UNIVERSIDAD NACIONAL DE INGENIERÍA

Recinto Universitario "Simón Bolívar"

FACULTAD DE ELECTROTECNIA Y COMPUTACIÓN

TRABAJO MONOGRAFICO PARA OPTAR AL TITULO DE INGENIERO EN ELECTRONICA

"APLICACIONES DE PROCESAMIENTO DIGITAL DE IMÁGENES Y VIDEOS CON PROPÓSITOS ACADÉMICOS"

Autores:

Br. Oliver José Duarte Aguirre

Br. José Carlos González Roque

Tutor:

TkneL. Marco Munguía Mena

Managua, Nicaragua, Enero 2017.

TITULO

Aplicaciones de Procesamiento Digital de Imágenes y Videos con Propósitos Académicos.

Agradecimiento

Agradezco primeramente a Dios por haberme acompañado y guiado a lo largo de mi carrera, por darme siempre fortaleza en los momentos que más lo necesitaba y brindarme la capacidad de adquirir los conocimientos necesarios que me trajeron hasta aquí, a mis padres Leonidas y Josefa por siempre apoyarme en todos los aspectos y enseñarme a nunca rendirme a pesar de todas las adversidades por las que pase a lo largo de mi carrera, a mi novia Vicky que siempre estuvo allí para apoyarme cuando hizo falta y ayudarme a no perder las esperanzas cada que lo necesite y a los amigos y familiares que me ofrecieron su apoyo incondicional siempre que lo necesite. También a mi tutor Marco Munguía que siempre nos apoyó y nos dio la información y el soporte que necesitamos y a José porque a pesar de todo ha sido un buen compañero de Tesis.

-*Oliver José Duarte Aguirre.*

Quiero agradecer a nuestro tutor Marco Munguía, que sin su ayuda y conocimientos no hubiese sido posible realizar este proyecto. A mi madre Leyla, por haberme proporcionado la mejor educación y lecciones de vida que me han permitido ser una mejor persona, por hacerme ver las cosas siempre desde una perspectiva distinta y especialmente por confiar en mis decisiones. A mis hermanos Luis, Andrés y demás familiares, por su apoyo incondicional. A mis amigos, por estar siempre a mi lado. A mi compañero de tesis Oliver por su paciencia y buena disposición.

-*José Carlos González Roque.*

RESUMEN

El presente trabajo monográfico fue realizado en el área de procesamiento digital de señales, específicamente en el área de imágenes y videos. A lo largo del documento se encontrará información acerca del procesamiento digital de imágenes y videos, además de información básica para comprender mejor, realizamos también la formación de una imagen que se realiza mediante la unión de pixeles que son representados como una matriz y pueden tomar valores desde 0 a 255 a lo largo de toda la imagen, así mismo el concepto de video el cual consiste en una serie de imágenes en una secuencia que van una tras otra a una velocidad determinada como fotogramas por segundo o fpm (frames per second), entre otras herramientas que facilitaran la comprensión en el área de procesamiento digital de imágenes y videos.

La herramienta computacional Matlab® nos permite realizar el desarrollo de software en lenguaje de alto nivel mediante el desarrollo de scripts. Matlab® posee un toolbox o librería (conocido el término más popularmente) para el desarrollo de procesamiento digital de imágenes y videos el cual nos ayudara a la utilización de comandos para realizar análisis de imágenes, también a realizar el histograma de una imagen así como modificar su escala de grises, entre otras cosas que son de utilidad para el desarrollo de software que realizaremos.

En este documento además de profundizar en temas relacionados al procesamiento digital de imágenes y videos. Desarrollamos aplicaciones en el área de imágenes así como de video, también realizamos un análisis de escala de grises de una imagen y la implementación de filtros al estilo de la popular aplicación llamada Instagram en lo que respecta a imágenes. En la parte de video se realiza una aplicación para la detección de objetos mediante la identificación de colores y se introduce un poco el tema de la detección de rostros. Todo esto para que posteriormente con el desarrollo de estas aplicaciones se realicen guías de laboratorio que tendrán como objetivo contribuir al desarrollo de los estudiantes en el área de procesamiento digital de señales.

