1.2: Representación del funcionamiento del algoritmo Template Matching

Un match está basado en la cercana proximidad entre los detalles de la plantilla y los de la imagen en cada una de sus coordenadas (i,j).

Limitaciones:

- Las plantillas no tienen en cuenta variaciones en su escala o rotación.
- Ligeros cambios de tamaño u orientación pueden causar problemas.

Para que sea eficaz se suelen usar varias plantillas para representar un mismo objeto, variando su tamaño y rotando la misma plantilla.

Pero por supuesto esto hace que el algoritmo sea una operación costosa computacionalmente. Especialmente si se busca en la imagen entera o se usan varias plantillas.

- **Background Subtraction**

Para varias aplicaciones CV, Background Subtraction (BS) es una “rápida y sucia” manera de localizar objetos en movimiento en un video capturado por una cámara estática según (Pierre-Marc Jodoin, 2012) en “Comparative Study of Background Subtraction Algorithms”.

Este algoritmo es usualmente requerido para ser tan rápido y simple como sea posible. Consecuentemente, la mayoría de los métodos de etiquetado BS “en movimiento” se fija en las diferencias entre cada pixel, en un mismo tiempo t , y el fondo de la imagen. Esta solución ha sido exitosamente probada siempre y cuando la cámara utilizada permanezca rigurosamente estática y su fondo esté libre de ruido (ej: olas del mar, copas de los arboles movidas por el viento, etc.). De hecho, muchos videos con baja calidad, o ambiente ruidoso son causa de numerosos falsos positivos. Los falsos positivos pueden ser incluso inducidos por cambios de iluminación o temblor de la cámara, solo por nombrar algunos.

Aunque con algunas diferencias, la mayoría de las técnicas BS comparten un criterio en común, ellas asumen que la frecuencia de video observada está compuesta por un fondo estático delante del cual, objetos en movimiento son observados. Asumiendo también que cada objeto en movimiento está compuesto por colores diferentes del observado en el fondo, de lo contrario, se estaría en presencia del efecto camuflaje.

La siguiente (figura 1.3) ilustra acerca del resultado obtenido aplicando el algoritmo BS. Gracias al cual se puede detectar fácil y rápidamente los autos que transitan por una carretera, pudiendo a su vez contabilizarlos, medir su velocidad, el nivel de congestionamiento etc.



Figura 1.3: Ejemplo de aplicar el método BS para detectar los autos que transitan por una carretera.

- **Haar Cascade**

Muchas aproximaciones basadas en machine-learning tienen ventaja al ser computacionalmente más eficientes en la detección de objetos. Aunque la detección genérica (por rasgos) a menudo realiza mejor los trabajos con localización, escala y formas.

Un método mucho más sofisticado es requerido, según expone (Sander Soo, 2015) en su trabajo "Object detection using Haar-cascade Classifier". Uno que pueda hacer la detección de objetos desde imágenes usando rasgos o estructuras específicas del objeto en cuestión. De todas formas, permanece un problema y es el de trabajar solamente con intensidades de las imágenes, lo que significa tener en cuenta los valores RGB de cada pixel de la imagen, haciéndolo muy costoso computacionalmente. El problema fue dirigido hacia los llamados rasgos Haar-like, (desarrollado por Viola and Jones).

Un rasgo Haar-like considera regiones rectangulares vecinas como locaciones específicas en una ventana, suma las intensidades por pixel en cada región y calcula la diferencia entre esas sumas. Dicha resta es entonces usada para categorizar sub-secciones de la imagen.

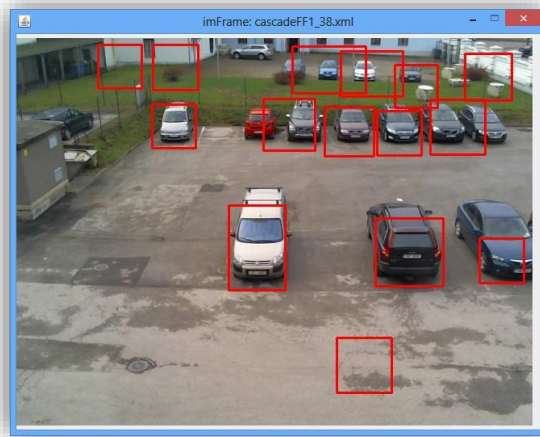


Figura 1.4: Ejemplo de reconocimiento realizado por el algoritmo Haar-Cascade.

El más conocido ejemplo puede ser el de detección de rostros humanos. Donde comúnmente, las áreas alrededor de los ojos (también llamadas ojeras) son más oscuras que las áreas de las mejillas. Un rasgo Haar-like sería la colocación de dos áreas rectangulares vecinas entre las regiones de las ojeras y las mejillas.

Haar-Classification es una técnica de árbol donde en la fase de entrenamiento, una cascada basada en descartes por estadística es creada (figura 1.5). Que sea por estadística significa que un clasificador potente es creado a partir de otros más débiles (en cascada). Y un clasificador débil, a su vez, es uno que toma la clasificación acertada en al menos el 50% de los casos. Este avance hacia mejores clasificadores desde otros muchos menos potentes es posible incrementando el peso (penalización) en los ejemplos de errores de clasificación, significando que en la siguiente ronda o iteración de entrenamiento las hipótesis más acertadas serán escogidas.

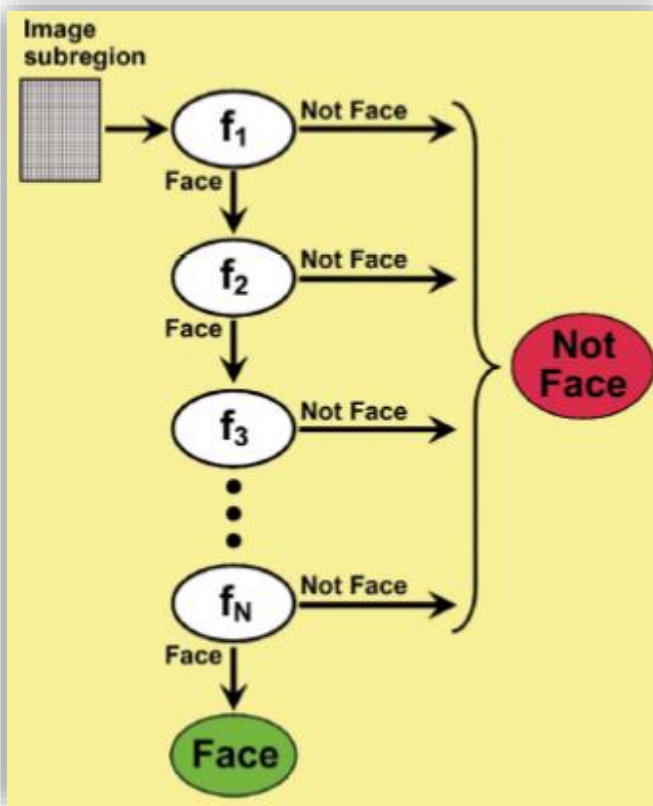


Figura 1.5: Ejemplificación del trabajo en cascada del algoritmo

Que Haar-Classification use una cascada de descarte significa que el clasificador final en una gran cascada de muchos clasificadores más simples o sencillos, embebidos, y una región de interés debe entonces pasar por todos los estados o pasos de esa cascada para poder ser reconocido como tal. El orden de esos nodos es a menudo colocado según su dificultad de procesamiento, entonces muchos rasgos candidatos son reducidos y eliminados tempranamente, mejorando sustancialmente el tiempo de computación.

Existen cuatro métodos que usan el Haar-classification y están disponibles en la herramienta OpenCV: Real Adaboost, Discrete Adaboost, Logitboost y Gentle Adaboost.

1.3.2 Selección del algoritmo.

Con lo aprendido hasta este punto acerca de tres de los algoritmos que más se están utilizando en la actualidad y con buenos resultados en el mundo del reconocimiento de objetos en CV, es lógico seleccionar el algoritmo Haar-Cascade. Basándonos sobre todo en las limitaciones que caracterizan a los otros dos, como lo son: el no poder descartar variaciones del entorno, (específicamente de un background en movimiento) que no tienen que ver con el objeto de interés o por lo contrario, no poder asociar al objeto en cuestión con instancias de el mismo en diferentes tamaños o rotaciones. Todo esto sumado a las ventajas, sobre todo en costo computacional que ofrece el Haar-Cascade y su fiabilidad gracias a su potencia de descarte según el entrenamiento, lo convierten en el candidato final por el cual apostar.

1.4 Herramientas utilizadas

En cuanto a las herramientas utilizadas se propone seguir una línea que contemple dichas herramientas bajo software libre. Ya que se trata de un proyecto de bases académicas y con propósito de seguir desarrollándose por distintos equipos o individuos colaboradores en el futuro que no deben estar sujetos a límites privativos de ningún tipo. Asimismo, las utilidades descritas debajo gozan de robustez, referenciadas, bien documentadas y ampliamente utilizadas por la comunidad internacional de desarrolladores.

1.4.1 Lenguaje de programación Java

Java, que vio la luz desde 1995, es un lenguaje de programación orientado a objetos y pensado para dar dentro del mundo de la característica es la intención de programa y poder ejecutarlo *once, run anywhere*). Lo que multiplataforma.

