UNIVERSIDAD JOSÉ ANTONIO PÁEZ

DESARROLLO DE UN SOFTWARE DE RECONOCIMIENTO DE COLORES DE OBJETOS EN IMÁGENES DIGITALES ORIENTADO A PERSONAS CON DALTONISMO DEL TIPO DEUTERANOMALÍA

Autores: Caicedo Cristhian
C.I.: 19.919.340
Pérez Manuel
C.I.: 25.047.747
Tutor Académico: Ing. Cárdenas George

Urb. Yuma II, calle Nº 3. Municipio San Diego
Teléfono: (0241) 8714240 (master) – Fax: (0241) 8712394

REPÚBLICA BOLIVARIANA DE VENEZUELA
UNIVERSIDAD JOSÉ ANTONIO PÁEZ
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA DE COMPUTACIÓN

DESARROLLO DE UN SOFTWARE DE RECONOCIMIENTO DE COLORES DE OBJETOS EN IMÁGENES DIGITALES ORIENTADO A PERSONAS CON DALTONISMO DEL TIPO DEUTERANOMALÍA

Trabajo de grado presentado como requisito para optar al título de
INGENIERO DE COMPUTACIÓN

Autores: Caicedo Cristhian
C.I.: 19.919.340
Pérez Manuel
C.I.: 25.047.747
Tutor Académico: Ing. Cárdenas George

San Diego, Marzo de 2018

Universidad José Antonio Páez
Facultad de Ingeniería

FI-C-065-2018-1

Valencia, 25 de Enero de 2018

Ciudadanos:
Caicedo Cristhian
C.I. 19.919.340
Pérez Manuel
C.I. 25.047.747
Presente.-

Cumplo con informarle que la Comisión de Trabajo de Grado y Pasantías de la Facultad de Ingeniería en su reunión N° **1-2018** de fecha **25/01/2018** aprobó el proyecto de trabajo de grado titulado **"DESARROLLO DE UN SOFTWARE DE RECONOCIMIENTO DE COLORES DE OBJETOS EN IMÁGENES DIGITALES ORIENTADO A PERSONAS CON DALTONISMO DEL TIPO DEUTERANOMALÍA"** presentado por usted(es) como requisito para optar al título de Ingeniero de Computación

Se ratifica la designación del Ing. George Cárdenas, C.I. 16.897.193 y la Ing. Alicia Yanez de Pizzella, C.I. 4.598.880 como Tutotes Académicos que lo asesorarán en el desarrollo de este proyecto.

Atentamente,

Prof. Zulay Saleedo
Decana de la Facultad de Ingeniería

c. c. Coordinación de Pasantías y Trabajo de Grado (1).

ZS/lr

REPÚBLICA BOLIVARIANA DE VENEZUELA
UNIVERSIDAD JOSÉ ANTONIO PÁEZ
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA DE COMPUTACIÓN

ACEPTACIÓN DEL TUTOR

Quien suscribe, Ingeniero George Cárdenas portador de la cédula de identidad N° 16.897.193, en mi carácter de tutor del trabajo de grado presentado por el(los) ciudadano(s) Cristhian Armando Caicedo Panqueva y Manuel Antonio Pérez Herrera, portador(es) de la cédula de identidad N° 19.919.340 y N° 25.047.747, (respectivamente), titulado **DESARROLLO DE UN SOFTWARE DE RECONOCIMIENTO DE COLORES DE OBJETOS EN IMÁGENES DIGITALES ORIENTADO A PERSONAS CON DALTONISMO DEL TIPO DEUTERANOMALÍA** presentado como requisito parcial para optar al título de Ingeniero de Computación, considero que dicho trabajo reúne los requisitos y méritos suficientes para ser sometido a la presentación pública y evaluación por parte del jurado examinador que se designe.

En San Diego, a los dieciséis días del mes de marzo del año dos mil dieciocho

Ing. George Cárdenas
C.I.: 16.897.193

AGRADECIMIENTOS

Le agradezco primeramente a Dios, por darme la oportunidad de vivir este momento, por darme la fortaleza y resistencia para superar aquellos pasos difíciles en el camino hasta esta meta.

Seguidamente, también quiero agradecerles a mis padres Aitxa y William, por guiarme y apoyarme, por los valores que me han inculcado que me ha moldeado como buena persona y por haberme dado la oportunidad de tener una excelente educación a lo largo de mi vida.

De igual manera le agradezco a Alberto por ser un hermano, por aconsejarme en todo momento lo cual me ayudó a crecer tanto personal como profesionalmente.

Gracias a mi compañero y amigo Cristhian, por su comprensión, paciencia, y por el esfuerzo realizado durante todo el proyecto. Y a sus padres por recibirme en su casa, y permitirnos trabajar hasta altas horas de la noche.

A nuestro tutor George Cárdenas, por su orientación y comprensión en todo momento, quien nos ayudó con su experiencia para lograr la culminación de este proyecto.

A mi bae, gracias a por apoyarme y motivarme a no bajar los brazos, por guiarme en esta última etapa y siempre estar pendiente de lo que me faltaba. Gracias por todo el cariño y la confianza.

A todos mis tíos, tías, primos y primas, por apoyarme y no dudar de que podría conseguir este objetivo. Así como también a mis amigos y futuros colegas, por ayudarme a crecer durante la carrera, a esforzarme más y motivarme a siempre aprender cosas nuevas.

MANUEL ANTONIO, PEREZ HERRERA

DEDICATORIA

A Dios.

Por llenarme de bendiciones como culminar esta experiencia, haberme dado salud brindarme la sabiduría para avanzar día a día durante toda la carrera.

A mis padres y hermano.

Por su amor y apoyo incondicional, por ser pilar fundamental en todo lo que soy hoy, en toda mi educación, tanto académica como en la vida. Esto es por ustedes, los quiero.

A mis familiares.

Por siempre estar conmigo y apoyarme en todo momento, además de representar un soporte emocional por siempre estar ahí cuando los necesite.

A mis amigos.

Por compartir momentos inolvidables durante toda la carrera, puedo decir que aprendí algo de todos y cada uno de ustedes.

MANUEL ANTONIO, PEREZ HERRERA

AGRADECIMIENTOS

Le agradezco primeramente a Dios, por darme la salud y bienestar a lo largo de toda la carrera, por guiarme en el camino correcto de este trabajo de grado y en mi vida diaria. Por su protección y compañía constante.

También agradezco a mis padres Armando Caicedo y Mireya Panqueva, por brindarme su apoyo continuo e incansable en todo el transitar de mis estudios hasta este momento; por la paciencia, tolerancia y confianza que han tenido en mí para la culminación de este trabajo de grado. Igualmente, agradezco enormemente la ayuda y el ejemplo que me ha dado mi hermana Nohora Caicedo, por su apoyo y cariño desde pequeño hasta el momento de hoy, por haber sido un ángel guardián, preocuparse y ocuparse de mí en tantas ocasiones.

Agradezco especialmente a la comunidad científicos y especialistas en las áreas de estudio con las que se trabajaron, por su arduo esfuerzo e incontables horas dedicadas para poder brindar todo el conocimiento valioso, el cual fue de suma utilidad en esta investigación.

Gracias a mi compañero y amigo Manuel, por su comprensión, paciencia y por el esfuerzo realizado durante todo el proyecto a pesar de las adversidades.

A nuestro tutor George Cárdenas, por su orientación y comprensión en todo momento, quien nos ayudó con su experiencia para lograr la culminación de este proyecto.

A todos mis familiares por parte de mamá, quienes me apoyaron y brindaron ayuda cuando más la necesitaba. Asimismo, agradezco a mi compañero de carrera Eladio Mejías, el cual me brindó su apoyo moral y compañía durante todo el transcurso de la realización de este trabajo de grado.

CRISTHIAN ARMANDO, CAICEDO PANQUEVA

DEDICATORIA

A mis padres y hermana, por su apoyo y amor incondicional a lo largo de estos años, así como toda la ayuda que me han brindado en los momentos más duros de mi vida.

A mi tio Jorge Luis y tía Natalie Prevost, por ser otra fuente adicional de cariño y soporte, en especial a mi tío por inspirarme a escoger esta carrera como profesión. Por su apoyo financiero para la compra del Raspberry Pi.

A mis familiares, por brindarme su apoyo, compañía y confianza en todo momento.

CRISTHIAN ARMANDO, CAICEDO PANQUEVA

ÍNDICE GENERAL

LISTADO DE TABLAS

LISTADO DE FIGURAS

REPÚBLICA BOLIVARIANA DE VENEZUELA

UNIVERSIDAD JOSÉ ANTONIO PÁEZ
FACULTAD DE INGENIERÍA
ESCUELA DE COMPUTACIÓN
INGENIERIA DE COMPUTACIÓN

DESARROLLO DE UN SOFTWARE DE RECONOCIMIENTO DE COLORES DE OBJETOS EN IMÁGENES DIGITALES ORIENTADO A PERSONAS CON DALTONISMO DEL TIPO DEUTERANOMALÍA

Autores: Manuel, Pérez
Cristhian, Caicedo
Tutor: George Cárdenas
Fecha: Marzo, 2018

RESUMEN

La mayor cantidad de información adquirida por el ser humano es través del sentido de la visión, mediante la cual se puede percibir el entorno físico que le rodea y cuyos elementos poseen características únicas, como por ejemplo, el color. Actualmente, existen una diversidad de maneras de representar gráficamente este entorno, dentro de las cuales se encuentran las imágenes digitales y cuya interacción es diariamente requerida, lo que las hace de suma importancia para obtener información pertinente en un momento dado. Sin embargo, algunas personas no poseen la capacidad de detectar algunas de estas características, como es el caso de aquellas que padecen de daltonismo, las cuales están limitadas a reconocer un espectro de colores más reducido de lo normal. Es por ello, que la presente investigación plantea el desarrollo de un software de reconocimiento de colores de objetos en imágenes digitales, orientado a personas con daltonismo del tipo deuteranomalía y con el objetivo de mejorar artificialmente su percepción cromática, para así poder interpretar correctamente la información que contienen éstas dentro de su contexto. Para llevar a cabo esto, la actual investigación posee un diseño no experimental de tipo campo, sometido a la modalidad de proyecto factible y de nivel proyectivo. Asimismo, se utilizó la metodología AUP (Proceso Unificado Ágil) a lo largo del proyecto, el cual se enfocó en la línea de investigación de desarrollo de nuevas tecnologías. Se concluyó que las técnicas fueron suficientes para reconocer y detectar de manera aceptable los colores problemáticos, de tal forma que es posible encerrar estos mediante regiones o segmentos que poseen distintas formas físicas.

Descriptores: Software, Colores, Imagen digital, Daltonismo, Deuteranomalía.

INTRODUCCIÓN

A lo largo de la historia, las imágenes han ido evolucionando constantemente, de ser una simple imitación de la vida real o la imaginación del ser humano a ser un objeto de estudio en muchas ciencias, las cuales han sido transportadas del plano físico al digital. A partir de esto, surge una rama de la ciencia de la computación denominada como "procesamiento de imágenes digitales", la cual se encarga de proveer un medio (conjunto de técnicas) que se aplican a las imágenes digitales con el objetivo de mejorar la calidad o facilitar la búsqueda de información. Gracias a esto, en la actualidad se puede contar con diversas herramientas que plasman esta realidad, las cuales son invaluables en áreas como la robótica, el control industrial y la medicina. De esta última, se destaca un sistema que es irremplazable: el de la visión humana.

La visión es un sentido muy avanzado y no es de sorprenderse que las imágenes juegan un papel primordial en la percepción humana. No obstante, el ojo humano está condicionado a cambios anormales imposibles de controlar y es propenso a fallar, como es el caso de los individuos con ceguera del color o mejor conocido como daltonismo. A diferencia de los ojos humanos, los dispositivos electrónicos como las cámaras pueden ser reemplazadas cuando se averían o incluso reparadas, además de que sirven de soporte para extraer información del entorno físico. Por tal motivo, nace el interés de mejorar artificialmente la percepción visual del color de una persona respecto a las imágenes digitales, mediante el procesamiento de estas últimas con el fin de mejorar la interpretación de la información que contienen. Dicho esto, la estructura de este trabajo se presenta por capítulos como se indica a continuación:

Capítulo I: El Problema. Comprende el planteamiento de la situación actual y la identificación del problema existente. Se explica detalladamente la limitación que está presente en los individuos con esta deficiencia visual, al

igual que sus causas. En este capítulo se describen, además, los objetivos que se desean lograr con la realización del sistema propuesto, la razón por la cual se lleva a cabo la investigación y el alcance que tendrá esta.

Capítulo II: Marco Teórico. Se dan a conocer los antecedentes del problema, es decir, aquellos que servirán de referencias para contribuir y complementar el proceso de alcanzar los objetivos y resultados esperados. Además, se exponen ciertos conceptos fundamentales en las áreas de estudio pertinentes, los cuales permitirán un análisis y explicación más profunda de la problemática.

Capítulo III: Marco Metodológico. Este capítulo describe el tipo, nivel y diseño de la investigación, determina la población y muestra, así como las técnicas y los instrumentos de recolección de datos los cuales son: entrevista mediante un guion del tipo informal y observaciones científicas auxiliadas por una cámara fotográfica. También se describen las distintas fases de desarrollo paso a paso, las cuales servirán para llevar a cabo la ejecución de cada uno de los objetivos específicos propuestos.

Capítulo IV: Resultados. Se presenta en detalle todas las actividades que se realizaron durante cada fase para el desarrollo del sistema, en donde están incluidas las diferentes tareas ejecutadas y el resultado de las mismas como diagramas, figuras y cuadros. Después de esto, se procede a plantear las conclusiones y recomendaciones de la investigación.

Capítulo V: Conclusiones y Recomendaciones. Dentro de este marco se plantean las conclusiones basadas en los resultados ya obtenidos durante las fases de la investigación y de acuerdo a los objetivos alcanzados. Asimismo, se indican las recomendaciones que se pueden tomar en cuenta para un futuro proyecto que involucre los mismos temas de interés.

CAPÍTULO I
EL PROBLEMA

1.1 Planteamiento del problema

Uno de los principales sentidos del cuerpo humano es la vista, la cual según Pérez y Merino (2008) considera que "...le brinda a distintos organismos la posibilidad de detectar la luz y reconocer lugares, personas y objetos"; por lo tanto, es correcto destacar que por medio de ella se percibe la mayor cantidad de información posible del entorno físico a diario, en comparación con el resto de los sentidos. Sin embargo, esta capacidad sensorial sólo puede ser proporcionada por el sistema visual humano, cuyo órgano principal es el ojo y es definido según González (2014) como "...una estructura prácticamente esférica en la que entra la luz sólo por un pequeño agujero...".

A su vez, el ojo está compuesto en su interior por una parte fundamental denominada retina, de la cual el mismo autor se refiere como "...una membrana sensible a la luz..." en donde "...se distribuyen dos tipos de receptores de luz llamados conos y bastones"; adicionalmente, Nave (2000) establece que de "...los 6 a 7 millones de conos se pueden dividir en conos rojos (64%), conos verdes (32%), y conos azules (2%)... Ellos proveen la sensibilidad del color del ojo". Partiendo de esta definición, cabe mencionar que el entorno físico que rodea al ser humano se encuentra fuertemente vinculado y dominado por la presencia de una amplia gama de colores con un conjunto infinito de tonalidades, crominancias y valores, los cuales aportan una característica única y algunas veces, una relación inequívoca entre los objetos y las personas que cohabitan dentro de él. Por lo cual es relevante destacar que las personas deben poseer todos los conos y que los mismos funcionen de manera correcta para poseer una visión normal.

A diario las personas se ven en la necesidad de obtener información acerca de aquellos elementos que se encuentran en sus alrededores, con el fin de poder extraer e interpretar las características de los mismos para establecer una relación o identificar una ya existente, bien sean referentes al color como se mencionó anteriormente o, a su forma, distancia, posición y tamaño; estos atributos son los que facilitan el entendimiento del contexto en el que están ubicados. Generalmente, esta capacidad visual de adquisición y asociación entre un conjunto de datos debería estar presente en un ser humano con una visión normal.

Desafortunadamente, no todas las personas cuentan con la cantidad necesaria de conos o con el correcto funcionamiento de los mismos, lo que en consecuencia produce una limitación al momento de reconocer las propiedades o atributos de los colores con precisión, dificultando la interpretación de las relaciones entre los colores de los elementos que conforman el entorno físico, ya sea real o digital. De igual manera, éste inconveniente se mantiene presente incluso en el mundo de la informática, ya que la visualización de representaciones gráficas computarizadas, o mejor conocidas como imágenes digitales, posee una disposición de los elementos del mundo real llevados al mundo virtual con una similitud aproximadamente exacta a su versión original, tomando en cuenta que algunas características se mantienen inmutables para ambos casos.

Esta limitación puede significar para aquellos individuos una desventaja en el desarrollo diario de actividades cotidianas. Tal es el caso del ámbito laboral, en el cual la probabilidad de conseguir un trabajo en ciertas áreas profesionales se vean reducidas, por ejemplo: los ingenieros electrónicos al momento de trabajar con cables y componentes; los pintores cuando deben elaborar una obra con pinturas; los diseñadores que necesitan ajustar los elementos a sus ideas concebidas para poder implementarlas luego sobre algún plano, entre otros. Aunado a esto, se considera necesario exponer la situación en la cual las ideas percibidas y transmitidas entre varios individuos no coinciden, estando dentro de este grupo algunas personas que

posean este impedimento; en consecuencia, se puede generar confusión o malentendidos en torno a una temática.

Por su parte, en el ámbito de estudio también se pueden ver afectadas otras disciplinas durante el proceso de aprendizaje que requieran de una correcta percepción de los colores, como es el caso de: La descripción por primera vez de los colores a los alumnos por parte de los profesionales en educación inicial; el reconocimiento de las diferentes bacterias y demás seres vivos en estudios biológicos; la identificación de las partes de un sistema del ser humano en la medicina; distinguir los ingredientes necesarios para un platillo en referencia a las artes culinarias; por mencionar algunos ejemplos.

A partir de esto, es pertinente resaltar que existe una enfermedad visual denominada daltonismo o ceguera del color, la cual según Pérez y Marino (2012) es "...una enfermedad de carácter genético y perfil congénito que, según los expertos, no es progresiva. Se trata de una anomalía que afecta a la vista...". Sin embargo, su origen no es necesariamente congénito sino que también puede ser adquirida por otros factores, tales como: el consumo o contacto con sustancias tóxicas; provocada por enfermedades como la degeneración vascular y la diabetes; generada por traumas severos en el área del ojo.

Ahora bien, el daltonismo puede clasificarse en cuatro categorías: acromático, monocromático, dicromático y tricromático anómalo. En referencia a esta clasificación, Castillero (2017) menciona que en el caso del tipo tricromático anómalo "...el individuo posee los tres tipos de pigmentos, pero al menos uno funciona de manera anómala y no puede percibir el color de la misma manera que un tricromático.", lo que indica que existen tres tipos diferentes de anomalías con esta característica, las cuales están determinadas por el pigmento que funcione incorrectamente. De igual forma, el autor anterior describe a éstas por su nombre y color: en la "Deuteranomalía...el pigmento verde no funciona correctamente.", mientras que en la "Protanomalía...el rojo no es percibido en su totalidad por parte

del ojo.” y por último, en la “Tritanomalía…el color que no se capta correctamente es el azul.”.

Por otro lado, según un estudio realizado a nivel mundial en hombres y mujeres por Kalloniatis y Luu (2017), las deficiencias de la visión del color que son congénitas afectan aproximadamente un *8%* y *0.5%*, correspondiendo en parte un *1%* y *0.01%* a la Protanomalía; *5%* y *0.4%* a la Deuteranomalía; y una cantidad despreciable a la Tritanomalía debido a su rareza, respectivamente para cada género. No obstante, otras fuentes de información expresan estos datos por grupo étnico para el género masculino, repartidos de la siguiente manera: *5.6%* a los caucásicos, *3.1%* a los asiáticos, *2.6%* a los hispanos y *1.4%* a los de color (Georgiev, 2017). Teniendo en cuenta las estadísticas anteriores, queda en evidencia que la deuteranomalía se establece indiscutiblemente como el común denominador de acuerdo a las proporciones del resto, lo que la posiciona como una de las discapacidades más urgentes a tratar.

1.2 Formulación del problema

De acuerdo a lo anteriormente planteado se formula la siguiente interrogante: ¿cómo mejorar la percepción cromática sobre objetos de las personas que padecen daltonismo del tipo deuteranomalía?

1.3 Objetivos

1.3.1 Objetivo General

Desarrollar un software de reconocimiento de colores de objetos mediante el procesamiento de imágenes digitales para mejorar la percepción cromática a las personas que padecen de deuteranomalía.

1.3.2 Objetivos Específicos

- Determinar el espectro de color y los colores de confusión correspondientes a las personas con deuteranomalía a través de una búsqueda bibliográfica que sustente la situación actual relacionada con esta deficiencia visual.

- Establecer los requerimientos funcionales y no funcionales del software mediante el uso de la ingeniería de requerimientos.
- Analizar los métodos y técnicas existentes en la librería OpenCV orientados a detectar los colores de un objeto en una imagen digital, mediante pruebas de ejecución a los algoritmos pertinentes.
- Desarrollar los algoritmos para la detección de colores de un objeto en una imagen digital mediante el uso de los módulos pertinentes a la librería de OpenCV.
- Verificar los algoritmos para la detección de colores de objetos en una imagen digital mediante pruebas de caja blanca.

1.4 Justificación

Esencialmente, la necesidad del ser humano en reconocer las características de su entorno es sumamente importante, ya que éste se encuentra plagado de elementos cuya gama de colores es amplia y muy variada, la cual debe ser tomada en cuenta en múltiples ocasiones de la vida cotidiana. Estas actividades diarias podrían ir desde aquellas con el menor porcentaje de ocurrencia hasta las de mayor frecuencia; tal es el caso de las luces de tráfico, en donde dos de los colores utilizados son el color rojo y verde, los cuales se ven afectados por la ceguera del color. De manera similar, existen otras situaciones que perjudican de una manera más sutil y engañosa: una quemadura de la piel que no puede ser realmente vista; no poder determinar si una carne está cocinada o no por su color; distinguir con exactitud la madurez de una fruta o vegetal; descifrar correctamente mapas coloreados y gráficas puede ser bastante difícil, entre otros (Flück, 2016; 22).

Es un hecho que el daltonismo es una enfermedad que afecta aproximadamente al 8% de la población mundial, tanto a hombres como a mujeres de distintas razas y pertenecientes a distintos continentes. Asimismo, los estudios estadísticos anteriormente mostrados señalan a la deuteranomalía como la discapacidad visual

predominante entre los tipos de daltonismo, motivo por el cual la actual investigación gira en torno a ésta.

Por otra parte, aun cuando esta anomalía apunta al fallo presente en los conos con pigmentación verde, la realidad muestra que varias partes del espectro visible del color se ven perjudicadas por este defecto, alterando a diversas tonalidades pertenecientes al mismo (Flück, 2016; 14). Por ende, el encontrar una manera de brindar un soporte a una persona con deuteranomalía abarcaría un conjunto más extenso de situaciones involucradas a esta condición de lo que parece, en las cuales se podría obtener un beneficio extra en comparación a un escenario en el cual no exista semejante provecho.

Por tal motivo, por el conjunto de personas que presentan este problema y debido a la demanda del entorno físico en reconocer visualmente las variables pertinentes a los colores, la presente investigación plantea el estudio y desarrollo de una aplicación informática que asista a aquellas personas que padecen de deuteranomalía, para que mediante de ésta se les devuelva la asociación intrínseca que hay entre coloración y objetos en imágenes digitales; cuestión que a su vez permite habilitar una concordancia entre dos o más individuos acerca de la percepción cromática afectada.

Adicionalmente, el procesamiento de imágenes digitales es un área de investigación que ha sido poco explorada por los estudiantes y profesionales pertenecientes a la Universidad José Antonio Páez, lo que da como resultado que no exista ningún trabajo elaborado para responder a problemáticas relacionados con este tópico. Por esta razón, el estudio y desarrollo de esta investigación podría incentivar el interés y curiosidad por parte de otros, para abordar no solo el procesamiento de imágenes digitales, sino también temas que estén relacionados a este y muchos más de las ciencias de la computación.

En un mismo orden de ideas, cabe mencionar que en referencia a una población más generalizada se otorga la capacidad de obtener una nueva percepción más exacta de los objetos de su ambiente al mismo tiempo que se mejora su calidad de vida, lo

cual permite concientizar a estas personas del verdadero valor de la disposición de un software con este fin, esto debido a que la mayoría de las personas suele conformarse con lidiar con esta deficiencia visual y esto a su vez, por la principal razón de que usualmente hay un desconocimiento de las tecnologías contemporáneas emergentes.

Por otro lado, aun cuando en la actualidad existen varias tecnologías enfocadas a resolver este problema, algunas de ellas no están disponibles a un precio accesible o solo trabajan para resolver ciertas áreas muy específicas, lo cual deja un amplio escenario de técnicas y aplicaciones que todavía no hayan sido contempladas, las cuales posiblemente pueden proporcionar un nuevo panorama de soluciones a problemas que nunca se pensaron.

1.4 Alcance

Se desarrollará un software basado en los métodos y técnicas relativas al área de estudio de procesamiento de imágenes digitales, a través de la librería OpenCV 3.0 bajo el lenguaje de programación Python 3 y, orientado al sistema operativo Linux para la distribución Raspbian, el cual opera sobre un ordenador Raspberry Pi versión 3, modelo B. La interacción con el usuario se proporcionará mediante una interfaz gráfica, elaborada a través del uso de la librería PyGTK. Simultáneamente, la investigación abarcará solamente a las personas con daltonismo del tipo deuteranomalía, centrándose solo en los colores espectrales y sus luminosidades restringidas a los términos claro y oscuro.

CAPÍTULO II
MARCO TEÓRICO

2.1 Antecedentes

Villarroel, M. (2003) egresada de la Universidad Central de Venezuela, realizó una investigación titulada: **Reconocimiento de colores en una imagen digital utilizando máquinas de aprendizaje no supervisado**. El objetivo principal fue emplear máquinas de aprendizaje no supervisado para la formación automática de grupos de colores a partir de las imágenes digitales de un campo de juego de fútbol para robots, con el fin de lograr calibrar en tiempo real, el rango en el cual se encuentran los colores de los objetos presentes en el área de juego para un determinado instante, permitiendo su correcta identificación. Dicha investigación se fundamentó en un diseño no experimental y del tipo campo, bajo la modalidad de proyecto factible.

Se concluyó que los métodos estudiados para este proyecto, K-Medios y Gas Neuronal con crecimiento dinámico, logran resolver el problema de identificación de colores en una imagen digital con un porcentaje de acierto en más de un 98%; de igual manera, se recomendó que deberían mejorar el equilibrio entre la velocidad de procesamiento y la precisión en la clasificación, para así obtener los resultados con mayor rapidez. Dicho trabajo de investigación es de gran ayuda, debido a que aborda un punto central en común: el de reconocer colores en una imagen digital; además, propone el uso de técnicas novedosas que dan pruebas fehacientes de su utilidad para la aplicación propuesta en este proyecto.

Asimismo, Lozano, G. y Orduz, J. (2015) egresados de la Universidad Autónoma del Caribe ubicada en Colombia, realizaron un trabajo de grado para optar por el título de Ingeniero de Mecatronica, titulado: **Diseño de un sistema de visión artificial para la revisión del nivel de llenado de bebidas**

embotelladas. El objetivo principal fue diseñar y desarrollar un sistema para la inspección del nivel de llenado y ubicación de bebidas embotelladas mediante el uso de visión artificial. Tal investigación se fundamentó en un diseño no experimental del tipo campo, de nivel exploratorio y proyectivo bajo la modalidad de proyecto factible; simultáneamente, se basó en un diseño bibliográfico del tipo documental. Se concluyó que el sistema logró identificar el nivel de líquido en las botellas tipo PET de POSTOBON S.A. de 400ml, clasificándolas según cumplan los parámetros de control de calidad establecidos.

De igual forma, se hicieron ciertas recomendaciones tales como: revisar la compatibilidad técnicas entre los componentes; realizar el proceso en un ambiente de luz controlada; trabajar con cámaras de alta resolución, entre otras. La anterior investigación es de utilidad, ya que se investigan y proponen métodos y técnicas orientadas a la segmentación de objetos, así como el uso de herramientas que posiblemente podrían ser contempladas. Aunado a esto, también se plantea el funcionamiento de éstas en un entorno operativo distinto al popularmente usado Windows.

Seguidamente, Juarez, J., Solis, L., Castañeda, R., Ortiz, J. y Gamboa, H. (2014), provenientes de la Universidad Autónoma de Zacatecas ubicada en México, realizaron una investigación titulada: **Algoritmo de procesamiento digital de imágenes para la detección y evaluación de heridas de pie diabético**. El objetivo principal fue convertirse en un primer eslabón para la optimización de los resultados en la evaluación del pie diabético, mediante la introducción de técnicas de procesamiento de imágenes digitales. Esta investigación se fundamentó en un diseño no experimental del tipo campo, de nivel exploratorio y proyectivo. Se concluyó que la aplicación del algoritmo sobre un grupo de imágenes de prueba dio resultados aceptables en la detección de las heridas, así como su tamaño y ubicación.

La investigación previa conlleva un significativo aporte, debido a que en ella se contemplan y utilizan técnicas avanzadas de: preprocesamiento, tales como, la conversión del espacio de color; segmentación, con la transformada Watershed; y

postprocesamiento de imágenes digitales, como por ejemplo, binarización de la imagen. Además, se extraen características mediante el etiquetado de objetos y cálculo de propiedades.

Por su parte, Gonzales, T. (2015) proveniente de la Universidad de Carabobo, realizó un trabajo de ascenso para optar a la categoría de profesora agregada, la cual fue titulada: **Conocimiento sobre la tiflotecnología aplicable a los estudiantes con ausencia total de visión o disfunción visual que poseen los docentes en formación de la mención matemática de la facultad de ciencias de la educación de la Universidad de Carabobo**. El objetivo principal fue determinar el conocimiento sobre la tiflotecnología aplicable a los estudiantes con ausencia total de visión o disfunción visual que poseen los docentes en formación de la mención matemática de la facultad de ciencias de la educación de la Universidad de Carabobo. Dicha investigación se fundamentó tanto en un diseño de campo, no experimental y transeccional, como del tipo descriptiva.

Del trabajo anterior, se concluyó que solo 9,16% de los sujetos conoce las tiflotecnologías y, que una mayoría representada por el 90,83% de estos las desconoce; por lo tanto, se recomendó revisar el pensum de estudios de la Mención Matemática, con la pretensión de incorporar unidades curriculares o contenidos dentro de las ya existentes, que impliquen el estudio, la investigación, puesta en práctica y difusión de estas tecnologías, como coadyuvante al proceso de capacitación que requieren los docentes en formación de la precitada mención. La anterior investigación fue de utilidad, debido a que existe un gran desconocimiento de este tipo de tecnología por parte de profesionales, en especial en el área de educación; por lo tanto, el desarrollo del software este proyecto debe estar orientado a brindar una herramienta intuitiva, de tal forma que contribuya a facilitar su uso para cualquier tipo de usuario.

2.2 Bases Teóricas

2.2.1 El espectro electromagnético y el espectro visible

El espectro electromagnético se puede definir como el rango de todas las radiaciones electromagnéticas posibles, las cuales son medidas mediante una

El Espectro visible

Figura No 1. Fuente: Suárez R. (2012)

2.2.1.1 Colorimetría

Este término se puede definir de manera general como la ciencia que estudia la medida de los colores y que desarrolla métodos para la cuantificación del color, es decir, la obtención de valores numéricos de este último. Con el fin de crear un sistema estandarizado que refleje tales medidas, el sistema CIE fue presentado por la Comisión Internacional de Iluminación, del cual Nave (2000) describe con más detalle de la siguiente manera:

> El sistema CIE caracteriza los colores mediante un parámetro de luminancia Y y dos coordenadas del color x y y, las cuales especifican el punto en el diagrama de cromaticidad. Este sistema ofrece mayor precisión en la medida del color que los sistemas de Munsell y Ostwald, ya que los parámetros están basados en la distribución de potencia espectral (SPD) de la luz emitida desde un objeto colorido y están factorizados por las curvas de sensibilidad, las cuales han sido medidas por el ojo humano.

De manera similar pero con otra perspectiva, para Madore este sistema está compuesto de varias funciones las cuales denomina individualmente "X, Y Z. La función Y se supone que sea precisamente la función de luminosidad. La función Z es... la respuesta a los conos cortos...Y la función X es una convención más o menos arbitraria" (ver Figura 2). Adicionalmente, el mismo autor destaca que:

El sistema CIE X, Y, Z, aunque es usado universalmente como la base fundamental de toda colorimetría moderna, es en verdad muy viejo… y sigue en pie sobre terreno científico dudoso...En ciertas regiones, las funciones son conocidas por ser severamente incorrectas.

Representación 3D del sistema CIE X,Y,Z

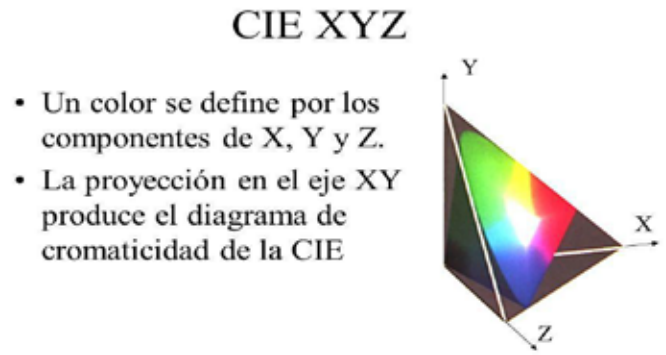


Figura No 2. Fuente: Catalina D. (2014)

2.2.1.2 Modelos del color

Son modelos matemáticos abstractos que permiten representar los colores en forma numérica, utilizando típicamente tres o cuatro valores o componentes cromáticos. A partir esto es que surge lo que se denomina espacio del color, el cual es un sistema de interpretación que tiene como objetivo identificar una combinación particular del modelo y de la función de mapeo. También se puede conceptualizar de una manera más sencilla, describiéndolo como una organización específica de los colores. Para complementar esto, MacEvoy (2005) establece que desde una perspectiva moderna, un modelo del color debe cumplir con los siguientes requerimientos:

> (1) una especificación del color que analice cada luz o superficie de color en una mezcla fundamental de atributos (como los colores "primarios"... o tonalidades únicas); (2) una infraestructura geométrica que localice todos los posibles colores en relación del uno al otro y a los atributos

fundamentales; (3) un identificador de color único o notación de color (actualmente el valor numérico de los tres atributos para la elaboración del color —brillo/claridad, tonalidad y pureza de la tonalidad) para cada posible color, y (4) una definición de modelos físicos, mezclas específicas de luces o pinturas, que recrean la percepción del color medida cuando son vistos dentro de un entorno estándar bajo condiciones de luz estándar.