Contenido

1. Introducción

La electrónica avanza día a día de forma acelerada, por lo que todos los aparatos electrónicos (e.g. celulares, tabletas, computadoras) han pasado a formar parte inherente en nuestro diario vivir ya que permiten un sin número de facilidades tanto en la realización de nuestras tareas cotidianas así como en la comunicación.

A partir del siglo XXI estos pasos han sido cada vez más agigantados y como predice la ley de Moore [1] cada dos años se duplica la potencia de cálculo de nuestros ordenadores. En la carrera de ingeniería electrónica que ofrece la Universidad Nacional de Ingeniería, se instruye a los estudiantes sobre las diferentes ramas que esta posee, entre las cuales destaca el procesamiento digital de señales [2] la cual se imparte parcialmente en la clase de señales y sistemas. Cuando decimos parcialmente nos referimos a que solo se da la componente teórica.

Como estudiantes somos conscientes de las limitaciones que conlleva esto en el plano profesional donde los conocimientos tanto teóricos como prácticos nos darían una ventaja competitiva sobre el resto de profesionales de otras universidades. En un intento de mejorar este aspecto, planteamos la realización de guías de laboratorio que sirvan como herramienta para la efectividad de los futuros ingenieros en situaciones que conlleven la utilización de procesamiento digital de imágenes o videos [3]. Algunos ejemplos pueden ser la detección de tumores en tomografías, utilización de efectos especiales en el mundo del cine o en series de televisión, revisión de errores en tarjetas electrónicas, hasta lo que vivimos a diario con la utilización de teléfonos o tablets para revisar Facebook, Instagram, Snapchat y otras redes sociales que permiten compartir imágenes y videos.

Estas guías servirán como introducción a los estudiantes en la materia de procesamiento digital de señales con la ayuda del componente de imágenes y videos. Se hará uso de la herramienta computacional Matlab®, la cual nos provee de los recursos que necesitemos para realizar un código fuente o programa el cual mediante una interfaz gráfica ayude al estudiante a tener una experiencia más intuitiva y amena, además de comprender los conocimientos básicos sobre procesamiento de imágenes y videos, tomando en cuenta lo necesario que esto es hoy en día y hacemos énfasis en que esta materia debe ser estudiada ya que cada vez tiene mayor número de aplicaciones en situaciones cotidianas.

En las siguientes páginas introduciremos y explicaremos un poco más sobre este tema, tanto en términos utilizados en este campo de estudio, así como la estrategia planteada para aplicar de forma académica estas herramientas.

2. Objetivo General

- Contribuir a la mejora del proceso de enseñanza-aprendizaje mediante la implementación de prácticas de laboratorio que faciliten el adiestramiento de los estudiantes de Ingeniería Electrónica en el área de procesamiento digital de imágenes y videos.

3. Objetivos Específicos

- Comprender los fundamentos del procesamiento digital de imágenes y videos.
- Desarrollar el software de procesamiento digital de imágenes utilizando la herramienta computacional Matlab®.
- Desarrollar el software de procesamiento digital de videos utilizando la herramienta computacional Matlab®.
- Elaborar guías de laboratorio para que los estudiantes reafirmen sus conocimientos de procesamiento digital de imágenes y videos.

4. Justificación

Las imágenes y los vídeos forman parte inherente de nuestras vidas. Los tenemos en exámenes médicos e.g. tomografía, en nuestros celulares y tabletas, en los sistemas de vigilancia, en el cine, en las redes sociales, etc. Una de las razones que han llevado a las imágenes y videos a formar parte de nuestro diario vivir es el procesamiento digital de señales. La disciplina de procesamiento digital de señales forma parte de la Ingeniería Electrónica. Ésta podríamos definirla como el almacenamiento, compresión, transmisión, transformación e interpretación de las señales con el propósito que queramos [2].

Actualmente, en la carrera de Ingeniería Electrónica se aborda el tópico de procesamiento digital de señales en la asignatura de señales y sistemas. No obstante, se adolece de la componente experimental o de laboratorio. Por tal motivo, proponemos el desarrollo de prácticas de laboratorio para contribuir a la mejora del proceso de enseñanza-aprendizaje en los estudiantes de Ingeniería Electrónica en esta disciplina. Así mismo, esperamos que este trabajo sea el punto de partida para cimentar el desarrollo de esta disciplina entre los docentes y estudiantes de la carrera.