Actualmente y desde ya lenguajes de programación



cumplimiento a diversos problemas computación. Su más famosa codificar una sola vez un en cualquier dispositivo (*write lo* convierte en un lenguaje varios años es uno de los más usados y cada vez parece adquirir

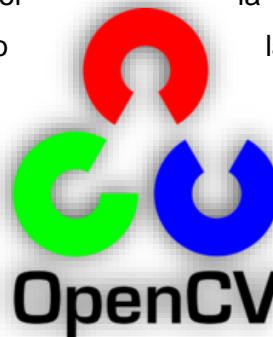
más relevancia. Su robustez y cantidad de librerías asociadas lo hace ser favorito entre desarrolladores de diversas índoles. Siendo tan grande su espectro de potencialidades que casi cualquier aplicación hecha por cualquier otro lenguaje puede ser copiado también por Java y obtener ventaja.

Una de sus más perceptibles influencias es en la industria de la telefonía móvil. Otros lenguajes como C#, J#, JavaScript, PHP, Python también se han visto fuertemente influenciados sobre todo por aspectos como la manera simple de orientarse a objetos, la política de librerías y la eliminación de funciones de bajo nivel, culpables de errores en muchos casos.

Los dos entornos de desarrollo más significativos en la comunidad de desarrolladores son NetBeans y Eclipse, aunque existen otros. En este trabajo se utilizó Eclipse en su versión Mars 4.5.

1.4.2 Librería de visión por computador: OpenCV

En (James A. Ross, 2008) se encuentra descrito a OpenCV como una librería multiplataforma orientada a objetos para funciones de procesamiento de imágenes y originalmente desarrollado por la empresa Intel. Desde su lanzamiento disímiles han sido las aplicaciones en las cuales ha probado ser útil. Se ha utilizado desde sistemas de seguridad, detección de movimiento o reconocimiento de objetos, entre ellos el reconocimiento facial. También encuentra utilidad práctica para la calibración de cámaras, la visión estereoa y la visión robótica.



En el trabajo de (Staffan Reinus, 2013) del departamento de información y tecnología de una de las universidades más prestigiosas de Europa: La UPPSALA UNIVERSITET de Suecia se encuentra descrito a OpenCV como una librería de visión por computador escrito en C y C++, conteniendo más de 500 funcionalidades asociadas con la visión. Esta librería es utilizada para usos académicos y comerciales desde que está bajo licencia BSD (Berkeley Software Distribution) y uno de los factores de su popularidad es que ha sido ampliamente usada y documentada (<http://sourceforge.net/projects/opencv/>).

1.4.3 Adaptación para java de la librería OpenCV: JavaCV

JavaCV es una librería Java que sirve como intermediaria para poder hacer uso del núcleo de las funcionalidades de OpenCV que están escritas en C. Pero esta librería no solo se limita a llamar dichos métodos desde un entorno Java, sino que es una potente herramienta gracias a que integra en sí otras librerías útiles para el tratamiento de imágenes como lo son: FFmpeg, GeometricCalibrator, ImageTransformer, ObjectFinder, CameraDevice y muchas otras que no encontramos por defecto en OpenCV. Cabe destacar, que contrario a la creencia de que existiendo una capa Java para llamar al núcleo de C, el proceso entero se ralentizaría, JavaCV logra optimizaciones y provee al proyecto de los beneficios de la programación al estilo de Java.

1.4.4 Plataforma de Hardware Libre: Arduino.

Según (Michael Margolis, 2011) Arduino es una familia de microcontroladores y entorno de creación de software que facilita la creación de programas (Sketch) que pueden interactuar con el mundo real. Gracias a los sensores es posible responder a toques, sonidos, posiciones, calor, luz etc. Este tipo de tecnología, referenciada como “*physical computing*”, es usada en todo tipo de proyectos, desde telefonía móvil hasta los sistemas electrónicos de automóviles. Arduino hace posible que incluso personas sin mucho conocimiento de electrónica o programación pueda enriquecerse con el uso y aplicación de esta tecnología.



1.5 Conclusiones parciales

Hasta ahora ha sido necesario definir los conceptos de imagen, así como los distintos procesos que son posibles de llevar a cabo para su transformación, mejoramiento etc. Todos estos conceptos encausados en la rama de CV y el reconocimiento de objetos ofrecen una base sólida teórica en la que poder proyectar trabajos que requieran ocupar dichas tecnologías, como es el caso que ocupa esta tesis donde va a ser necesario el pre-procesamiento de las imágenes, ya sean capturadas o no por el propio sistema, y posteriormente analizar para determinar si el objeto, para el cual ha sido entrenado el algoritmo seleccionado (Haar-Cascade), está contenido en la imagen.

Capítulo 2: Descripción del sistema.

El presente capítulo detalla los algoritmos y técnicas investigados, ahondando en sus principales conceptos, métodos, ventajas y desventajas. Todo esto apoyándose en ilustraciones y pseudocódigos que ayudan a una mejor abstracción y comprensión. Seguidamente se realiza un cálculo de su complejidad temporal, así como se brinda de manera general una propuesta para el sistema. Al final se describe el proceso de obtención de la base de imágenes, paso fundamental que sirve como material para el entrenamiento de dicho algoritmo.

2.1 Descripción de algoritmos y técnicas que se proponen

El objetivo de esta investigación es encontrar posibles soluciones para detectar al objeto de interés contando únicamente con las imágenes capturadas y gracias a una cámara digital. Usar Visión por Computador (CV por sus siglas del inglés) para detectar objetos en las imágenes es un tema ampliamente tratado en investigaciones de renombre como (Viola & Jones, 2001), (Dalal & Triggs, 2005), (Fabisch, 2010), (Reinhardt, 2011) entre otros. Pero aún muchos desafíos necesitan ser superados.

Son dos las maneras de clasificar objetos con CV según (Duncan ten Velthuis, 2014), haciendo uso del *reconocimiento de objetos* o de la *detección de objetos*, el primero pudiéndonos decir si un objeto está contenido en la imagen y el otro especificando donde específicamente se encuentra el objeto en la imagen.

Un algoritmo probado satisfactoriamente para la detección de rostros ha sido el algoritmo propuesto por (Viola & Jones, 2001). Dicho algoritmo proporciona respuesta a ambas tareas, el reconocimiento y la detección de una manera computacionalmente eficiente. Seguidamente será explicado el algoritmo en profundidad y cómo podemos aplicarlo a nuestro trabajo.

2.1.1 El algoritmo Viola y Jones

Uno de los más tempranos éxitos en la identificación de dónde en la imagen está localizado un objeto fue gracias a una variante del algoritmo de aprendizaje AdaBoost (ver pseudocódigo 2) empleado por (Viola & Jones, 2001) usando cascadas de rasgos simples.

El algoritmo Viola & Jones Machine Learning entrena al clasificador usando rasgos Haar-Like encontrados en los ejemplos del entrenamiento (en la base de imágenes

usadas para entrenar). Los rasgos simples usados por este algoritmo ven sus orígenes en (Papageorgiou, 1998). Para el reconocimiento de objetos en tiempo real, la extracción de rasgos necesita ser computada rápidamente y los rasgos Haar-Like son uno de los más simples y fáciles de computar según (Flynn, 2009)

El algoritmo entrena con una base de imágenes, el conjunto de imágenes positivas contiene al objeto de interés y el conjunto de las imágenes negativas sin el objeto es usado para discriminar entre los rasgos útiles y los que no lo son.

Las imágenes positivas son todas segmentadas a una misma medida de región pequeña haciendo posible el ir escalando en la búsqueda del objeto con una segmentación cada vez mayor recorriendo la imagen. Entrenando con regiones menores va a traducirse en contar con menos cantidad de rasgos, pero al mismo tiempo acelera la fase de entrenamiento del algoritmo.

Si se quiere como resultado una buena detección de objetos en tiempo real, el clasificador debe ser lo más rápido posible. Computacionalmente puede ser costoso cuando todos los rasgos son usados para clasificar en grandes imágenes. Considerando que en las imágenes de gran tamaño existen muchos píxeles que pueden ser redundantes, por consiguiente, es posible aproximar sus valores sin perder mucha información. Si las imágenes entonces son reducidas a imágenes más pequeñas, la detección podrá ocurrir con mayor velocidad sin pérdida significativa en la calidad en el proceso.

Como una representación intermedia de la imagen, Viola & Jones transforma la imagen en lo que ellos llaman “una imagen integral”. Los valores de los píxeles de la imagen integral son la suma de los píxeles a la izquierda y debajo de la imagen original. Por ejemplo, los nuevos valores son denotados por coordenadas (x,y) , donde $i(x,y)$ son los valores de la imagen original y $ii(x,y)$, la imagen integral.

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} i(x',y')$$

(Ver Figura 2.1)

Usando la formula recurrente, donde $s(x,y)$ denota suma acumulativa,
 $s(x, -1) = 0$ and $ii(-1, y) = 0$

$$s(x, y) = s(x, y - 1) + i(x, y)$$

$$ii(x, y) = ii(x - 1, y) + s(x, y)$$

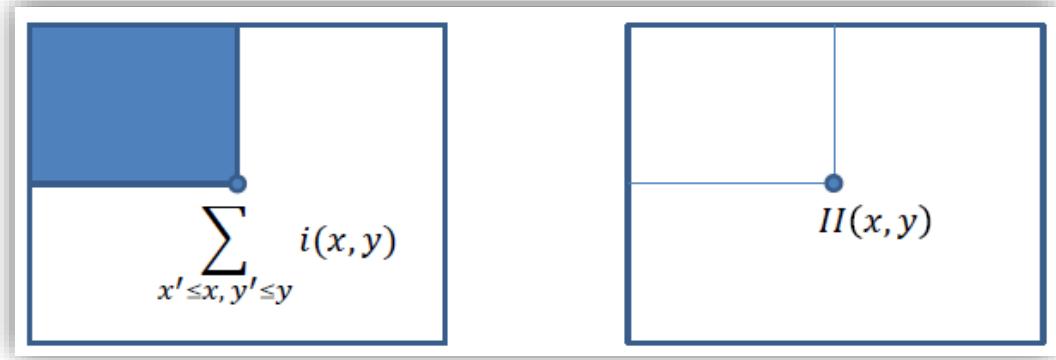


Figura 2.1: Izquierda: imagen original con gran cantidad de pixeles muchos de ellos redundantes y que afectan los recursos en el proceso. Derecha: representación intermedia de la imagen original llamada “imagen integral” calculada a partir de la imagen original y menos costosa de procesar

La imagen integral puede ser calculada partiendo de la imagen original. Los valores de la imagen integral $ii(x, y)$ contienen la suma de los pixeles debajo y a la izquierda de la imagen original $i(x, y)$, (figuras 2.1 y 2.2).