- **Modelo RGB**

El modelo RGB (ver Figura 3) se basa en la composición del color en términos de la intensidad de los colores primarios de la luz (rojo, verde y azul); a través de la combinación de estos es que se pueden obtener el resto de colores que están representados en el respectivo espacio del color. En referencia a esto, Villarroel (2003) explica que:

> ...se muestra la geometría del modelo de color RGB para la especificación de colores utilizando un sistema de coordenadas cartesianas. El espectro de grises se ubica en la línea que une los vértices blanco y negro, estos colores poseen la misma cantidad de los tres componentes. Este modelo es usado para monitores de color y por la mayor parte de las cámaras de video.

Representación geométrica del modelo RGB

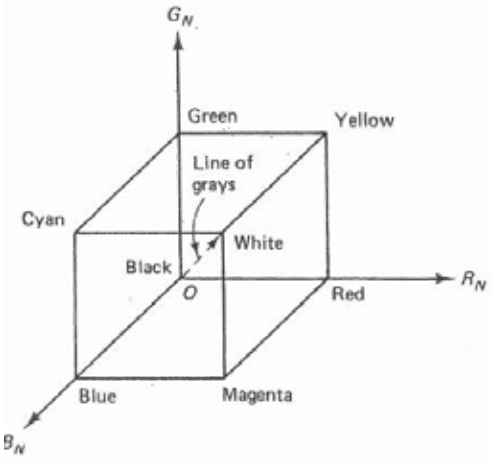


Figura No 3. Fuente: Wang R. (2013)

- **Modelo HSV**

Este modelo puede describirse fácilmente mediante cada una de las letras que componen su nombre. La tonalidad (H) hace referencia a cuál color puro se parece, por ejemplo, todas las tintas, tonos y sombras del color rojo tienen la misma tonalidad. También a ésta última se le describe con un número que especifica la posición correspondiente a un color puro dentro de su diagrama respectivo. La saturación (S) de un color describe que tan blanco o claro es. Un rojo puro está completamente saturado, con una saturación de 1; tintas rojas tienen saturaciones menores a 1; y el blanco tiene una saturación de 0. Consecuentemente, el valor (V) de un color, también llamado claridad, describe que tan negro es.

Con el fin de mostrar una representación gráfica de este modelo, se elaboró un diagrama al cual se le llama el modelo hexaconal único del espacio del color, el cual puede ayudar a visualizar el significado de los parámetros H, S y V (ver Figura 4). Shipman (2012) detalla otros aspectos específicos para dar una mejor explicación de esto:

> El borde externo de la parte más alta del cono es la rueda del color, con todos los colores puros. El parámetro H describe el ángulo alrededor de la rueda. La S (saturación) es cero para cualquier color en el eje del cono; el centro del círculo en la parte más alta es blanco. Un incremento en el valor de S corresponde al movimiento lejos del eje. El V (valor o claridad) es cero para negro. Un incremento en el valor de V corresponde a un movimiento lejos del negro y en dirección a la parte más alta del cono.

Diagrama del modelo del color HSV

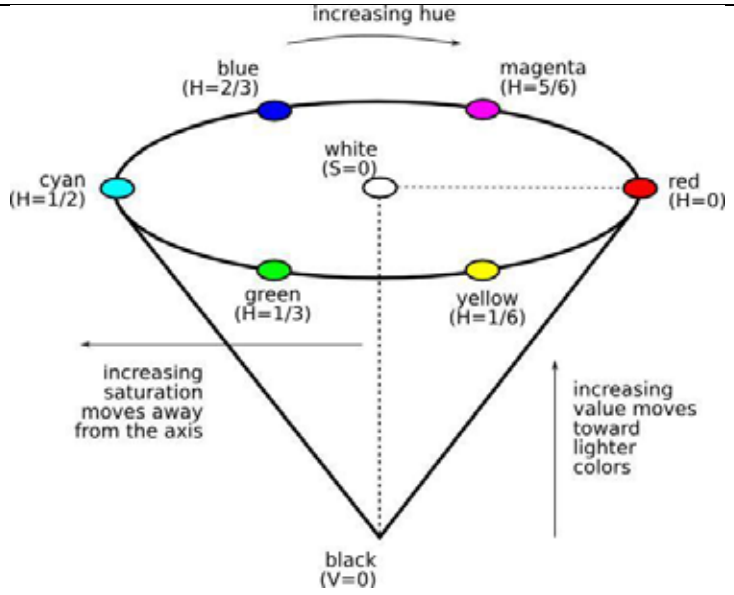


Figura No 4. Fuente: Shipman J. (2012)

- **Modelo CIELAB**

CIELAB es un modelo adoptado por la Commission Internationale de l'Eclairage (CIE) en 1976, el cual se basa en un sistema de color oponente. Esta oposición tiene relación con los descubrimientos a mediados de 1960, a partir de los que se establece que en algún lugar entre el nervio óptico y el cerebro, los estímulos del color retinal son trasladados en distinciones entre lo claro y lo oscuro, rojo y verde y azul y amarillo. En baso a lo anterior, CIELAB indica estos valores con tres ejes: L*, a* y b*. El eje vertical central representa claridad (representada como L*), cuyos valores van desde 0 (negro) hasta 100 (blanco).

Los ejes de color están basados en el hecho de que un color no puede ser ambos verde y rojo o ambos azul y amarillo, debido a que estos colores se oponen el uno al otro. En cada eje los valores van desde positivo hasta negativo. En el eje a-a', los valores positivos indican cantidades de rojo mientras que valores negativos indican cantidades de verde. En el eje b-b', el amarillo es positivo y el azul es negativo. Para ambos ejes, cero es gris neutral: por lo tanto, los valores son

únicamente necesarios para los ejes de colores y para el eje de claridad o escala gris (L*), lo cual es por separado (ver Figura 5). Cabe destacar que estas características hacen que CIELAB sea independiente del dispositivo electrónico que se use, a diferencia del modelo RGB y CMYK.

Representación gráfica del modelo CIELAB

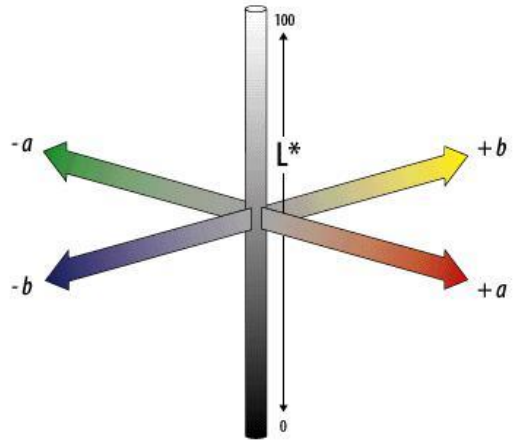


Figura No 5. Fuente: Adobe Systems incorporated (2000)

2.2.1.3 Teorías de la visión del color

Existen dos teorías principales que explican y guían la investigación en el área de la visión del color: la teoría tricromática también conocida por la teoría Young-Helmholtz, y la teoría del proceso oponente. Estas dos teorías son complementarias y explican los procesos que operan en los diferentes niveles del sistema visual humano.

- **Teoría Tricromática**

La existencia de la teoría tricromática proviene de unos estudios realizados para combinar e igualar colores por parte de Thomas Young y Hermann von Helmholtz, quienes llevaron a cabo experimentos con individuos que ajustaban la intensidad relativa de una, dos o hasta tres fuentes de luz de diferentes longitudes de onda, para que el área mezclada resultante igualase un área de prueba adyacente compuesta de una sola longitud de onda. Los resultados demostraron que los

individuos con visión del color normal necesitaron tres diferentes longitudes de onda (i.e., colores primarios) para igualar cualquier otra longitud de onda en el espectro visible (ver Figura 6). En base a esto, Kokotailo & Kline (2002) afirman que:

> Este descubrimiento llevó a la hipótesis de que la visión del color normal está basada en la actividad de tres tipos de receptores, cada uno con un pico de sensibilidad diferente. Consistente con la teoría tricromática, ahora sabemos que el balance general de actividad en los conos S (longitud de onda corta), M (longitud de onda media), y L (longitud de onda larga) determina nuestra percepción del color.

Colores resultantes del experimento de Young & Helmholtz

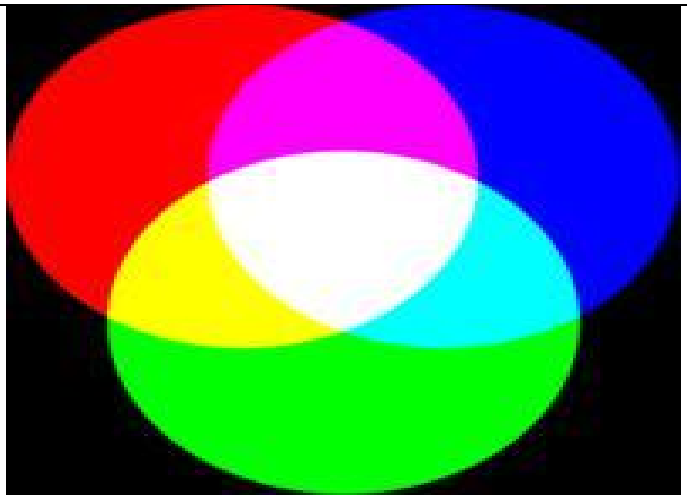


Figura No 6. Fuente: Fuente: Adams D. (2003)

- **Teoría del proceso oponente**

Esta teoría fue inicialmente propuesta por Ewald Hering en 1878, la cual establece que el sistema visual humano interpreta información acerca del color mediante el procesamiento de señales provenientes de los conos y bastones en una manera antagonista. Los tres tipos de conos tienen algo de solapamiento en las longitudes de onda de la luz visible, lo que permite que sea más eficiente por parte del sistema visual el registrar diferencias entre las respuestas de conos, en vez de

cada una de las respuestas individuales de cada tipo de cono. Según Kokotailo & Kline (2002), esta teoría se puede explicar de otra manera peculiar:

> ...La teoría del proceso oponente establece que los… conos están vinculados conjuntamente para formar tres pares de colores opuestos: azul/amarillo, rojo/verde, y blanco/negro. La activación de un miembro del par inhibe la actividad del otro. Consistente con esta teoría, ninguno de los dos miembros de un par pueden ser vistos en un mismo lugar, lo que explica porque no vemos la experiencia de tales colores como "amarillo azulado" o "verde rojizo" (ver Figura 7).

Combinaciones posibles entre los tres tipos de conos basados en la teoría del proceso oponente

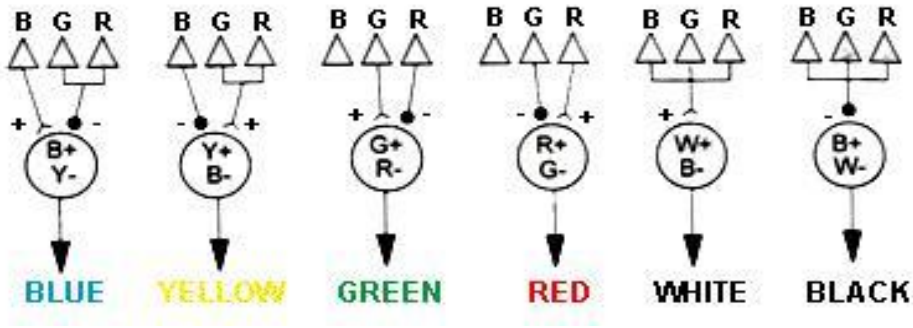


Figura No 7. Fuente: Fuente: Vita M. (2016)

Adicionalmente, estos autores afirman que esta teoría ayuda a explicar algunos tipos de deficiencias de la visión del color, explicando que "las personas con deficiencias dicromáticas son capaces de emparejar un área de prueba usando solo dos colores primarios. Dependiendo del tipo de dicromata que sean, éstos confunden ya sea rojo y verde o azul y amarillo". Por otro lado, los mismos autores destacan el caso especial del color amarillo, para así dar una mejor explicación de cómo funciona esta teoría:

> La teoría del proceso oponente explica como vemos el amarillo aun cuando no hay cono de color amarillo. Esto resulta de las excitaciones e inhibiciones entre los tres tipos de conos. Específicamente, la estimulación simultánea de rojo (conos L) y verde (conos M) es sumada

y por turnos inhibe B+Y-, lo que resulta en la percepción de amarillo. Sin embargo, cuando la luz azul está presente, el cono S es activado, la célula B+Y- recibe una entrada excitatoria y el azul es percibido.

2.2.2 Daltonismo

El daltonismo o popularmente conocido como ceguera del color es una discapacidad visual relacionada con la percepción de los colores, la cual ocasiona una pérdida de discriminación y por lo tanto, una confusión entre algunos de éstos. Su origen proviene de una condición anormal presente en alguno de los tres tipos de conos que son sensibles a ondas electromagnéticas de diferentes longitudes: conos L para los de longitud de onda larga, conos M para aquellos de longitud de onda mediana y conos S para los de longitud de onda corta (ver Figura 8). Estos se definen como pequeñas células muy sensibles a la luz que están contenidas en una membrana interior del ojo denominada retina. Esta condición puede ir desde la ausencia hasta el malfuncionamiento de algunos de los conos.

Curvas de absorción de los fotopigmentos humanos

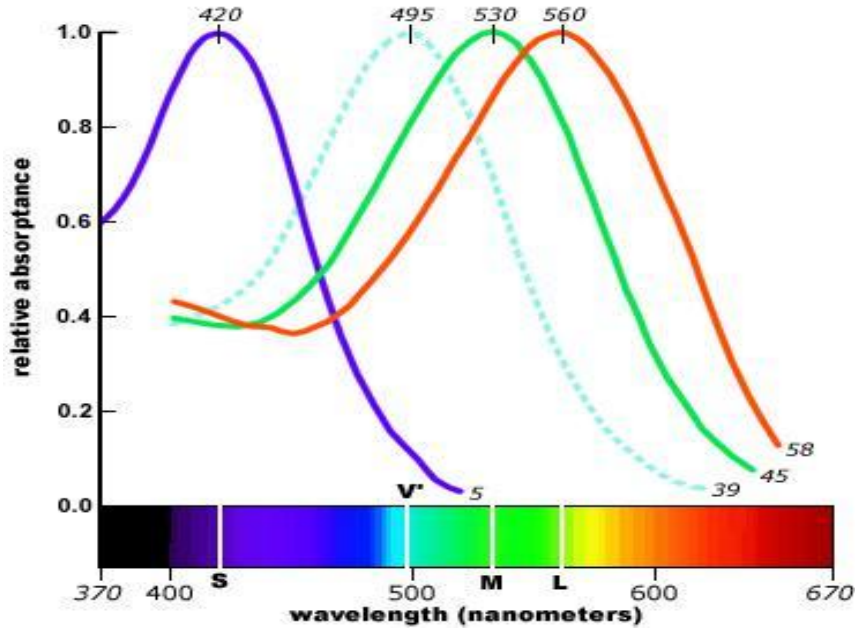


Figura No 8. Fuente: Fuente: MacEvoy B. (2015)

Aunque se le es referenciado frecuentemente como ceguera del color, este apodo es relativo a ciertas áreas distorsionadas correspondientes a la región del espectro electromagnético que es visible para el ojo del ser humano, lo que significa que no necesariamente una persona daltónica se encuentra excluida en su totalidad al reconocimiento de los colores espectrales y no espectrales. En apoyo a esto, el National Eye Institute (NEI, 2015) establece que:

> Existen tres principales tipos de ceguera del color, basados en los defectos del fotopigmento en los tres diferentes tipos de conos que responden a la luz azul, verde, y roja. La ceguera del color rojo-verde es la más común, seguida de la ceguera del color azul-amarillo. Una completa ausencia de la visión del color —ceguera del color total– es rara.

2.2.2.1 Tricromatismo anómalo y Deuteranomalía

Para que un ser humano posea una deficiencia visual del color debe estar presente alguno de las dos condiciones a continuación: la ausencia completa de uno o más de los tres tipos de conos dentro de la retina o, el mal funcionamiento de alguno de ellos estando presentes en su totalidad. Este último caso corresponde a lo que se denomina tricromatismo anómalo, del cual la compañía Color Blind Awareness afirma que:

> La gente con una visión tricromática defectuosa será ciega ante el color a un cierto grado y son conocidos como tricromáticos anómalos. En la gente con esta condición todos sus tres tipos de conos son usados para percibir los colores claros, pero un tipo de cono percibe la luz un poco fuera de alineación, por lo que existen tres diferentes tipos de efectos producidos dependiendo de cuál tipo de cono sea defectuoso.

Como resultado de esto, existen tres vertientes únicas para describir mejor el efecto generado para cada condición, pero solo una de ellas es la más predominante en la población mundial y de la cual el autor anterior se refiere como "...deuteranomalía, la cual es una sensibilidad reducida de la luz verde y es la más común forma de ceguera del color". Aunado a esto, también el mismo autor establece que "los efectos de la visión tricromática anómala puede oscilar desde casi una percepción del color normal

hasta una total ausencia de la percepción del color defectuoso" y en consecuencia, comparte relativamente los mismos defectos que posea una persona con deuteranopia, la cual corresponde a la ausencia por completo de los conos con fotopigmento verde y es un derivado del dicromatismo.

Por otro lado, el previo autor resalta dos realidades importantes a tomar en cuenta referente a los colores de una manera cualitativa: Primero que las "personas con deuteranomalía son colectivamente conocidas como ciegos del color rojo-verde y ellas generalmente tienen dificultad distinguiendo entre rojos, verdes, marrones y naranjas. Ellas también comúnmente confunden diferentes tipos de tonalidades azul y púrpura", lo que revela que es un par de colores el que se ve afectado, por lo tanto, los demás que surjan de la combinaciones de estos dos también estarán involucrados dentro de esta anomalía. Segundo, también establece que:

> Alrededor de la mitad de las personas con tricromatismo anómalo verán el mundo en una manera similar a aquellos con dicromatismo, pero sus habilidades para percibir los colores mejorarán con abundante luz y se deteriorara con escasa luz. Frecuentemente su percepción del color puede ser tan pobre como la de aquellos con dicromatismo.

2.2.2.2 Representación de la visión defectuosa del color en el diagrama de cromaticidad CIE (1931)

Desde un punto de vista técnico, es posible representar cuantitativamente la confusión entre varios colores que se produce en la percepción visual de los dicrómatas. Esto se lleva a cabo mediante la representación de lo que se denomina líneas isocromáticas o de confusión. Para definir a éstas últimas, el National Research Council (US) Committee on Vision (1981) establece que "para los dicrómatas, algunas líneas de mezcla en el diagrama de cromaticidad x,y representan una serie de colores que no pueden ser discriminados uno del otro...". De igual forma, este autor proporciona una descripción aún más específica para dar a entender este punto:

> Al asociar las coordenadas en el diagrama de cromaticidad x,y con las series normales de apariencias de los colores para los tricromáticos

anómalos, podemos describir aproximadamente cuáles colores son confundidos por los dos tipos de dicrómatas. Para ambos protanopes y deuteranopes, una línea isocromática yace en la ubicación del espectro de los 540 nanómetros hasta los 700 nanómetros.

El diagrama de cromaticidad CIE (x,y)

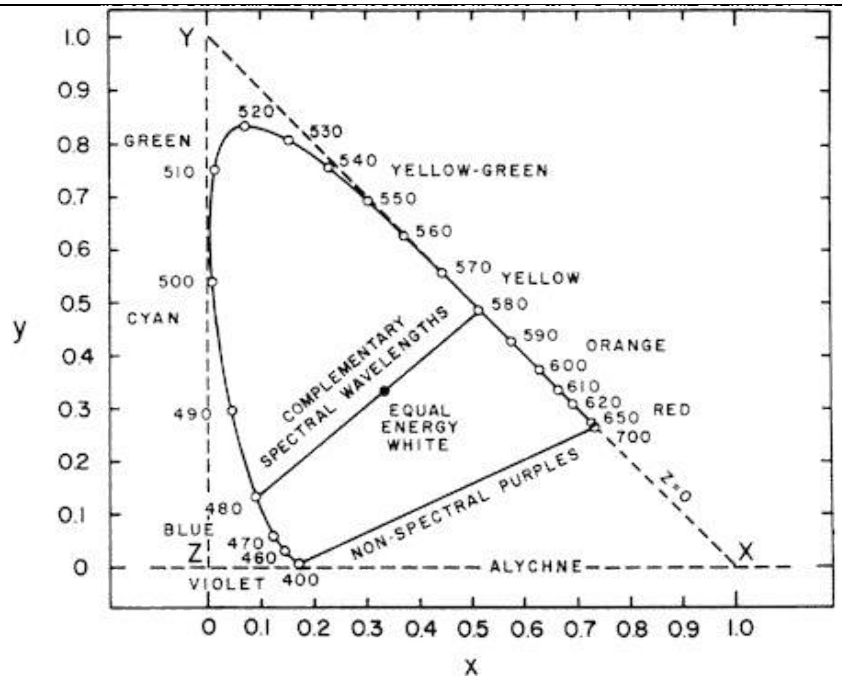


Figura No 9. Fuente: Fuente: National Research Council (US) Committee on Vision (1981)

Líneas de confusión en el diagrama CIE para los protanopes (izquierda) y para los deuteranopes (derecha)

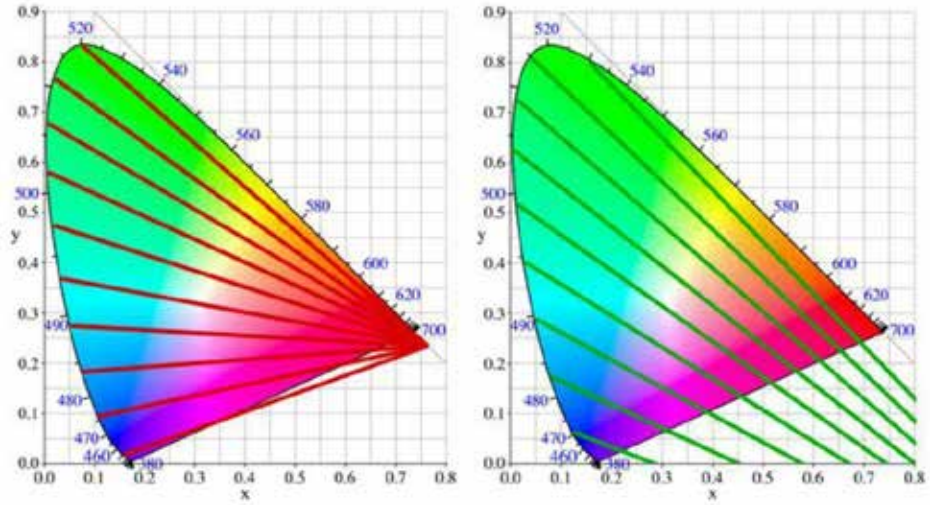


Figura No 10. Fuente: Colblindor (2009)

Además de estas aseveraciones, el anterior autor habla acerca de un punto especial en el diagrama CIE y una línea, cuyas especificaciones sirven de soporte para exponer con una mayor exactitud la percepción visual de los colores dentro de la región gráfica por parte de un dicromata (ver Figura 11 y 12). Tal supuesto se presenta a continuación:

> Una línea de confusión pasa a través de una igualdad de energía blanca. Esto indica precisamente cuales cromaticidades y en particular cual luz monocromática (punto neutral) puede ser completamente emparejada con la igualdad de energía blanca. De las cromaticidades representadas por encima de esta línea se dice que parecen "amarillas" con un incremento de saturación; aquellas por debajo de la línea son "azules", también con el incremento de saturación. Por lo tanto, del espectro visible se dice que parece como sombras de amarillos y azules a los observadores con protanopia y deuteranopia.

Punto neutral (N) y punto de igualdad de energía (W) para los protanopes

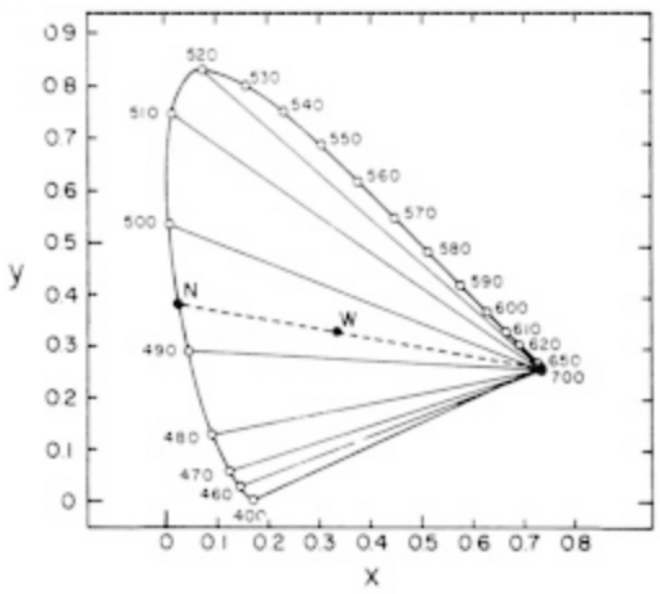


Figura No 11. Fuente: National Research Council (US) Committee on Vision (1981)

Punto neutral (N) y punto de igualdad de energía (W) para los deuteranopes

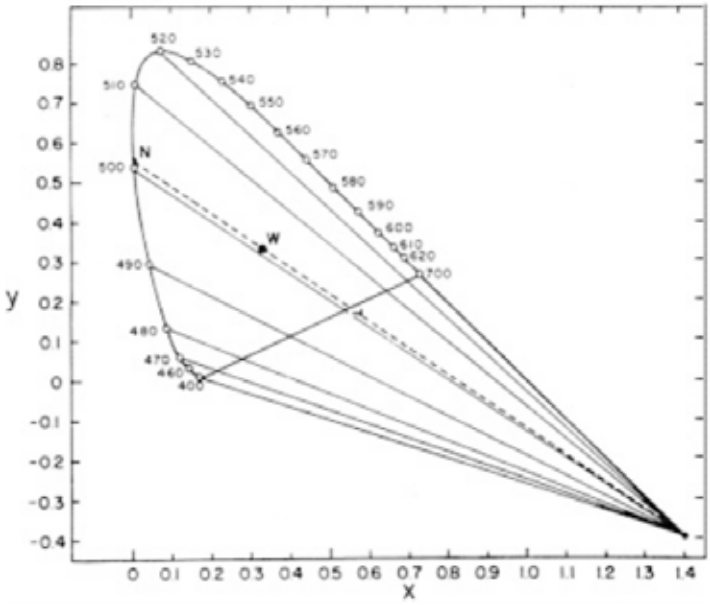


Figura No 12. Fuente: National Research Council (US) Committee on Vision (1981)

Por otro lado, este rango de colores relacionados con los dos tipos de dicrómatas ya expuestos, también son aplicables para los otros dos tipos de tricrómatas anómalos que presentan en común la afección de un mismo cono (ya sea de mediana o de larga longitud de onda), con respecto a lo que Farnsworth (1943) afirma que "de los tricromáticos protanomalía y deuteranomalía se dice que hacen confusión de los colores que son cualitativamente similares a, aunque con menos severidad que, los dicromáticos correspondientes".

2.2.3 Tecnologías de apoyo

En la actualidad, las tecnologías brindan una importante fuente de recursos para el acceso a la información y la comunicación, pero en el caso de algunas personas con discapacidades, surgen dificultades para su utilización. Además, para que la computadora no se convierta en una nueva barrera para el aprendizaje y la participación, es necesario establecer soluciones para adaptarla al usuario, de manera que sea capaz de utilizar, dentro de lo posible, todos los recursos y programas de manera autónoma. Es a partir de estas necesidades que se desarrolla el campo de las Tecnologías de Apoyo o Tecnologías Adaptativas.

En referencia a esto, Abadin y Álvarez (2014, 6) mencionan que las tecnologías de apoyo "…hacen referencia a cualquier producto (incluyendo dispositivos, equipos, instrumentos y software)…" cuya función sea "…proteger, apoyar, entrenar, medir o sustituir funciones/estructuras corporales y actividades o prevenir deficiencias, limitaciones en la actividad o restricciones en la participación.".

2.2.3.1 Tiflotecnología

La tiflotecnología, cuyo nombre proviene del griego tiflos que significa ciego, es el conjunto de técnicas, conocimientos y recursos encaminados a procurar, diseñar y adaptar los medios convenientes en favor a las personas con ceguera o deficiencia

visual, para la correcta utilización de las tecnologías de la información y la comunicación (TIC). Para Martinez (2012), la tiflotecnología se refiere a:

> …todas las tecnologías electrónicas (hardware y software) que permitan el acceso a la información y a la comunicación de estas personas con el objeto de facilitar su autonomía personal. Sin la tiflotecnología, este colectivo no podría acceder a una gran cantidad de ámbitos cotidianos, así como a determinada información o la posibilidad de comunicarse e interaccionar con otras personas de su entorno.

En determinados casos y para determinadas personas con discapacidad visual, la tecnología se convierte en un elemento imprescindible, donde las TIC en la mayoría de las veces constituyen un elemento fundamental para la inclusión educativa, social, afectiva, cultural, formativa o laboral, entre otros ámbitos. Existen dos tipos de tiflotecnologías: las específicas, que serían los dispositivos tecnológicos creados para el uso exclusivo de invidentes, y las adaptadas que se centran en el hardware y el software diseñados para que un ciego o deficiente visual pueda utilizar un equipo estándar. En base a lo anterior, la misma autora establece que:

> La adaptación de estos dispositivos facilita la orientación, la movilidad y contribuyen a la autonomía de las personas ciegas. La tiflotecnología abre el horizonte de las personas ciegas, de tal forma que mejora su integración a las exigencias de la vida cotidiana.

2.2.4 Software de aplicación

El software de aplicación, mayormente conocido solamente como aplicación, es simplemente un programa informático creado para llevar a cabo o facilitar una tarea en un dispositivo electrónico, éste permite a los usuarios llevar a cabo una o varias actividades específicas. En el mismo orden de ideas, para el Instituto Tecnológico de Celaya destacan que es en las aplicaciones donde “...se aprecia en forma más clara la ayuda que puede suponer un computador en las actividades humanas, ya que la máquina se convierte en un auxiliar del hombre...”, de este modo se puede mencionar que el uso de aplicaciones facilita la interacción de los usuarios con un computador.

Es importante tomar en cuenta que todas las aplicaciones son considerados programas informáticos pero está consideración no es conmutativa, ya que en el mundo del software existe una multitud de programas destinados a cumplir otras funciones; no se califica a un sistema operativo como una aplicación ya que este tiene un propósito más general, por mencionar un ejemplo.

2.2.4.1 Interfaz gráfica de usuario

Todo software de aplicación cumple con la característica de que cuenta con una interfaz gráfica, y en este caso en particular se destacan las interfaces gráficas de usuarios, conocidas en el mundo de la informática como GUI, acrónimo en inglés de Graphical User Interface, su función principal consiste en facilitar un entorno visual sencillo que permita la comunicación con el sistema operativo de un ordenador. La interfaz gráfica aparece como evolución de las interfaces de líneas de comando (CLI), que se utilizaban en los primeros sistemas operativos.

Carbonell (2013) destaca que "...en definitiva GUI es una interfaz de usuario en la que una persona interactúa con la información digital a través de un entorno gráfico de simulación. Este sistema de interacción con los datos se denomina WYSIWYG (What you see is what you get, 'lo que ves es lo que obtienes')..."; cabe mencionar que la interfaz que se le presenta al usuario está compuesta por ventanas, cuadros de diálogo, barras de herramientas, botones, listas desplegables y otros elementos, denominados widgets, los cuales producirán eventos que se traducen en instrucciones dentro del software.

Ahora bien, al momento de crear una interfaz de usuario se deben tener en cuenta ciertos aspectos para que la misma no resulte conflictiva con el usuario y éste se oponga menos al uso del software. Por eso una interfaz gráfica de usuario debe ser autónoma, tanto el computador, la interfaz gráfica como el espacio de trabajo deben estar a disposición del usuario, dejando un ambiente flexible para facilitar el aprendizaje del uso del software; debe poseer un diseño ergonómico estableciendo menús, barras de acciones e iconos de fácil acceso; y otros dos características

importantes son la tipografía y el tratamiento del color ya que hay que prestar especial importancia a la hora de establecer una buena interfaz, poniendo especial cuidado en el diseño de las formas y la coherencia interna entre ellas, para mejorar la accesibilidad y crear una interfaz más amigable.

- **GTK**

Hoy día existe un gran conjunto de herramientas y lenguajes para el desarrollo de interfaces de usuario, sin embargo en el presente proyecto se quiere hacer mención acerca de GTK, la cual es una biblioteca que forma parte del grupo GTK+ y se utiliza para desarrollar interfaces gráficas de usuario, principalmente para los entornos gráficos: GNOME, XFCE y ROX, aunque también se puede usar en el escritorio de Windows, MacOS y otros. Posee una licencia LGPL, lo cual permite que mediante GTK se puedan desarrollar programas con licencias abiertas, gratuitas, libres y hasta licencias comerciales no libres sin mayores problemas.

GTK es esencialmente una interfaz para la programación de aplicaciones orientadas a objetos (API). Aunque está completamente escrito en C, también está implementado haciendo uso de la idea de clases y de funciones respuesta o de callback (punteros o funciones), lo que proporciona una gran variedad de lenguajes de programación con los cuales se puede usar GTK, y aunque no en todos está disponible en su última versión, destacan entre los más usados lenguajes como: C++, C#, Java y Python.

2.2.5 Raspberry Pi

El uso de cualquier software siempre estará sujeto a que sea ejecutado en algún dispositivo que funcione como hardware. Para efectos de la investigación se hará uso de un pequeño dispositivo electrónico llamado Raspberry Pi; este dispositivo se clasifica como ordenador de placa reducida, ya que tiene el tamaño aproximado al de una tarjeta de crédito. El dispositivo fue ideado en 2006 pero no fue lanzado al mercado hasta febrero de 2012, fue desarrollado por un grupo de la Universidad de

Cambridge con la misión de fomentar la enseñanza de las ciencias de la computación a los jóvenes.