Dichas guías se plantean desarrollar con ayuda de la herramienta computacional Matlab® ya que es la más utilizada en nuestra carrera, por lo que les permitirá a los estudiantes sentirse familiarizados con el ambiente de desarrollo. No obstante, al no tener antecedentes de guías de laboratorio sobre este tema, las mismas serán hechas de forma fácil de entender y accesible para los estudiantes.

Debido a que este es un campo que en la actualidad posee un sin número de aplicaciones en diferentes áreas de trabajo, tener conocimientos sobre esta temática le permitirá al estudiante destacar sobre otros profesionales, tomando en cuenta que las guías sirvan como punto de partida y motivación al estudiante para que indague y estudie más sobre el tema.

5. Procesamiento digital de señales

Las señales eléctricas son tensiones o corrientes que contienen información. Además de las señales eléctricas existen otras, de naturaleza magnética, hidráulica, neumática, luminosa, etc. Las señales pueden ser generadas en forma natural o artificial [2]. Algunos ejemplos de señales naturales son la radiación electromagnética de una estrella, la presión atmosférica y la velocidad del viento. Algunos ejemplos de señales artificiales son la emisión de un canal de TV, las ondas emitidas y recibidas por radares, teléfonos celulares, sonares, etc. Las señales se representan matemáticamente como funciones de una o más variables independientes. La variable independiente más común es el tiempo, y algunas señales que dependen de él son, por ejemplo, la voz, una onda de radio, un electrocardiograma, etc.

Otras señales, tales como las imágenes, son funciones de 2 variables independientes, ya que contienen información de brillo o de colorido en función de las coordenadas X e Y de un plano.

Procesamiento de Señales es un área de la Ingeniería Electrónica que se concentra en la representación, transformación y manipulación de señales, y de la información que ellas contienen. El primer tipo de procesamiento electrónico que se desarrolló y se aplicó extensivamente fue el procesamiento análogo, el cual se lleva a cabo mediante circuitos compuestos por resistores, capacitores, inductores, amplificadores operacionales, etc.

Procesamiento de Señales en Tiempo Discreto (Discrete-Time Signal Processing) se refiere al procesamiento de señales discretas en el tiempo o en el espacio. Esto implica que sólo se conoce el valor de la señal en instantes o en puntos específicos. Sin embargo, la amplitud de la señal es continua, es decir, puede tomar infinitos valores diferentes [2].

Procesamiento Digital de Señales (Digital Signal Processing o DSP) añade a la característica anterior la de manejar la amplitud en forma discreta, la cual es una condición necesaria para que la señal pueda ser procesada en un computador

digital [2]. La amplitud de la señal sólo puede tener un número finito de valores diferentes.

En el ejemplo 5.1 se ilustra la diferencia entre los distintos tipos de procesamiento.

Ejemplo 5.1: en la Figura 5.1 se muestra un filtro paso bajo implementado con 3 tecnologías diferentes, que procesan la señal en las 3 formas descritas anteriormente.

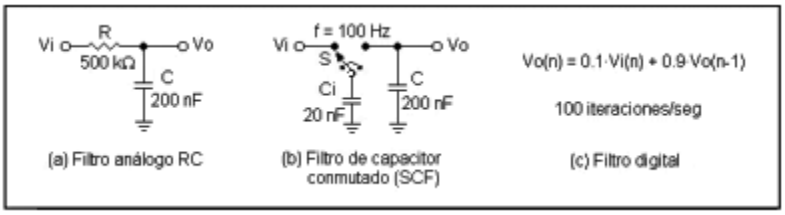


Figura 5.1. Diferentes filtros pasa bajo

Para describir el funcionamiento de los 3 filtros se supondrá que todos los voltajes son cero hasta el instante inicial, momento en el cual se aplica una tensión de 1 V en la entrada (Vi). Esto se conoce como la “respuesta escalón” del filtro.

(a) Filtro análogo RC: la tensión de entrada hace fluir una corriente a través del resistor R, cargando al capacitor C. A medida que Vi aumenta, disminuye la diferencia de potencial en R, disminuyendo la corriente y la velocidad de crecimiento de Vo, el cual se aproxima asintóticamente a 1 V, siguiendo una curva exponencial creciente.