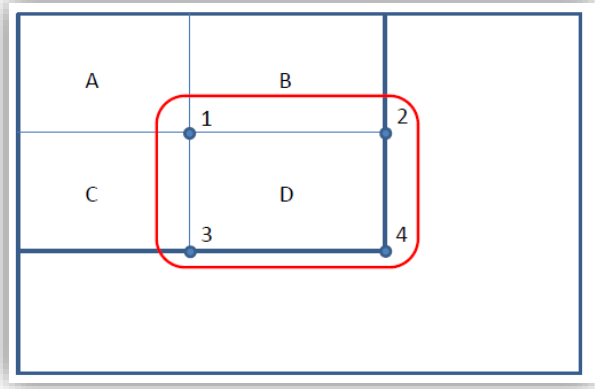


Figura 2.2: La suma de los pixeles en el espacio D pueden ser extraídos con los valores de las posiciones del 1 al 4.

$$(4+1) - (2+3) = D$$

Algorithm Integral Image

- 1: **Input:** an image I of size $N \times M$.
 - 2: **Output:** its integral image Π of the same size.
 - 3: Set $\Pi(1, 1) = I(1, 1)$.
 - 4: **for** $i = 1$ to N **do**
 - 5: **for** $j = 1$ to M **do**
 - 6: $\Pi(i, j) = I(i, j) + \Pi(i, j - 1) + \Pi(i - 1, j) - \Pi(i - 1, j - 1)$ and Π is defined to be zero whenever its argument (i, j) ventures out of I 's domain.
 - 7: **end for**
 - 8: **end for**
-

Pseudocódigo 1: Creación de la imagen integral (Duncan ten Velthuis, 2014)

Los rasgos Haar-Like son rasgos rectangulares basados en la sustracción entre la suma de los valores de los pixeles encontrados en la región blanca del rectángulo y la suma de los valores encontrados en la parte oscura del rectángulo (figura 2.3(a) y 2.3(b)). Esos rasgos son encontrados usando regiones rectangulares que son colocadas encima de toda la imagen integral. La cual es una de las razones del por qué el algoritmo es tan computacionalmente eficiente, dado que la computación de los rasgos Haar-Like solo necesita cuatro operaciones por rasgo, una vez que se ha calculado la imagen integral.

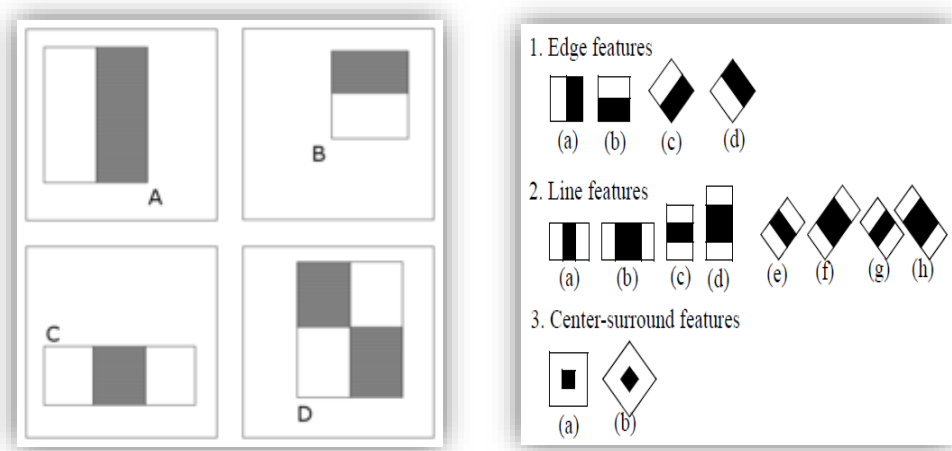


Figura 2.3 (a): Rasgos rectangulares originales de Viola & Jones Figura 2.3(b): Otros rasgos utilizados en nuestro clasificador

Estos rasgos son colocados y escalados independientemente en las todas coordenadas (x, y) de la imagen integral, generando así un muy completo set de rasgos.

2.1.2 Boosting

Gracias a (Duncan ten Velthuis, 2014) que ha elaborado un muy buen trabajo, resumiendo las técnicas relacionadas con los rasgos Haar-Like y los trabajos anteriores de los que se ha nutrido entendemos que para encontrar rasgos importantes se usa la técnica Boosting que puede ser traducida como: impulsar, empujar, promover, etc. El término Boosting viene del modelo “Probably approximately correct” (PAC). Se cree en la idea de que un conjunto de clasificadores débiles, (cada uno de los cuales puede realizar ligeramente mejor la tarea de clasificar que una estimación aleatoria), pueden convertirse en un clasificador mucho más fuerte. La técnica Boosting fue descrita y creada por (Freund, 1999) para la combinación de clasificadores débiles cuya calidad en conjunto era significativamente mejor que la de un clasificador individual.

Como podemos observar, en el algoritmo, se van aumentando los pesos de los rasgos significativos (re-encontrados por los clasificadores individuales) de manera que la conclusión es un clasificador que también posee mayor capacidad de descarte si no cumple la imagen que se analiza con estos rasgos conseguidos y que ahora tienen gran peso.

Viola & Jones usa el método AdaBoost, el cual es adaptable en favor de los rasgos encontrados por los clasificadores anteriores. Los pesos son asignados a los rasgos encontrados por los primeros clasificadores y son entonces usados para entrenar al próximo clasificador. Una vez que se finaliza, (basado en que se logra un porcentaje de falso positivo lo bastante bajo), todos los clasificadores son combinados según la mayor coincidencia en la distribución de los pesos. Resultando entonces en un clasificador fuerte que puede detectar con alta exactitud si una nueva imagen (diferente a las usadas para entrenar) contiene al objeto de interés. En el método de Viola & Jones, un clasificador débil es restringido a seleccionar un rasgo Haar-Like que mejor discrimine entre un ejemplo positivo de uno negativo.

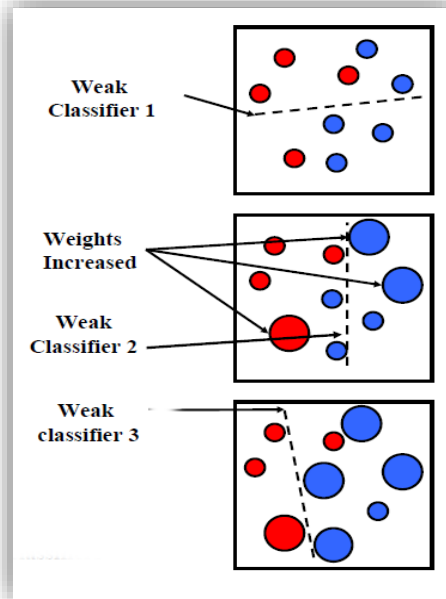


Figura 2.4: Se muestra la construcción de 3 clasificadores débiles cada uno basado en las clasificaciones previas

Algorithm Adaboost

- 1: **Input:** n training examples $(x_i, y_i) \in \mathbb{R}^d \times \{-1, 1\}$, $1 \leq i \leq n$, number of training rounds T .
- 2: **Parameter:** the initial probabilistic weights $w_i(1)$ for $1 \leq i \leq n$.
- 3: **Output:** a strong learner/committee.
- 4: **for** $t = 1$ to T **do**
- 5: Run Algorithm 5 to train a decision stump h_t using the weights $w_i(t)$ and get its weighted error ϵ_t

$$\epsilon_t = \sum_{i=1}^n w_i(t) 1_{h_t(x_i) \neq y_i},$$

- 6: **if** $\epsilon_t = 0$ and $t = 1$ **then**
- 7: training ends and return $h_1(\cdot)$.
- 8: **else**
- 9: set $\alpha_t = \frac{1}{2} \ln(\frac{1-\epsilon_t}{\epsilon_t})$.
- 10: update the weights

$$\forall i, \quad w_i(t+1) = \frac{w_i(t)}{2} \left(\frac{1}{\epsilon_t} 1_{h_t(x_i) \neq y_i} + \frac{1}{1-\epsilon_t} 1_{h_t(x_i) = y_i} \right).$$

- 11: **end if**
- 12: **end for**
- 13: Return the rule

$$f^T(\cdot) = \text{sign} \left[\sum_{t=1}^T \alpha_t h_t(\cdot) \right].$$

Pseudocódigo 2: Método AdaBoost integral (Duncan ten Velthuis, 2014)

2.1.3 Cascada

Viola & Jones introduce un método para construir la cascada de clasificadores fuertes (cada uno construido de varios débiles). Para así poder lograr un alto grado de detección y con un reducido tiempo de computación. La idea principal es que la primera capa sea entrenada desde los pequeños pero eficientes clasificadores promovidos, los cuales pueden rápidamente rechazar a la mayoría de imágenes por no tener los rasgos positivos de interés. Después, con el resto de las imágenes que lograron pasar por estos primeros clasificadores se continua a las siguientes rondas de clasificación dentro de la cascada que son cada vez más exigentes hasta declarar que realmente existe en la imagen el objeto que se está buscando.