En cuanto a la estructura física del Raspberry Pi, en su modelo B versión 3, está compuesto a rasgos generales por un procesador ARM Cortex A53 de cuatro núcleos con 1.2GHz de 64 bits, una GPU Broadcom VideoCore IV que es capaz de reproducir vídeos en 1080p a 30fps, también posee una memoria RAM de 512MB, salidas de audio y video con puerto HDMI, dos puertos USB, un puerto Ethernet, el consumo energético es de 700mA y 3,5W y se alimenta a través de un puerto micro USB con una fuente de 5V (ver Figura 13).

Raspberry Pi modelo B versión 3

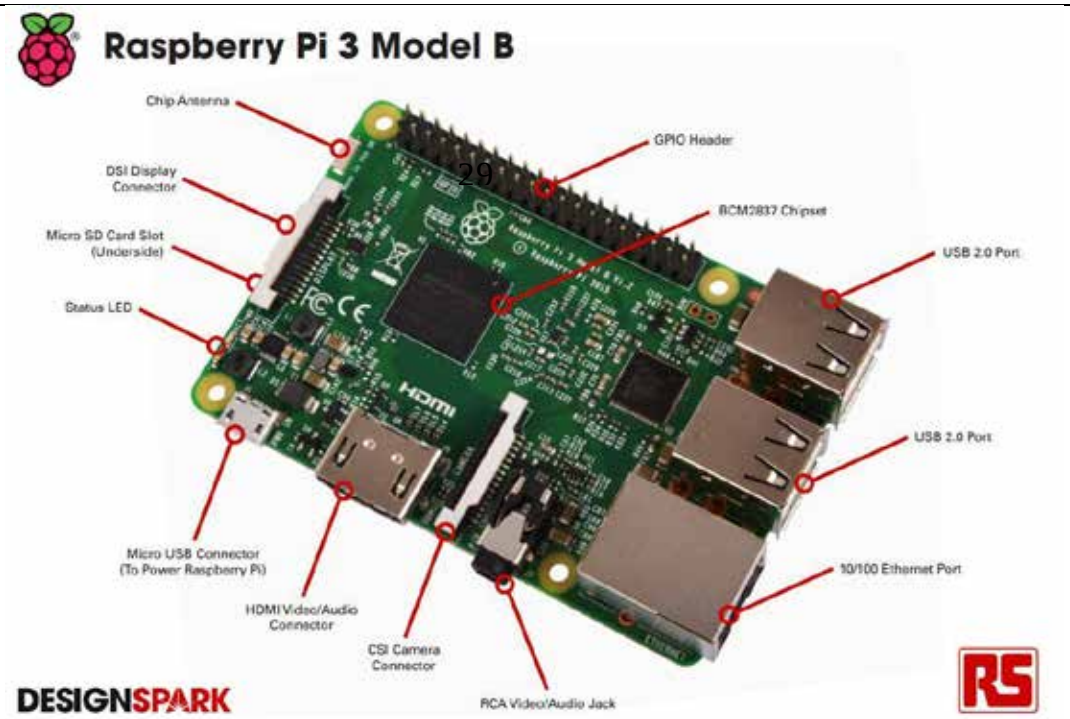


Figura No 13. Fuente: Smith (2016)

El software de arranque del Raspberry Pi puede variar ya que al ser un computador en sí, puede soportar diferentes sistemas operativos, entre ellos podemos mencionar algunos como: Android, hasta la versión 4.0; Arch Linux ARM; Firefox OS; Gentoo Linux; Google Chromium OS; Kali Linux; Pidora; QtonPi; Raspbian, el cual será utilizado durante la realización de este proyecto; entre otros.

- **Cámara Raspberry Pi v2**

Uno de los componentes adicionales que se le puede agregar al Raspberry Pi es una placa que cuenta con una cámara de alta definición (HD), la cual se conecta a cualquier dispositivo Raspberry Pi para crear fotografías y videos HD. Utiliza el sensor de imagen IMX219PQ de Sony que ofrece imágenes de vídeo de alta velocidad y alta sensibilidad. El módulo de la cámara del Raspberry dispone de funciones de control automático como el control de exposición, el balance de blancos y la detección de luminancia.

La cámara se conecta al dispositivo a través de un cable plano de 15 cm fijado a las ranuras del módulo directamente en el puerto de interfaz serie de la cámara (ver Figura 14). Una vez conectado, se puede acceder a la placa de cámara a través de la capa de abstracción multimedia (MMAL) o el vídeo para API de Linux (V4L). También hay bibliotecas como Picamera Python y muchas otras para el manejo de la cámara.

Raspberry Pi Camera V2

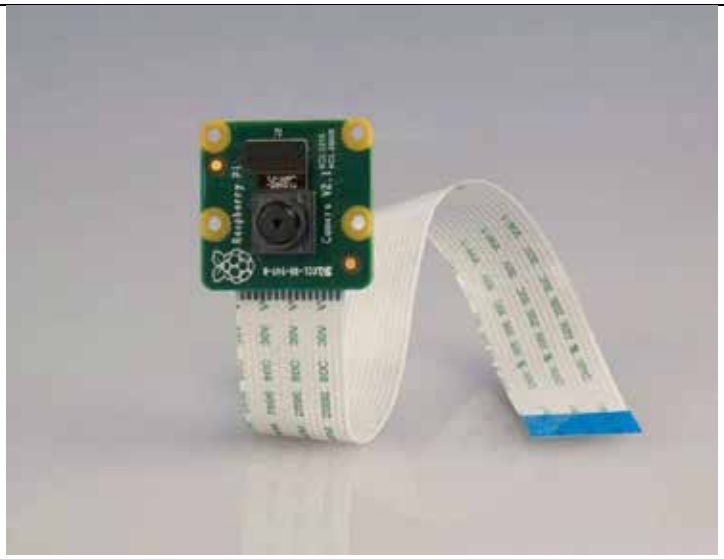


Figura No 14. Fuente: Michelone (2016)

Entre las características de este componente, Michelone (2016) destaca el hecho de que "...posee 8 megapíxeles de resolución, lo cual permite tener imágenes de alrededor de 2280 x 2464 píxeles…", también que puede "...trabajar como cámara de alta definición (HD) a 1080p y 30 cuadros por segundo...", o "...60 cuadros por segundo a 720p…", además del sensor IMX219 de Sony "...el cual es mucho mejor en calidad de imagen, fidelidad de color y desempeño cuando hay poca luz que las versiones anteriores".

2.2.6 Metodologías ágiles

El desarrollo de software no es una tarea fácil y la prueba de ello es que existen numerosas propuestas metodológicas, las cuales inciden en distintas dimensiones del proceso de desarrollo. Por una parte tenemos aquellas propuestas más tradicionales que se centran especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, y las herramientas y notaciones que se usarán. Estas propuestas han demostrado ser efectivas y

necesarias en un gran número de proyectos, pero también han presentado problemas en otros muchos.

Una posible mejora es incluir en los procesos de desarrollo más actividades, más artefactos y más restricciones, basándose en los puntos débiles detectados. Sin embargo, el resultado final sería un proceso de desarrollo más complejo que puede incluso limitar la propia habilidad del equipo para llevar a cabo el proyecto. Otra aproximación es centrarse en diferentes dimensiones, como por ejemplo, el factor humano o el producto software. Esta es la filosofía de las metodologías ágiles, las cuales dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software.

El mundo del desarrollo, para bien o para mal, ha evolucionado desde un modelo en el que se planificaba y estructuraba minuciosamente todas las fases a un modelo en el que el desarrollo debe ser lo más rápido y eficiente posible, lo que ha provocado que las metodologías tradicionales de desarrollo de software han quedado obsoletas en determinados sectores, en los que la propia demanda de los usuarios es más rápida que la capacidad de producción de los desarrolladores. Existen muchos métodos de desarrollo ágil, son especialmente útiles para sistemas en los que los requisitos cambian rápidamente durante el proceso de desarrollo, la mayoría de ellos minimiza riesgos desarrollando software en cortos lapsos de tiempo.

2.2.6.1 AUP

Una vez dicho esto, cabe destacar que para el desarrollo de la investigación se hará uso de la metodología de Proceso Unificado Ágil, también conocida por su nombre en inglés Agile Unified Process (AUP). Ésta metodología fue desarrollada en septiembre de 2005 por Scott Ambler y es una versión simplificada del Proceso Racional Unificado (RUP); describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP.

El proceso unificado ágil se preocupa especialmente de la gestión de riesgos. Propone que aquellos elementos con alto riesgo tengan prioridad en el proceso de desarrollo y sean abordados en etapas tempranas del mismo. Para ello, se crean y mantienen listas identificando los riesgos desde etapas iniciales del proyecto. Especialmente relevante en este sentido es el desarrollo de prototipos ejecutables durante la fase de elaboración del producto, donde se demuestre la validez de la arquitectura para los requisitos clave del producto y de los cuales se determinan los riesgos técnicos. Así como también sucede en el proceso unificado racional (RUP), en el proceso unificado ágil se establecen cuatro fases que transcurren simultáneamente, tal como lo muestra la gráfica de la arquitectura de la metodología (ver Figura 15), donde se muestran los flujos de trabajo que agrupan de forma lógica las actividades a realizar en cada una de las siguientes etapas:

- Origen: en esta fase, el objetivo es lograr obtener una comprensión común entre el cliente y el equipo de desarrollo, y de este modo poder identificar el alcance inicial del proyecto, una arquitectura potencial para el sistema, y obtener la financiación inicial del proyecto y la aceptación de las partes interesadas.
- Elaboración: la meta de esta etapa es que el equipo de desarrollo profundice en la comprensión de los requisitos del sistema y en validar la arquitectura y realizar las mejoras pertinentes.
- Construcción: Durante la fase de construcción el sistema es desarrollado y probado en el ambiente de desarrollo. La finalidad de esta fase es construir un software funcional en una base regular e incremental, la cual cumpla con las necesidades de prioridad más alta de los involucrados de su proyecto.
- Transición: en esta fase, el sistema se lleva a los entornos de preproducción donde se somete a pruebas de validación y aceptación, con el objetivo de validar y finalmente desplegar el sistema en el ambiente de producción.

Arquitectura de la metodología AUP

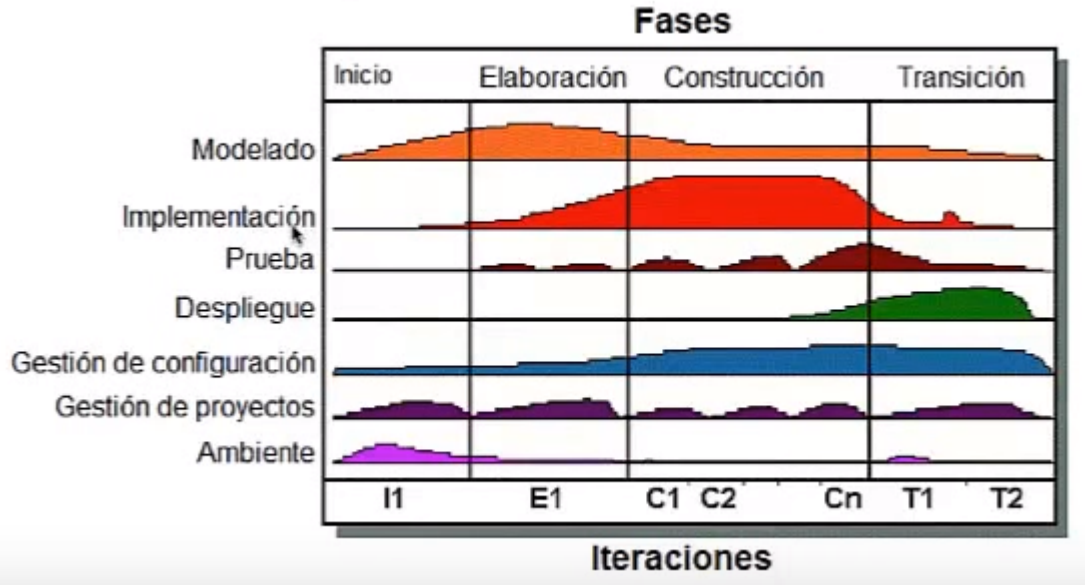


Figura No 15. Fuente: Ambler S. (2005)

2.3 Bases Legales

Estas constituyen un conjunto de leyes y normas que establecen el basamento jurídico que sustenta una investigación. Según Palella y Martins (2012) "La fundamentación legal o bases legales se refiere a la normativa jurídica que sustenta el estudio. Desde la Carta Magna, las Leyes Orgánicas, las resoluciones, decretos, entre otros." (p. 63). Para efectos del presente trabajo, las bases legales están representadas en la Constitución de la República Bolivariana de Venezuela, la Ley Orgánica de Ciencia, Tecnología e Innovación, el Reglamento Parcial de la Ley Orgánica de Ciencia Tecnología e Innovación y, el Decreto Nº 825.

Constitución de la República Bolivariana de Venezuela (1999)

Artículo 108. Los medios de comunicación social, públicos y privados, deben contribuir a la formación ciudadana. El Estado garantizará servicios públicos de radio, televisión y redes de bibliotecas y de informática, con el fin de permitir el acceso universal a la información. Los centros educativos deben incorporar el conocimiento y aplicación de las nuevas tecnologías, de sus innovaciones, según los requisitos que establezca la ley.

Artículo 110. El Estado reconocerá el interés público de la ciencia, la tecnología, el conocimiento, la innovación y sus aplicaciones y los servicios de información necesarios por ser instrumentos fundamentales para el desarrollo económico, social y político del país, así como para la seguridad y soberanía nacional. Para el fomento y desarrollo de esas actividades, el Estado destinará recursos suficientes y creará el sistema nacional de ciencia y tecnología de acuerdo con la ley. El sector privado deberá aportar recursos para los mismos. El Estado garantizará el cumplimiento de los principios éticos y legales que deben regir las actividades de investigación científica, humanística y tecnológica. La ley determinará los modos y medios para dar cumplimiento a esta garantía. El uso de la tecnología de la información y comunicación en provecho del desarrollo educativo y progreso social y económico de la de la sociedad, está plenamente justificado en el marco constitucional, por lo que todo programa o propuesta a mejorar y capacitar al personal docente y la formación integral de la población, a través de los medios tecnológicos serán respaldado constitucionalmente.

Ley Orgánica de Ciencia, Tecnología e Innovación (2006)

Artículo 1. La presente Ley tiene por objeto desarrollar los principios orientadores que en materia de ciencia, tecnología e innovación y sus aplicaciones, establece la Constitución de la República Bolivariana de Venezuela, organizar el Sistema Nacional de Ciencia, Tecnología e Innovación, definir los lineamientos que orientarán las políticas y estrategias para la actividad científica, tecnológica, de innovación y sus aplicaciones, con la implantación de mecanismos institucionales y operativos para la promoción, estímulo y fomento de la investigación científica, la apropiación social del conocimiento y la transferencia e innovación tecnológica, a fin de fomentar la capacidad para la generación, uso y circulación del conocimiento y de impulsar el desarrollo nacional.

Artículo 18. La autoridad nacional con competencia en materia de ciencia, tecnología, innovación y sus aplicaciones, ejercerá la dirección en el área de tecnologías de información. En tal sentido, deberá:

1. Establecer políticas sobre la generación de contenidos en la red, respetando la diversidad, así como el carácter multiétnico y pluricultural de nuestra sociedad.

2. Resguardar la inviolabilidad del carácter confidencial de los datos electrónicos obtenidos en el ejercicio de las funciones de los órganos y entes públicos.

3. Democratizar el acceso a las tecnologías de información. Las oportunidades de crecimiento tecnológico y apropiación del conocimiento científico serán

impulsadas por esta ley, de manera que la generación de contenidos en la red y su acceso no tendrá limitaciones.

Reglamento Parcial de la Ley Orgánica de Ciencia Tecnología e Innovación. Referido a los aportes e Inversión (2010)

Artículo 3. Actividad de Innovación:

Es el conocimiento, procesamiento, aplicabilidad o materialización de una idea con un componente de nivel inventivo o desarrollada durante el desempeño de actividades de investigación, que va encaminada a dar como resultado un bien, proceso o producto nuevo o una mejora de lo existente, que pueden ser desarrollados o utilizados en la industria, en el comercio o en un nuevo enfoque de un servicio social.

Artículo 4. Formación de Talento Humano:

Son los procesos cognoscitivos o educativos en las diferentes modalidades orientadas a la formación, actualización o capacitación de personas, encaminados al desarrollo de actividades de ciencia, tecnología, innovación, gestión o aplicación del conocimiento. Se reconocerá como aporte e inversión en la actividad de capacitación de talento humano, aquella dirigida a estudios de Alto Nivel, entendidos estos como aquellos superiores al bachillerato.

Artículo 5. Transferencia de Tecnología:

Proceso e interrelación que se establece entre un sujeto, persona o empresa que posee la tecnología o los conocimientos para producir, utilizar o manejar un bien, negocio, producto o servicio y que traslada, intercambia, entrega, vende o negocia a otra persona, sujeto o empresa, dichos conocimientos, procedimientos o formas de hacer, para su captación, aplicación, producción y aprovechamiento por el entorno social y económico del país, procurando la apropiación del conocimiento por parte de la colectividad.

Artículo 6. Laboratorio:

Unidad organizada para la actividad científica y tecnológica de carácter experimental, computacional o de procesamiento de datos o cualquier otra forma que indique el desarrollo de investigaciones, pruebas, ensayos, requeridas o necesarias en el campo de las actividades científicas, tecnológicas o de innovación. Quedan incluidos en estos artículos, todas las instituciones del sector académico vinculadas

con formación y desarrollo de talento y de investigación que coadyuven en el fortalecimiento de las áreas de Ciencia y Tecnología.

Decreto Nº 825 (Gaceta Oficial Nº 36.955. 22 de mayo de 2000)

La Constitución reconoce como de interés público la ciencia, la tecnología, el conocimiento, la innovación y sus aspiraciones y los servicios de información, a los fines de lograr el desarrollo económico, social y político del país, y que el Ejecutivo Nacional a través del Ministerio de Ciencia y Tecnología, debe velar por el cumplimiento del mencionado precepto constitucional.

Artículo 1. Se declara el acceso y el uso de Internet como política prioritaria para el desarrollo cultural, económico, social y político de la República Bolivariana de Venezuela.

Artículo 5. El Ministerio de Educación, Cultura y Deportes dictará las directrices tendentes a instruir sobre el uso de Internet, el comercio electrónico, la interrelación y la sociedad del conocimiento. Para la correcta implementación de lo indicado, deberán incluirse estos temas en los planes de mejoramiento profesional del magisterio.

Artículo 8. En un plazo no mayor de tres (3) años, el cincuenta por ciento (50%) de los programas educativos de educación básica y diversificada deberán estar disponibles en formatos de Internet, de manera tal que permitan el aprovechamiento de las facilidades interactivas, todo ello previa coordinación del Ministerio de Educación, Cultura y Deportes. Los artículos y contenidos descritos dan sustento a la propuesta presentada, por cuanto están acorde con los lineamientos legales para el acceso y uso de internet y de la plataforma tecnológica para el desarrollo de contenidos educativos, previstos en la legislación venezolana, indispensables para el progreso cultural, económico, social y político del país.

2.4 Definición de Términos Básicos

- **Brillo o claridad:** Se refiere a la cantidad de luz que emite una fuente o refleja un objeto. En el aspecto físico la claridad va asociada a la luminancia que presenta dicho objeto.
- **Color espectral:** Un color espectral es aquel que se puede emparejar con una única longitud de onda presente en el espectro luminoso. Los principales

colores espectrales son los colores del arcoíris que tienen nombre: Violeta, azul, verde, amarillo, naranja y rojo.

- **Color no espectral:** Es cualquier color que no provenga del espectro visible, es decir, que no pueda ser producido por una sola onda electromagnética perteneciente a este espectro.
- **Colores primarios:** Son aquellos colores que no pueden obtenerse mediante la mezcla de ningún otro por lo que se consideran absolutos y únicos. Generalmente, el grupo está conformado por tres de éstos.
- **Cromático(a):** Se dice de aquello que es perteneciente o relativo a los colores.
- **Distribución de potencia espectral:** Es aquella que contiene toda la data física básica acerca de la luz y sirve de punto de inicio paras el análisis cuantitativo del color.
- **Fotopigmento:** Es un pigmento que sufre un cambio físico o químico bajo la acción de la luz.
- **Longitud de onda:** Es la distancia que recorre el pulso (onda originada por un impulso único) mientras un punto realiza una oscilación completa.
- **Luminancia:** Es la cantidad de energía luminosa (luz) emitida o reflejada por una superficie en el rango de longitudes de onda del espectro visual.
- **Luminosidad:** Es la medida estándar de la respuesta del ojo a la luz monocromática de diversas longitudes de onda.
- **Onda:** Es una perturbación que se propaga desde el punto en que se produjo hacia el medio que rodea ese punto.
- **Ordenador de placa reducida:** Es una computadora completa en un sólo circuito. Su diseño se centra en agregar todos los componentes necesarios de un computador funcional en una sola tarjeta, la cual suele ser de tamaño reducido y tiene todo lo que necesita en la placa base.

- **Python:** Es un lenguaje de programación poderoso y fácil de aprender que cuenta con estructuras de datos eficientes y de alto nivel, además de un enfoque simple pero efectivo a la programación orientada a objetos.
- **Radiación electromagnética:** Es la que está formada por la combinación de campos eléctricos y magnéticos, que se propagan a través del espacio en forma de ondas portadoras de energía.
- **Raspbian:** Es una distribución del sistema operativo Linux basada en Debian Jessie desarrollada para el ordenador de placa reducida Raspberry Pi.
- **Software:** Es un conjunto de instrucciones o programas que le permiten al computador realizar tareas, este es de gran importancia ya que representa la interacción entre el usuario y el computador.
- **Visión del color:** Es la habilidad para hacer discriminaciones basadas en la composición de longitud de onda de la luz independiente de su intensidad. Es usada para determinar la ubicación y forma de objetos, así como su identidad y características.

CAPÍTULO III
MARCO METODOLÓGICO

3.1 Tipo de investigación

De acuerdo a Palella y Martins (2012), el tipo de investigación se refiere a "...la clase de estudio que se va a realizar. Orienta sobre la finalidad general del estudio y sobre la manera de escoger las informaciones o datos necesarios" (p. 88). Por un lado, entre los diferentes tipos de investigación existentes se destaca la de campo, de la cual los autores previos afirman que:

> Consiste en la recolección de datos directamente de la realidad donde ocurren los hechos, sin manipular o controlar variables… El investigador no manipula variables debido a que esto hace perder el ambiente de naturalidad en el cual se manifiesta y desenvuelve el hecho (p. 88).

A partir de esto, se determinó que la presente investigación es de campo, ya que se indaga sobre los efectos de la interrelación entre los diferentes tipos de variables en el lugar de los hechos, como por ejemplo, la percepción cromática de las personas con deuteranomalía y las diferentes tonalidades de los colores. Adicionalmente, el actual estudio está sometido a la modalidad de proyecto factible, ya que el mismo cumple con el propósito de utilización inmediata; respecto a lo anterior, el Manual de Tesis de Grado y Especialización y Maestría y Tesis Doctorales de la Universidad Pedagógica Experimental Libertador (2006) menciona que:

> Consiste en la investigación, elaboración y desarrollo de un modelo operativo viable para solucionar problemas, requerimientos necesidades de organizaciones o grupos sociales que pueden referirse a la formulación de políticas, programas, tecnologías, métodos o procesos. El proyecto debe tener el apoyo de una investigación de tipo documental y de campo, o un diseño que incluya ambas modalidades. (p. 13)

3.2 Diseño de la investigación

Según Palella y Martins (2012) el diseño se refiere a "...la estrategia que adopta el investigador para responder al problema, dificultad o inconveniente planteado en el estudio. Para fines didácticos, se clasifican en diseño experimental, diseño no experimental y diseño bibliográfico" (p. 86). En cuanto al no experimental, los autores ya mencionados lo definen de la siguiente manera:

> Es el que se realiza sin manipular en forma deliberada ninguna variable. El investigador no sustituye intencionalmente las variables independientes. Se observan los hechos tal y como se presentan en su contexto real y en un tiempo determinado o no, para luego analizarlos. Por lo tanto, en este diseño no se construye una situación específica sino que se observan las que existen. Las variables independientes ya han ocurrido y no pueden ser manipuladas, lo que impide influir sobre ellas para modificarlas (p. 87).

En base a esto, la presente investigación posee un diseño no experimental, debido a que no se pretende alterar la percepción o reconocimiento de los colores en las personas con deuteranomalía, sino más bien analizar esta condición anormal a la que están sujetas; de igual forma, solo se utilizaran métodos y técnicas previamente inventados para el procesamiento de imágenes digitales sin modificar su esencia, incluyendo aquellos que se encargan de detectar objetos y trabajar en base a los colores de estos.

3.3 Nivel de la investigación

La investigación es considerada de nivel proyectivo, la cual según Hurtado (2000) “...consiste en la elaboración de una propuesta o de un modelo, como solución a un problema o necesidad de tipo práctico, ya sea de un grupo social, o de una institución, en un áreaparticular del conocimiento, a partir de un diagnóstico preciso de las necesidades del momento, los procesos explicativos o generadores involucrados y las tendencias futuras.” (p.325).

Teniendo en cuenta lo anteriormente dicho, cabe resaltar que el desarrollo del software planteado para este proyecto pretende mejorar la percepción cromática a

las personas que padecen de deuteranomalía, lo que permitirá a estas personas poder comprender el contexto del entorno físico en el que se encuentran, gracias a la información que obtienen de los colores de los objetos que los rodean.

3.4 Población y Muestra

3.4.1 Población

Para Arias (2006) la población es "...un conjunto finito o infinito de elementos con características comunes, para los cuales serán extensivas las conclusiones de la investigación. Esta queda limitada por el problema y por los objetivos del estudio." (p.81). De lo anteriormente dicho, se destaca que la población constituye en sí al objeto de la investigación, y al considerarse como el centro de la investigación, debe ser desde este conjunto del cual sea extraída la información requerida para el proyecto.

Basado en esto, se toman en cuenta dos poblaciones para el presente proyecto: la primera está delimitada por los profesionales en el área de procesamiento de imágenes digitales, así como aquellos que trabajen en áreas estrechamente relacionadas, como por ejemplo, la visión del computador o el aprendizaje de la máquina. Debido a que ésta es una población con cantidad desconocida de individuos, se considera como población infinita, de la cual el autor anterior menciona que "es aquella en la que se desconoce el total de elementos que la conforman, por cuanto no existe un registro documental de estos debido a que su elaboración sería prácticamente imposible." (p.82). Asimismo, se tiene en cuenta que las personas especializadas en estas áreas son aquellas que podrán servir como fuente de información, con el fin de obtener ayuda sobre el uso de los métodos y técnicas que pueden contribuir al desarrollo del software propuesto para la investigación.

Por otro lado, la segunda población está compuesta por una cantidad estimada de 100 obras artísticas, delimitada por las imágenes digitales provenientes de los alrededores de la universidad José Antonio Páez, la cual se dividirá en dos grupos distintos: aquellas que estén compuestas sólo de objetos geométricos coloridos y

aquellas que no contengan estos necesariamente, en las cuales se representen situaciones, momentos o escenarios de la vida real o imaginarios. Tales imágenes serán extraídas de las obras que se encuentran presentes dentro del campus de la universidad, con la finalidad de utilizarlas tanto para las pruebas evaluativas de los métodos y técnicas candidatos a formar parte del software de aplicación, como para aquellas con los cuales se verificará la funcionalidad de este.

3.4.2 Muestra

Según Arias (2006), la muestra se define como “...un subconjunto representativo y finito que se extrae de la población accesible.” (p.83). Partiendo de esto, la muestra de la población de individuos profesionales es no probabilística del tipo intencional, de la cual Palella y Martins (2012) consideran relevante mencionar que "...el investigador establece previamente los criterios para seleccionar las unidades de análisis, las cuales reciben el nombre de tipo..." (p. 114). En base a lo anterior, solo se contemplarán los profesionales con experiencia que puedan ser ubicados por Internet y ser contactados por un medio de comunicación directo y en tiempo real.

En referencias a las imágenes digitales, su muestra es no probabilística del tipo por cuotas, de la cual los mismos autores mencionan que "...para ejecutar este tipo de selección, se divide la población en sectores, tomando en cuenta ciertos aspectos prefijados; pero la elección de las unidades de cada sector se realiza de manera arbitraria, sin atender a ningún procedimiento de selección" (p. 114). A partir de lo anterior, se tomarán 24 individuos mediante el criterio estadístico de elección simple, el cual consiste en seleccionar n elementos de la muestra entre k estratos o sectores, lo que da como resultado un número de 12 imágenes para el grupo que contenga figuras geométricas y para aquel que no necesariamente posea las mismas.

3.5 Técnicas e Instrumentos de recolección de datos

Según Arias (2006) “se entenderá por técnica, el procedimiento o forma particular de obtener datos o información” (p.67), tales como la observación, la encuesta o la entrevista, los cuales son ejemplos de técnicas de recolección de datos. En cuanto a los instrumentos, el mismo autor menciona que “...es cualquier recurso, dispositivo o formato (en papel o digital), que se utiliza para obtener, registrar o almacenar información” (p.69). Entre las técnicas de recolección de datos para el trabajo de investigación, se utilizará la entrevista, la cual según Palella y Martins (2012):

> ...es una técnica que permite obtener datos mediante un dialogo que se realiza entre dos personas cara a cara: el entrevistador “investigador” y el entrevistado; la intención es obtener información que posea este último. La ventaja esencial de la entrevista reside en que son los mismos actores sociales quienes proporcionan los datos relativos a sus conductas, opiniones, deseos, actitudes, expectativas, en fin, informaciones que, por su misma naturaleza, es casi imposible obtener desde afuera (p. 119).

Es importante resaltar que esta técnica será aplicada a la muestra de individuos profesionales, mediante el uso de un guion de entrevista no estructurada del tipo informal. Acerca de este, los autores anteriores afirman que "...es aquel en que no existe una estandarización formal dejando, por lo tanto, un margen más o menos grande de libertad para formular las preguntas y proporcionar las respuestas." (p. 129). Para efectos de la investigación, el guion a aplicar está disponible en inglés (ver anexo A, guion de entrevista), debido a que se desea entrevistar a personas de comunidades internacionales; asimismo, las preguntas son abiertas debido a la informalidad que esta presenta, de la cual los autores antes citados establecen que:

> …suele utilizarse en las fases iniciales de investigaciones de cualquier naturaleza, recurriendo a informantes claves que pueden ser expertos sobre el tema en estudio, lideres formales o informales, personalidades destacadas. Lo principal es dar la completa sensación al entrevistado de que puede hablar libremente, alentándolo y estimulándolo con cautela para evitar influir en sus actitudes (p. 129).

Por otro parte, se utilizará la observación como técnica para la recolección de imágenes digitales, la cual según los mismos autores se define como aquella que "...consiste en estar a la expectativa frente al fenómeno, del cual se toma y se registra información para su posterior análisis..." (p. 116). Además, estos autores afirman que:

> Existen dos clases de observación: la científica y la no científica. La diferencia básica entre una y otra está en la intencionalidad; la primera significa observar con un objetivo claro y preciso: el investigador sabe lo que desea notar y para qué quiere hacerlo, lo cual implica una preparación previa al proceso de observación. Observar sin intención científica consiste en mirar sin objetivo definido y, por tanto, sin preparación previa (p. 116).

En base a lo anterior, la observación es científica, ya que se desea obtener sólo aquellas imágenes que cumplan con las características correspondientes a los grupos estipulados para esta población. Es conveniente resaltar que para los autores anteriores "la observación científica presenta varias modalidades: directa o indirecta, participante o no participante, estructurada o no estructurada, de campo o de laboratorio, e individual o de equipo" (p. 118). De acuerdo a lo mencionado, la modalidad es directa, de campo y de equipo, la cual será auxiliada en la toma de imágenes mediante una cámara fotográfica y, serán recolectadas de acuerdo a una lista de cotejo (ver Anexo B), de la cual Arias (2006, p. 70), menciona que "…es un instrumento en el que se indica la presencia o ausencia de un aspecto o conducta a ser observada."

3.6 Fases metodológicas

Con la finalidad de complementar las diferentes fases metodológicas de la presente investigación y mantener una congruencia entre las actividades propuestas y los objetivos específicos, se escogió la metodología de desarrollo de software denominada proceso unificado ágil, cuyas fases internas se alinean adecuadamente para el desarrollo del software de aplicación planteado, mediante el uso de técnicas agiles.

Fase I: Determinar el espectro de color y los colores de confusión correspondientes a las personas con deuteranomalía

Inicialmente, se plantea realizar una investigación bibliográfica para descubrir las interrelaciones entre temas presentes en tres ramas de estudio principales: la óptica, la cual describe el origen y naturaleza del color; la colorimetría, la cual abarca subtemas de interés como los modelos del color, sistemas de mezcla de colores y teoría de los colores; y el daltonismo, con un enfoque hacia el tipo deuteranomalía y sus efectos. Adicionalmente, se desea comprender estos aspectos anteriores para poder identificar el alcance inicial del proyecto y lograr la aceptación de las partes interesadas, armonizando al mismo tiempo con la metodología AUP.

- **Actividad 1:** Se realizará una búsqueda bibliográfica por Internet con respecto a los tópicos ya mencionados, seleccionando y almacenando solo los puntos de interés que tengan relación con los colores.
- **Actividad 2:** Se identificarán las interrelaciones entre estos tópicos de estudio, almacenando solo aquellas que aporten beneficios para el desarrollo de la investigación.
- **Actividad 3:** Se determinará de manera cualitativa y cuantitativa el rango de colores afectados por el daltonismo del tipo deuteranomalía, con el fin de identificar claramente solo aquellos que sean pertinentes a esta discapacidad.

Fase II: Establecer los requerimientos funcionales y no funcionales del software

Una vez que se conozcan y comprendan los aspectos que afectan a la percepción cromática de las personas que padecen de deuteranomalía, en este caso los colores espectrales y aquellos que son confundidos, es necesario aplicar el modelo de ciclo de vida de la ingeniería de requerimientos para poder establecer una definición completa, consistente y no ambigua de los objetivos, funciones y restricciones que debe poseer el sistema, ya que esto representa un paso fundamental en el desarrollo de software. Aunado a esto, se considera también la etapa de

elaboración de la metodología AUP, en donde se plantean las siguientes actividades acordes al ciclo de vida mencionado anteriormente.