(b) Filtro de capacitor conmutado (SCF): cuando el conmutador S se encuentra en la posición izquierda, el capacitor Ci se carga con Vi; cuando S conmuta a la posición derecha, Ci transfiere parte de su carga a C, elevando el voltaje de este último. Como Ci es bastante menor que C, cada conmutación de S eleva Vo en un peldaño de pequeña magnitud. Además, a medida que Vo aumenta, la transferencia de carga desde Ci a C es cada vez menor, haciendo que Vo se asemeje a una escalera, con una velocidad de elevación decreciente.

(c) Filtro digital: está constituido por una fórmula y una máquina calculadora. La fórmula dice: la salida actual se obtiene sumando un 10% de la entrada actual con

un 90% de la salida anterior. Por lo tanto, la primera salida será 0.1 V, la segunda será 0.1·1 + 0.9·0.1 = 0.19 V, etc. En este ejemplo la máquina recalcula la fórmula 100 veces por segundo.

Las salidas de los 3 filtros están graficadas en la Figura 5.2. Se aprecia que las respuestas son virtualmente idénticas.

El circuito (a) es un filtro análogo. Las señales están definidas para todo instante de tiempo, y pueden tomar infinitos valores diferentes.

El circuito (b) discretiza la señal en el tiempo, pero no en la amplitud, ya que el voltaje en el capacitor C puede tomar infinitos valores diferentes, dependiendo de la entrada aplicada. Figura 5.1. Filtro pasabajos análogo (a), en tiempo discreto (b) y digital (c).

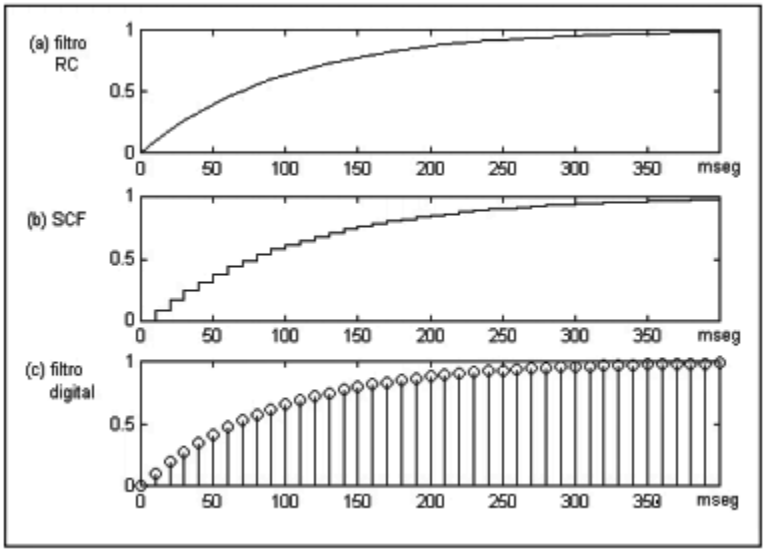


Figura 5.2. Salida de filtros pasa bajo de la figura anterior

La ecuación del filtro (c) se ejecuta en un computador digital, por lo que existe discretización en el tiempo y en la amplitud.

La discretización en el tiempo es la diferencia más importante entre el procesamiento digital y el procesamiento análogo. La discretización en el tiempo modifica las fórmulas de las transformadas, convolución, correlación, etc., e introduce un posible problema que no existe en el mundo análogo, denominado

aliasing, el cual se origina cuando la tasa de muestreo es insuficiente, generando una pérdida irrecuperable de la información contenida en la señal. La discretización en la amplitud puede ser casi imperceptible, como cuando se efectúan los cálculos en punto flotante con doble precisión (alrededor de 15 decimales) en un lenguaje de programación de alto nivel, o notoria, si se cuantiza la señal con pocos bit.

La discretización en la amplitud puede provocar algunos efectos indeseados, tales como:

• Si proviene de la conversión A/D de la señal, es equivalente a sumarle un cierto tipo de ruido, el cual se denomina “ruido de cuantización”.

• Si afecta a los cálculos, y es significativa, puede producir errores importantes, e incluso inestabilidad en algunos sistemas.