La cascada final es capaz entonces de rechazar a la mayoría de las imágenes/subregiones con pocas operaciones.

- Evaluar los rasgos de las regiones.

- Ejecutar los clasificadores débiles por cada rasgo.
- Convertir a los clasificadores débiles en uno más fuerte.

El proceso de la cascada ha ordenado al final una estructura de árbol de descarte.

Cada capa tiene la capacidad de descartar por completo a una región por no contener los rasgos de interés, logrando que sea más rápido el algoritmo en general. (Ver figura 2.5)

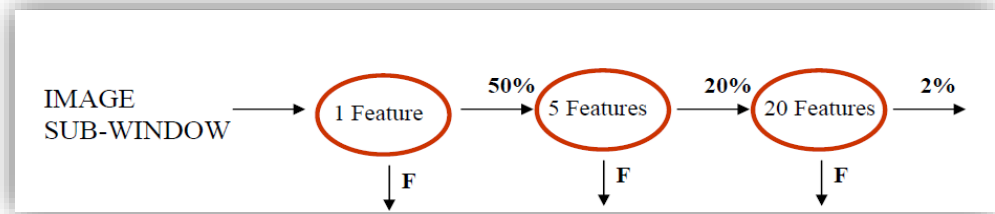


Figura 2.5: El primer clasificador débil logra un 100% de detección y cerca de un 50% de falsos positivos. El segundo logra 100% de detección y 40% de falsos positivos. el tercer clasificador logra 100% de detección con 10% de falsos positivos (2% para el árbol)

```

1: Input: an  $M \times N$  grayscale image  $I$  and an  $L$ -layer cascade of shifted classifiers trained using
   Algorithm 10
2: Parameter: a window scale multiplier  $c$ 
3: Output:  $\mathcal{P}$ , the set of windows declared positive by the cascade
4: Set  $\mathcal{P} = \{[i, i + e - 1] \times [j, j + e - 1] \subset I : e = \lceil 24c^\kappa \rceil, \kappa \in \mathbb{N}\}$ 
5: for  $l = 1$  to  $L$  do
6:   for every window in  $\mathcal{P}$  do
7:     Remove the windowed image's mean and compute its standard deviation.
8:     if the standard deviation is bigger than 1 then
9:       divide the image by this standard deviation and compute its features required by the
       shifted classifier at layer  $l$  with Algorithm 3
10:      if the cascade's  $l$ -th layer predicts negative then
11:        discard this window from  $\mathcal{P}$ 
12:      end if
13:    else
14:      discard this window from  $\mathcal{P}$ 
15:    end if
16:  end for
17: end for
18: Return  $\mathcal{P}$ 
  
```

Pseudocódigo 3: Algoritmo de detección de Viola & Jones integral (Duncan ten Velthuis, 2014)

Resumiendo, podemos ver que las principales características que hacen al algoritmo Viola & Jones un buen algoritmo de detección son:

- Robusto: Un siempre alto porcentaje de detección (verdaderas detecciones positivas) en comparación con el bajo porcentaje de falsos positivos.
- Tiempo Real: Para aplicaciones prácticas al menos 2 frames por segundo pueden ser procesados con pocos recursos de cómputo.

Y que los cuatro pasos fundamentales que distinguen al algoritmo según el propio (Viola & Jones, 2001) son:

- Selección de los rasgos Haar-Like
- Creación de la Imagen integral
- Entrenamiento AdaBoost
- Clasificación en Cascada.

2.1.4 Principales ventajas y desventajas de Viola & Jones

Ventajas (Viola & Jones, 2001)

- Rápido en el cálculo de los rasgos.
- Eficiente selección de los rasgos.
- En vez de un genérico esquema de detección, puede ser entrenado para la detección de otros tipos de objetos.

Desventajas (Duncan ten Velthuis, 2014)

- El detector es más efectivo solamente para imágenes frontales.
- Es muy sensible a cambios en la iluminación.

2.2 Cálculo de la complejidad temporal y espacial de los algoritmos

Para detectar si el objeto de interés está contenido en la imagen, las regiones y subregiones de la imagen integral pasan por la ya mencionada cascada. Siendo cada región $O(FSf)$, donde F es el número de rasgos Haar-Like usados por la cascada y Sf es el tamaño de los píxeles contenidos por cada rasgo. Gracias a la cascada, la mayoría de las regiones pueden ser rápidamente descartadas y solo las positivas (o las falsas positivas) recorrerán la cascada en su totalidad. El número total de operaciones es $O(NFSf)$ donde N es la cantidad total de

regiones y está en función del tamaño de imagen y del número de escalas incluidas en la búsqueda dentro de esta (Flynn, 2009)

El algoritmo de Viola & Jones realiza sus predicciones para la detección basándose en sus propios resultados previos, esto es computacionalmente eficiente usando las imágenes integrales y los clasificadores débiles construidos con el algoritmo AdaBoost para reducir el número rasgos. Todo esto logra un algoritmo de clasificación que puede ser usado para la detección en tiempo real sin mayores inconvenientes.

2.3 Propuesta del sistema

El sistema procura una aplicación de escritorio multiplataforma que esté desarrollada con el lenguaje de programación Java y el IDE Eclipse. Su interfaz es minimalista, lo más sencilla posible permitiéndole así ser intuitiva a cualquier usuario. Este sistema contará con varios módulos, entre ellos, dos dedicados a la carga de las imágenes, uno desde una cámara digital y otro desde los archivos de sistema. Un tercero es dedicado a la aplicación de filtros para el pre-procesado de las imágenes, un cuarto para la selección del archivo clasificador entrenado para el reconocimiento del objeto de interés y el quinto y último dedicado a la localización y seguimiento de dicho objeto en la secuencia de imágenes. El sistema en general permite la detección y seguimiento de objetos de interés en imágenes digitales obtenidas de diversas fuentes y pre-procesadas por el propio sistema a elección del usuario. Para lograr el reconocimiento del objeto, el sistema se vale de cualquier clasificador entrenado con dicho objetivo por el algoritmo Haar-Cascade de Viola & Jones.

2.4 Especificación de los requisitos de software

Resulta oportuno recordar que la finalidad medular de este proyecto de tesis es la selección viable de algoritmos capaces y necesarios para el reconocimiento de los objetos de interés desde perspectivas aéreas y su validación por simulación. Por tal motivo no se detallan en el informe los requisitos que debe cumplir el software ya que este solo servirá como apoyo para probar los resultados teóricos que arroje dicha investigación.

2.4.1 Requisitos Funcionales

RF#1 Reconocimiento y detección del objeto de interés dentro de la imagen.

2.4.2 Requisitos no funcionales

- Software
 - ✓ Máquina Virtual de Java (JRE o JDK) versión 1.8 o superior.
 - ✓ Sistema Operativo: Windows
- Hardware
 - ✓ Memoria RAM: mínimo 1 GB.
 - ✓ Procesador: Dual Core 2.0 GHz o superior.
 - ✓ Espacio en HDD: mínimo 500 Mb.

- Interfaz de usuario

Deberá disponer de una interfaz minimalista, haciendo buen uso del espacio disponible. El menú contendrá las principales funciones del sistema encargadas de las distintas fases del proceso final de detección. Todas sus opciones deben estar señaladas en el lenguaje inglés ya que está pensado como un proyecto en colaboración con otras instituciones e investigadores extranjeros.