- **Actividad 1:** Se realizará la elicitación de los requerimientos mediante el uso de técnicas como la tormenta de ideas y la elaboración de prototipos, con el fin de identificar el flujo y la estructura de la información.
- **Actividad 2:** Seguidamente se procederá con la fase de negociación, en la que se realizará un estudio de factibilidad acerca de las herramientas y tecnologías disponibles que sean más favorables para el desarrollo del software.
- **Actividad 3:** Posteriormente, en la especificación se describe el comportamiento funcional deseado por parte del software diseñando un documento de requerimientos.
- **Actividad 4:** Para finalizar, se aplicarán verificaciones a los requerimientos especificados con una revisión de éstos, para así poder certificar que los mismos sean consistentes con lo planteado en el documento descrito anteriormente.

Fase III: Analizar los métodos y técnicas existentes en la librería OpenCV orientados a detectar los colores de un objeto en una imagen digital

En esta etapa se pretende recolectar datos e información acerca de los métodos y técnicas enmarcados dentro del área de procesamiento de imágenes digitales y campos de estudio relacionados, con la finalidad de ser usados como base para el desarrollo posterior de los algoritmos alineados a los requerimientos funcionales de la aplicación de software. Asimismo, se continúa trabajando en la fase de elaboración según la metodología AUP, con el motivo de profundizar en la comprensión de los requisitos del sistema y validar su arquitectura.

- **Actividad 1:** Se aplicará el guion de entrevista no estructurada elaborado para los profesionales del área de estudio o relacionadas a esta, ya sean complementarias o fundamentales.

- **Actividad 2:** Se ejecutarán los algoritmos que implementan los métodos y técnicas recomendados por los profesionales a un grupo de imágenes digitales, previamente obtenidas del muestreo realizado a esta población. También se tomarán en cuenta los comentarios y recomendaciones dadas en la entrevista.
- **Actividad 3:** Posteriormente, se revisará la eficacia y eficiencia de estos algoritmos. Después, se seleccionará uno o más algoritmos que hayan pasado las pruebas evaluativas y se consideren necesarios.

Fase IV: Desarrollar los algoritmos para la detección de colores de un objeto en una imagen digital

Después de haber recolectado y analizado todos los datos, informaciones y conocimientos indispensables, se procederá en base a estos a la construcción de los diferentes módulos del sistema, cuyos objetivos individuales son atómicos y complementarios entre sí. Esta fase está compuesta de tres módulos principales: el preprocesamiento, procesamiento y postprocesamiento de las imágenes digitales, los cuales serán desarrollados con la ayuda de los módulos de la librería OpenCV. Posteriormente, estos se unificaran para construir un software funcional inicial a través de una interfaz gráfica de usuario, para que cumpla con las necesidades de prioridad más alta, según lo establecido por la metodología AUP.

- **Actividad 1:** Se elaborará el módulo de preprocesamiento de imágenes digitales, en el cual se recibirán todos los parámetros de entrada.
- **Actividad 2:** Se construirán los módulos pertinentes y necesarios que reconozcan los colores de los objetos en una imagen digital, retornando información correspondiente que será sometida a un postprocesamiento.
- **Actividad 3:** Se elaborará el módulo de postprocesamiento de imágenes digitales, el cual se encargará de arrojar la respuesta de salida.

- **Actividad 4:** Se desarrollará la interfaz gráfica de usuario usando la librería de PyGTK que integre a todos los módulos de la aplicación desarrollados anteriormente.

Fase V: Verificar los algoritmos para la detección de colores de objetos en una imagen digital

Finalmente, se llevará a cabo la verificación funcional de las distintas etapas de procesamiento de una imagen digital así como la de su ejecución colectiva, mediante pruebas de caja blanca que buscarán detectar errores no previstos. De ser posible y factible se realizarán las correcciones respectivas, mostrando posteriormente una comparación de un antes y después de la reparación. De acuerdo a la metodología AUP, se continuará trabajando en la fase de construcción mediante las pruebas realizadas en el ambiente de desarrollo hasta pasar a la fase de transición, en donde se busca llevar la aplicación al entorno de preproducción, que en este caso será el Raspberry Pi.

- **Actividad 1:** Se aplicarán las pruebas de caja blanca a los algoritmos desarrollados en la fase cuatro, en las cuales se tomarán en cuenta los fallos para realizar correcciones.
- **Actividad 2:** Se mostrarán cada uno los resultados arrojados por las pruebas de caja blanca y aquellos después de haber hecho las correcciones. Una vez finalizado esto, se procederá a su correcta instalación en el Raspberry Pi.

CAPÍTULO IV
RESULTADOS

4.1 Fase I: Determinar el espectro de color y los colores de confusión correspondientes a las personas con deuteranomalía

Inicialmente, se realizó una búsqueda bibliográfica profunda, a partir de la cual se logró entender ciertos aspectos relevantes en las áreas de estudio de interés: la óptica, la colorimetría y el sistema visual humano junto con la ceguera del color o daltonismo. Adicionalmente, también se llevó a cabo una identificación de interrelaciones entre los temas de estudio, permitiendo generar un diagnostico acerca de la problemática en cuestión. A partir de la recolección de estos conocimientos, se procedió cautelosamente a determinar los colores, sus escalas y rangos de valores que corresponden con la afección asociada a la deuteranomalía.

4.1.1 Actividad I: Búsqueda bibliográfica en Internet

Por el lado de la óptica, se encontró que además de los colores convenidos en el espectro óptico de la luz (ver Figura 1, Punto 2.2.1), otras fuentes también consideran al color cian como parte del conjunto espectral pero no incluyen al color violeta (ver Figura 16). A estos colores se les puede diferenciar fácilmente del resto, debido a su clara posición dentro del espectro ya mencionado.

Colores espectrales con sus rangos de valores en nanómetros

Red	700...630 nm
Orange	630...600 nm
Yellow	600...570 nm
Green	570...530 nm
Cyan	530...490 nm
Indigo	490...450 nm
Blue	450...400 nm

Figura No 16. Fuente: Giangrandi I. (2004)

Por otro lado, se halló que aunque el diagrama de cromaticidad de la CIE (ver Figura 2, Punto 2.2.1.1) surge a partir del espacio de color RGB (ver Figura 3, Punto 2.2.1.2), éste es la representación gráfica ideal para establecer la gama de colores que el ser humano puede distinguir, no sólo de manera cualitativa sino también cuantitativa, con una alta exactitud. Tal característica permite que sea abordado para la realización de estudios e investigaciones a nivel científico y tecnológico. Asimismo, establece una norma universal para poder distinguir entre colores espectrales del resto en general, encontrándose estos a lo largo de la curva denominada locus espectral (ver Figura 17).

Los colores espectrales en la curva o locus espectral

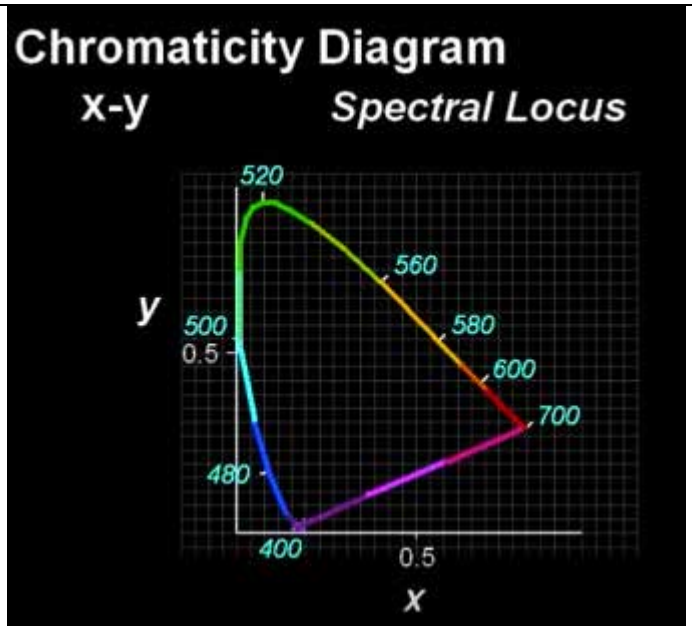


Figura No 17. Fuente: Blackwell C. (2013)

Dentro del ámbito de la colorimetría, se pudo identificar las ventajas y desventajas de varios modelos o espacios del color que pudiesen ser útiles dentro de la investigación. En el modelo RGB, el uso es provechoso en su mayoría para los dispositivos electrónicos que pueden realizar cálculos vectoriales a altas velocidades,

mientras que para una persona esto se vuelve tedioso. Con respecto al modelo CIELAB (ver Figura 5, Punto 2.2.1.2), aun cuando se basa en la visión del color humana, no permite a primera instancia utilizar su espacio de manera fácil, debido a su abstracta implementación en tres dimensiones y con ocho cuadrantes. A diferencia de los anteriores, el modelo HSV (ver Punto 2.2.1.2) presenta una manera intuitiva y fácil al implementar conceptos más amigables y no tan abstractos.

En otro orden de ideas, de acuerdo a la teoría del color oponente (ver Punto 2.2.1.3), se identificó que la deuteranomalía se ve afectada por el par de colores rojo-verde, lo que en consecuencia permite únicamente el reconocimiento de los colores amarillo y azul con sus respectivas variaciones limitadas. Junto con esto, se logró entender el rol que cumplen las líneas de confusión o isocromáticas (ver Figura 10, Punto 2.2.2.2) en relación a la afección generada por la deuteranopia y deuteranomalia. Cualquier color que se sitúe o se encuentre próximo a ellas será interpretado por el par amarillo-azul dentro del diagrama CIE, ubicados a lo largo de una línea recta que intersecta al punto de igualdad de energía (W) o color blanco.

Finalmente, se conoció que el punto de energía anterior representa a la cantidad de luz blanca ideal, por lo que también se le denomina como un punto neutro dentro del diagrama CIE. No obstante, también se determinó que esta misma luz puede tomar cuatro estados diferentes, cuyo color dependerá de la temperatura que estas contengan: la de tungsteno se percibe naranja, la directa que proviene del sol como un poco amarilla, aquella un poco azulada y una que representa al promedio de luz estándar en un día normal (ver Figura 18).

Los colores espectrales en el locus espectral de acuerdo a la temperatura

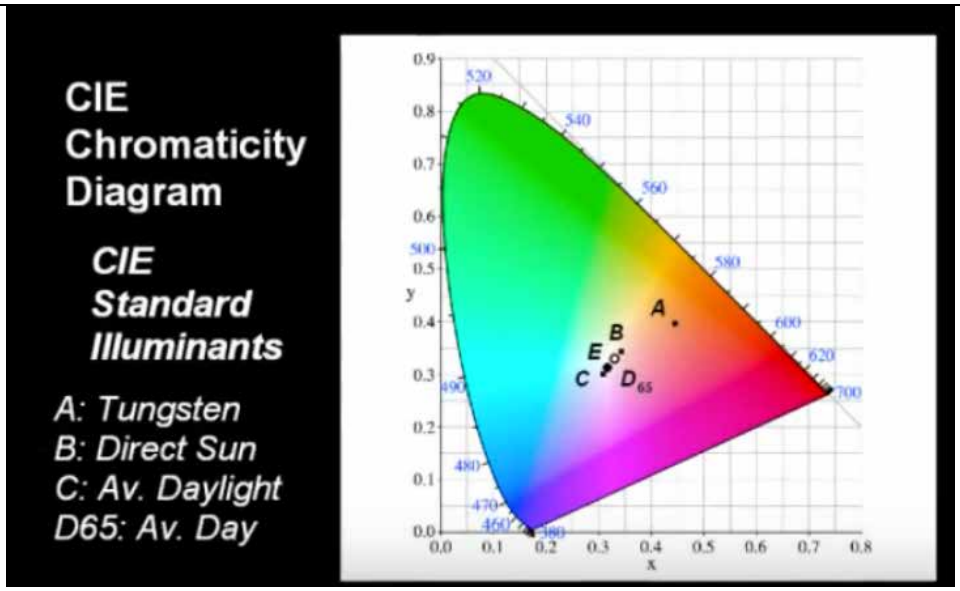


Figura No 18. Fuente: Blackwell C. (2013)

4.1.2 Actividad II: Identificar las interrelaciones entre los tópicos de estudio

Una vez realizada la búsqueda bibliográfica, se pudo determinar que dentro de toda la gama de colores y sus combinaciones, los que más resaltan al momento de realizar una investigación son los colores espectrales y no espectrales. Esto se debe a su íntima relación con el diagrama de cromaticidad de la CIE, el cual permite mapear a éstos con una gran exactitud y coherencia; dicha característica se sincroniza a su vez con los conceptos de la teoría del color oponente y la ceguera del color.

En relación al diagrama CIE, se determinó que las distintas características de la luz blanca pueden alterar las tonalidades de los colores, afectándolos al momento de agregar aquella variación de color inherente a esta luz, según los tipos existentes que están próximos al punto de energía de igualdad (W). Esto ocasionara que la detección del color esté condicionada a esta variable adicional, generando resultados indeseados en ciertas ocasiones.

Por otro lado, las líneas de confusión en conjunto con el par de colores amarillo-azul, permitieron delimitar el radio de acción en cuanto a los colores que deben ser filtrados dentro del software, tales son: rojo, naranja, verde, cian, violeta, magenta y rosa. Aunado a esto, el modelo de color HSV provee un manejo fácil y apropiado que se alinea justamente a estos requerimientos; a su vez, éste sugiere la utilización de dos colores más que pueden ser tomados en cuenta: verde amarillo y verde primavera (ver Figura 19). Aun cuando estos dos tipos de verdes son muy parecidos, la incorporación de estos permitirá abarcar un conjunto más amplio de escenarios en las imágenes. Debido a estas prestaciones y ventajas, se decidió elegir el anterior modelo para trabajar con el software.

Colores primarios, secundarios y terciarios en la rueda del color en HSV

Figura No 19. Fuente: Howard J (2016)

4.1.3 Actividad III: Determinar rango de colores afectados por la deuteranomalía

En base a los colores elegidos anteriormente, se decidió utilizar la escala cromática para dar un nombre calificativo a las distintas combinaciones de colores; en

ésta los valores del tono se obtienen mezclando los colores puros con blanco o con negro. A su vez, dicha escala se subdivide en dos partes más: alta o versión clara y baja o versión oscura. Estas dos últimas fueron elegidas de acuerdo al alcance inicial del proyecto, agregando como complemento la unión de ambas.

Las tonalidades de cada color se eligieron según su ángulo en el modelo HSV: 0º para el rojo, 30º para el naranja, 90º para el verde amarillo, 120º para el verde, 150º para el verde primavera, 180º para el cian, 270º para el violeta, 300º para el magenta y 330º para el rosa. Estos valores sirvieron como punto medio, a partir de los cuales se eligió una variación de 15º en ambos sentidos, formando un conjunto de rangos en la tonalidad para cada color (ver Cuadro 1), exceptuando al naranja y al violeta debido a que se distorsionan hacia al amarillo y al azul respectivamente. Se optó por usar tal conjunto debido a que puede atrapar a una mayor cantidad de colores.

Cuadro No 1: Rango de valores del modelo HSV para las tonalidades (H)

Fuente: Caicedo C. y Pérez M. (2018)

Por otro lado, el rango para la saturación (S) y el valor (V) fue establecido según la percepción subjetiva de los investigadores con la ayuda del software GIMP; primero se procedió a delimitar visualmente las regiones para cada escala de cada color (ver Anexo C, gamas de colores). Para los colores claros y oscuros se determinó una segmentación fija a través de rectángulos y cuadrados, a excepción nuevamente del naranja y el violeta, para los cuales se llevó a cabo una revisión visual individual y exhaustiva, observando las áreas respectivas de cada color que fuesen las más apropiadas. Seguidamente, se recolectaron los valores que delimitaron a cada región, respetando la separación entre una y otra (ver Cuadros 2-3).

345	40	55	15	55	69	16	50	90	40	89	100
345	40	70	15	79	100						
75	40	55	105	55	69	106	40	55	135	55	69
75	40	70	105	79	100	106	40	70	135	79	100
136	40	55	165	55	69	166	40	55	195	55	69
136	40	70	165	79	100	166	40	70	195	79	100

260	30	75	285	69	79	286	40	55	315	55	69
260	30	80	285	79	100	286	40	70	315	79	100
316	40	55	344	55	69						
316	40	70	344	79	100						

Cuadro No 2: Límites de los rangos para los colores claros en HSV

Fuente: Caicedo C. y Pérez M. (2018)

345	40	40	15	55	54	16	80	80	40	89	89
345	56	40	15	79	69	16	90	75	40	100	100
345	80	40	15	100	100						
75	40	40	105	55	54	106	40	40	135	55	54
75	56	40	105	79	69	106	56	40	135	79	69
75	80	40	105	100	100	106	80	40	135	100	100

136	40	40	165	55	54	166	40	40	195	55	54
136	56	40	165	79	69	166	56	40	195	79	69
136	80	40	165	100	100	166	80	40	195	100	100

260	30	50	285	79	74	286	40	40	315	55	54
260	70	75	285	79	79	286	56	40	315	79	69
260	80	50	285	100	100	286	80	40	315	100	100

316	40	40	344	55	54
316	56	40	344	79	69
316	80	40	344	100	100

Cuadro No 3: Límites de los rangos para los colores oscuros en HSV

Fuente: Caicedo C. y Pérez M. (2018)

4.2 Fase II: Establecer los requerimientos funcionales y no funcionales del software

Luego de haber realizado la investigación respectiva para poder entrar en contexto, conocer los aspectos relacionados a la percepción cromática del color y de qué manera esta afecta a las personas que sufren de deuteranomalía, se procedió a

aplicar los pasos de la ingeniería de requerimientos para asegurar la generación de especificaciones correctas que describan con claridad, sin ambigüedades y en forma compacta las necesidades del software.

4.2.1 Actividad I: Elicitación de requerimientos

En primer lugar, se procedió a ejecutar técnicas como la tormenta de ideas, la cual buscaba establecer el alcance de la investigación, por lo que se decidió que se segmentaria y dividiría por puntos que ayuden a determinar el flujo del trabajo de investigación, mediante el documento de definición de requerimientos (ver Anexo D-1, documentos de requerimientos).

Luego de esto, se elaboraron los casos de uso correspondientes al sistema, para identificar las acciones que puede realizar el usuario (ver Figura 20), tales como: tomar una foto (ver Cuadro 4), subir una imagen (ver Cuadro 5), leer una imagen (ver Cuadro 6), procesar una imagen (ver Cuadro 7), guardar la imagen de respuesta (ver Cuadro 8) y ver la ventana de ayuda (ver Cuadro 9). Se determinó que el procesamiento de imágenes digitales es el área de aplicación en la cual se orientó el desarrollo del proyecto, por lo que ese fue el enfoque técnico que se le dio al mismo y seguidamente se procedió a realizar un estudio que permitió la comprensión del dominio en relación a: conceptos, términos, teorías y técnicas que se utilizan dentro de esta área. Cabe mencionar que se consideraron otras opciones como la visión del computador y el procesamiento de videos.

Diagrama de casos de uso

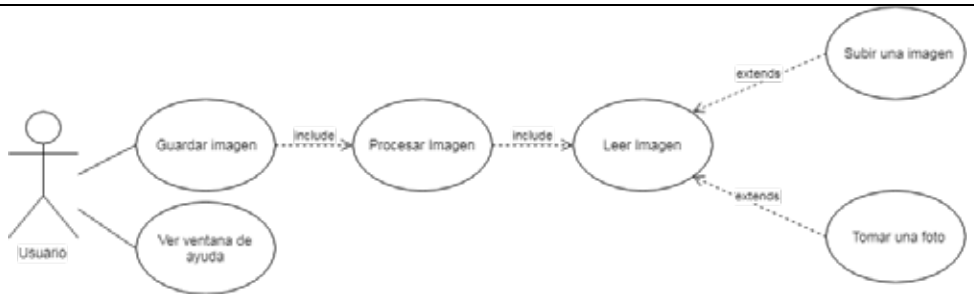


Figura No 20. Fuente: Caicedo C. y Pérez M. (2018)

Caso de uso: Tomar una foto
Actor participante: Usuario
Flujo de eventos:
1. El usuario presiona el botón de tomar foto 2. Se captura la imagen que se visualiza en la vista previa 3. Se selecciona el directorio local en donde se guardará la imagen 4. Se guarda la imagen tomada en la dirección indicada

Cuadro No 4: Descripción del caso de uso Tomar una foto

Fuente: Caicedo C. y Pérez M. (2018)

Caso de uso: Subir una imagen
Actor participante: Usuario
Flujo de eventos:
1. El usuario presiona el botón de subir una imagen 2. Se selecciona el directorio del fichero local en donde se encuentra la imagen

Cuadro No 5: Descripción del caso de uso Subir una imagen

Fuente: Caicedo C. y Pérez M. (2018)

Caso de uso: Leer imagen
Actor participante: Usuario
Flujo de eventos:
1. Se obtiene la dirección la imagen guardada 2. Se muestra la imagen en la ventana principal

Cuadro No 6: Descripción del caso de uso Leer imagen

Fuente: Caicedo C. y Pérez M. (2018)

Caso de uso: Procesar imagen
Actor participante: Usuario
Flujo de eventos:
3. Se obtiene la dirección la imagen guardada 4. Se muestra la imagen en la ventana principal

Cuadro No 7: Descripción del caso de uso Procesar imagen

Fuente: Caicedo C. y Pérez M. (2018)

Caso de uso: Guardar foto de respuesta
Actor participante: Usuario
Flujo de eventos:
1. El usuario presiona el botón de guardar foto 2. Se selecciona el directorio local en donde se guardará la imagen 3. Se guarda la imagen de respuesta en la dirección indicada

Cuadro No 8: Descripción del caso de uso Guardar respuesta

Fuente: Caicedo C. y Pérez M. (2018)

Caso de uso: Ver ventana de ayuda
Actor participante: Usuario
Flujo de eventos:
1. El usuario presiona el botón de Ayuda 2. Se muestra la ventana con la información del sistema

Cuadro No 9: Descripción del caso de uso ver ventana de ayuda

Fuente: Caicedo C. y Pérez M. (2018)

Además de esto, se procedió a revisar la documentación acerca de las tecnologías modernas disponibles para el procesamiento de imágenes digitales, en donde se encontró información referente al área de estudio, en donde destacaron como opciones principales la librería OpenCV y la caja de herramientas para el procesamiento de imágenes en MATLAB.

4.2.2 Actividad II: Negociación del alcance

Para realizar esta actividad se inició un debate de preferencias en donde se discutieron puntos para establecer el alcance del software, en el cual se seleccionó el tipo de daltonismo a tratar, siendo la deuteranomalía el tipo escogido en el que se enfocó este proyecto; la selección anterior se llevó a cabo tomando en consideración las investigaciones realizadas previamente durante la Fase I. Además, se estableció cuáles fueron los tipos de entrada que tendría el software, siendo solo permitida una imagen en los formatos .jpeg, .jpg, .jpe y .png; por su parte, la salida consiste en devolver imágenes del mismo tipo que en la entrada, a diferencia de que se muestra de acuerdo al color y a la escala seleccionada a filtrar.

Asociado a lo anterior, fue indagado acerca de las características funcionales y no funcionales, tanto de la caja de herramientas de MATLAB como de la librería de OpenCV; se determinó que son capaces de llevar a cabo las mismas tareas, sin

embargo, se optó por usar OpenCV debido a su alta optimización, menor consumo de memoria RAM y robustez, además de sus otras ventajas incluidas (ver Cuadro 10). Según el manual de OpenCV, lo anterior es gracias a que todo el código fuente está escrito en el lenguaje de programación de C++, al cual se le hace un llamado a sus algoritmos mediante el uso de generadores de enlaces o bindings generators, los cuales crean un puente entre C++ y Python (Documentación de OpenCV, 2016).

	MATLAB	**OpenCV**
Facilidad de uso	9	3
Velocidad	2	9
Recursos necesitados	4	9
Costo económico	4	10
Ambiente de trabajo	8	6
Manejo de la memoria	9	4
Portabilidad	3	8
Desarrollo de conocimientos útiles	3	8
Depuración de código	9	5
Ayuda y código de ejemplo	8	9
Total	**59**	**71**

Cuadro No 10: Tabla comparativa entre MATLAB y OpenCV

Fuente: Khatua, D. (2015)

Una vez se establecieron las entradas y salidas del software, se inició un estudio de factibilidad del proyecto (ver Anexo E, estudio de factibilidad). El estudio se realizó con la finalidad de evaluar si la ejecución del proyecto, con los recursos y tiempos estimados, obtendría un resultado exitoso; este documento se redactó en base a tres etapas.

La primera correspondía a la factibilidad técnica, en donde se evaluaron si se disponían de los recursos y conocimientos técnicos necesarios para que el proyecto cumpliera con sus objetivos, entre los cuales destacan: los computadores de desarrollo del proyecto, el Raspberry Pi, las librerías a utilizar y la documentación apropiada. Luego se contempló la factibilidad económica, en la cual se buscó hacer un balance entre los costos y beneficios asociados al proyecto. Y por último, la tercera etapa que corresponde a la factibilidad operativa, en donde se tomó en cuenta la utilización correcta del software por parte de los usuarios.

4.2.3 Actividad III: Especificación de requerimientos

Se comenzó con la redacción del documento de especificación de requerimientos, el cual está conformado por dos secciones. La primera consta de una tabla en donde se muestra la información del proyecto, tal como: el título, la descripción, la fecha de inicio y la versión del software a desarrollar. Luego, se muestra una segunda tabla en donde cada fila representa a un requerimiento del sistema, estos constituyen las características funcionales y no funcionales que deben implementarse en el software (ver Anexo D-2, documentos de requerimientos).

Adicionalmente a esto se inició el modelado inicial de la interfaz gráfica de usuario del sistema, para lo que se utilizó un diagrama de transición de estados (ver Figura 21) que mostrase el comportamiento de la misma, mediante la representación gráfica de los estados que pueden tomar sus componentes y mostrar los eventos que implican el cambio de un estado a otro. Todo esto se fundamentó en los casos de uso previamente descritos durante la Actividad I.

Diagrama de transición de estados

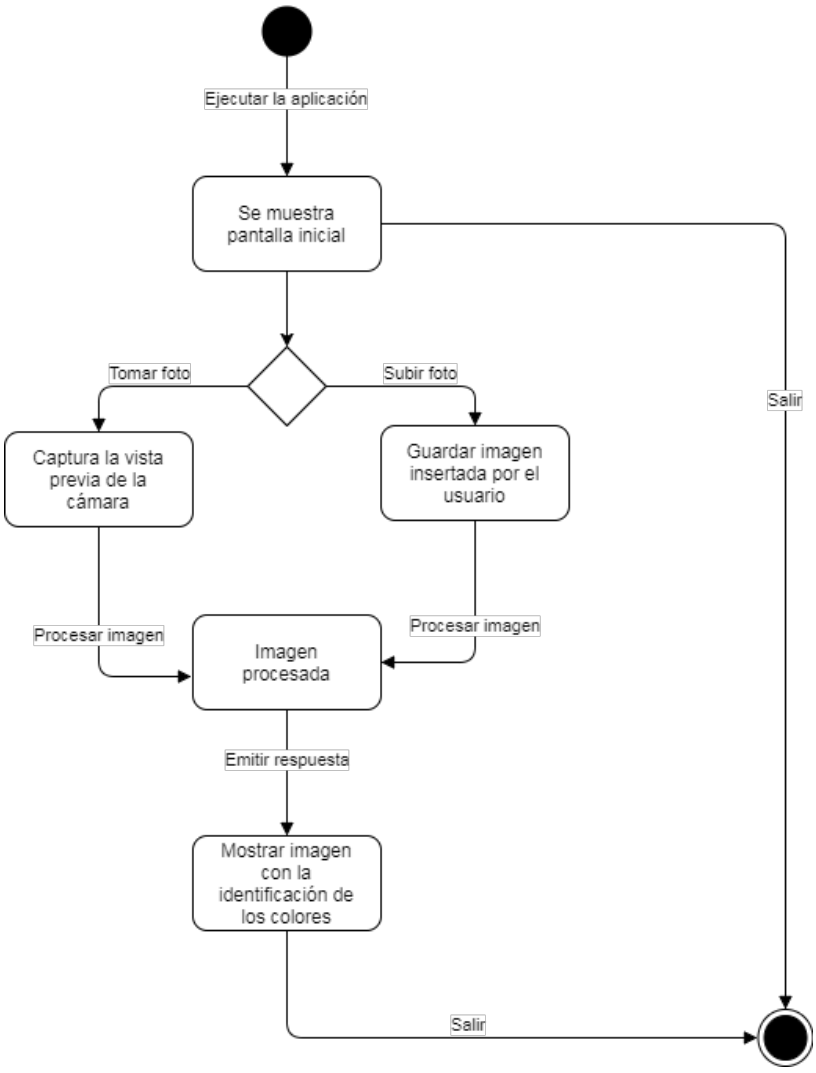


Figura No 21. Fuente: Caicedo C. y Pérez M. (2018)

4.2.4 Actividad IV: Verificación de requerimientos

Para la verificación de los requerimientos, se comprobó la trazabilidad de los mismos, aplicando una matriz de trazabilidad (ver Cuadro 11), para revisar que exista una asociación entre un requisito y una funcionalidad del sistema, lo que permite mantener la consistencia entre los requerimientos y monitorizar su ciclo de vida. Esto

se encuentra relacionado a los casos de uso elaborados anteriormente en la Actividad I.

	MATRIZ DE TRAZABILIDAD					
Requisito N	Caso de Uso 1: Tomar una Foto	Caso de Uso 2: Subir una imagen	Caso de Uso 3: Leer imagen	Caso de Uso 4: Procesar imagen	Caso de Uso 5: Guardar imagen	Caso de Uso 6: Abrir ayuda
1	X	X	X	X	X	X
2	X					
3		X	X			
4				X		
5				X		
6					X	
7						X

Cuadro No 11: Matriz de Trazabilidad de Requerimientos

Fuente: Caicedo C. y Pérez M. (2018)

4.3 Fase III: Analizar los métodos y técnicas existentes en la librería OpenCV orientados a detectar los colores de un objeto en una imagen digital

Durante esta etapa se aplicó un guion de entrevista a expertos en el área de procesamiento de imágenes digitales y campos de estudio relacionados. Después de haber recolectado las respuestas, se revisó las mismas en busca de técnicas o métodos, algoritmos e incluso recomendaciones, cuyos casos de uso sean representativos para llevar a cabo la detección de colores de objetos dentro de una imagen. De igual forma, se recabaron otros algoritmos que fuesen complementarios para llevar a cabo esta última tarea, mediante una búsqueda adicional a parte en otras fuentes de Internet.

Seguidamente, éstos fueron probados mediante algoritmos ya implementados en la librería OpenCV y otros creados por algunos autores externos. Los resultados consecuentes demostraron que todos los algoritmos fueron de utilidad, indicando que su uso colectivo podría cumplir con los objetivos planteados dentro del proyecto. A pesar de lo anterior, se llevó a cabo una evaluación de su desempeño en ejecución, de tal forma que se pudiese comprobar la factibilidad de su uso, tanto en el consumo de tiempo como en el de memoria RAM.

4.3.1 Actividad I: Aplicar el guion de entrevista a expertos en el área

Inicialmente, se procedió a enviar las solicitudes por correo electrónico a los expertos en el área de procesamiento de imágenes digitales, de tal forma que se pudiese concertar el día y la hora a realizar la entrevista. En respuesta a lo anterior, cinco (5) personas respondieron a la solicitud, por lo que cada una de ellas fue entrevistada. A continuación se presenta un análisis cualitativo acorde a la similitud de las respuestas dadas en cada una de las preguntas realizadas en las entrevistas, las cuales tienen como objetivo el encontrar algún algoritmo o técnica que pueda ser de utilidad dentro de la investigación.

- **Al momento de la detección del color de un objeto en el procesamiento de imágenes, ¿debería yo trabajar con el color independientemente del objeto?**

Las respuestas dadas por los profesionales indicaron que si solo se desea identificar el color, entonces la detección del objeto no es necesaria, dando uno de ellos el siguiente ejemplo: al detectar áreas de alta temperatura en un mapa de calor, el contorno de la figura es de menor importancia debido a que el color es el factor predominante.

Debido a que el interés principal de la investigación es trabajar con los colores de un objeto y no con este último, se tomó esto en cuenta, por lo que se adoptó la recomendación a lo largo de todo el desarrollo del proyecto.

- **En la detección del color de un objeto dentro de una imagen, ¿consideraría usted trabajar con reconocimiento o detección de objetos?**

La mayoría indicó que tal opción depende de los datos reales, justificando que si el objeto abarca la imagen por completo o por lo menos 80% del espacio, entonces se debería aplicar reconocimiento. Sin embargo, si el objeto de interés está localizado en una pequeña porción de la imagen, entonces el proceso de detección viene al caso.

Aun cuando la respuesta fue acertada y precisa, no se tomó en cuenta, debido a que se acordó anteriormente que la detección o reconocimiento de un objeto no es necesario.

- **¿Es la segmentación de imágenes basada en el color buena para la detección del color de un objeto? ¿Por qué?**

Algunos de los entrevistados señalo que no es buena, debido a que los colores en los computadores son diferentes a la intuición humana respecto al color, justificando que el ojo humano se ajusta automáticamente a las condiciones de iluminación y dando como ejemplo, a aquellos objetos estén expuestos parcialmente a la luz y a la sobra, lo que da como resultado una interpretación doble y distinta para cada caso, por parte de un computador o cámara fotográfica. Sin embargo, otros indicaron que aunque este era un problema real e inevitable, se podía tratar en ciertas ocasiones cambiando localmente el brillo de los colores, alterando el valor gamma de una imagen digital.

En base a lo anterior, se decidió implementar la funcionalidad de corrección de gamma a las imágenes digitales, de tal forma de que se pueda brindar una solución a este inconveniente.

- **¿Cuál modelo del color es ideal para la detección de este en el procesamiento de imágenes?**

Varios entrevistados sugirieron el modelo del color HSV como una opción fácil e intuitiva, resaltando el hecho de que en algunos casos puede que no funcione, ya que el color depende mucho de otros factores, como la luz, la calidad de la cámara y el ángulo de visión. Sin embargo, también objetaron que por ser un proyecto novato, el uso de HSV dará buenos resultados inmediatos sin tener la necesidad de aumentar la complejidad del mismo.