Como se había mencionado anteriormente, el filtro digital del ejemplo 5.1 está constituido por una fórmula y por una máquina calculadora. Si se modifica la fórmula, cambia la respuesta del filtro, pero si se reemplaza la máquina calculadora, la respuesta se mantiene (siempre que la máquina no introduzca errores significativos, y que sea capaz de realizar los cálculos en el tiempo disponible).

Por lo tanto, el elemento más importante del filtro digital es la fórmula, no la máquina usada para resolverla, la cual puede ser un microprocesador de propósito general, un procesador DSP especializado, un computador personal, o incluso el cerebro humano, si el proceso es suficientemente lento.

6. Imágenes

De acuerdo a algunas definiciones provenientes del diccionario el término imagen significa: copia, figura, dibujo, fotografía, grabado, ilustración, representación mental de un objeto, siendo correctas todas ellas [3]. Para obtener una imagen no es necesario percibir directamente el fenómeno a representar, por ejemplo el desarrollo tecnológico de las últimas décadas ha permitido la generación de imágenes empleando radiación invisible a la visión humana, imágenes acústicas, magnéticas, de radar [4].

Sin embargo las imágenes más importantes para el ser humano son las imágenes ópticas las cuales pueden percibirse directamente por el ojo humano, éstas se pueden clasificar de muchas maneras pudiéndose ser imágenes fijas, en movimiento, continuas o discretas por mencionar solo algunas de sus características. Una definición más rigurosa de imágenes continuas y discretas es la siguiente [5]:

Una imagen continua es aquella donde la variación de tonos de gris o color se presenta sin discontinuidades, sin líneas o fronteras aparte de las que pudiera tener la escena misma, una imagen discreta por su parte es la que está compuesta por elementos definidos y diferenciados como puntos o cuadrados.

6.1 Imagen digital

Para nuestro trabajo nos interesan solamente las imágenes discretas, como un subconjunto de ellas se pueden encontrar las imágenes digitales; el hecho de que una imagen sea digital implica que los elementos que la forman solo podrán tener valores formados por las combinaciones de 0 y 1. Al digitalizar una imagen, se produce una pérdida de información con respecto a la imagen continua.

El hecho de que la información contenida en una imagen digital, sean combinaciones de unos y ceros permite que se pueda hacer referencia a cualquier cosa, de ahí que la información numérica de una imagen almacenada en un archivo pueda ser teóricamente idéntica a la de un sonido o un texto [5].

6.1.1 Tipos básicos de imágenes digitales

Puesto que la información digital es discontinua toda imagen de este tipo ha de estar dividida en unidades claramente identificables que contengan cada una un conjunto de información determinada, con respecto a esto existen dos tipos de imágenes digitales.

• Las creadas mediante porciones gráficas de la imagen.

• Las creadas mediante elementos definidos matemáticamente.

A las primeras se les denomina imágenes de mapas de bits a las segundas imágenes vectoriales [3].

6.1.2 Imágenes de mapas de bits

Las imágenes de mapas de bits (bitmaps) están formadas por una rejilla de celdas a cada una de las cuales se les denomina píxel (elemento de imagen por sus siglas en ingles), a dichos elementos se les asigna un valor propio (dependiendo del modo de color utilizado) de tal forma que su agrupación crea la ilusión de una imagen en tono continuo.

Los píxeles son unidades de información mas no de medida, significando que contienen información independientemente de su tamaño, por ejemplo un píxel puede ser muy pequeño (0.1mm) o muy grande (1 m).

Una imagen de mapa de bits se modifica por grupos de píxeles, no por los objetos o figuras que contiene, por lo que estos suelen deformarse o perder algunos de los píxeles que los definen; por lo tanto una imagen de bits está diseñada para un tamaño determinado perdiendo calidad si se modifican sus dimensiones [5].

6.1.3 Resolución

La resolución se define como el número de píxeles que tiene una imagen por unidad de longitud, es decir la densidad de píxeles en la imagen, una forma común de clasificar imágenes según su resolución es aquella que las divide en imágenes de alta resolución e imágenes de baja resolución.

A mayor resolución existen más píxeles en una imagen y por lo tanto su mapa de bits es más grande, contiene mayor información y es mayor su capacidad de distinguir los detalles espaciales finos por lo que tendrá más definición, permitiendo transiciones de color más suaves y una mayor calidad de reproducción [6].