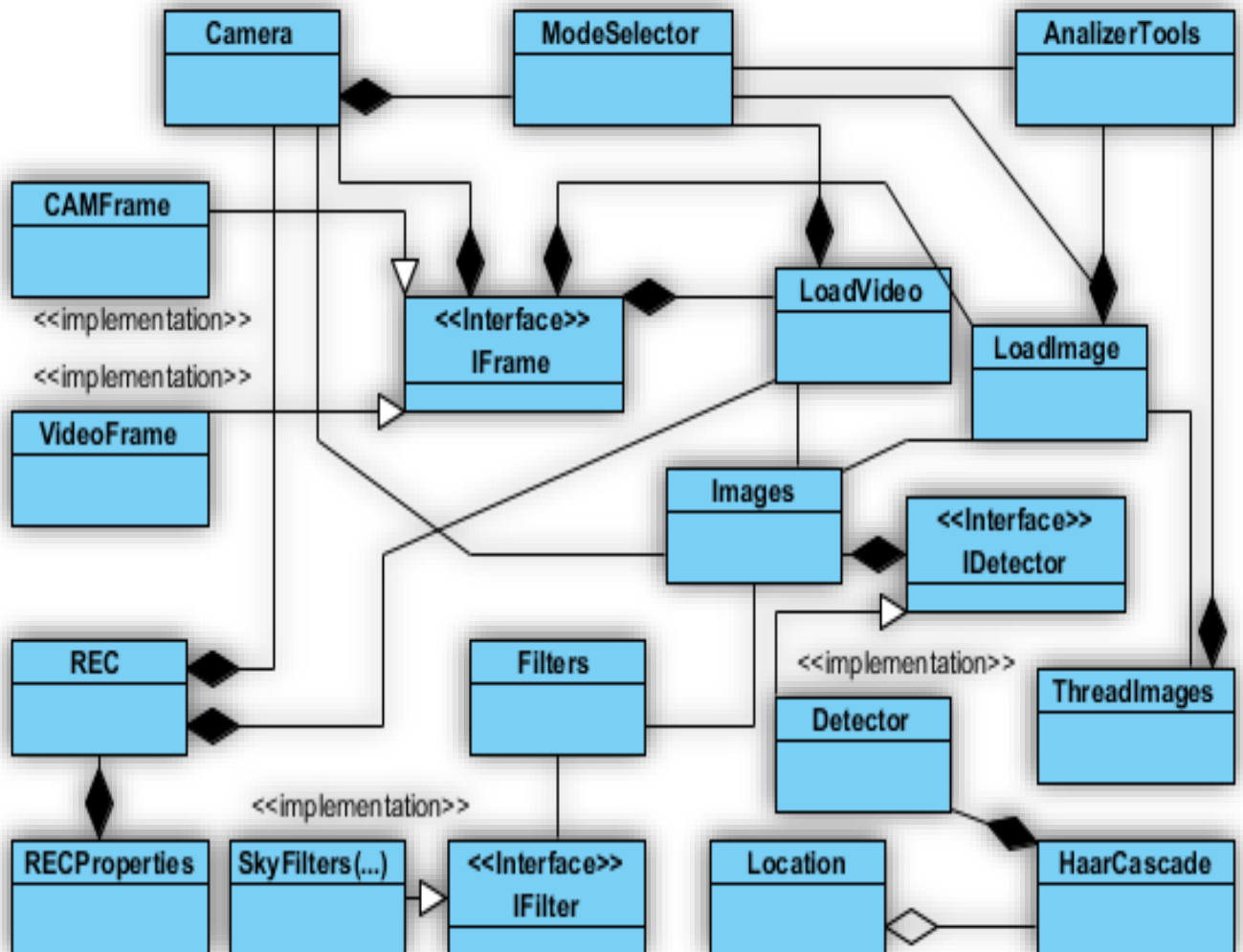
- Usabilidad

El software está dirigido a los mismos desarrolladores e investigadores del proyecto como herramienta útil a la hora de probar principalmente la eficacia de los distintos clasificadores en cascada.

- Rendimiento

En general debe desempeñarse lo más fluido posible pues se trata en ocasiones de operaciones de detección en tiempo real. Está comprobado que este factor está influenciado directamente por la cantidad de filtros utilizados en el pre-procesamiento de la imagen y en la cantidad de objetos de interés encontrados en un mismo frame.

2.5 Diagrama de clases del sistema.



2.6 Captura de las imágenes

Existen muchísimos clasificadores en cascada como el que necesitamos en internet y pueden ser fácilmente encontrados y usados. Pero la mayoría de ellos están entrenados para reconocer rostros, ojos, oídos y bocas sin embargo y gracias a la herramienta OpenCV podemos entrenar uno propio para reconocer cualquier objeto de interés. En el caso de este trabajo de tesis, es necesario un clasificador en cascada capaz de reconocer un prototipo de pruebas, en este caso, un vehículo no tripulado LEGO controlado a distancia vía Bluetooth gracias a su módulo NXT Mindstorms.



Figura 2.6: Módulo NXT Mindstorms de LEGO

La única función de este prototipo es servir como objetivo de interés móvil para ser reconocido y seguido por el clasificador.

2.7 Conclusiones parciales

Hasta aquí se ha profundizado en la teoría necesaria para poder emprender correctamente el camino hacia los resultados deseados. Se ha estudiado a fondo el algoritmo responsable del reconocimiento y la detección de nuestro objeto de interés. Se han descrito sus detalles para la mayor comprensión objetiva de la utilidad de su uso, de sus fortalezas y debilidades. Para con todo esto poder explotarlo de manera conveniente e ir previendo un resultado al alcance real y que satisfaga tanto las expectativas como de las necesidades reales del proyecto.

Capítulo 3: Experimentación y prueba.

En este último y crítico capítulo abordaremos temas que tienen que ver con la creación de la base de imágenes (nuestro conocimiento), el entrenamiento del algoritmo con dicha base y los resultados logrados por el clasificador, fruto de dicho entrenamiento. Asimismo, aplicamos pruebas de calidad a nuestra propuesta de software y describimos de manera sencilla su interfaz de usuario.

3.1 Experimentación.

La fase de experimentación es el núcleo fundamental de este proyecto. Todo lo investigado hasta ahora nos conduce hasta este punto, donde volcamos toda la teoría recopilada en un ejercicio que va desde la colección variada de imágenes de interés hasta la comprobación de la efectividad del algoritmo entrenado a la hora de detectar objetos.

3.1.1 Confección de la base de imágenes.

Fundamental entonces es recolectar una considerable cantidad de imágenes que contengan el objeto de interés (Figura 3.1) desde una perspectiva aérea.



Figura 3.1: Prototipo de objeto de interés para esta tesis. Vehículo Lego.

La literatura recomienda que el número de ejemplos de imágenes positivas (las que contienen al objeto de interés y se usan para entrenar al algoritmo) debe estar por encima de las 1800 unidades según los experimentos de (Jakub Domanski, 2006) mientras que las imágenes negativas (que no contienen al objeto de interés y que sirven para robustecer el descarte de rasgos que pueden ocasionar falsos positivos) deben estar siempre en mayor cantidad, por encima de las 2500 imágenes.

Otra fuente (Duncan ten Velthuis, 2014) utiliza otras cantidades por encima de las 700 imágenes positivas y de las 1700 negativas para algunos experimentos. En otras fuentes podemos encontrar distintas configuraciones incluso usando solo 40 imágenes positivas y 600 negativas (Thorsten Ball, 2013) o 5000 imágenes positivas e igual cantidad de negativas. Obteniéndose mejores resultados en los casos en que las bases de imágenes son mayores y más variadas.

Todos están de acuerdo en que la cantidad de imágenes negativas debe ser igual o mayor que las positivas y esto para garantizar un mayor poder de descarte, que al algoritmo le sea más fácil descartar otros objetos que no son de interés. En favor de la efectividad se prefirió tener dificultades para detectar al objeto que encontrarnos con falsos positivos muy frecuentemente.

Para la selección de las imágenes positivas se recomienda que se tenga en cuenta diferentes fondos e iluminaciones. Y para las imágenes negativas es importante que exista una gran variedad entre las cuales se encuentren algunas con objetos contenidos en ellas que se parezcan bastante, casi iguales, al objeto de interés pero que no lo sean. Esto reduce la probabilidad de que objetos muy parecidos al objetivo sean confundidos y tratados como tal. Con esta fuerte acción estamos especificando los rasgos que son casi iguales pero que no pertenecen al objetivo.

La totalidad de las imágenes positivas (182) para el entrenamiento en este proyecto fueron obtenidas mediante el empleo de un Smartphone Samsung Galaxy S II I777 (SkyRocket 2011). El dispositivo ofrece una cámara trasera de 8 megapíxeles con flash LED simple. Es necesario recordar que, mejorando la calidad del hardware, en este caso de la cámara, se influye directamente en el resultado del algoritmo. Pues la captura con una mayor calidad permite una mejor colección de rasgos visibles para ser analizados.

Las imágenes han sido obtenidas también teniendo en cuenta una amplia variedad de entornos para fomentar que la detección pueda ser posible en distintos ambientes. Desde espacios abiertos bañados de luz solar, ya sea directamente o no y en distintos horarios, hasta espacios bajo techo y con distintas configuraciones de iluminación. Ya sean luces cálidas o frías, habitaciones con ventanas abiertas o cerradas etc. Pero, debemos exponer que si lo que se desea es una mayor eficacia en un entorno controlado, entonces las

tomas deberían enfocarse en ese entorno, hacerse fuertes allí, y no intentar abarcar distintos escenarios.



Figura 3.2: Muestra de las imágenes positivas tomadas para el entrenamiento del algoritmo.

En cuanto a las imágenes negativas (2173) fueron obtenidas en parte del mismo entorno donde fueron capturadas las positivas, con las mismas variaciones de iluminación y localidades, con el propósito de asemejar los rasgos distintivos lo más posible y así mejorar el descarte de falsos positivos. El resto de las imágenes pertenecen a un banco de diseño industrial de productos que se asemejan de alguna manera a nuestro objeto de interés.