Dicha respuesta fue de utilidad, ya que permitió seleccionar un modelo de color definitivo entre un abanico de opciones adicionales; además, debido a que el proyecto a desarrollar esta en sus etapas iniciales, este modelo es el ideal para poder arrancar la

investigación. También su manejo fácil e intuitivo a nivel de código lo convierte en el candidato ideal para el desarrollo del software.

- **¿Cuál técnica recomendaría usted para la detección del color de un objeto en el procesamiento de imágenes?**

Con respecto a dicha pregunta, todos los expertos coincidieron similarmente en usar una técnica básica para un favorable manejo y detección del color, la cual es convertir una imagen al modelo del color HSV y después pasarla por el método del valor del umbral, basándose en el valor de la tonalidad (H). Asimismo, algunos entrevistados destacaron la necesidad de utilizar una función de transformación entre modelos del color, debido a que los dispositivos electrónicos que capturan imágenes usan en su mayoría el modelo RGB.

Dicha recomendación sirvió para determinar el paso inicial en el desarrollo de la funcionalidad principal y más importante del software, la cual es detectar los colores. Igualmente, permitió comprender otros aspectos de suma importancia para poder llevar a cabo lo anterior, como lo es la transformación de un modelo de color a otro. Finalmente, tales puntos permitieron comenzar la exploración de los algoritmos candidatos a ser integrados con esta técnica ya mencionada.

4.3.2 Actividad II: Pruebas técnicas de los algoritmos

En base a las respuestas obtenidas en la actividad anterior, se encontró una función en la librería OpenCV para transformar una imagen digital de tamaño 8-bit en el modelo RGB a HSV (ver Figura 22). En ella se debieron comprender tres aspectos importantes: el modelo RGB se usa por defecto para la lectura de las imágenes y de manera inversa, es decir, sus valores siguen el orden BGR; en segundo lugar, la transformación resultante se basa en el espacio de color anterior y por lo tanto, los valores de los pixeles corresponden a una representación gráfica abstracta y para nada intuitiva del modelo HSV; y en tercer lugar, la escala de valores del modelo HSV en OpenCV delimita a la tonalidad de 0 a 180 y, la saturación y el valor de 0 a 255. A

causa de esta modificación especial, se debieron ajustar estos tres tipos de valores en los rangos de los colores ya establecidos en la fase I.

Imagen transformada de BGR a HSV

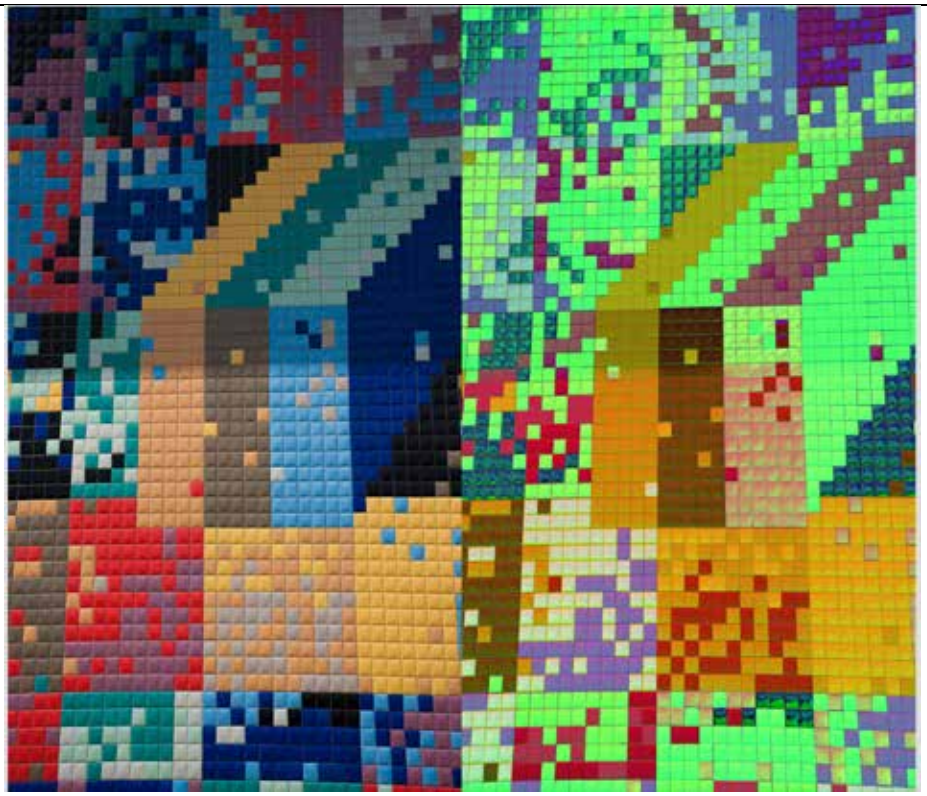


Figura No 22. Fuente: Caicedo C. y Pérez M. (2018)

Con respecto a la corrección de gamma, se halló que existe una fórmula matemática usada para aplicar esta técnica, denominada la transformación de la ley de potencia (ver Figura 23), con la cual se implementó un algoritmo para que fuese probada. Esta consiste en cambiar las intensidades de los pixeles de tal forma que aumente o disminuya el brillo en una imagen (ver Figura 24). Cabe destacar que el punto medio o de equilibrio es cuando el valor de la variable gamma es igual a uno; valores menores a uno tienden a oscurecer la imagen mientras que los mayores a uno la hacen más clara.

Transformación de la ley de potencia. I es el valor de entrada, g es el valor de gamma y O es el valor de salida

$$O = I \wedge (1 / G)$$

Figura No 23. Fuente: Caicedo C. y Pérez M. (2018)

Imagen original a la izquierda y a la derecha la corregida con un valor de gamma igual a 2

Figura No 24. Fuente: Caicedo C. y Pérez M. (2018)

A pesar de los algoritmos ya obtenidos, todavía no se había definido una manera de detectar los colores de una imagen. Por tal motivo, se procedió a investigar más a fondo y se encontró el método del valor umbral, el cual consiste en separar los objetos de interés de una imagen del resto, mediante la asignación de cada píxel a un cierto grupo, llamado comúnmente segmento. El anterior método se modificó a conveniencia para que hiciera esta misma agrupación, de tal forma que sólo se recolectaron aquellos píxeles con los valores cromáticos deseados, transformando a la imagen que los contiene en una representación binaria: asigna el color blanco a aquellos pixeles que cumplan con las condiciones ya planteadas y el color negro para

los que no (ver Figura 25). A dicha alteración se le podría apodar como un filtro espacial de colores que se aplica sobre una imagen.

Método del valor umbral aplicado al color rojo, escala oscura

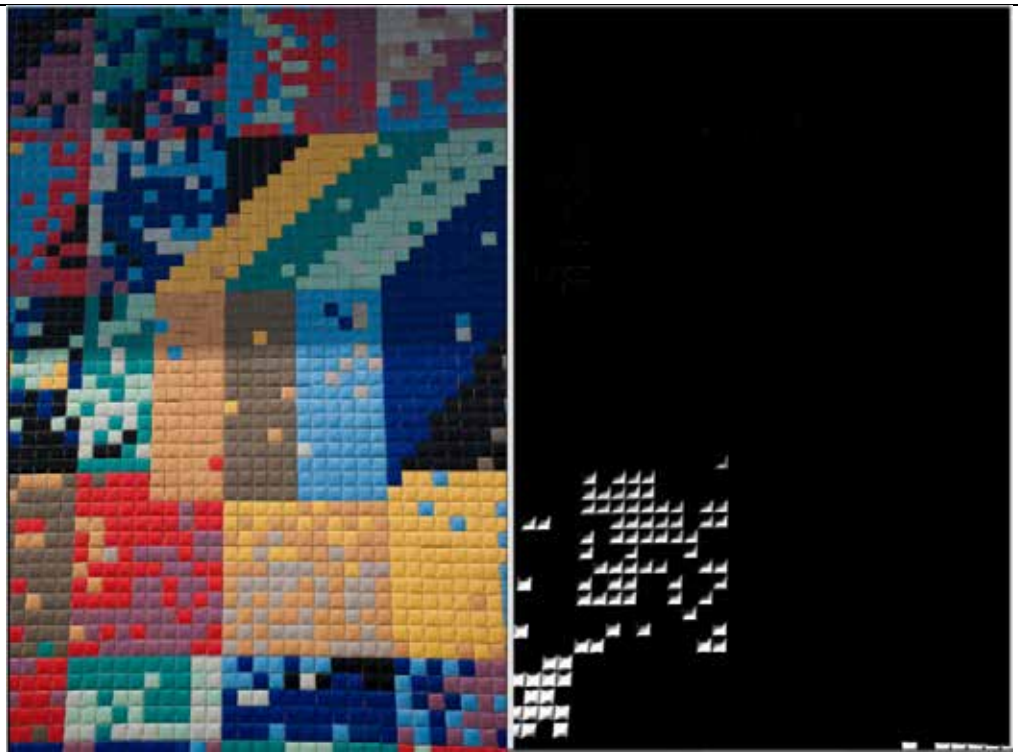


Figura No 25. Fuente: Caicedo C. y Pérez M. (2018)

Seguidamente, se hizo una búsqueda acerca de alguna técnica que pudiese conectar de manera más fluida a los segmentos obtenidos por el filtrado anterior, de tal forma que se lograse agrupar a un conjunto grande de píxeles en áreas más grandes y continuas. En base a la necesidad de satisfacer tal demanda, se utilizó una técnica muy usada en el procesamiento de imágenes denominada como suavizado o difuminación de píxeles, la cual consiste en promediar el valor de un pixel en torno a un grupo reducido de pixeles vecinos a este (ver Figura 26). Dicho método busca obtener varios resultados, los cuales son: suavizar los bordes de los segmentos

contenidos dentro de un filtro de colores, unir aquellos que estén relativamente bastante cerca entre sí y eliminar el ruido generado por diminutos grupos de píxeles aislados, que no es otra cosa sino cambiar el valor de sus intensidades a cero.

A la izquierda se encuentra el filtro generado por el método del valor umbral y a la derecha el resultado de suavizar este último

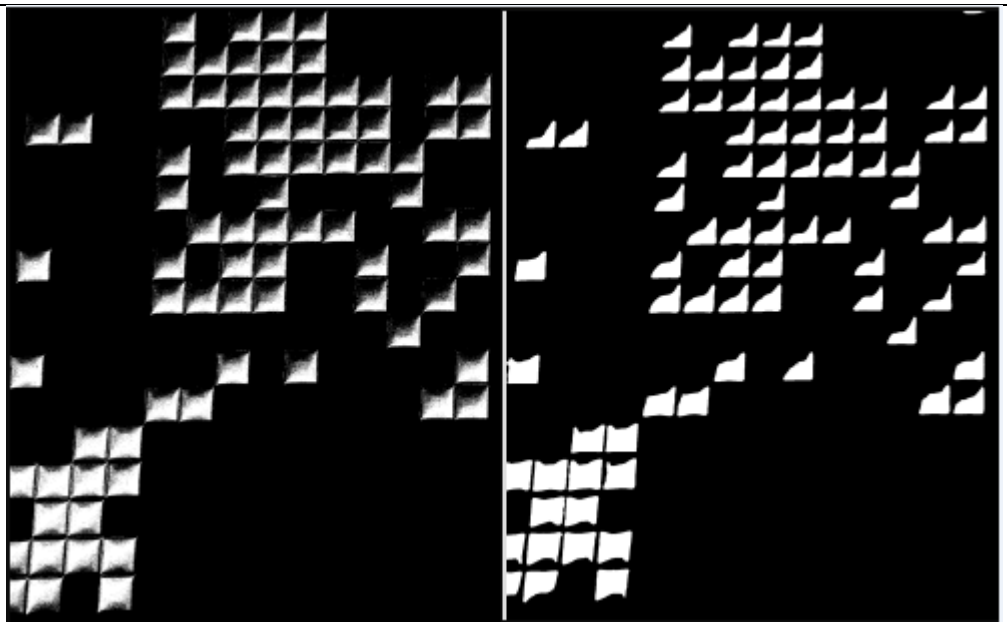


Figura No 26. Fuente: Caicedo C. y Pérez M. (2018)

Otro motivo adicional para escoger la técnica anterior es la delimitación gráfica de contornos, cuya función principal es encerrar en regiones grandes compuestas de varios segmentos aledaños entre sí. Con la finalidad de realizar esto, se estudió y probó un algoritmo que lleva a cabo un análisis estructural topológico de imágenes binarias digitalizadas, enfocado en los bordes contenidos en éstas. Los consecuentes resultados fueron de suma utilidad al momento de atrapar las zonas en las cuales se encuentran los colores problemáticos, por lo que se eligió como opción definitiva e indudable para llevar a cabo la anterior tarea planteada. De igual forma se probó un

algoritmo que basado en la localización de estas áreas espaciales, procede a dibujarlas encima de la imagen procesada (ver Figura 27).

Dibujo de contornos basado en el análisis estructural topológico de imágenes binarias digitalizadas

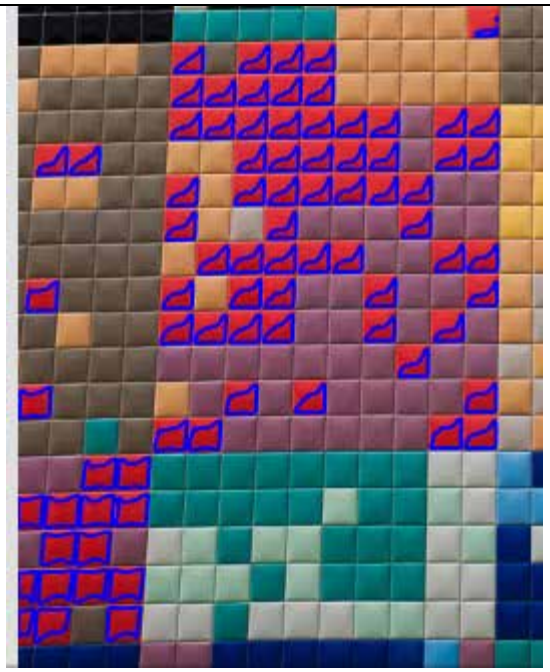


Figura No 27. Fuente: Caicedo C. y Pérez M. (2018)

4.3.3 Actividad III: Evaluar el desempeño de ejecución

Una vez realizadas las pruebas técnicas de los algoritmos, se procedió a estudiar y verificar el desempeño de ejecución de estos, los cuales en su mayoría ya estaban implementados como funciones en la librería OpenCV. Esta tarea se llevó a cabo con la ayuda de dos módulos creados en python version 3 para tal propósito: uno llamado line profiler (perfilador de línea), el cual arroja un perfil determinístico con datos de un programa en python mediante un archivo, basado principalmente en la unidad de tiempo en microsegundos; otro denominado memory profiler (perfilador de memoria), cuyos procedimientos usados son similares al primer módulo, pero utiliza la memoria en bytes como unidad de medida. Ambos pueden especificar resultados

para cada línea de código dentro del script y los reflejan en su totalidad mediante una tabla sencilla, graficada en una interfaz de línea de comandos.

En lo que respecta a las pruebas basadas en el tiempo, se recolectaron datos tales como: número de ocasiones que se ha ejecutado una línea de código (Hits), tiempo total consumido (Time), tiempo consumido por cada iteración (Per Hit), un porcentaje del tiempo utilizado por cada instrucción (% Time) y el tiempo de ejecución total de la función. Los resultados correspondientes demostraron que la mayoría de funciones se ejecutaron rápidamente, siendo la más lenta de todas la función de suavizado de bordes, la cual tomó en total ocho segundos en completarse (ver Anexo F-1, pruebas que miden el tiempo de ejecución). No obstante, la suma acumulada de la duración de todas estas pruebas no llegó a representar una cantidad significativa que pudiese ralentizar el uso de ellas de manera individual o en grupo.

Por otro lado, para las pruebas basadas en el uso de memoria únicamente se recolectaron datos como el uso de ésta y su incremento por cada línea de código, arrojando como resultado final la cantidad total que fue consumida. Los resultados respectivos indicaron que el uso de memoria promedio utilizado para cada función es alto (ver Anexo F-2, pruebas que miden la memoria consumida), en comparación relativa con la cantidad de RAM que posee el Raspberry Pi (1GB). Sin embargo, el ordenador deja disponible 600 MB aproximados para uso libre, además de que cada función de OpenCV está implementada en C++, por lo que se libera la cantidad de memoria correspondiente que se tomó al finalizar la ejecución de cada una de éstas.

4.4 Fase IV: Desarrollar los algoritmos para la detección de colores de un objeto en una imagen digital

En el comienzo de esta fase, se importaron sólo aquellas clases, funciones y constantes necesarias a lo largo de la construcción del software, de tal forma que se utilice la memoria RAM del Raspberry eficientemente (ver Cuadro 12). Luego, se procedió a desarrollar los algoritmos de cada etapa en el procesamiento de imágenes: en primer lugar, se elaboró aquel que realiza la preparación previa de una imagen a la

etapa principal; luego se crearon aquellos que se encargan de la detección y el reconocimiento de colores, en base a los cuales se ejecutaron unas pruebas de funcionalidad; y en último lugar, se agregó una función que pueda devolver el resultado final para que sea usado por la interfaz gráfica de usuario. Finalmente, se llevó a cabo la creación de la interfaz gráfica de usuario de acuerdo a los requerimientos funcionales previamente planteados.

LUT	array	imread	PixbufLoader
cvtColor	arange	imencode	

imread	preproc	array
inRange	postproc	
medianBlur	rangos_color	
findContours		
drawContours		
RETR_EXTERNAL		
CHAIN_APPROX_NONE		
COLOR_BGR2HSV		
COLOR_HSV2BGR		

Cuadro No 12: Especificación de los paquetes y sus componentes a utilizar para la construcción de los distintos módulos del software

Fuente: Caicedo C. y Pérez M. (2018)

Por otro lado, se establecieron ciertas convenciones para dar las características de uniformidad y compactación al código. Una de ellas fue el usar el estilo lowerCamelCase para el nombre de las funciones, así como el Snake case para las

variables que tengan nombres compuestos. Igualmente, se nombraron a las constantes de los rangos de los colores con letras mayúsculas.

4.4.1 Actividad I: Desarrollar módulo de preprocesamiento de imágenes

Al desarrollar este módulo, se tomó en cuenta una función que prepara a la imagen de entrada para la posterior inspección de sus colores: la corrección de gamma (ver Anexo F-1, módulo de preprocesamiento). Esta se encarga de aumentar o disminuir el brillo que contenga una imagen, cambiando las intensidades de los pixeles que puedan pertenecer a la escala de grises, con la finalidad de obtener tanto colores que hayan sido muy oscurecidos por la captura de una cámara, como aquellos que por defecto tengan mucha luminosidad.

4.4.2 Actividad II: Desarrollar los métodos para el reconocimiento de colores en imágenes

En relación a la presente actividad, se elaboraron dos algoritmos fundamentales para el reconocimiento y la detección de colores, a los cuales se llama según un orden estricto (ver Anexo F-2, módulo de procesamiento principal). El primero se encarga de filtrar el color y la escala indicada por el usuario a partir de la imagen dada y mediante el método del valor de umbral, retornando como respuesta una imagen binaria que representa a dicho filtro; cabe destacar que también se creó un módulo a parte que contiene los colores con sus respectivas escalas, en donde se incluyeron los valores literales de las regiones correspondientes a la saturación y el valor de cada tonalidad. Tal módulo será requerido como parámetro de entrada para el algoritmo anterior, como para los siguientes a describir.

Con respecto al segundo, se creó una función que toma como parámetro al filtro ya generado, el cual ha sido posteriormente procesado por el algoritmo de suavización o difuminación de píxeles. Dentro de ésta se implementa el algoritmo de análisis estructural topológico de imágenes binarias, con la finalidad de obtener los contornos externos de los segmentos contenidos dentro del filtro. Asimismo, toma la imagen original y el color indicado para los bordes, de tal forma que dibuja los

contornos adquiridos previamente sobre ésta. Debido a que los bordes se pueden confundir con los colores problemáticos, se establecieron varios valores que permitan reflejar el suficiente contraste para su reconocimiento visual, previamente señalado por el usuario.

Posteriormente, se agregó una función adicional que engloba todos los procesos anteriores, incluyendo a aquel que transforma un espacio del color de una imagen a otro distinto, para así poder trabajar los colores de acuerdo al modelo HSV y devolver al modelo RGB, con la intención de sobrescribir el resultado final sobre ésta. Esta última función es llamada desde el módulo de la interfaz gráfica de usuario, recibiendo como parámetros la ruta absoluta de la imagen, el color a reconocer y detectar, la escala respectiva, el color de los bordes de los contornos y el valor de gamma. En dicho espacio de código se agruparon todos los algoritmos anteriores, siguiendo el orden respectivo de cada uno. De igual forma, se adicionaron otros métodos necesarios para continuar amenamente con el flujo de ejecución, como lo son el de lectura de la imagen y el de conversión de formato de la misma a uno compatible con la interfaz.

Debido a que esta función engloba al resto y es la que será llamada por el usuario, se aprovechó el momento para realizar una serie de pruebas que verifiquen la correcta funcionalidad del mismo, utilizando como entrada a las imágenes que corresponden a ambos tipos de muestra, extraídas de la población de la investigación. Para el desarrollo de estas pruebas, se estableció por defecto que el color del borde sería azul debido a que no es uno de los colores problemáticos, así como también el valor de la gamma igual a uno, ya que las imágenes recién tomadas por una cámara poseen dicho valor.

Una vez realizadas estas pruebas, se logró determinar que siete (7) de doce (12) imágenes con elementos geométricos y seis (6) de doce (12) imágenes con elementos no necesariamente geométricos, arrojaron resultados aceptables en varias regiones (ver Anexo H, pruebas funcionales), por lo cual se decidió que ya estaba lista para ser implementada.

4.4.3 Actividad III: Desarrollar módulo de post-procesamiento de imágenes

Para el último módulo, se desarrolló un algoritmo que permite cambiar el formato de una imagen leída con una librería al de otra (ver Anexo F-3, módulo de postprocesamiento); en este caso, se transforma un objeto de la librería numpy a uno de la librería Gdk. Esto se lleva cabo con la finalidad de poder integrar el resultado final del procesamiento con la interfaz gráfica de usuario, de tal manera que se pueda mostrar en una ventana de ésta los colores problemáticos que ya fueron detectados, en base a la imagen inicialmente dada.

4.4.4 Actividad IV: Desarrollar interfaz gráfica de usuario

Dentro de este marco se presentan los detalles de la interfaz de usuario, tomando en consideración diferentes factores como: los colores del sistema, el tipo de letra y la distribución de los contenedores, ya que el objetivo principal del diseño es crear una interfaz gráfica intuitiva y amigable para los usuarios. Para empezar el desarrollo de la interfaz se hizo uso de un wireframe (ver Figura 28) como una herramienta importante en el prototipado del software para definir la posición de los diversos contenedores de la ventana.

Wireframe de la interfaz

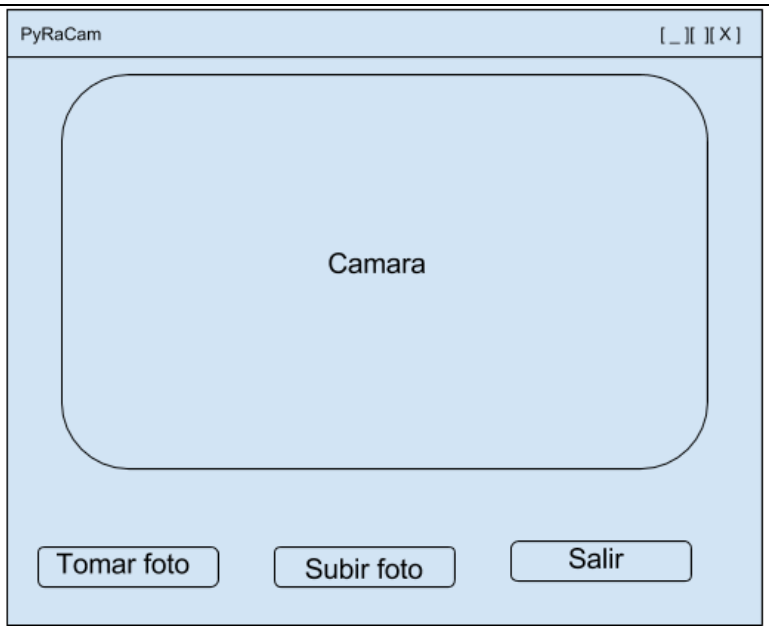


Figura No 28. Fuente: Caicedo C. y Pérez M. (2018)

En este mismo orden de ideas, luego de tener el boceto inicial de la ventana, se procedió a programar el prototipo que se había diseñado anteriormente; para esto se utilizó la librería PyGObject, la cual proporciona un acceso limpio y consistente a la plataforma de software de GNOME y los métodos para el desarrollo de interfaces gráficas con GTK.

El resultado de este primer prototipo fue una ventana segmentada horizontalmente en dos áreas. El lado izquierdo con dos botones que llamarían a cada una de las funcionalidades iniciales del software: el primero para seleccionar una imagen del sistema de archivo local del Raspberry Pi, a través de una nueva ventana (ver Figura 29); el segundo se agregó para dar la orden de procesamiento, el cual genera una ventana que trae como respuesta la imagen procesada (ver Figura 30), cuyos botones brindan una serie de funcionalidades como por ejemplo, guardar la imagen obtenida o acercar la misma. Por otra parte, en el lado derecho se delimitó un área que le permite al usuario ver cuál fue la imagen elegida (ver Figura 31).

Ventana del sistema de archivos

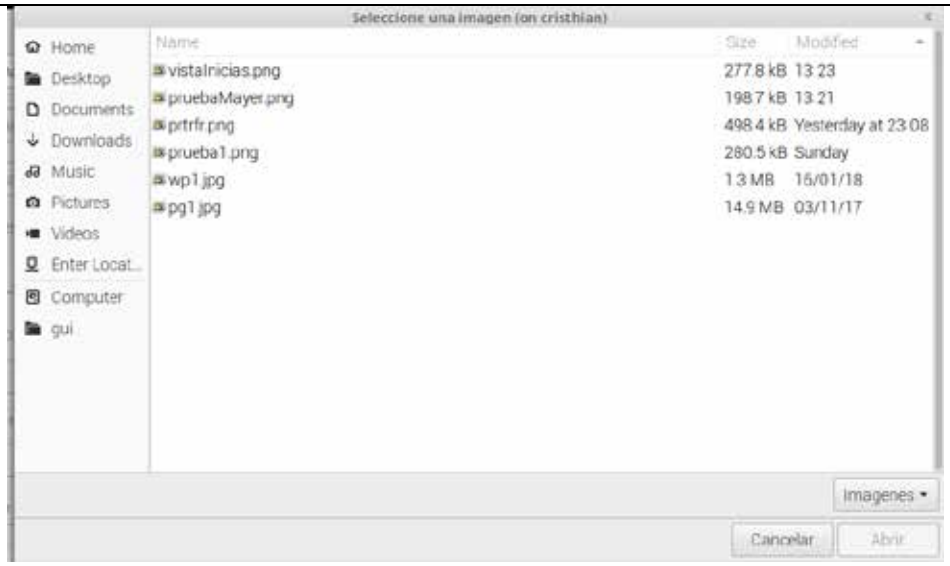


Figura No 29. Fuente: Caicedo C. y Pérez M. (2018)

Ventana con la imagen de respuesta

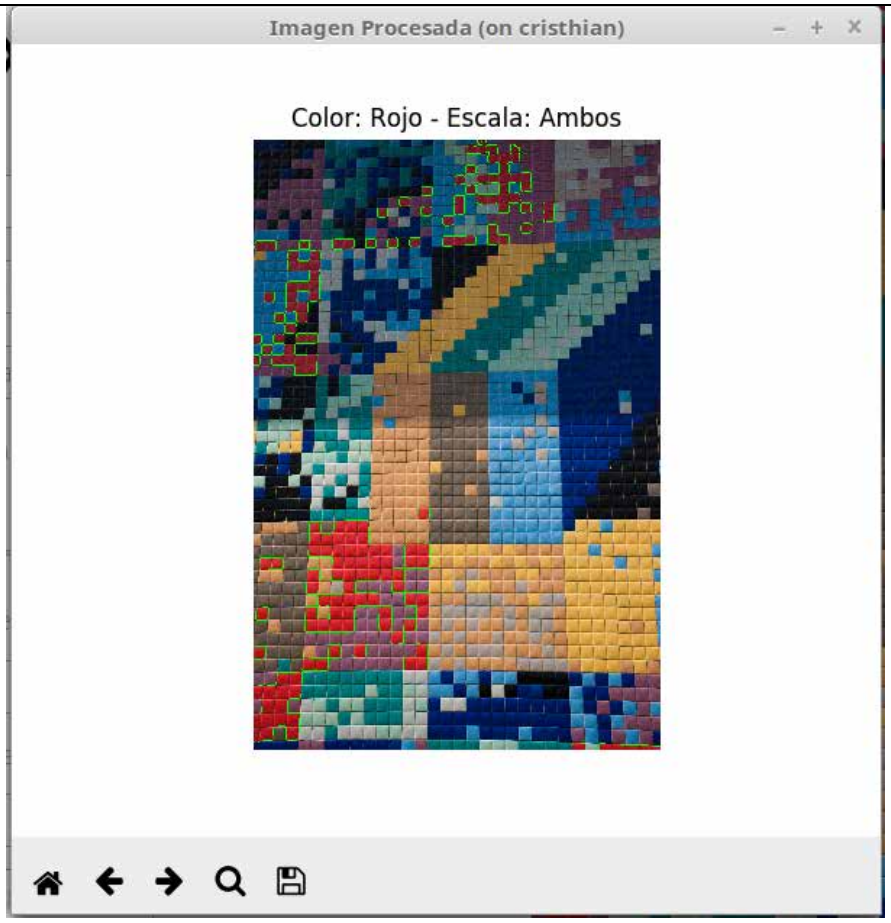


Figura No 30. Fuente: Caicedo C. y Pérez M. (2018)

Prototipo de la interfaz

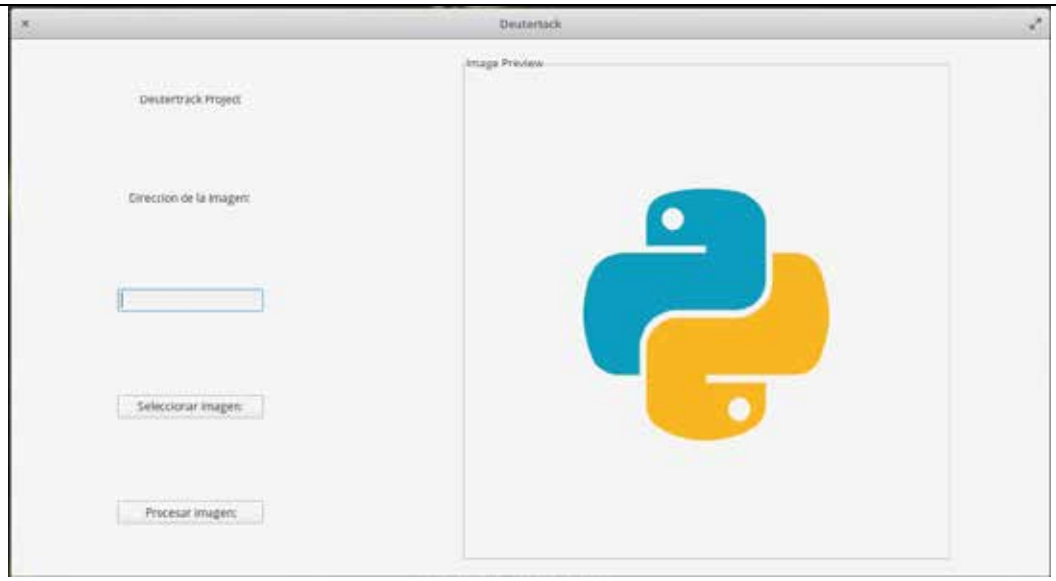


Figura No 31. Fuente: Caicedo C. y Pérez M. (2018)

En la medida en que el desarrollo de la interfaz fue tomando forma, se agregaron otros componentes como el botón para obtener una fotografía del entorno que se está mostrando actualmente desde la cámara del Raspberry Pi, en donde primeramente se abre una nueva ventana con la vista previa de lo que está observando la cámara (ver Figura 32). Esta ventana cuenta con un grupo de botones tipo radio que permiten al usuario seleccionar la calidad de la foto, así como un botón para iniciar la toma de captura por segundo. Además, se agregó un componente deslizador con el rango de valores para seleccionar la gamma que se le aplicará a la imagen escogida y así ajustar el brillo según lo que considere el usuario.

Ventana de vista previa

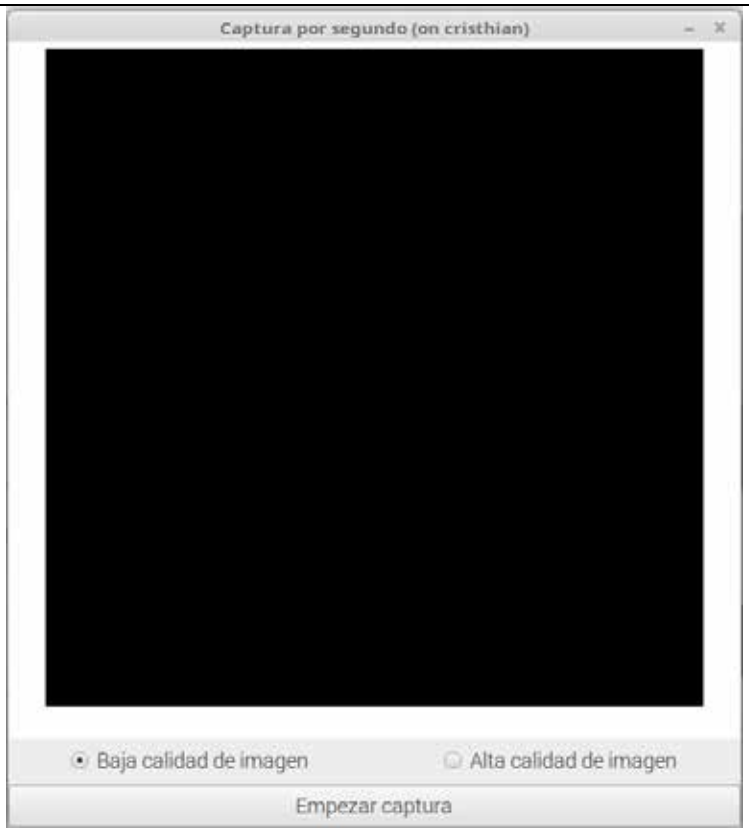


Figura No 32. Fuente: Caicedo C. y Pérez M. (2018)

Al mismo tiempo, se incluyeron las listas para seleccionar el color a filtrar y su respectiva escala, donde el usuario tiene la opción de elegir entre mostrar la escala clara, oscura o ambas y, el color del borde a dibujar; entre los colores disponibles para esta opción se encuentran el rojo, el verde y el azul, además del color blanco y el negro; los primeros tres se escogieron debido a que estos colores se encuentran muy separados entre sí dentro del espectro visible de la luz, a diferencia de los dos últimos, en este caso el blanco y el negro, debido a que representan extremos opuestos. Finalmente, se añadió un botón de ayuda cuya función es mostrar una nueva ventana, la cual hace referencia a la documentación de uso del software. Cada uno de los

elementos descritos conforman la interfaz gráfica de usuario del sistema (ver Figura 33).