6.1.4 Profundidad de bits

Como parte de la información que contiene un píxel para representar la imagen original se le asigna una cantidad determinada de bits, a esta cantidad se le denomina profundidad de bits. Se trata de un concepto importante porque a mayor profundidad de bits más información contiene la imagen y por consiguiente se puede tener un mayor número de colores.

• Si la profundidad es de un solo bit solo existe la posibilidad de tener dos niveles o tonos.

• Si la profundidad es de dos bits es posible tener cuatro niveles o tonos. Los niveles que podrá contener una imagen se encuentran mediante la siguiente relación 2L, siendo L el número de profundidad de bits; para imágenes en tono real se tiene una profundidad de 24 bits generalmente lo cual genera 16,777,216 colores posibles para su representación [7].

6.1.5 Peso

Los mapas de bits pueden estar definidos en un número variable de colores, cuantos más colores tenga la imagen, mayor calidad tendrá, así mismo si su resolución es elevada, mayor peso requerirá para su almacenamiento. El peso de una imagen digital sin ninguna técnica de compresión aplicada se define en la ecuación como [8]:

$$\frac{(\text{Resolución ancho} \times \text{Resolución alto}) \times \text{profundidad bits}}{8192} \tag{6.1}$$

Esta proporciona el peso de la imagen en Kbytes, como ejemplo se tiene una imagen con resolución de 256 píxeles x 256 píxeles, con una profundidad de bits de 24 (imagen en tono continuo) su peso es:

((256 x 256) x 24) / 8192 = 192 Kbytes

6.1.6 Frecuencia espacial

Un concepto íntimamente ligado con las imágenes es el de frecuencia espacial, para su comprensión es necesario abordar el concepto de frecuencia temporal, en general las señales unidimensionales de alta frecuencia cambian su valor en un período corto de tiempo, si se supone una señal senoidal con periodo 2π y otra con periodo π, está última con menor periodo cambia más rápidamente que la primera es decir tiene mayor frecuencia.

Si se considera una imagen como la señal a tratar, se define que una imagen con alta frecuencia espacial, cambia el valor de los niveles de intensidad en un intervalo pequeño, o lo que es lo mismo en distancias pequeñas de la imagen, el resultado visual es que los niveles de intensidad cambian de forma abrupta de un nivel a otro, por el contrario las bajas frecuencias corresponden a cambios más lentos en la variación de los niveles, donde los cambios ocurren gradualmente de una posición a otra de la imagen [8].

6.1.7 Espacios de color

Desde que Sir Isaac Newton descubriera la descomposición de la luz en el espectro se han propuesto numerosos modelos sobre qué es el color, el más importante de todos ellos, y que aún sirve de fundamento a la mayoría de los sistemas de color usados en la actualidad fue el de Thomas Young, que en el siglo XVIII propuso que el ojo creaba todos los matices mediante una mezcla de tres colores básicos; esta hipótesis es conocida como teoría tricromática de la luz [7].

La teoría tricromática señala que idealmente tres arreglos de muestras (tres componentes) deben ser suficientes para representar una imagen en color. El sistema RGB (rojo, verde y azul) es un ejemplo de la representación en color que

requiere tres valores independientes para describir los colores, cada uno de los valores puede ser variado independientemente y por lo tanto se puede crear un espacio tridimensional con las tres componentes. Como coordenadas independientes, los colores son representados como puntos en este espacio los tonos de gris desde el negro hasta el blanco se encuentran en la línea diagonal de la Figura 6.1 [9].

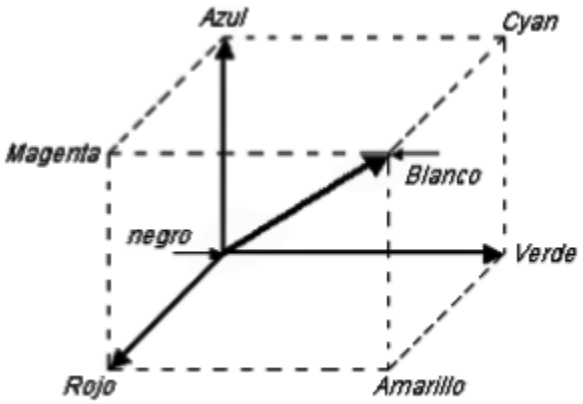


Figura 6.1.Espacio coordenado de RGB

6.1.8 Transformaciones lineales del color

Las representaciones tales como RGB no siempre son las más convenientes, otras representaciones que usan componentes de color están relacionadas directamente con la forma visual de percepción.