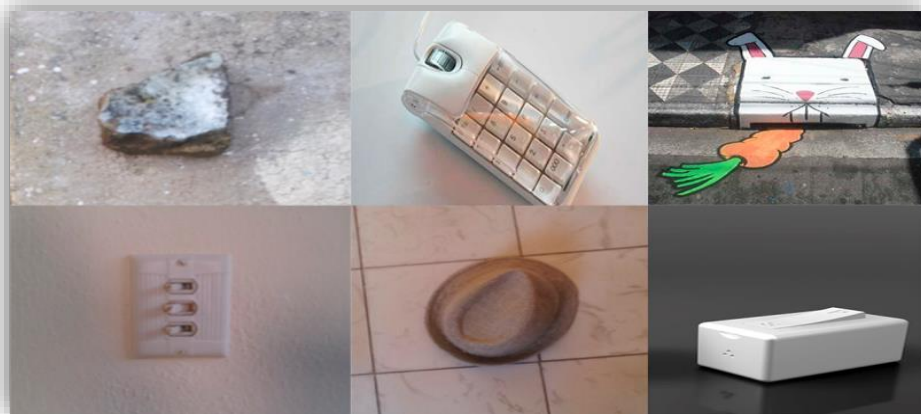
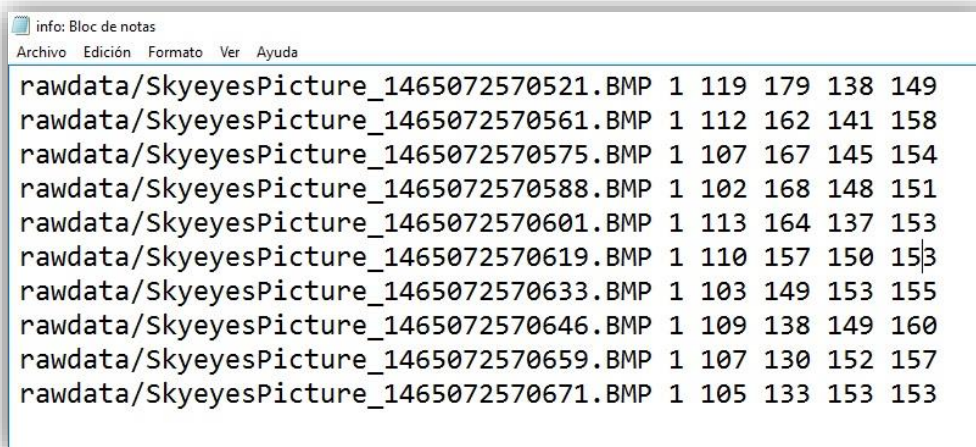


Figura 3.3: Muestra de las imágenes negativas usadas para entrenar al algoritmo

3.1.2 Entrenamiento.

Una vez lograda la base de imágenes lo siguiente es entrenar el algoritmo seleccionado con ellas con el fin de que sea capaz de reconocer a nuestro objeto de interés contenido en las imágenes positivas de la base. Para lograr esto, son necesarios algunos pasos descritos de manera más detallada en (Florian Adolf, 2003). Dichos pasos y las herramientas necesarias para su realización (*Haartraining Stuff*) pueden ser encontradas en la instalación de OpenCV.

La primera fase del entrenamiento es la preparación, donde se deben separar las imágenes positivas de las negativas y crear para ambas una lista con sus nombres de fichero que servirá para luego ser ubicadas durante el entrenamiento. En el caso de la lista de imágenes positivas (figura 3.4), esta además contiene las coordenadas donde se encuentra el objeto de interés dentro de cada imagen.



```
rawdata/SkyeyesPicture_1465072570521.BMP 1 119 179 138 149
rawdata/SkyeyesPicture_1465072570561.BMP 1 112 162 141 158
rawdata/SkyeyesPicture_1465072570575.BMP 1 107 167 145 154
rawdata/SkyeyesPicture_1465072570588.BMP 1 102 168 148 151
rawdata/SkyeyesPicture_1465072570601.BMP 1 113 164 137 153
rawdata/SkyeyesPicture_1465072570619.BMP 1 110 157 150 153
rawdata/SkyeyesPicture_1465072570633.BMP 1 103 149 153 155
rawdata/SkyeyesPicture_1465072570646.BMP 1 109 138 149 160
rawdata/SkyeyesPicture_1465072570659.BMP 1 107 130 152 157
rawdata/SkyeyesPicture_1465072570671.BMP 1 105 133 153 153
```

Figura 3.4 Lista de las imágenes positivas con las coordenadas correspondientes a la ubicación en cada caso del objeto de interés

Dicha lista es fácil de obtener gracias a la utilización de la herramienta *objectmarker.exe*, parte del paquete *Haartraining Stuff*. En la figura 3.5 se puede apreciar un ejemplo del proceso de creación de una lista de imágenes positivas.

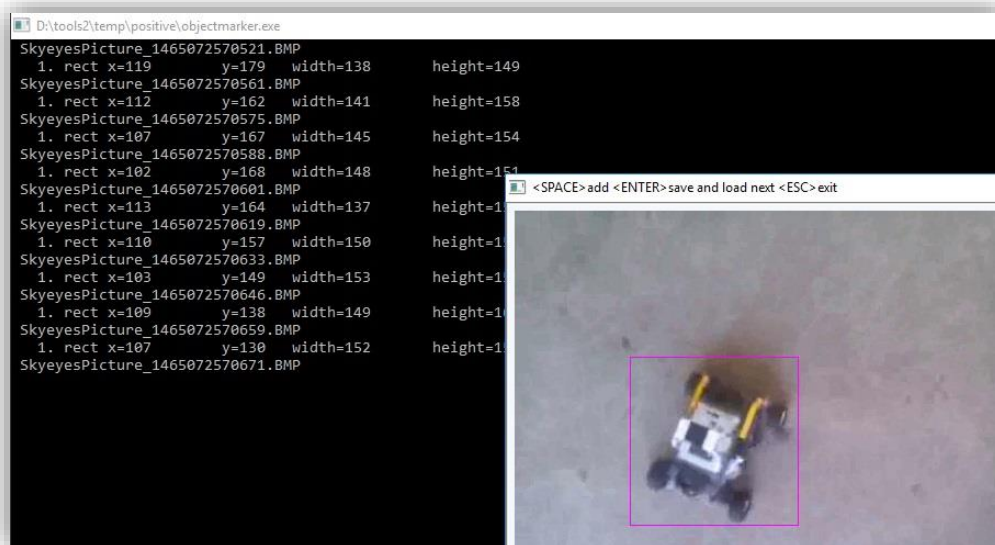


Figura 3.5 Ejemplo para la creación de una lista de imágenes positivas con la herramienta *objectmaker.exe*

La segunda etapa del entrenamiento se encarga de crear los archivos *.vec, estos vectores no son más que el compilado de imágenes positivas y sus propiedades, todas ajustadas a unas medidas comunes y dispuestas para ser usadas en el entrenamiento. La herramienta para la obtención de los vectores es *createsamples.exe* (figura 3.6). En dicha herramienta se debe especificar la cantidad de imágenes positivas que se utilizan, así como las dimensiones.

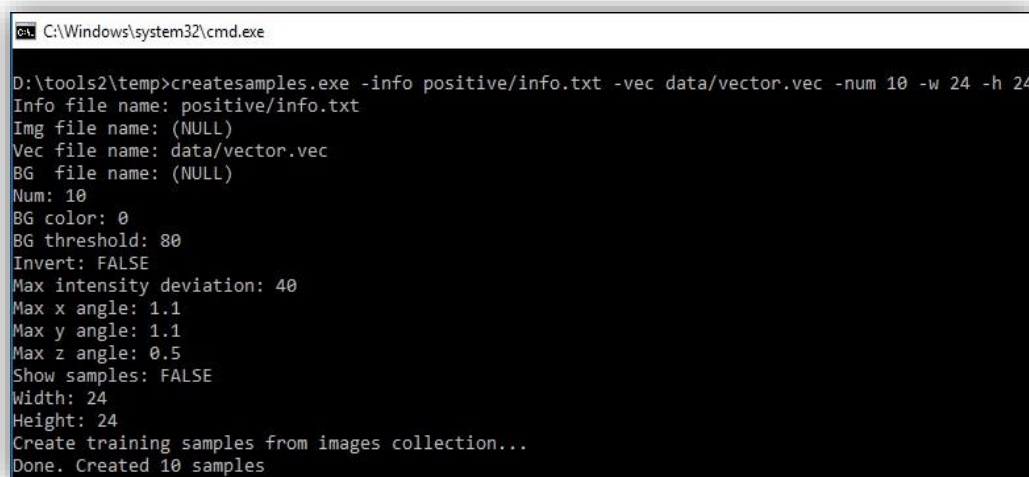


Figura 3.6 Ejemplo de creación del fichero vector.

Luego de obtener el vector, solo resta el entrenamiento en cuestión. Para ello se utiliza la herramienta *haartraining.exe* (figura 3.7). En ella destacamos que se especifican nuevamente la cantidad de imágenes positivas y negativas a utilizar, la cantidad de

memoria RAM que se quiere designar para la tarea y la cantidad de fases de entrenamiento deseadas. El parámetro *-nonsym* se especifica cuando el objeto de interés no es simétrico, como en nuestro caso. Un clasificador para un objeto simétrico requiere menos tiempo de entrenamiento que uno asimétrico para lograr un mismo nivel de calidad.

```

C:\> haartraining.exe -data data\cascade -vec data\vector.vec -bg negative\infofile.txt -npos 10 -nneg 20 -nstages 20 -mem 6000 -mode ALL -w 24 -h 24 -nonsym

Parent node: 3

*** 1 cluster ***
POS: 10 10 1.000000
NEG: 20 4.63787e-005
BACKGROUND PROCESSING TIME: 5.69
Precalculation time: 1.58
+-----+
| N |%SMP|F| ST.THR | HR | FA | EXP. ERR|
+-----+
| 1|100%|-| 1.000000| 1.000000| 0.000000| 0.000000|
+-----+
Stage training time: 0.49
Number of used features: 1

Parent node: 3
Chosen number of splits: 0
Total number of splits: 0

Tree Classifier
Stage
+-----+
| 0| 1| 2| 3| 4|
+-----+

0---1---2---3---4

```

Figura 3.7 Ejemplo de entrenamiento del clasificador en cascada haciendo uso de la herramienta *haartraining.exe*

Con ello se ha conseguido obtener un clasificador en cascada por cada fase indicada, los cuales deben ser convertidos a un único fichero *.XML con la herramienta *haarconv.exe* para ser usado posteriormente por cualquier aplicación para la detección de objetos.