Ventana principal del sistema

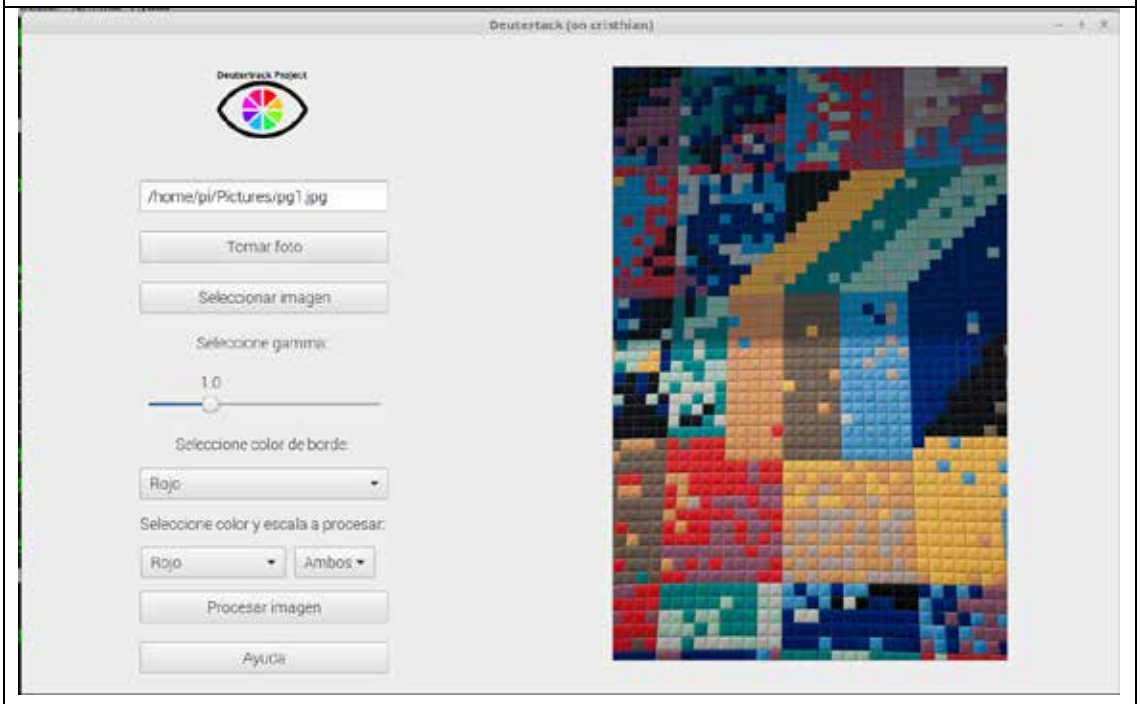


Figura No 33. Fuente: Caicedo C. y Pérez M. (2018)

4.5 Fase V: Verificar los algoritmos para la detección de colores de objetos en una imagen digital

Para concluir con las fases de la investigación, se procedió a realizar pruebas de caja blanca que verificaron la correcta implementación de las unidades internas, las estructuras y sus relaciones. Para esto, se utilizaron los tres tipos de pruebas más comunes: la de la ruta básica, la de ciclos y la de datos. Estas se acomodaron según los criterios de suficiencia basados en las propiedades estructurales del programa, es decir, tanto estructuras lógicas y de control como flujo de datos.

Por último, se instalaron una serie de librerías en el Raspberry Pi que son indispensables para el correcto funcionamiento de la aplicación informática, tanto las dependencias requeridas por OpenCV como aquellas de PyGObject. Seguidamente, se procedió a instalar estas dos últimas y a descargar el código del software del proyecto, el cual está alojado en un repositorio de git publicado en el sitio web Github.

4.5.1 Actividad I: Aplicar pruebas de caja blanca al software

Durante el desarrollo del código, es común por parte de los programadores el no contemplar ciertas situaciones que pueden generarse al ejecutar el programa. Es por esto que, se deben llevar a cabo pruebas unitarias de cada una de las funciones implementadas. Por consiguiente, se realizaron una serie pruebas de tipo caja blanca a cada uno de los módulos del procesamiento de imágenes, las cuales se derivaron según el criterio de rutas básicas, ciclos y datos (ver Cuadro 13), cuya finalidad es detectar las posibles fallas que indiquen los defectos a ser erradicados. A éstas se le proporcionaron unos valores de entrada simulados para cada función, los cuales forman parte de los casos de prueba y son los que indican las salidas deseadas al final de la ejecución de las mismas.

preproc	corrGama	Datos
imgproc	filtImagen	Datos, Ciclos
	obtContornos	Datos
	procImagen	Datos, Rutas
postproc	convPixbuf	Datos

Cuadro No 13: Criterio de pruebas para cada función

Fuente: Caicedo C. y Pérez M. (2018)

En lo tocante al módulo de preprocesamiento, se probó la función de corrección de gamma denominada como corrGamma, la cual se enfocó en el resultado que daba el dato gamma al pasar dos valores esenciales: uno igual a cero y cualquier otro que sea mayor a éste (ver Cuadro 14). Al ejecutar la prueba, se encontró con un error de división entre cero que se originó de la división realizada para obtener el valor inverso de gamma (ver Anexo I-1, casos de prueba para corrGamma). En base a esta falla, se procedió a corregir el conjunto de valores permitidos a seleccionar en la interfaz gráfica de usuario, estableciendo como límite inferior a 0,1 en vez de 0.

array(uint8)	0	array(uint8)	ZeroDivisionError	Falla
array(uint8)	0.4	array(uint8)	array(uint8)	OK

Cuadro No 14: Casos de prueba para la función de corrección de gamma

Fuente: Caicedo C. y Pérez M. (2018)

Con respecto al módulo de procesamiento principal, se llevaron a cabo tres pruebas para los módulos contenidos dentro de éste. La primera hizo referencia a la función de obtención de contornos (ver Cuadro 15), en donde se quiso verificar si se podían utilizar sin problema todos los colores de los bordes, determinados a través de las llaves de un diccionario de datos. La segunda, se realizó para los casos de prueba del método principal de procesamiento de una imagen denominado como procImagen (ver Cuadro 16), en donde se buscó tanto pasar por todas las rutas independientes de éste fragmento de código según su complejidad ciclomática (ver Anexo J, grafo de flujo de control), como verificar que se retorna correctamente un objeto Pixbuf. Se obtuvo que ambas funciones lograron pasar sin error las pruebas de caja blanca (ver Anexo I-2 y I-3, casos de prueba para obtContornos y procImagen respectivamente).

array(uint8)	Rojo	(0, 0, 255)	(0, 0, 255)	OK
array(uint8)	Verde	(0, 255, 0)	(0, 255, 0)	OK
array(uint8)	Azul	(255, 0, 0)	(255, 0, 0)	OK
array(uint8)	Negro	(0, 0, 0)	(0, 0, 0)	OK
array(uint8)	Blanco	(255, 255, 255)	(255, 255, 255)	OK

Cuadro No 15: Casos de prueba para la función de obtención de contornos

Fuente: Caicedo C. y Pérez M. (2018)

prueba.png	ROJO	Claro	Rojo	0.5	Pixbuf	Pixbuf	OK
prueba.png	ROJO	Oscuro	Verde	0.5	Pixbuf	Pixbuf	OK
prueba.png	ROJO	Oscuro	Azul	0.5	Pixbuf	Pixbuf	OK
prueba.png	ROJO	Ambas	Negro	1	Pixbuf	Pixbuf	OK

Cuadro No 16: Casos de prueba para la función principal de procesamiento de una Imagen

Fuente: Caicedo C. y Pérez M. (2018)

En relación a la tercera prueba asociada a la función de creación de filtros llamada filtImagen, se hizo énfasis en la lista de rangos de colores pasados a la misma, así como en el ciclo for involucrado en el recorrido de esta lista. Los casos de pruebas se basaron principalmente en dicho ciclo, planteando varias condiciones

mínimas: saltarlo por completo, solo pasar una vez a través del mismo, permitir menos iteraciones de las que sean posibles y proveer una lista que tenga n cantidad de elementos impar (ver Cuadro 17). Gracias a los resultados de la prueba, se pudo detectar que habían errores provocados por: la inapropiada definición de una variable y por la indeseada indexación de un rango que no sea de longitud par (ver Anexo I-4, casos de prueba para filtImagen). Después de haber analizado las fallas generadas, se procedieron a eliminar los defectos en el código.

array(uint8)	[ ]	array(uint8)	AttributeError	Falla
array(uint8)	[0,0,0], [0,0,255]	array(uint8)	array(uint8)	OK
array(uint8)	[0,0,0], [0,127,0], [0,255,0]	array(uint8)	IndexError	Falla
array(uint8)	[0,0,0], [0,127,0], [0,200,0], [0,255,0]	array(uint8)	array(uint8)	OK

Cuadro No 17: Casos de prueba para la función de creación de filtros a través de una Imagen

Fuente: Caicedo C. y Pérez M. (2018)

Como prueba final, se tomó en cuenta a la función que cambia el formato de una imagen definida como convPixbuf, la cual se basa en un arreglo de tipo entero sin signo de tamaño 8 bits. A partir de este arreglo, se esperaba obtener una instancia de la clase Pixbuf de GdkPixbuf (ver Cuadro 18), la cual será retornada al módulo de la

interfaz gráfica para que esta la muestre al usuario como respuesta. El resultado de la ejecución de la prueba arrojó que la conversión de los tipos de datos no produjo ningún error (ver Anexo I-5, casos de prueba para convPixbuf).

array(uint8)	Pixbuf	Pixbuf	OK

Cuadro No 18: Caso de prueba para la función de cambio de formato de una imagen en OpenCV a GdkPixbuf

Fuente: Caicedo C. y Pérez M. (2018)

4.5.2 Actividad II: Instalar el software en el Raspberry Pi

Para empezar, se procedió a instalar una imagen del sistema operativo liviano en la tarjeta SIM que se introdujo al Raspberry Pi, en este caso, la distribución de Raspbian Jessie versión 8. Seguido a esto, se procedió a ejecutar una serie de comandos para poder descargar e instalar las librerías necesarias (ver Anexo K-1, instalación de dependencias), tanto para configuración del proceso de construcción de unas como para el uso interno de las subsecuentes. Asimismo, solo se descargó los paquetes de Python 3, debido a que esta es la versión que se usó a lo largo del proyecto. Consecutivamente, se realizó la descarga de OpenCV desde su repositorio público en github, para luego ejecutar unos comandos de configuración previa, compilar su código fuente e instalarlo (ver Anexo K-2, instalación de OpenCV).

Por otro lado, para la instalación de PyGObject es necesario tener el paquete gi; teniendo en cuenta que el Raspberry Pi trabaja con la distribución de Linux Raspbian y esta es basada en Debian, se debe instalar mediante el uso del sistema de gestión de paquetes de este sistema operativo. Se debe ejecutar el comando apt install (ver

Figura 34) en el terminal del ordenador, lo cual instalará las librerías necesarias para que el dispositivo pueda abrir una ventana de GTK.

Comando para instalar PyGObject en el Raspberry Pi

```
$ sudo apt install python-gi python-gi-cairo gir1.2-gtk-3.0
```

Figura No 34. Fuente: Caicedo C. y Pérez M. (2018)

Estas librerías forman parte de Glib, la biblioteca principal utilizada para construir en GTK+ y GNOME. La primera, python-gi, contiene el generador de enlace Python 2.x para bibliotecas que admiten gobject-introspection, por ejemplo, que envían un paquete gir1.2-X; con estos paquetes las bibliotecas se pueden usar desde Python. Seguidamente está python-gi-cairo: este paquete contiene los enlaces de Python Cairo para GObject, es utilizado principalmente por otros enlaces para mapear los objetos GObjects con Python; asimismo, se instaló gir1.2-gtk-3.0, la cual contiene los enlaces a la librería de interfaz de usuario de GTK+. Finalmente, se debió instalar la librería de matplotlib para poder trabajar con las imágenes ya procesadas y el ambiente de GTK (ver Figura 35), para así poder mostrar en una ventana la respuesta final generada por el sistema.

Comando para instalar matplotlib en el Raspberry Pi

```
sudo apt-get install python3-matplotlib
```

Figura No 35. Fuente: Caicedo C. y Pérez M. (2018)

CAPÍTULO V
CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

En el presente trabajo de investigación se desarrolló un software de reconocimiento de colores de objetos mediante el procesamiento de imágenes digitales para mejorar la percepción cromática a las personas que padecen de deuteranomalía, a partir del cual se sentaron las bases de una área de investigación poco explorada en la Universidad José Antonio Páez, en las cuales futuros proyectos pudiesen basar su estudio utilizando este trabajo como antecedente.

Adicional a lo anterior, se determinó el espectro de color y los colores de confusión correspondientes a las personas con deuteranomalía, gracias al fácil manejo de estos a través del modelo del color HSV. Asimismo, se observó que la delimitación de rango en las gamas de los colores no fue suficiente para capturar regiones altamente exactas dentro de las imágenes, debido a que en la mayoría de los casos los segmentos cromáticos en éstas no son homogéneos en sus tonalidades.

Seguidamente, se establecieron los requerimientos funcionales y no funcionales, en donde el uso de una metodología ágil permitió que el desarrollo del software se adapte con facilidad a los cambios de requerimientos que surgieron a través de la construcción de los módulos de procesamiento y ejecución de pruebas, los cuales contribuyeron a complementar las especificaciones del sistema, ajustándolo a las necesidades del usuario.

Por otro lado, se analizaron los métodos y técnicas existentes en la librería OpenCV orientados a detectar los colores de un objeto en una imagen digital. En base a estos, se halló que el reconocimiento de un color a través de un umbral de valores binarios es de suma importancia, tanto para la detección de éste mediante el análisis de segmentos comprendidos por pixeles, como para la delimitación de los bordes de

éstos. De igual manera, se apreció que existe una diferencia en la percepción del brillo entre una cámara y el ojo humano, por lo que se implementó la corrección de gamma. Con respecto a las pruebas para los algoritmos correspondientes a estos métodos, se demostró que los tiempos de ejecución fueron cortos y su uso de memoria fue razonable. Por consecuente, se evidenció que es posible desarrollar software de este tipo para ordenadores de placa reducida, como lo es el Raspberry Pi.

En el mismo orden de ideas, se notó que la construcción de la arquitectura del software con Python fue sencilla y práctica al mismo tiempo, siendo esta modular para brindar escalabilidad y así poder agregar nuevas funcionalidades en futuros proyectos. También de manera concurrente, la prueba funcional ejecutada en grupo para las funciones de procesamiento demostró que se logró el objetivo general de la investigación, a un grado aceptable referente a exactitud, obteniendo unos resultados similares tanto en imágenes con elementos geométricos, como en aquellos que no necesariamente los contengan.

Asimismo, se determinó que el problema de exactitud en los resultados está vinculado a la distribución de luz blanca en la composición interna de los objetos, generada por el entorno de manera natural o artificial. Igualmente, la calidad de una imagen digital puede afectar en este mismo sentido, de tal forma que: esta contenga mucho ruido y separe las intensidades de los pixeles considerablemente o; porque haya sido tomada en movimiento y su representación sea muy abstracta.

Con el motivo de asegurar que el anterior programa de software no contenga errores a la hora de ejecución, se verificaron los algoritmos implementados para la detección de colores de objetos. Para esto, las pruebas de caja blanca ayudaron a descubrir fallas no contempladas y contribuyeron a la mejora del código fuente del sistema. Sin embargo, hay que tener en cuenta muchos criterios para poder establecer los casos de prueba correctamente, por lo que se ve en la necesidad de clasificar éstos para reducir sus números.

5.2 Recomendaciones

Dentro de la realización de un proyecto tan ambicioso como lo fue éste, siempre se desea que exista una mejora continua del mismo, por lo tanto, con el fin de mantener la escalabilidad del software y ampliar el alcance de la investigación, se conciben las siguientes recomendaciones:

- Segmentar de manera más precisa y detallada las escalas de los colores, ya sea con la ayuda de un estudio profundo cuantitativo, cualitativo o ambos; se indica esto con la finalidad de poder identificar con mayor exactitud los colores claros y los colores oscuros, así como una combinación de ambos.
- Continuar el desarrollo de la presente aplicación informática al añadir nuevas funciones y características, con el propósito de que se tome en cuenta el resto de las deficiencias visuales del color existentes.
- Utilizar un estándar de asignación de nombres de colores que permita a los usuarios comprender las diferentes variaciones de los mismos, basándose en un lenguaje familiar a este. Por ejemplo, incluir a los colores pasteles.
- Implementar una función que se encargue de eliminar el ruido en las imágenes, de manera eficaz y eficiente.
- Jugar con los parámetros de entrada de la función para la difuminación o suavizamiento de píxeles, de tal forma que contemple segmentos con distintas formas, con la ayuda de un algoritmo inteligente.
- Hacer una mayor inclusión y estudio del efecto añadido que aporta la luz blanca a los colores, tanto en magnitud como de tipo.
- Usar el software como prototipo para ser llevada a otras plataformas como la web, dispositivos móviles u otros ordenadores de placa reducida.
- Orientar el desarrollo del software a que sea más centrado en el usuario, mediante una instalación de éste directa y sencilla, por ejemplo, en forma de un ejecutable.

- En caso de que se desee sacar al mercado una versión de este software, ya sea libre o propietario, se recomienda añadirle una licencia acorde a las intenciones del proveedor.

REFERENCIAS

Abadin, D. y Álvarez, L. (2014). **Tecnologías de apoyo, mercado y nuevos sistemas de información.** Extraído desde: http://www.ceapat.es/InterPresent1/groups/imserso/documents/binario/tecnologiaapoyo.pdf

Adams, D. (2003). **Color Systems.** Sistemas del color. Extraído desde: http://www.davidadamsonline.com/newsletter_young_helmholtz.htm

Adobe Systems Incorporated (2000). **Color Models CIELAB.** Modelos del color CIELAB. Extraído desde: http://dba.med.sc.edu/price/irf/Adobe_tg/models/cielab.html

Adrian Rosebrock (2015). **OpenCV Gamma Correction.** Correción de gamma con OpenCV. Extraído desde: https://www.pyimagesearch.com/2015/10/05/opencv-gamma-correction/

Ambler, S. (2005). **The Agile Unified Process (AUP).** El Proceso Unificado Ágil (AUP). Extraído desde: http://www.ambysoft.com/unifiedprocess/agileUP.html

Anusha, I. (2016). **Object Detection using blob tracing.** Detección de objeto usando el rastreo de masa amorfa. Extraído desde: http://layer0.authentise.com/object-detection-using-blob-tracing.html

Arévalo, J. (2005). **La librería de visión artificial OpenCV aplicación a la docencia e investigación.** Extraído desde: http://mapir.isa.uma.es/varevalo/drafts/arevalo2004lva1.pdf

Arias, F. (2006). **El Proyecto de Investigación.** Introducción a la metodología científica (5a ed.). Venezuela, Caracas: Editorial Episteme

Blackwell, C. (2013). **Color Vision 2: Color Matching.** La Visión del Color 2: Coincidencia de Color. Extraído desde: https://www.youtube.com/watch?v=82ItpxqPP4I

Carbonell, M. (2013). **Interfaz gráfica de usuario (GUI).** Extraído desde: http://www.fundeu.es/escribireninternet/interfaz-grafica-de-usuario-gui/

Castillero, O. (2017). **Daltonismo: causas, síntomas, tipos y características.** Extraído desde: https://psicologiaymente.net/clinica/daltonismo

Catalina, D. (2014). **Color.** Extraído desde: http://slideplayer.es/slide/2302666/8/images/23/CIE+XYZ+Un+color+se+define+por+los+componentes+de+X,+Y+y+Z..jpg

Centro de Escritura Javeriano (2018). **Normas APA** (6a ed.). Extraído desde: https://www.um.es/documents/378246/2964900/Normas+APA+Sexta+Edici%C3%B3n.pdf/27f8511d-95b6-4096-8d3e-f8492f61c6dc

Color espectral (s.f). Extraído desde: http://www.glosariografico.com/color_espectral

Colour Blind Awareness (s.f). **Types of Colour Blindness.** Tipos de ceguera del color. Extraído desde: http://www.colourblindawareness.org/colour-blindness/types-of-colour-blindness/

DeValois, K. & Webster, A. (2011). **Color Vision.** Visión del color. Extraído desde: http://www.scholarpedia.org/article/Color_vision

Documentación de OpenCV. (2016). **How OpenCV-Python Bindings Works?.** ¿Cómo funcionan los enlaces entre OpenCV y Python?. Extraído desde: https://docs.opencv.org/3.2.0/da/d49/tutorial_py_bindings_basics.html

Farnsworth, D. (1943). **The Farnsworth-Munsell 100-hue and dichotomous tests for color vision.** Los 100 tonos de Farnsworth-Munsell y las pruebas dicotómicas para la visión del color. Extraído desde: https://www.ncbi.nlm.nih.gov/books/NBK217816/

Flück, D. (2016). **Color Blind Essentials.** Fundamentos de la ceguera del color. Extraido desde: http://www.color-blindness.com/color-blind-essentials

Gato, J. (s.f). **Las ondas y sus características**. Extraído desde: http://teleformacion.edu.aytolacoruna.es/FISICA/document/fisicaInteractiva/Ondasbachillerato/ondasCaract/ondas-Caract_indice.htm#long

Georgiev, D., Alyoshin, V., & Krastev, S. (2017). **Statistics and How many people are Color blind.** Las estadísticas y cuantas personas son ciegas al color. Extraído desde: https://visiontechnology.co/statistics/

Giangrandi, I. (2004). **Visible spectrum**. Espectro Visible. Extraído desde: http://www.giangrandi.ch/optics/spectrum/spectrum.shtml

González, V. (2014). **El Sistema Visual Humano.** Extraído desde: https://w3.ual.es/~vruiz/Docencia/Apuntes/Perception/Sistema_Visual/index.html

Howard, J. (2016). **What colors mixed with blue for another color?.** ¿Qué colores se obtienen al mezclar con el color azul?. Extraído desde: https://www.quora.com/What-colors-mixed-with-blue-for-another-color

Hurado, J. (2000). **Metodología de la Investigación. Guía para la comprensión holística de la ciencia** (4a ed.). Venezuela, Caracas: Editorial Quiron

Instituto Tecnológico de Celaya (s.f.). **Arquitecturas de las computadoras.** Extraído desde: http:// www.iqcelaya.itc.mx/~vicente/Programacion/Arquitectura.pdf

Kalloniatis, M. & Luu, C. (2005). **The Perception of Color.** La percepción del color. Extraído desde: https://www.ncbi.nlm.nih.gov/books/NBK11538/

Khatua, D. (2015). **Which is better for more efficient and quicker Image-Processing, MATLAB, OpenCV, or Octave?.** ¿Qué es mejor para un procesamiento de imágenes más eficiente y más rápido, MATLAB, OpenCV u Octave?. Extraído desde: https://www.quora.com/Which-is-better-for-more-efficient-and-quicker-Image-Processing-MATLAB-OpenCV-or-Octave

Kokotailo, R. & Kline, D. (2002). **Theories of Colour Perception.** Teorías de la percepción del color. Extraído desde: http://psych.ucalgary.ca/PACE/VA-Lab/colourperceptionweb/default.htm

MacEvoy, B. (2005). **Modern color models.** Modernos colores de modelos. Extraído desde: http://www.handprint.com/HP/WCL/color7.html

MacEvoy, B. (2015). **Light and the eye.** La luz y el ojo. Extraído desde: http://www.handprint.com/HP/WCL/color1.html

Madore, D. (s.f). **Colors and Colorimetry.** Colores y colorimetría. Extraído desde: http://www.madore.org/~david/misc/color/#cie

Martinez, D. (2012). **Conoce todo acerca de la tiflotecnología.** Extraído desde: http://www.salud180.com/adultos-mayores/conoce-todo-acerca-de-la-tiflotecnologia

Merriam W. (s.f). **Medical Definition of photopigment.** Definición médica de fotopigmento. Extraído desde: https://www.merriam-webster.com/medical/photopigment

Michelone, M (2016). **La cámara de Raspberry Pi tiene ahora 8 megapíxeles de resolución.** Extraído desde: https://www.unocero.com/noticias/la-camara-de-raspberry-pi-tiene-ahora-8-megapixeles-de-resolucion/

National Eye Institute (2015). **Facts About Color Blindness.** Información acerca de la ceguera del color. Extraído desde: https://nei.nih.gov/health/color_blindness/facts_about

National Research Council (US) Committee on Vision (1981). **Appendix Understanding test design.** Apéndice Entendiendo el diseño de prueba. Extraído desde: https://www.ncbi.nlm.nih.gov/books/NBK217816/

Nave, O. (2000). **Bastones y Conos.** Extraído desde: http://hyperphysics.phy-astr.gsu.edu/hbasees/vision/rodcone.html

Nave, O. (2000). **The C.I.E color space.** El color del espacio C.I.E. Extraído desde: http://hyperphysics.phy-astr.gsu.edu/hbase/vision/cie.html

Nave, O. (2000). **Spectral Power Distribution.** Distribución de potencia espectral. Extraído desde: http://hyperphysics.phy-astr.gsu.edu/hbase/vision/spd.html#c1

Palella, S. y Martins, F. (2012). **Metodología de la investigación Cuantitativa.** Venezuela, Caracas: Fondo Editorial de la Universidad Pedagógica Experimental Libertador

Pérez, C. (2006). **Visión, Luz y Color.** Extraído desde: http://personales.unican.es/perezvr/pdf/Vision%20Luz%20y%20Color.pdf

Pérez, G. (s.f). **Espectro electromagnético.** Extraído desde: https://www.espectrometria.com/espectro_electromagntico

Pérez, J. y Merino, M. (2012). **Definición de Visión.** Extraído desde: http://definicion.de/vision/

Plogging Dev (2016). **Performance measurement in Python 3.** Medición del desempeño en Python 3. Extraído desde: https://www.ploggingdev.com/2016/12/performance-measurement-in-python-3/

Pressman, R. (2010). **Ingeniería del Software.** Un enfoque práctico (7a ed.). México, D.F.: Editorial McGraw-Hill

Radiaciones electromagnéticas (s.f). Extraído desde: http://astrojem.com/radiacionelectromagnetica.html

Real Academia Española (s.f). **Diccionario de la real Academia Española.** Extraído desde: http://dle.rae.es/?w=diccionario

Shipman, J. (2012). **The hue-saturation-value (HSV) color model.** El modelo del color tonalidad-saturación-valor (TSV). Extraído desde: http://infohost.nmt.edu/tcc/help/pubs/colortheory/web/hsv.html

Smith, B. (2016). **Setting up Raspberry Pi.** Configurando el Raspberry Pi. Extraído desde: https://github.com/bwasmith/Rhew-R-Pi/wiki/Setting-up-Raspberry-Pi

Sommerville, I. (2005). **Ingeniería del Software** (7a ed.). España, Madrid: Editorial Pearson Addison Wesley

Universo de Formulas. (2017). **Muestreo por cuotas**. Extraído desde: http://www.universoformulas.com/estadistica/inferencia/muestreo-cuotas/

Vita, M. (2016). **Trichromatic and Opponent Process theory.** Teoría Tricromática y del proceso oponente. Extraído desde: https://storify.com/MariaVita1/opponent-process-theory

Wang, R. (2013). **The RGB Cube and Cone Models.** El cubo RGB y los modelos de cono. Extraído desde: http://fourier.eng.hmc.edu/e180/lectures/color1/node29.html

Zapata, W. (s.f). **Los colores: Primarios, secundarios, terciarios.** Extraído desde: https://wilmardejzapatajaramillo.wordpress.com/2012/05/20/los-colores-primarios-secundarios-terciarios/

ANEXO A

Instrumento de investigación

REPÚBLICA BOLIVARIANA DE VENEZUELA
UNIVERSIDAD JOSÉ ANTONIO PÁEZ
FACULTAD DE INGENIERÍA
ESCUELA DE COMPUTACIÓN

ENTREVISTA

Objetivo: Conocer información relevante para los desarrolladores del software, de tal manera que se pueda construir un sistema que cumpla con los objetivos de la investigación.

1. When doing color detection of an object in image processing. ¿Should I work with the color independently from the object?

_____

_____

_____

___

2. In color detection of an object within an image. ¿Would you consider dealing with object recognition or detection?

_____

_____

_____

___

3. ¿Is image segmentation based on color good for color detection of an object? ¿Why?

_____

_

4. ¿Which color model is the ideal for color detection in image processing?

_____

_____

_____

_____

____

5. ¿Which technique would you recommend for color detection of an object in image processing?

_____

_____

_____

_____

____

ANEXO B

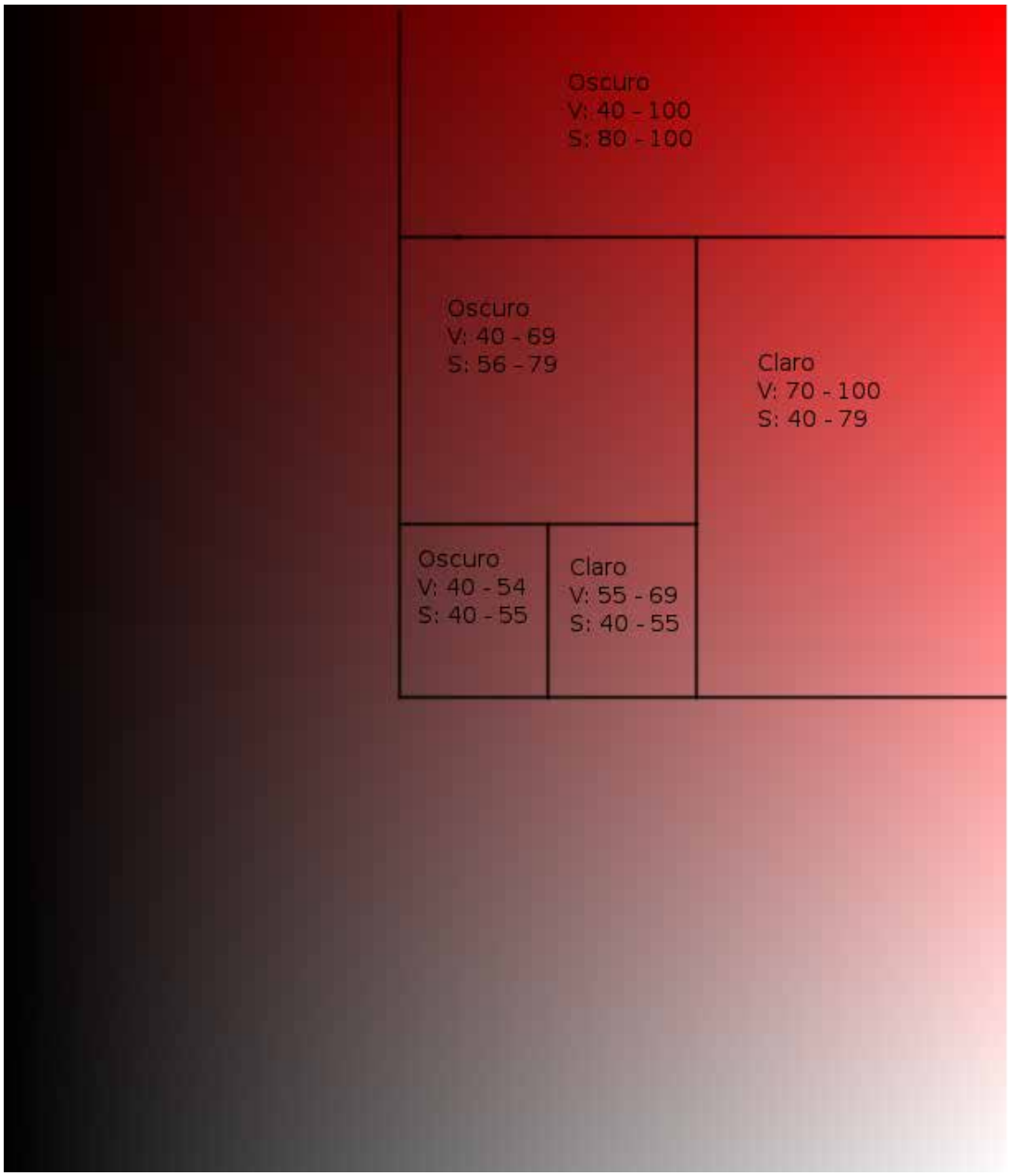
Lista de Cotejo

Aspecto a tomar en cuenta	Si	No
La imagen posee figuras geométricas		
La imagen no necesariamente posee figuras geométricas		
La imagen posee colores espectrales		
La imagen posee colores no espectrales		
La imagen está compuesta parcialmente por el color amarillo		
La imagen está compuesta parcialmente por el color azul		

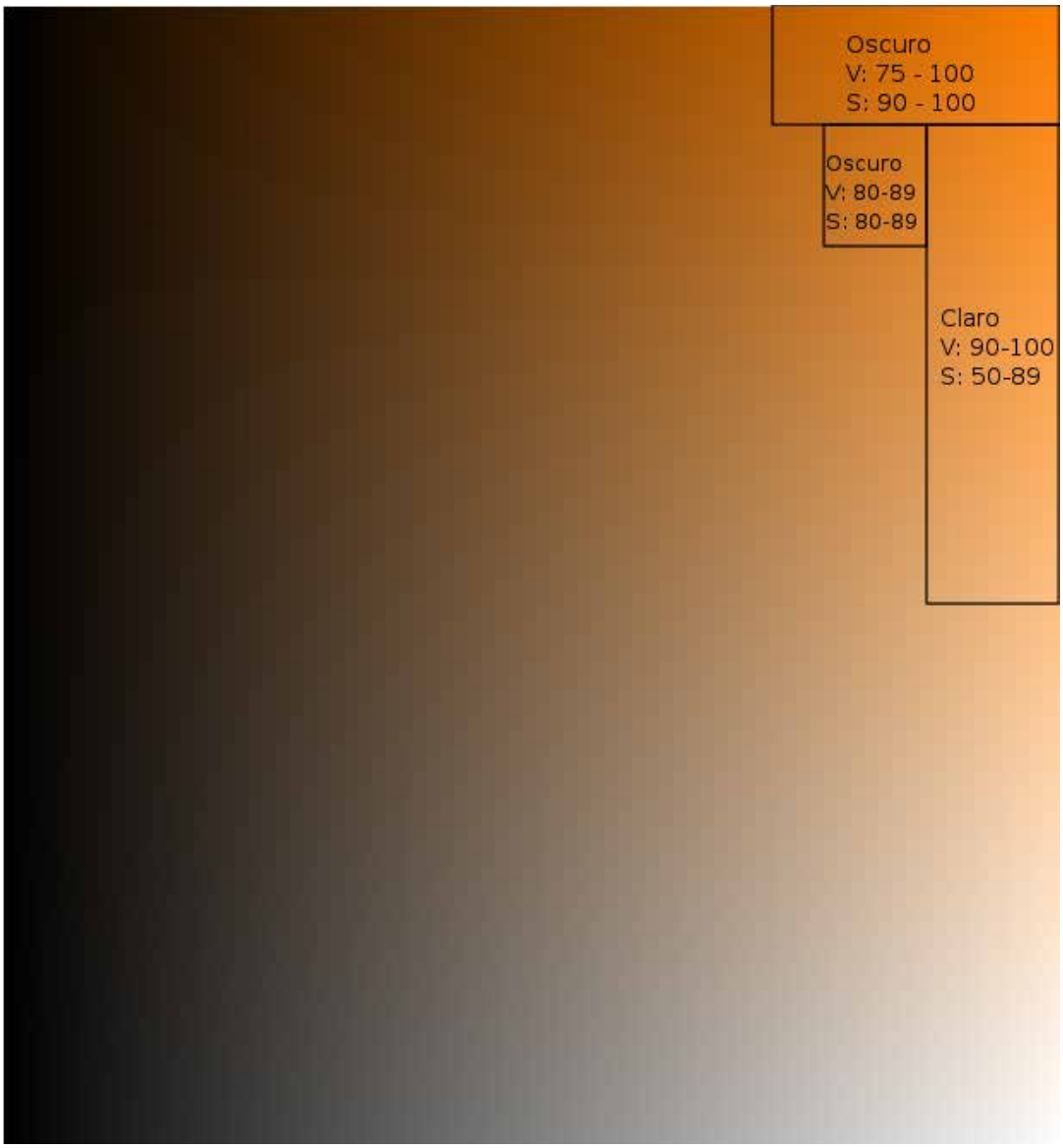
ANEXO C

Gamas de Colores

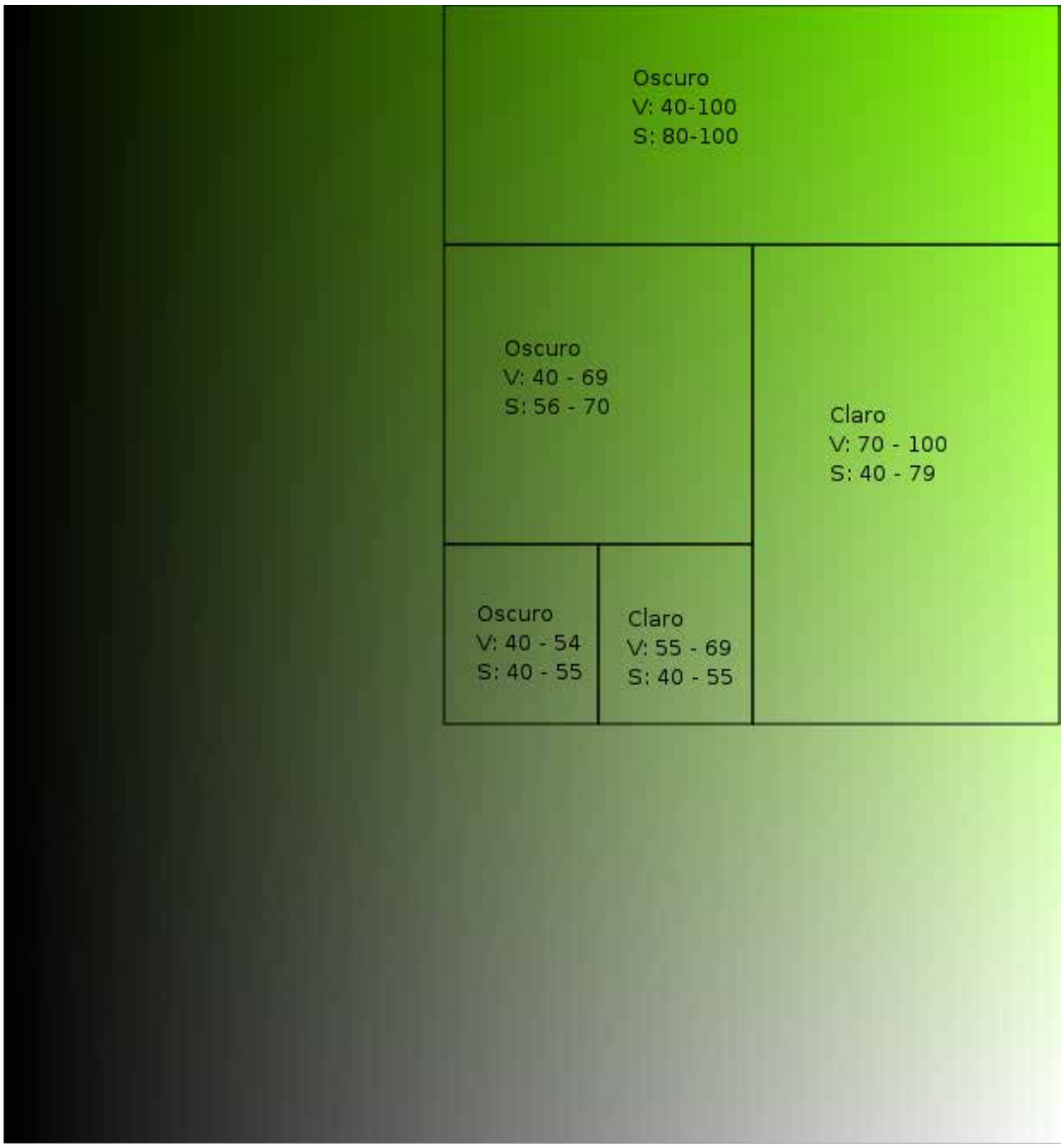
Gama del color rojo



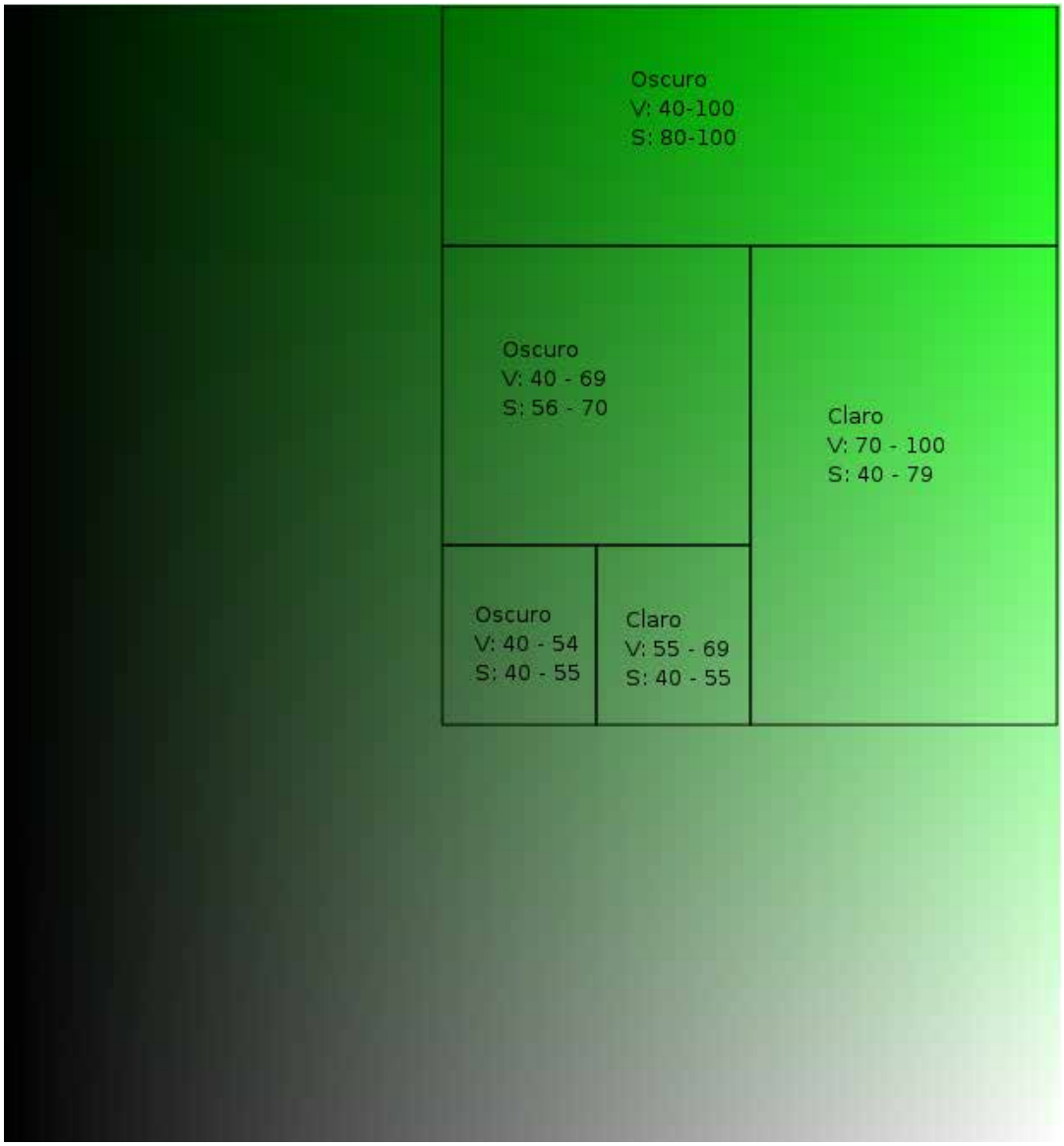
Gama del color naranja



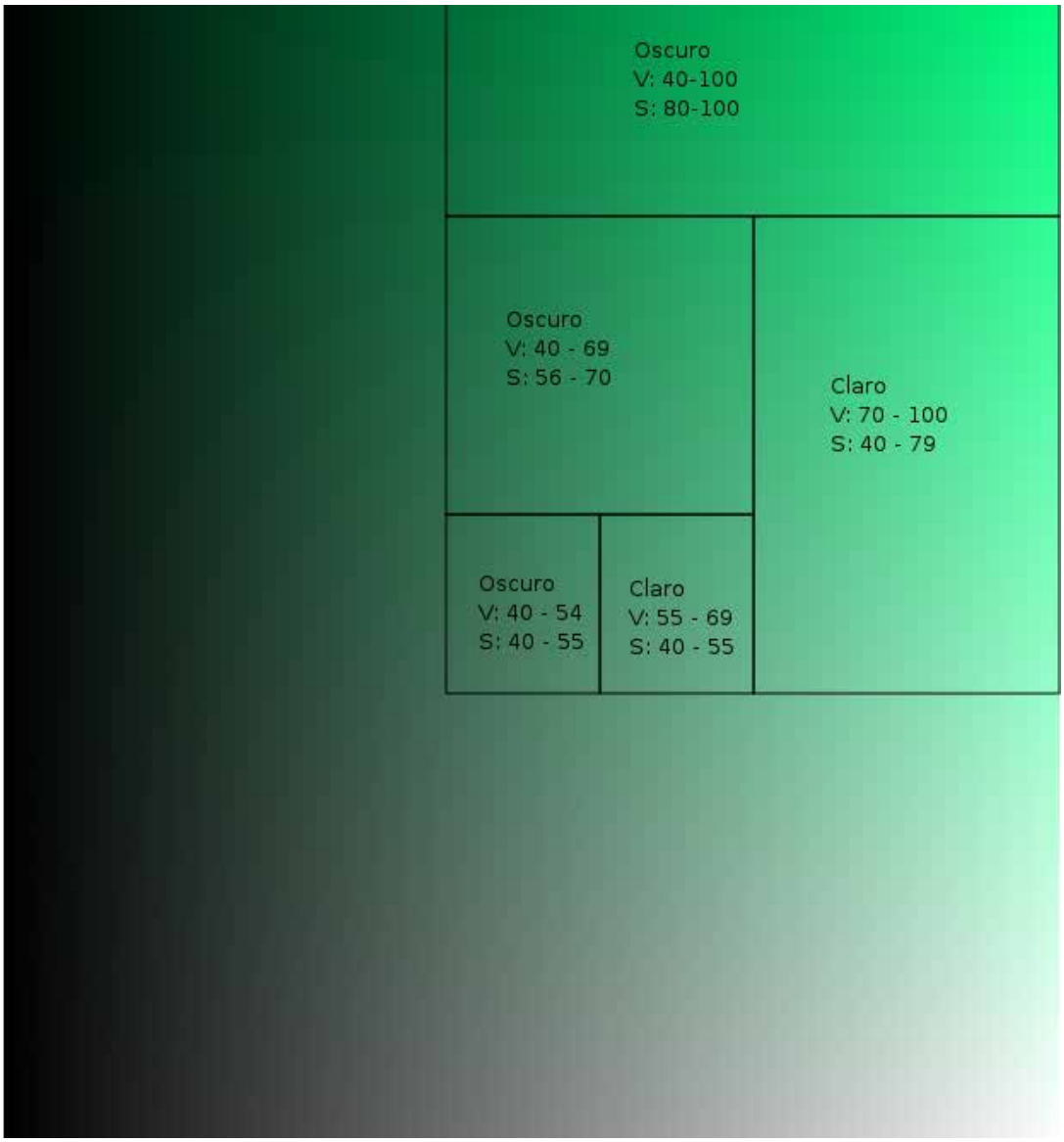
Gama del color Verde-Amarillo



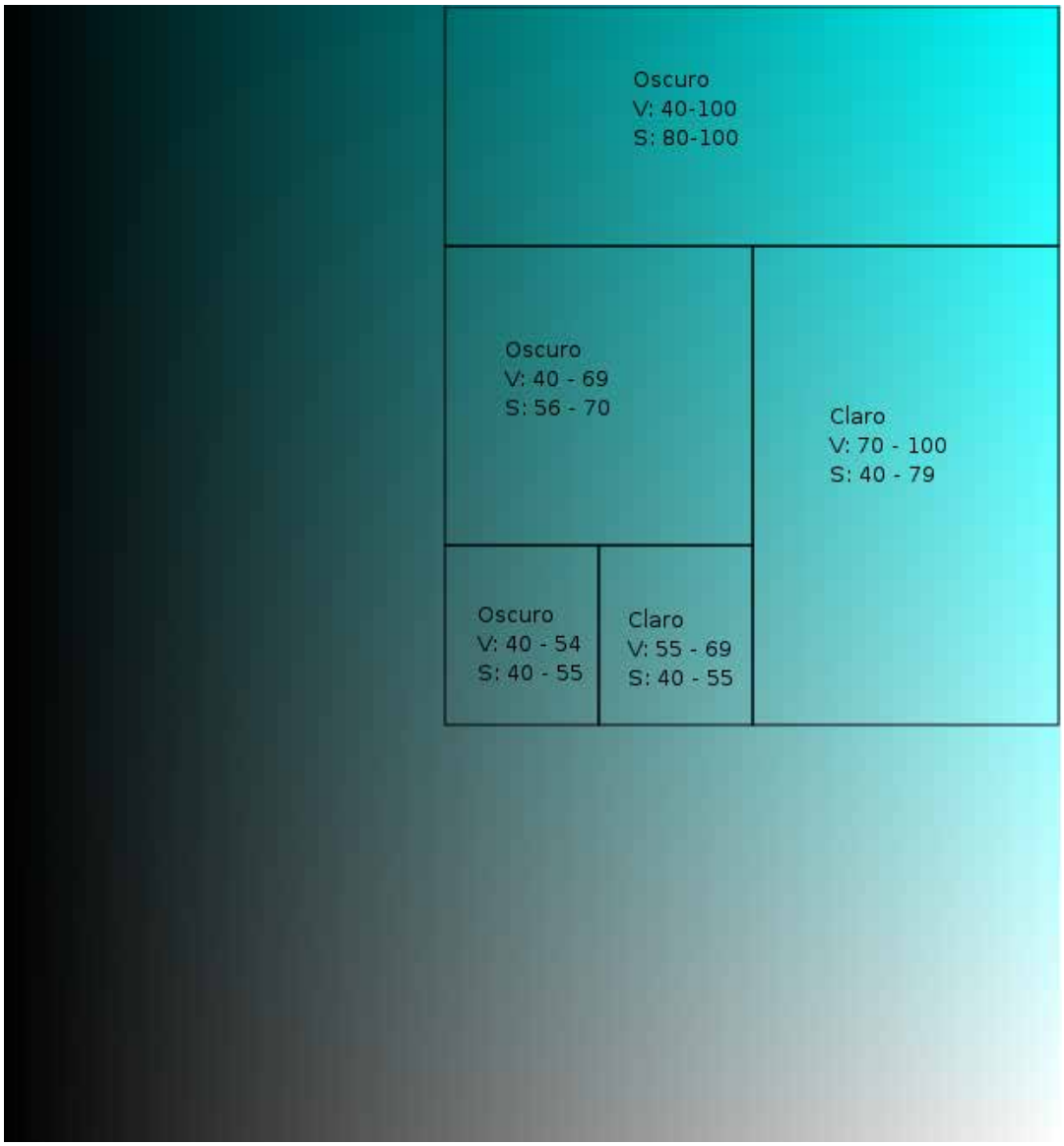
Gama del color Verde



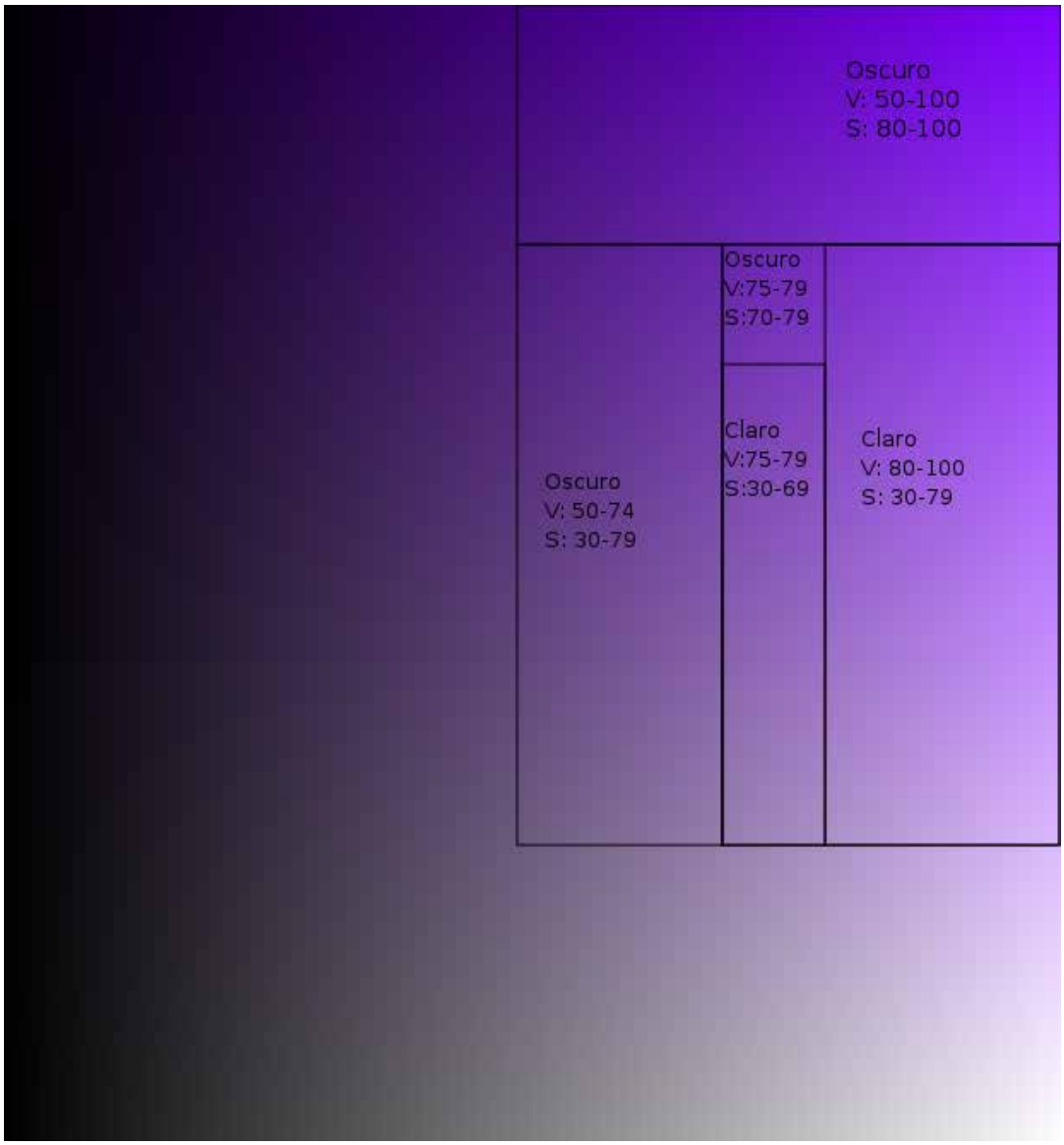
Gama del color Verde-Primavera



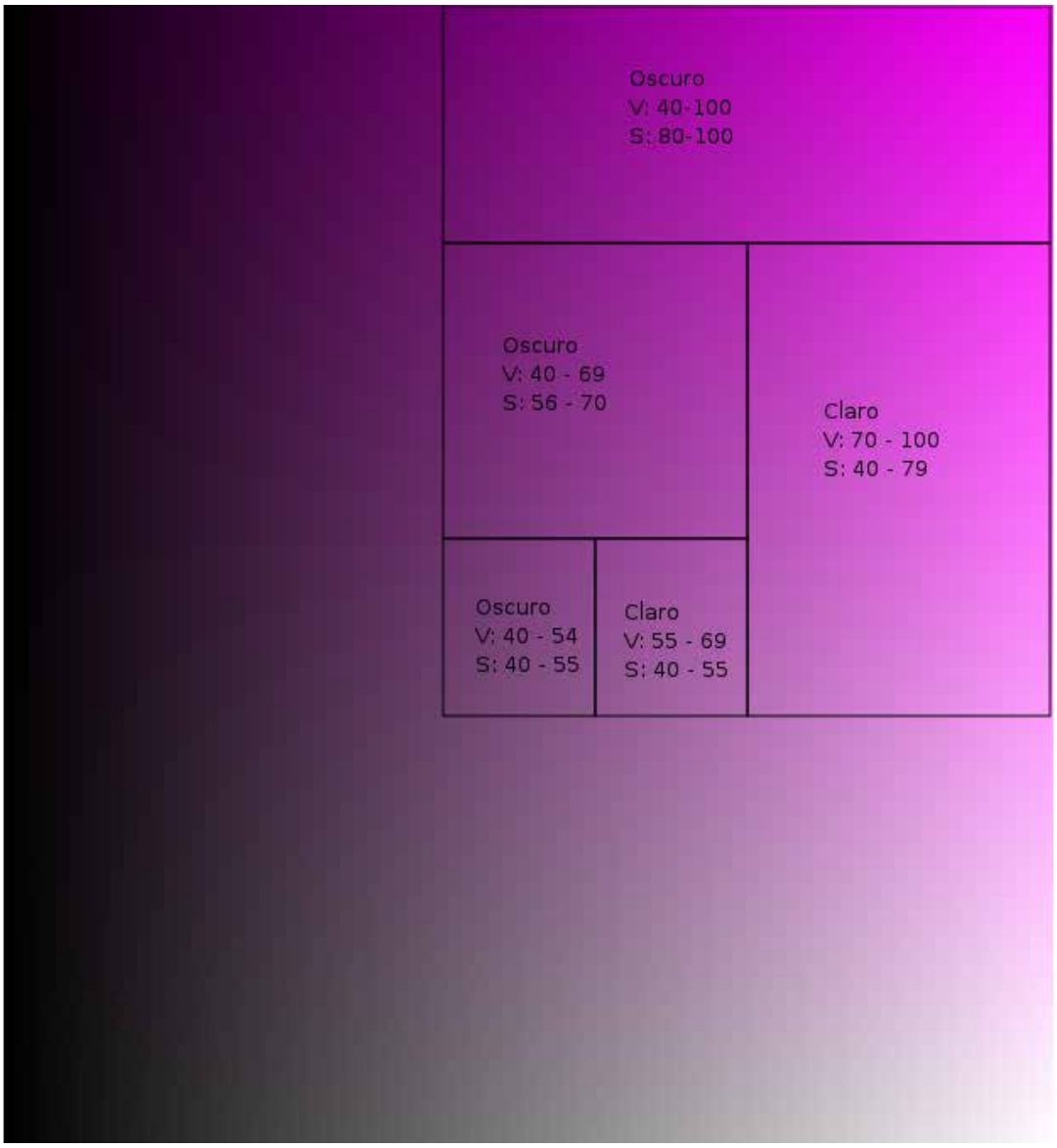
Gama del color Cian



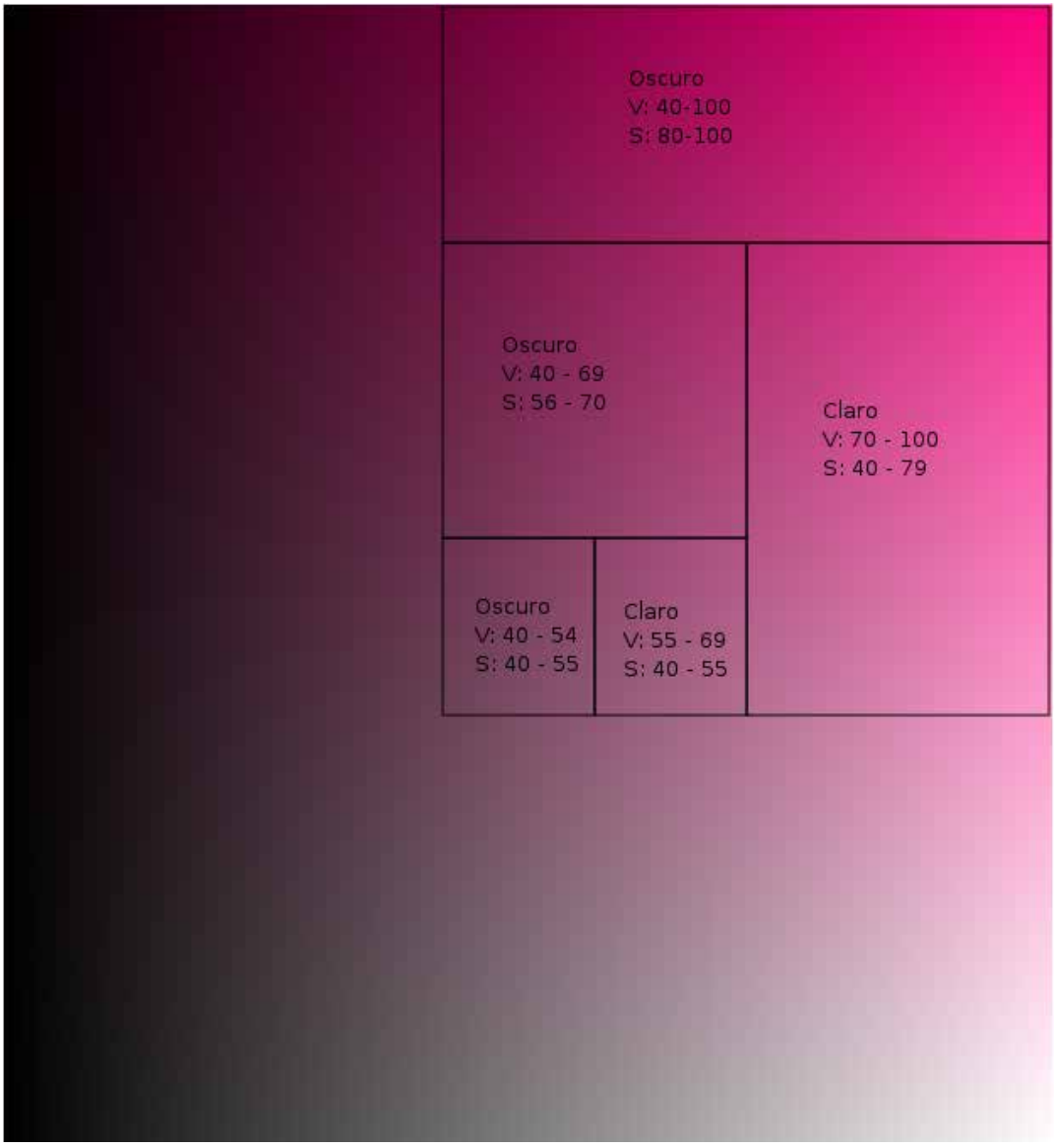
Gama del color Violeta



Gama del color Magenta



Gama del color Rosa



Documento de definición de requerimientos

Nombre del proyecto	Software de reconocimiento de colores de objetos en Imágenes Digitales orientado a personas con Daltonismo del tipo Deuteranomalía
Descripción	Se desarrollará un software que permita a las personas que padecen de daltonismo del tipo deuteranomalía reconocer aquellos colores a los que se le dificulta su percepción visual, esto mediante el procesamiento de imágenes digitales ya sean capturadas en el momento o seleccionadas por el usuario desde su directorio local.
Fecha de inicio	03/07/2017
Versión	1.0

	Construir una interfaz gráfica para el manejo del software
	Tomar una foto con la cámara del Raspberry Pi y guardarla en formato .jpg
	Subir imagenes en formato .jpg o .png ya guardadas en el directorio local
	Agregar filtros para los colores problemáticos a identificar en la imagen
	Devolver como respuesta la misma imagen procesada pero identificando la región en donde se encuentra el color problemático
	Guardar la imagen devuelta como respuesta
	Agregar opción de ayuda, para que el usuario conozca la información del software

Documento de especificación de requerimientos

Nombre del proyecto	Software de reconocimiento de colores de objetos en Imágenes Digitales orientado a personas con Daltonismo del tipo Deuteranomalía
Descripción	Se desarrollará un software que permita a las personas que padecen de daltonismo del tipo deuteranomalía reconocer aquellos colores a los que se le dificulta su percepción visual, esto mediante el procesamiento de imágenes digitales ya sean capturadas en el momento o seleccionadas por el usuario desde su directorio local.
Fecha de inicio	03/07/2017
Versión	1.0

---

1) <u>Construir una interfaz gráfica para el manejo del software</u>
   a) Desarrollar las ventanas de la interfaz
      i) Una ventana principal
      ii) Una ventana para tomar la foto
      iii) Una ventana para la navegación al sistema de archivos local
      iv) Una ventana para la imagen de respuesta
      v) Una ventana que contenga una breve documentación del software
   b) Agregar opciones de filtrado
      i) Lista de opciones para el color
      ii) Lista de opciones para las escalas del color
   c) Agregar los botones de la interfaz
      i) Un botón que active la cámara para tomar una foto
      ii) Un botón para seleccionar una imagen desde el directorio local
      iii) Un botón para procesar la imagen
      iv) Un botón para guardar la imagen de respuesta
      v) Un botón para mostrar la documentación

2) Tomar una foto con la cámara del Raspberry Pi y guardarla en formato .jpg
    a) Definir el formato .jpg para la imagen capturada
    b) Capturar una imagen
    c) Indicar la dirección del fichero destino
    d) Guardar la imagen en el fichero destino
3) Subir imágenes en formato .jpg o .png ya guardadas en el directorio local
    a) Indicar la dirección del fichero destino
    b) Comprobar que el formato de la imagen sea .jpg o .png
    c) Cargar la imagen a procesar
4) Agregar filtros para los colores problemáticos a identificar en la imagen
    a) Establecer los rangos de las escalas de los colores
    b) Corregir la gama de la imagen para adaptarla al ojo humano
    c) Crear una máscara con el rango de la escala seleccionada
    d) Aplicar la máscara a la imagen de entrada
5) Devolver como respuesta la misma imagen procesada pero identificando la región en donde se encuentra el color problemático
    a) Desplegar la ventana de respuesta
    b) Mostrar la imagen después de que se haya seleccionado un filtro determinado
6) Guardar la imagen devuelta como respuesta
    a) Indicar la dirección del fichero destino
    b) Guardar la imagen en el fichero destino
7) Agregar una opción de ayuda para que el usuario conozca la información del software
    a) Incorporar una función que llame a la ventana con la información referente a la descripción y uso del software

Estudio de Factibilidad

El estudio de factibilidad forma parte del proceso de evaluación al cual debe someterse todo nuevo proyecto, esto debido a que la aprobación de un proyecto no depende sólo de una buena idea, sino también de que se pueda demostrar su factibilidad y presentarla en forma vendedora. En ocasiones se afronta este tema desde un enfoque económico-financiero, olvidando así el resto de los análisis que deben ser tratados. De manera que, con la realización de este estudio se pretende mostrar un análisis de factibilidad basado en: un estudio técnico, un estudio económico, y un estudio operacional, donde cada uno de estos elementos poseen igual nivel de importancia para la toma de decisiones.

Factibilidad Técnica

Esta etapa del estudio se basa en tres aspectos fundamentales, la tecnología que requiere el sistema propuesto, la tecnología disponible actualmente por los investigadores, y la tecnología por adquirir para completar el desarrollo.