3.1.3 Resultados del algoritmo.

En el caso de este proyecto, el entrenamiento fue realizado sobre una máquina HP Pavilion dv7-6c95dx Entertainment con recursos de memoria RAM 8GB DDR3 y un procesador Intel(R) Core(TM)i7-2670QM 2.20GHz. El entrenamiento en sus 20 fases tardó 2 días en completarse. La prueba del clasificador obtenido fue realizada sobre un lote de 1402 imágenes que contenían al objeto de interés extraídas de un archivo de video, teniendo una efectividad de 84%. Gracias a la experimentación también podemos afirmar que cuanto mayor y mejor confeccionada es la base de imágenes, mejores resultados obtendremos del clasificador entrenado. Aunque entendemos este porcentaje es adecuado para la tarea a resolver, se sugiere incorporar algún mecanismo auxiliar en los casos donde falle la detección. Como trabajo futuro se pueden incorporar filtros como el de Kalman (Kalman, 1960) para mejorar el resultado.

3.2 Interfaz del Software.

La interfaz ha sido pensada para aprovechar al máximo sus espacios de diseño. Siendo así que en todo momento son visibles las dos áreas de visualización de la imagen (una con la captura original y otra con la imagen ya procesada (figura 3.8)). Las funciones principales de la aplicación (cargar ficheros de imágenes o videos, capturar desde una cámara, aplicar filtros de imagen a la captura (figura 3.9), determinar la posición con respecto al tiempo del objeto en la imagen (figura 3.10), cambiar el clasificador entrenado para detectar otro objeto y analizar un lote de imágenes para determinar la efectividad del clasificador), están desplegadas en la barra de menú y dependiendo del proceso que se este ejecutando se pueden usar otras utilidades secundarias (grabar video del resultado obtenido al procesar la captura original, detener o continuar la reproduccion de un video y/o crear un lote de imágenes a partir de un video).

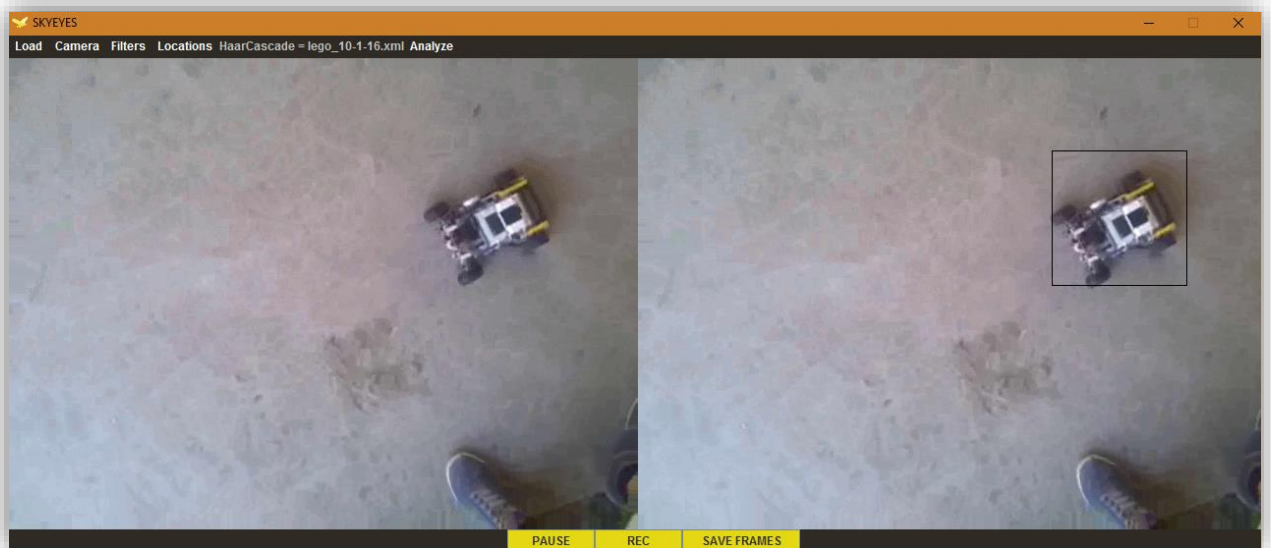


Figura 3.8 Ejemplo de detección del objeto en un video.

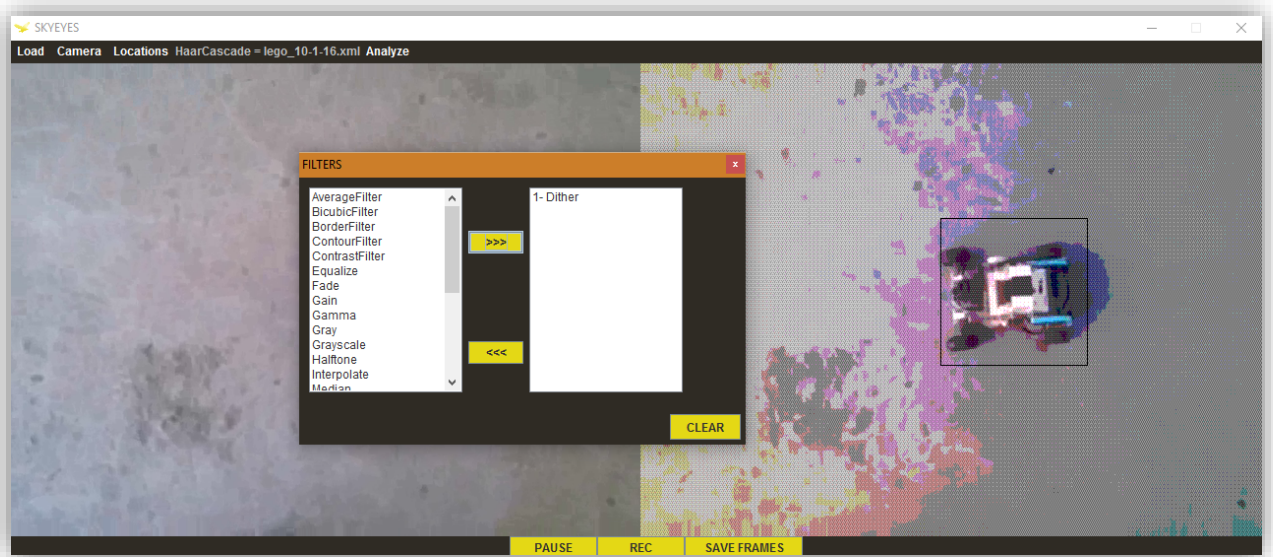


Figura 3.9 Aplicación de filtros a la imagen para mejorar el resultado de detección

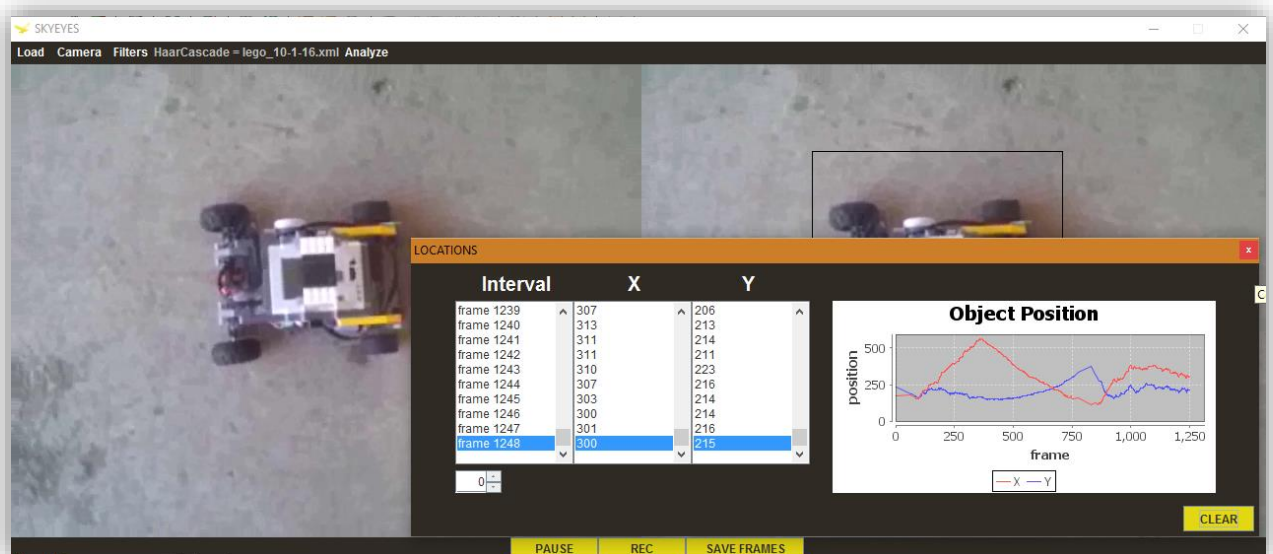


Figura 3.10 Módulo de ubicación del objeto en la imagen.

3.3 Ambiente de prueba.

En esta sección se explica un sencillo ejercicio que muestra lo que es posible realizar mediante el uso del software SkyEyes. En dicho ejercicio, interactúa con el software un esqueleto robótico de piezas LEGO con un motor de corriente directa acoplado (figura 3.11), ello sirve como base móvil a una placa Arduino UNO R3 que controla al motor y a una cámara HD USB. Tanto la cámara como la placa Arduino están conectadas mediante puertos USB a la PC que ejecuta el software.

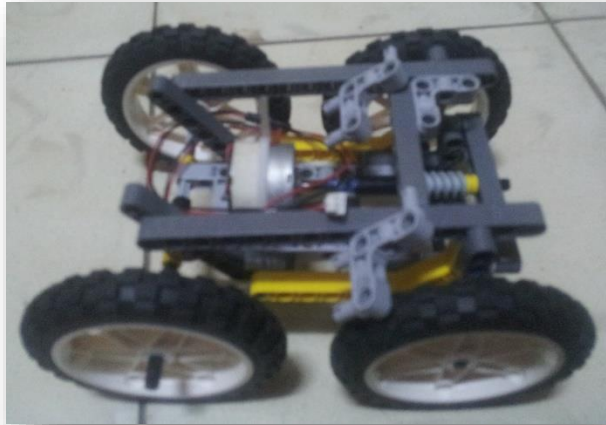


Figura 3.11 Base móvil compuesta por piezas LEGO y un Motor DC



Figura 3.12 Arduino UNO R3

El ejercicio funciona de la siguiente manera: La cámara captura las imágenes, estas son procesadas por SkyEyes para determinar en qué lugar de la imagen se encuentra (de ser detectado) el objeto de interés. Luego de determinar la posición del objeto en la imagen, se envía una señal de control al Arduino para que este accione el motor en dirección a centrar el lente de la cámara con el objeto y vuelve a empezar el ciclo (figura 3.13) creando así el efecto de seguimiento deseado.

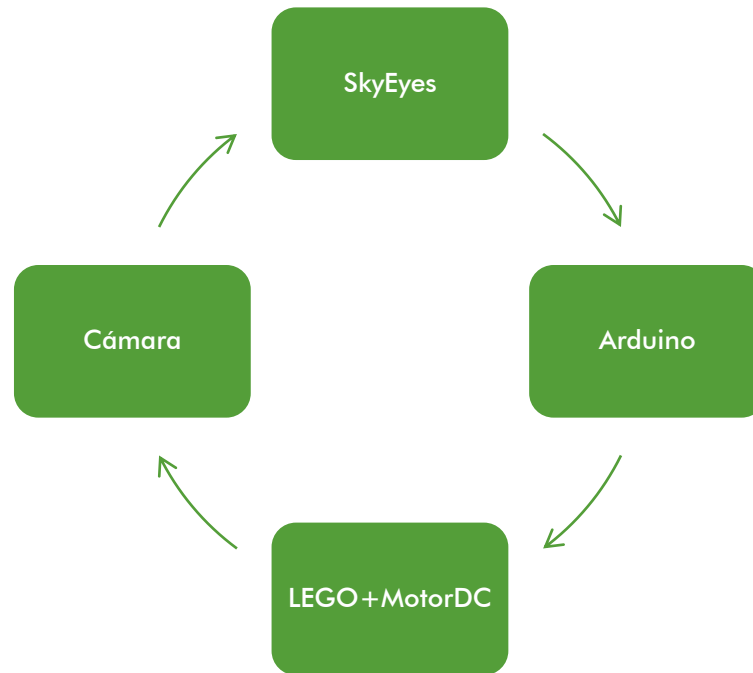


Figura 3.13 Ciclo del ejercicio de demostración de seguimiento.

3.4 Conclusiones parciales.

En este capítulo de experimentación hemos observado el crucial proceso de la obtención y confección de la base de imágenes, proceso crítico al cual se debe prestar especial atención pues influye directamente en el desempeño general del algoritmo. El posterior entrenamiento del algoritmo con dichas imágenes obteniendo resultados alentadores. Así como también se ha vinculado al sistema propuesto con un prototipo de hardware en un ejercicio de demostración con el fin de mostrar la utilidad de sus funcionalidades.

Conclusiones

Durante el desarrollo de este proyecto de tesis se ha obtenido como resultado, una investigación y posterior selección del algoritmo HaarCascade, algoritmo capaz de llevar a cabo las tareas de detección del objeto de interés. También se ha incurrido en la confección de una base de imágenes contenedora de dicho objeto, con propósito de llevar a cabo el crítico entrenamiento del algoritmo seleccionado. Finalmente se creó un sistema automático de reconocimiento que hace uso de dicho algoritmo para el ejercicio de la detección, probando que con la debida preparación (colección de imágenes y entrenamiento) llega a ser lo suficientemente eficiente.

Trabajo Futuro:

Para trabajos futuros que se desprendan de esta investigación se recomienda la incorporación de filtros predictivos como el de Kalman, que ayuden a eliminar falsos positivos o ruido a la hora de efectuar la detección. De igual manera se insta a incursionar en la vinculación del sistema automático de detección con otras gamas de hardware como drones, máquinas de industria etc.

Bibliografía:

1. Ball, T. (2013). *Train Your Own OpenCV Haar Classifier*. Recuperado 23 de Junio 2016, de <http://coding-robin.de/2013/07/22/train-your-own-opencv-haar-classifier.html>
2. Benezeth, Y., Jodoin, P. M., Emile, B., Laurent, H., & Rosenberger, C. (2010). Comparative study of background subtraction algorithms. *Journal of Electronic Imaging*, 19(3), 033003-033003.
3. Dalal, N., & Triggs, B. (2005, Junio). *Histograms of oriented gradients for human detection*. Ponencia presentada en 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, the Netherlands.
4. De Kok, P., Girardi, N., Gudi, A., Kooijman, C., Methenitis, G., Negrijn, S., ... & Visser, A. (2013). *Team description for RoboCup 2013 in Eindhoven*. the Netherlands: Dutch Nao Team, Universiteit van Amsterdam & TU Delft.
5. Domanski, J. (2006). *How to use tools to create your own cascade of boosted classifiers*. Recuperado el 23 de Junio 2016, de http://en.verysource.com/code/4587633_1/read_this_first.txt.html
6. Domenech M., A. (2009). *Reconocimiento gestual a través de webcam*. Disertación doctoral publicada en Universidad Pontificia de Comillas, Madrid.
7. Duncan, S., Intelligentie, B. O. K., y Visser, A. (2014). *Nao detection with a cascade of boosted weak classifier based on Haar-like features*. [formato electrónico] [s.l.]: [s.n.].
8. Fabisch, A., Laue, T., & Röfer, T. (2010). *Robot recognition and modeling in the robocup standard platform league*. [formato electrónico] [s.l.]: [s.n.].
9. Flynn, H. (2009). *Machine learning applied to object recognition in robot search and rescue systems*. Disertación de maestría publicada en Universidad de Oxford.
10. Freund, Y., Schapire, R., y Abe, N. (1999). A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14, 771-780.
11. Gonzalez, R. & Woods, R. (2010). *Digital image processing*. Upper Saddle River, N.J.: Prentice Hall.
12. Hardin, P. J., & Jensen, R. R. (2011). Small-scale unmanned aerial vehicles in environmental remote sensing: Challenges and opportunities. *GIScience & Remote Sensing*, 48(1), 99-111.
13. Hodgson, A., Kelly, N., & Peel, D. (2013). Unmanned aerial vehicles (UAVs) for surveying marine fauna: a dugong case study. *PloS one*, 8(11), 79556.
14. Jones IV, G. P., Pearlstine, L. G., & Percival, H. F. (2006). An assessment of small unmanned aerial vehicles for wildlife research. *Wildlife Society Bulletin*, 34(3), 750-758.
15. Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1), 35-45.
16. Margolis, M. (2012). *Arduino cookbook*. Sebastopol, Calif.: O'Reilly.
17. OpenCV | OpenCV. (2016). *OpenCV.org*. Recuperado el 23 de junio del 2016, de <http://opencv.org/>
18. Papageorgiou, C. P., Oren, M., & Poggio, T. (1998, Enero). A general framework for object detection. En *Computer vision, 1998. sixth international conference on* (pp. 555-562). USA: IEEE.
19. Pouli, T., & Reinhard, E. (2011). Progressive color transfer for images of arbitrary dynamic range. *Computers & Graphics*, 35(1), 67-80.
20. Ramírez, D. L. A., & Martínez, C. A. R. (2006). *Conceptos Básicos del Procesamiento Digital de Imágenes Usando OrquideaJAI*. Disertación doctoral publicada en Universidad Nacional de Colombia, Colombia.

21. Reinius, S. (2013). *Object recognition using the OpenCV Haar cascade-classifier on the iOS platform* [formato electrónico] [s.l.]: [s.n.].
22. Rodríguez M., R. (2004). *Fundamentos de visión por computadora*. [formato electrónico] [s.l.]: [s.n.].
23. Ross, J. A. (2008). *Computer vision and target localization algorithms for autonomous unmanned aerial vehicles* Dicertación Doctoral, The Pennsylvania State University.
24. Roth, P. M., & Winter, M. (2008). *Survey of appearance-based methods for object recognition*. Graz University of Technology, (Inf. Téc) Austria: [s.n.].
25. Soo, S. (2014). *Object detection using Haar-cascade Classifier*. [formato electrónico] [s.l.]: [s.n.].
26. Tamersoy, B. (2009). *Background subtraction*. [formato electrónico] USA: University of Texas.
27. Tatem, A. J., Lewis, H. G., Atkinson, P. y Nixon, M. (2000). *Land cover mapping at the sub-pixel scale using a Hopfield neural network*. [s.l.]: [s.n.].
28. Viola, P., & Jones, M. (2001). *Rapid object detection using a boosted cascade of simple features*. Ponencia presentada en Computer Vision and Pattern Recognition, USA.
29. Watts, A. C., Perry, J. H., Smith, S. E., Burgess, M. A., Wilkinson, B. E., Szantoi, Z. (2010). Small unmanned aircraft systems for low-altitude aerial surveys. *The Journal of Wildlife Management*, 74(7), 1614-1619.
30. Xu, G., Shen, W., & Wang, X. (2014). Applications of wireless sensor networks in marine environment monitoring: A survey. *Sensors*, 14(9), 16932-16954.