Tecnología Necesaria

Hardware

Entre los aspectos necesarios relacionados al hardware tenemos:

- 2 Laptops con requerimientos mínimos: procesador Intel Core i3 2.53 Ghz 4 GB de memoria RAM
- 1 Raspberry Pi versión 3 modelo B
- 1 Cámara de Raspberry Pi
- 1 Memoria flash 32 GB
- 1 Cable HDMI
- 1 Impresora EPSON Stylus TX110
- 1 Monitor LCD de 19”

Software

Entre los aspectos necesarios relacionados al software están:

- Raspbian
- Python 3.x
- PyGObject
- OpenCV 3.4.x
- NumPy
- picamera

Recursos Humanos

Para los recursos humanos se necesitan

- 2 programadores con conocimientos sobre lenguaje de programación Python

Tecnología Existente

Hardware

Entre los aspectos necesarios relacionados al hardware tenemos:

- 2 Laptop con procesador Intel Core i3 2.53 Ghz 4 GB de memoria RAM
- 1 Raspberry Pi versión 3 modelo B
- 1 Cámara de Raspberry Pi
- 1 Memoria flash 32 GB
- 1 Cable HDMI
- 1 Impresora EPSON Stylus TX110
- 1 Monitor LCD de 19”

Software

Entre los aspectos disponibles por los investigadores relacionados al software están:

- Raspbian
- Python 3.x
- PyGObject
- OpenCV 3.4.1
- NumPy
- picamera

Recursos Humanos

Entre los recursos humanos se tienen disponibles

- 2 programadores con conocimientos sobre lenguaje de programación Python

Tecnología por Adquirir

Gracias a que los investigadores poseen todas las herramientas a nivel tecnológico para la realización del proyecto, no fue necesaria la búsqueda de alternativas tecnológicas.

Factibilidad Económica

En esta etapa, se comprueba que el proyecto es sustentable económicamente, demostrando que si el sistema no cumple con su objetivo no habrá pérdidas económicas o serán las mínimas. Al iniciar el proyecto, se invirtieron Bs. 60.000 en la cámara del Raspberry Pi, lo cual representó la única inversión monetaria realizada en el proyecto.

En cuanto a costos operativos, el proyecto requiere de la disponibilidad de equipos tecnológicos como el Raspberry Pi y un monitor para mostrar lo que está procesando el computador de placa reducida, sin embargo, como estos equipos ya están disponibles en su totalidad por parte de los investigadores, no es necesario realizar alguna inversión adicional en alquiler de equipos o algo similar. A su vez, el software desarrollado es bastante sencillo e intuitivo de utilizar, por lo que no es necesario capacitar a los usuarios o contratar a alguien experto en el área para la operatividad del sistema.

En referencia a los beneficios, los usuarios que sufran de deuteranomalía que usen el sistema, se sentirán más cómodos sabiendo que podrán reconocer los colores en una imagen digital de manera correcta, evitando confusiones y posibles inconvenientes. Habiendo mencionado todo esto, se demuestra que el proyecto es económicamente factible gracias a que no se necesita de un gran músculo económico para la ejecución del mismo.

Factibilidad Operativa

Para el estudio de la factibilidad operativa, se tomaron en cuenta ciertos aspectos que suelen interferir en el uso correcto del software. Para empezar, se minimizó la complejidad del sistema, diseñando una interfaz de usuario fácil de usar e intuitiva, en donde las funcionalidades del sistema se ejecutan a través de botones que están todo el tiempo disponible en la ventana principal de la interfaz, además de las validaciones pertinentes para minimizar los errores en el sistema. Otro punto que se tomó en cuenta para la factibilidad fue el uso de tecnología disponible, considerando la posibilidad de la obsolescencia subsecuente, el sistema es desarrollado versiones actuales de las diferentes librerías y recursos tecnológicos utilizados para la fecha.

- Función de la transformación del color de un modelo a otro

```
Total time: 0.322471 s
File: pf3.py
Function: convertirModelo at line 5

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
     5                                           @profile
     6                                           def convertirModelo(img, color_m):
     7         1     322458.0 322458.0    100.0      img_hsv = cv.cvtColor(img, color_m)
     8         1         13.0     13.0      0.0      return img_hsv
```

- Función que crea un filtro a través del método del valor de umbral

```
Total time: 1.16805 s
File: pf3.py
Function: crearFiltro at line 9

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
     9                                           @profile
    10                                           def crearFiltro(img, color_r):
    11         1         49.0     49.0      0.0      if len(color_r) == 4:
    12         1         78.0     78.0      0.0          linf_1 = np.array(color_r[0])
    13         1         18.0     18.0      0.0          lsup_1 = np.array(color_r[1])
    14         1         16.0     16.0      0.0          linf_2 = np.array(color_r[2])
    15         1         15.0     15.0      0.0          lsup_2 = np.array(color_r[3])
    16         1     543799.0 543799.0     46.6          filtro_1 = cv.inRange(img, linf_1, lsu
p_1)
    17         1     507875.0 507875.0     43.5          filtro_2 = cv.inRange(img, linf_2, lsu
p_2)
    18         1     116190.0 116190.0      9.9          res_f = filtro_1 + filtro_2
    19                                               else:
    20                                                   linf_1 = np.array(color_r[0])
    21                                                   lsup_1 = np.array(color_r[1])
    22                                                   res_f = cv.inRange(img, linf_1, lsup_1
)
    23         1         13.0     13.0      0.0      return res_f
```

- Función que se encarga de corregir el valor gamma de una imagen

```
Total time: 0.119672 s
File: pf3.py
Function: corregirGamma at line 28

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
    28                                           @profile
    29                                           def corregirGamma(img, gamma=1.0):
    30         1         27.0     27.0      0.0      gamma_i = 1.0 / gamma
    31                                               tabla = np.array([((i / 255.0) ** gamma_i) * 255
    32         1       7350.0   7350.0      6.1                    for i in np.arange(0, 256)]) astype("uint8")
    33         1     112295.0 112295.0     93.8      return cv.LUT(img, tabla)
```

- Función que suaviza los bordes de una imagen

```
Total time: 8.30843 s
File: pf3.py
Function: suavizarBordes at line 48

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
    48                                           @profile
    49                                           def suavizarBordes(img):
    50         1    8308430.0 8308430.0    100.0      return cv.medianBlur(img,41)
```

- Función para detectar los contornos de áreas ya segmentadas

```
Total time: 0.306107 s
File: pf3.py
Function: detectarContornos at line 51

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
    51                                           @profile
    52                                           def detectarContornos(img_original, img_suavizada):
    53         1     187718.0 187718.0     61.3      __, contornos, jerarquia = cv.findContours(img_suavizada, cv.RETR_TRE
E, cv.CHAIN_APPROX_SIMPLE)
    54         1     118389.0 118389.0     38.7      cv.drawContours(img_original, contornos, -1, (255,0,0), 11)

root@cristhian:/home/pi/Documents/TDGP/pruebas_faseIII#
```

- Función que permite mostrar una imagen en una nueva ventana

```
Total time: 0.277001 s
File: pf3.py
Function: mostrarImg at line 55

Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
    55                                           @profile
    56                                           def mostrarImg(img, img_n):
    57         1      45324.0  45324.0     16.4      cv.namedWindow(img_n, cv.WINDOW_NORMAL)
    58         1     231677.0 231677.0     83.6      cv.imshow(img_n, img)
```

- Función de la transformación del color de un modelo a otro

```
Line #    Mem usage    Increment   Line Contents
================================================
     6    105.2 MiB    105.2 MiB   @profile
     7                             def convertirModelo(img, color_m):
     8    161.6 MiB     56.4 MiB       img_hsv = cv.cvtColor(img, color_m)
     9    161.6 MiB      0.0 MiB       return img_hsv
```

- Función que crea un filtro a través del método del valor de umbral

```
Line #    Mem usage    Increment   Line Contents
================================================
    10    162.2 MiB    162.2 MiB   @profile
    11                             def crearFiltro(img, color_r):
    12    162.2 MiB      0.0 MiB       if len(color_r) == 4:
    13    162.2 MiB      0.0 MiB           linf_1 = np.array(color_r[0])
    14    162.2 MiB      0.0 MiB           lsup_1 = np.array(color_r[1])
    15    162.2 MiB      0.0 MiB           linf_2 = np.array(color_r[2])
    16    162.2 MiB      0.0 MiB           lsup_2 = np.array(color_r[3])
    17    181.3 MiB     19.1 MiB           filtro_1 = cv.inRange(img, linf_1, lsup_1)
    18    200.4 MiB     19.1 MiB           filtro_2 = cv.inRange(img, linf_2, lsup_2)
    19    219.2 MiB     18.8 MiB           res_f = filtro_1 + filtro_2
    20                                   else:
    21                                       linf_1 = np.array(color_r[0])
    22                                       lsup_1 = np.array(color_r[1])
    23                                       res_f = cv.inRange(img, linf_1, lsup_1)
    24    219.2 MiB      0.0 MiB       return res_f
```

- Función que se encarga de corregir el valor gamma de una imagen

```
Line #    Mem usage    Increment   Line Contents
================================================
    25    180.7 MiB    180.7 MiB   @profile
    26                             def corregirGamma(img, gamma=1.0):
    27    180.8 MiB      0.1 MiB       gamma_i = 1.0 / gamma
    28    180.8 MiB      0.0 MiB       tabla = np.array([((i / 255.0) ** gamma_i) * 255
    29    180.8 MiB      0.0 MiB                         for i in np.arange(0, 256)]).astype("uint8")
    30    237.5 MiB     56.7 MiB       return cv.LUT(img, tabla)
```

- Función que suaviza los bordes de una imagen

```
Line #    Mem usage    Increment   Line Contents
================================================
    31    237.9 MiB    237.9 MiB   @profile
    32                             def suavizarBordes(img):
    33    257.2 MiB     19.3 MiB       return cv.medianBlur(img,41)
```

- Función para detectar los contornos de áreas ya segmentadas

```
Line #    Mem usage    Increment  Line Contents
================================================
    34    257.4 MiB    257.4 MiB  @profile
    35                            def detectarContornos(img_original, img_suavizada):
    36    257.8 MiB      0.4 MiB      __, contornos, jerarquia = cv.findContours(img_suavizada, cv.RETR_TREE, cv.CHAIN_AP
PROX_SIMPLE)
    37    257.8 MiB      0.0 MiB      cv.drawContours(img_original, contornos, -1, (255,0,0), 11)
```

ANEXO G-1

Función que corrige el valor gamma de una imagen:

```
def corrGamma(img, gamma=1):
    """
    Descripción:
        Corrige el brillo de una imagen mediante la formula
        de la transformación de la ley de potencia. Por de-
        fecto, el valor de gamma es igual a 1.

        Parametros:

            img: Imagen a la cual se le corregirá su brillo

            gamma: Valor de correción que se aplicará

        Retorna:
            La misma imagen pero con el brillo aumentado o
            disminuido.
    """
    gamma_i = 1.0 / gamma
    tabla = array([((i / 255.0) ** gamma_i) * 255
        for i in arange(0, 256)]).astype("uint8")
    return LUT(img, tabla)
```

ANEXO G-2

Función que procesa una imagen y crea un filtro a partir de ella:

```
def filtImagen(img, color_r):
    """
    Descripción:
        Filtra una imagen para detectar los segmentos de un
        color problematico especifico, de acuerdo a su res-
        pectiva escala.

        Parametros:

            img: Imagen que se desea filtrar

            color_r: Arreglo de rangos que delimitan las di-
                     ferentes regiones de la escala

        Retorna:
            El filtro generado en forma de imagen binaria, el
            cual contiene los segmentos compuestos de pixeles
            con los valores comprendidos dentro de la escala
            del color proporcionado.
    """
    filtro = array([]).astype('uint8')
    rango_cr = len(color_r)
    if rango_cr > 0 and rango_cr % 2 == 0:
        filtro = 0
        for region in range(0, rango_cr, 2):
            lim_inf = array(color_r[region])
            lim_sup = array(color_r[region+1])
            filtro += inRange(img, lim_inf, lim_sup)
    return filtro
```

Función que obtiene los contornos encontrados en un filtro y los dibuja sobre la imagen originalmente procesada:

```
def obtContornos(img, filtro, borde_c):
    """
    Descripción:
        Localiza los contornos de los segmentos ubicados en un
        filtro que ya ha sido suavizado. Luego procede a dibu-
        jarlos sobre la imagen original.

        Parametro:

            img: Imagen sobre la que se dibujaran los con-
                 tornos

            filtro: filtro previamente suavizado

            borde_c: Color de los bordes de los contornos
    """
    borde_c = {
        'Rojo': (0,0,255),
        'Verde': (0,255,0),
        'Azul': (255,0,0),
        'Blanco': (255,255,255),
        'Negro': (0,0,0)
    }.get(borde_c)
    _, conts, jer = findContours(filtro, RETR_EXTERNAL,
                                 CHAIN_APPROX_NONE)
    drawContours(img, conts,  -1, borde_c, 8)
```

Función que procesa una imagen originalmente referenciada desde la interfaz gráfica de usuario:

```
def procImagen(ruta, color, escala, borde_c, gamma):
    """
    Descripción:
    """
    img = imread(ruta)

    if gamma != 1:
        img = prep.corrGamma(img, gamma)

    img = cvtColor(img, COLOR_BGR2HSV)

    if escala == 'Claro':
        filtro = filtImagen(img, rc.RANGOS[color]['CL'])
    elif escala == 'Oscuro':
        filtro = filtImagen(img, rc.RANGOS[color]['OS'])
    else:
        filtro = filtImagen(img, rc.RANGOS[color]['CL'])
        filtro += filtImagen(img, rc.RANGOS[color]['OS'])

    img = cvtColor(img, COLOR_HSV2BGR)
    filtro = medianBlur(filtro, 31)
    obtContornos(img, filtro, borde_c)
    img = postp.convPixbuf(img)
    return img
```

ANEXO G-3

Función que lleva el formato de imagen en OpenCV al de GdkPixbuf:

```
def convPixbuf(img):
    """
    Descripción:
        Convierte un arreglo de numpy de tipo unsigned int
        de 8 bits (formato con el cual trabaja las imagenes
        OpenCV) en un objeto pixbuf.

        Parametros:

            img: Imagen que ha finalizado su etapa de proce-
                 samiento por completo

        Retorna:
            Un objeto de Gdk.pixbuf que sirve para la manipu-
            lacion de imagenes en la interfaz grafica.
    """
    _, jpg = imencode(".jpg", img)
    loader = PixbufLoader.new_with_type("jpeg")
    loader.write(jpg)
    loader.close()
    pixbuf = loader.get_pixbuf()
    return pixbuf
```

ANEXO H

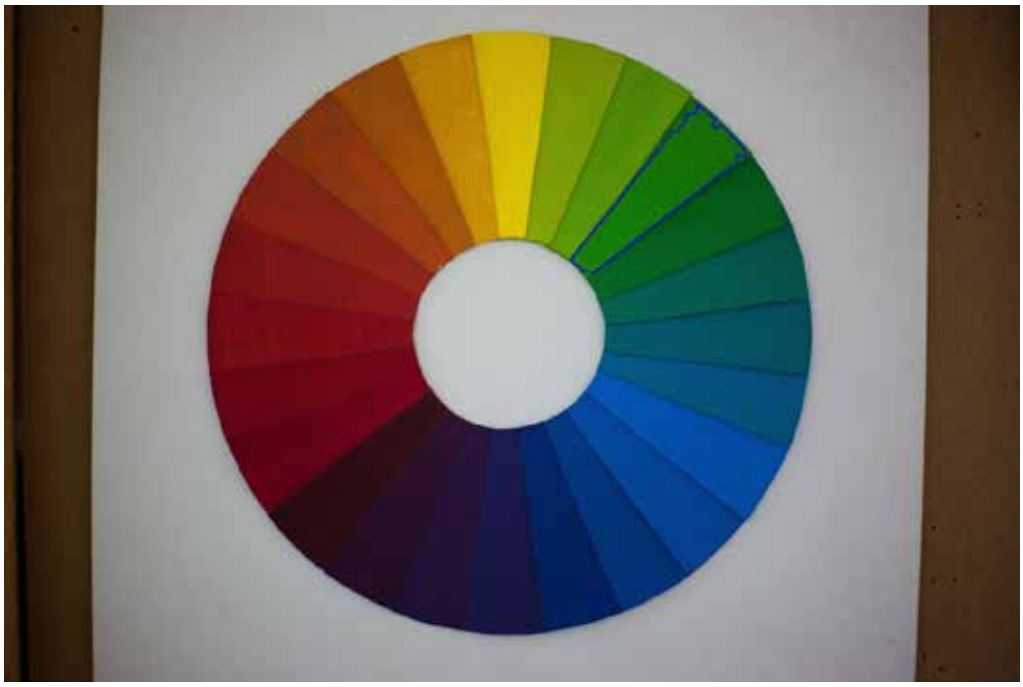
Las imágenes aquí descritas representan los resultados más significativos de las muestras. No obstante, se realizó una verificación a todas las imágenes de la muestra y se obtuvieron los siguientes resultados:

Grupo	1	2	3	4	5	6	7	8	9	10	11	12
Geométrica	Si	Si	No	Si	Si	No	Si	Si	No	No	No	Si
No geométrica	Si	No	No	Si	No	No	No	Si	Si	Si	Si	No

Resultados de las pruebas realizadas para el conjunto de muestras con elementos geométricos:

- Color a detectar: Rojo con ambas escalas

Color a detectar: Verde con ambas escalas



- Color a detectar: Cian con ambas escalas

Resultados de las pruebas realizadas para el conjunto de muestras con elementos no necesariamente geométricos:

- Color a detectar: Verde-Amarillo para ambas escalas

- Color a detectar: Rojo para ambas escalas

- Color a detectar: Rojo para ambas escalas

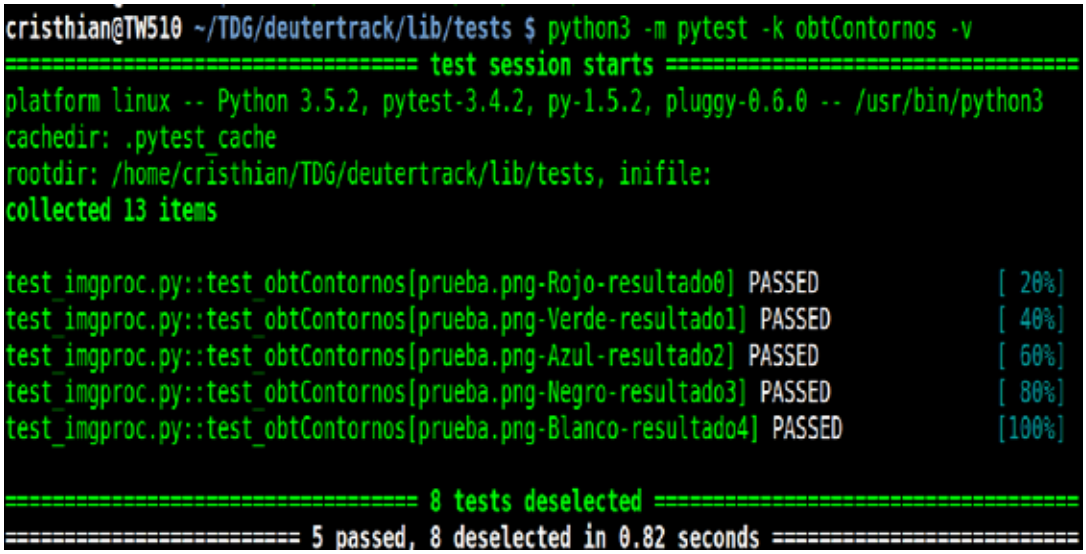
ANEXO I-1

Ejecución de pruebas para la función corregir gamma:

```
cristhian@TW510 ~/TDG/deutertrack/lib/tests $ python3 -m pytest -k corrGamma -v
============================================ test session starts ============================
=======
platform linux -- Python 3.5.2, pytest-3.4.2, py-1.5.2, pluggy-0.6.0 -- /usr/bin/python3
cachedir: .pytest_cache
rootdir: /home/cristhian/TDG/deutertrack/lib/tests, inifile:
collected 7 items


test_preproc.py::test_corrGamma[prueba.png-0-uint8] FAILED
[ 50%]
test_preproc.py::test_corrGamma[prueba.png-0.4-uint8] PASSED
[100%]
================================================== FAILURES ==============
_____________________________________ test_corrGamma[prueba.png-0-uint8] ___

ruta = 'prueba.png', gamma = 0, sal = 'uint8'

    @pytest.mark.parametrize("ruta, gamma, sal",
                             [
                                ('prueba.png', 0, 'uint8'),
                                ('prueba.png', 0.4, 'uint8')
                             ]
                            )
    def test_corrGamma(ruta, gamma, sal):
        img = imread(ruta)
>       img_c = pp.corrGamma(img, gamma)

test_preproc.py:16:
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

img = array([[[ 14,  14,  14],
        [ 14,  14,  14],
        [ 43,  12,   1],
        ...,
        [223, 223, 223],
    ... 72,  16,   0],
        ...,
        [255, 255, 255],
        [255, 255, 255],
        [255, 255, 255]]], dtype=uint8)
gamma = 0

    def corrGamma(img, gamma=1):
>       gamma_i = 1.0 / gamma
E       ZeroDivisionError: float division by zero

../preproc.py:4: ZeroDivisionError
```

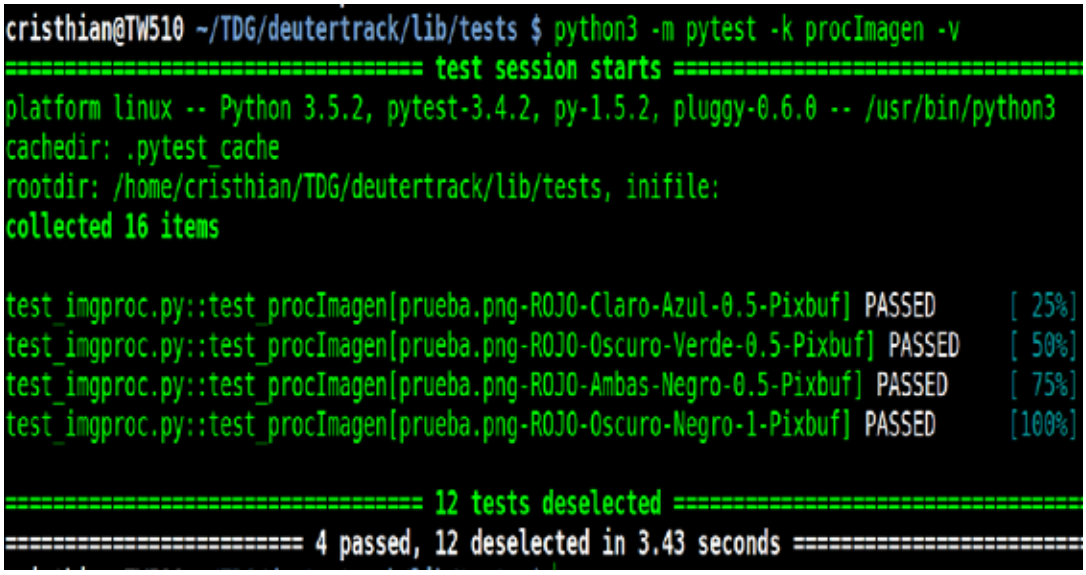
ANEXO I-2

Ejecución de pruebas para la función de obtención de contornos:



ANEXO I-3

Ejecución de pruebas para la función de procesamiento de una imagen:



ANEXO I-4

Ejecución de pruebas para la función de creación de un filtro a partir de una imagen

- **Previo a las correcciones del código:**

```
cristhian@TW510 ~/TDG/deutertrack/lib/tests $ python3 -m pytest -k filtImagen -v
============================= test session starts ==============================
platform linux -- Python 3.5.2, pytest-3.4.2, py-1.5.2, pluggy-0.6.0 -- /usr/bin/python3
cachedir: .pytest_cache
rootdir: /home/cristhian/TDG/deutertrack/lib/tests, inifile:
collected 9 items

test_imgproc.py::test_filtImagen[prueba.png-rango0-uint8] FAILED          [ 25%]
test_imgproc.py::test_filtImagen[prueba.png-rango1-uint8] PASSED          [ 50%]
test_imgproc.py::test_filtImagen[prueba.png-rango2-uint8] FAILED          [ 75%]
test_imgproc.py::test_filtImagen[prueba.png-rango3-uint8] PASSED          [100%]

=================================== FAILURES ===================================
_____________________ test_filtImagen[prueba.png-rango0-uint8] _____

ruta = 'prueba.png', rango = [], resultado = 'uint8'

    @pytest.mark.parametrize("ruta, rango, resultado",[
        ('prueba.png',[],'uint8'),
        ('prueba.png',[[0,0,0],[0,0,255]],'uint8'),
        ('prueba.png',[[0,0,0],[0,127,0],[0,255,0]],'uint8'),
        ('prueba.png',[[0,0,0],[0,127,0],[0,200,0],[0,255,0]],'uint8')
    ])
    def test_filtImagen(ruta, rango, resultado):
        img = imread(ruta)
        filtro = ip.filtImagen(img, rango)
>       assert filtro.dtype == resultado
E       AttributeError: 'int' object has no attribute 'dtype'

test_imgproc.py:17: AttributeError
```

```
_____ test_filtImagen[prueba.png-rango2-uint8] _____

ruta = 'prueba.png', rango = [[0, 0, 0], [0, 127, 0], [0, 255, 0]], resultado = 'uint8'

    @pytest.mark.parametrize("ruta, rango, resultado",[
        ('prueba.png',[],'uint8'),
        ('prueba.png',[[0,0,0],[0,0,255]],'uint8'),
        ('prueba.png',[[0,0,0],[0,127,0],[0,255,0]],'uint8'),
        ('prueba.png',[[0,0,0],[0,127,0],[0,200,0],[0,255,0]],'uint8')
    ])
    def test_filtImagen(ruta, rango, resultado):
        img = imread(ruta)
>       filtro = ip.filtImagen(img, rango)

test_imgproc.py:16:
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

img = array([[[ 14,  14,  14],
        [ 14,  14,  14],
        [ 43,  12,   1],
        ...,
        [223, 223, 223],
    ... 72,  16,   0],
        ...,
        [255, 255, 255],
        [255, 255, 255],
        [255, 255, 255]]], dtype=uint8)
color_r = [[0, 0, 0], [0, 127, 0], [0, 255, 0]]

    def filtImagen(img, color_r):
        filtro = 0
        for region in range(0, len(color_r), 2):
            lim_inf = array(color_r[region])
>           lim_sup = array(color_r[region+1])
E           IndexError: list index out of range

../imgproc.py:11: IndexError
```

- **Posterior a la corrección del código:**

```
cristhian@TW510 ~/TDG/deutertrack/lib/tests $ python3 -m pytest -k filtImagen -v
============================= test session starts ==============================
platform linux -- Python 3.5.2, pytest-3.4.2, py-1.5.2, pluggy-0.6.0 -- /usr/bin/python3
cachedir: .pytest_cache
rootdir: /home/cristhian/TDG/deutertrack/lib/tests, inifile:
collected 16 items

test_imgproc.py::test_filtImagen[prueba.png-rango0-uint8] PASSED             [ 25%]
test_imgproc.py::test_filtImagen[prueba.png-rango1-uint8] PASSED             [ 50%]
test_imgproc.py::test_filtImagen[prueba.png-rango2-uint8] PASSED             [ 75%]
test_imgproc.py::test_filtImagen[prueba.png-rango3-uint8] PASSED             [100%]

================================ 12 tests deselected ================================
======================== 4 passed, 12 deselected in 0.62 seconds ========================
```

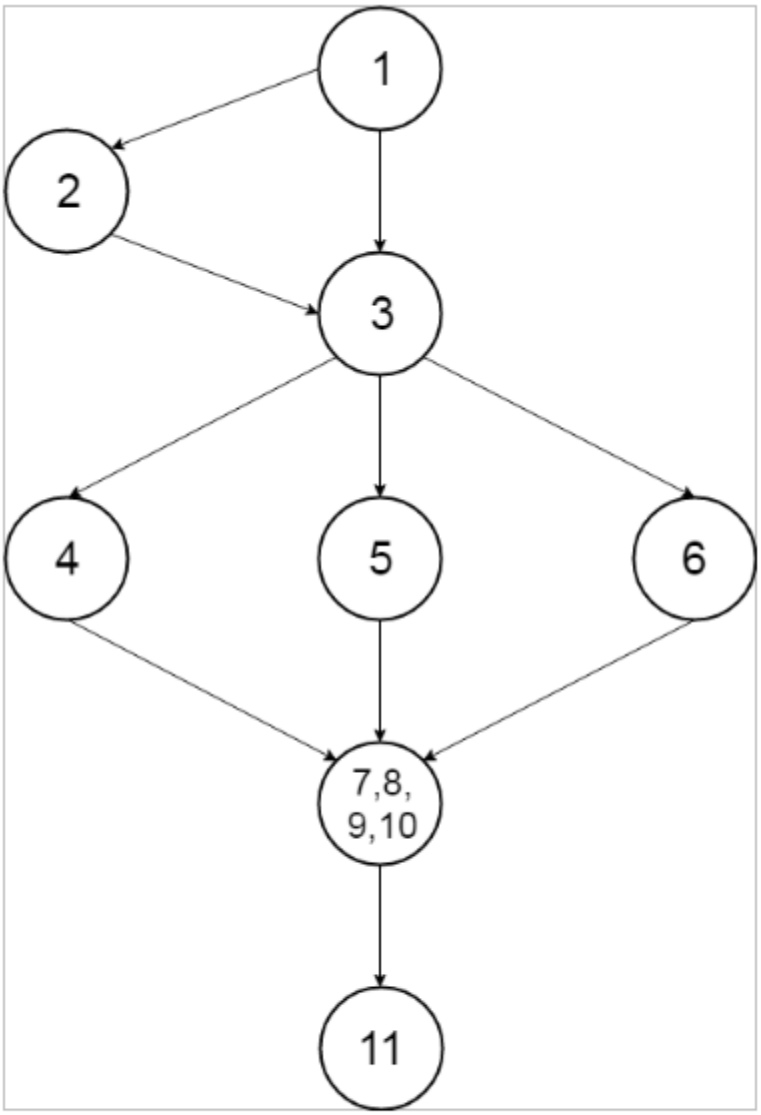
ANEXO I-5

Ejecución de pruebas para la función de conversión del formato de imagen usado en OpenCV al objeto Pixbuf:

```
cristhian@TW510 ~/TDG/deutertrack/lib/tests $ python3 -m pytest -k convPixbuf -v
============================= test session starts ==============================
platform linux -- Python 3.5.2, pytest-3.4.2, py-1.5.2, pluggy-0.6.0 -- /usr/bin/python3
cachedir: .pytest_cache
rootdir: /home/cristhian/TDG/deutertrack/lib/tests, inifile:
collected 7 items

test_postproc.py::test_convPixbuf[prueba.png-Pixbuf] PASSED                  [100%]

================================ 6 tests deselected =================================
======================== 1 passed, 6 deselected in 0.50 seconds ========================
```

ANEXO J

Grafo de flujo de control de la función principal de procesamiento de una imagen:



COMPLEJIDAD CICLOMATICA

M = E - N + 2 * P = 10 - 8 + 2 (1) = 4

Leyenda de Nodos		
Nodo	Línea	Descripción
1	img = imread(ruta)	Leer una imagen
2	if gamma != 1: img = prep.corrGamma(img, gamma)	Corregir el brillo de la imagen
3	img = cvtColor(img, COLOR_BGR2HSV)	Cambiar modelo del color
4	if escala == 'Claro': filtro = filtImagen(img, rc.RANGOS[color]['CL'])	Crear filtro para colores claros
5	elif escala == 'Oscuro': filtro = filtImagen(img, rc.RANGOS[color]['OS'])	Crear filtro para colores oscuros
6	else: filtro = filtImagen(img, rc.RANGOS[color]['CL']) filtro += filtImagen(img, rc.RANGOS[color]['OS'])	Crear filtro para ambas escalas de colores
7	img = cvtColor(img, COLOR_HSV2BGR)	Cambiar el modelo del color nuevamente
8	filtro = medianBlur(filtro, 41)	Suavizar el fitro
9	obtContornos(img, filtro, borde_c)	Dibujar el contorno de los segmentos
10	img = postp.convGdkPixbuf(img)	Cambiar el format de la imagen
11	return img	Devolver la imagen como respuesta

ANEXO K-1

Descarga e instalación de dependencias

- **Actualizar y renovar paquetes existentes:**

  $ sudo apt-get update

  $ sudo apt-get upgrade

- **Instalación de CMake para el proceso de construcción de OpenCV**

  $ sudo apt-get install build-essential cmake pkg-config

- **Instalación de algunos paquetes de entrada y salida de imágenes que permiten cargar varios formatos de archivos de imágenes del disco. Ejemplos: JPEG, PNG, entre otros:**

  $ sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev

- **Instalación de la librería de desarrollo de GTK, necesaria para compilar el módulo highgui de OpenCV:**

  $ sudo apt-get install libgtk2.0-dev

- **Instalación de dependencias extras para la optimización de operaciones internas en OpenCV:**

  $ sudo apt-get install libatlas-base-dev gfortran

- **Instalación de los archivos de cabecera de Python 3 para poder compilar OpenCV con vínculos de Python:**

  $ sudo apt-get install python3-dev

- **Instalación del instalador de paquetes de Python 3**

```
$ sudo python3 get-pip.py
```

- **Instalación de la librería numpy para el manejo de arreglos numéricos grandes:**

```
$ pip3 install numpy
```

ANEXO K-2

Descarga e instalación de OpenCV desde el código fuente

- **Descarga del repositorio que contiene la librería:**

  $ git clone https://github.com/opencv/opencv.git

- **Creación de la carpeta build dentro del repositorio recién descargado:**

  $ mkdir build

  $ cd build

- **Establece algunos parámetros de la construcción de OpenCV usando CMake:**

  $ cmake -D CMAKE_BUILD_TYPE=RELEASE \

  -D CMAKE_INSTALL_PREFIX=/usr/local \

  -D INSTALL_PYTHON_EXAMPLES=ON \

  -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-3.x.x/modules \

  -D BUILD_EXAMPLES=ON ..

- **Compila OpenCV para luego instalarlo dentro del Raspberry pi:**

  $ make

  $ sudo make install

- **Renombra los vinculos de Python 3.x y OpenCV a cv2.so:**

  $ cd /usr/local/lib/python3.x/site-packages/

  $ sudo mv cv2.cpython-3xm.so cv2.so

- **Conecta de manera simbólica los vínculos de OpenCV para Python 3.x:**

```
$ cd ~/.virtualenvs/cv/lib/python3.x/site-packages/
$ ln -s /usr/local/lib/python3.x/site-packages/cv2.so cv2.so
```