Los espacios de color o sistemas de color coordenado tienen una componente de luminancia y los otros dos de crominancia, la luminancia provee una versión en escala de grises de la imagen y los componentes de crominancia proveen la información extra que convierte la imagen en escala de grises a una imagen en color [9].

El sistema visual de luminancia es la suma de diferentes porciones de los componentes R, G, B por conveniencia los valores de estos tres colores son expresados por una escala relativa de 0 a 1, donde 0 indica que no existe

excitación y 1 indica máxima excitación; la luminancia Y puede ser calculada por la ecuación [9].

$$Y = 0.3\ R + 0.6\ G + 0.1\ B \tag{6.2}$$

El término crominancia está definido como la diferencia entre un color y la luminancia, la información de crominancia puede ser expresada por un conjunto de diferencias de color Cb e Cr que se definen de acuerdo a las ecuaciones (6.3) e (6.4) respectivamente [9].

$$Cr = ((R - Y)/1.6) + 0.5 \tag{6.3}$$

$$Cb = ((B - Y)/2) + 0.5 \tag{6.4}$$

Estas diferencias de color son cero si R = G = B lo que produce diferentes niveles de gris, que no contienen crominancia; junto con la luminancia estas coordenadas forman el sistema de color conocido como YCrCb. Otro espacio de color YIQ es usado en algunos sistemas de TV, el sistema YIQ presenta la ecuación (6.2) para su componente de luminancia, pero las ecuaciones para I e Q son definidas en (6.5) y (6.6).

$$I = 0.6\ R - 0.282\ G - 0.317\ B \tag{6.5}$$

$$Q = 0.213\ R - 0.534\ G + 0.321\ B \tag{6.6}$$

Independientemente del modo de color utilizado en la representación de la imagen existen algunos conceptos importantes relacionados con su tratamiento [7].

• Brillo: Es la intensidad de iluminación total o promedio que determínale nivel de fondo en la imagen reproducida.

• Contraste: Es la diferencia de intensidad entre las partes blancas y las negras de la imagen reproducida. El intervalo de contraste debe ser tan amplio como para producir una imagen intensa, con blanco brillante y negro oscuro para valores extremos.

• Saturación: Los colores saturados son brillantes, intensos o fuertes, los colores pálidos o débiles tiene poca saturación; la saturación indica cómo está diluido el color por el blanco; por ejemplo el rojo brillante o intenso está completamente saturado cuando éste se diluye por el blanco, el resultado es el rosa que en realidad es un rojo no saturado.

• Tinte: De manera más específica el color de un objeto se denomina tinte o matiz, las hojas verdes tienen matiz verde una manzana roja tiene matiz rojo.

6.2 Imágenes vectoriales

Las imágenes vectoriales también son conocidas como gráficos orientados a objetos, es el segundo grupo de imágenes digitales y son más simples que los mapas de bits, ya que se almacenan y representan por medio de trazos geométricos controlados por cálculos y fórmulas matemáticas, éstas no se construyen píxel a píxel, sino que se construyen a partir de vectores objetos formados por una serie de puntos y líneas rectas o curvas definidas matemáticamente.

Las principales ventajas que ofrecen las imágenes vectoriales derivadas de su naturaleza matemática son:

• Almacenan las imágenes en archivos muy compactos ya que solo se requiere la información necesaria para generar cada uno de los vectores, dado que no se ha de almacenar información para definir cada píxel de la pantalla, sino una serie de fórmulas matemáticas.

• Permiten modificar el tamaño de las imágenes y de sus objetos sin que se produzca pérdida de información ya que se analizan de forma matemática todas las nuevas relaciones y posiciones de los elementos [5].

6.3 Segmentación de imágenes

La segmentación es una de las tareas más importantes en el procesamiento de imágenes y visión por ordenador. La segmentación de imágenes es la operación

que marca la transición entre el procesamiento de imágenes de bajo nivel y análisis de imágenes: la entrada de un bloque de segmentación en un sistema de visión artificial es una imagen pre procesado, mientras que la salida es una representación de las regiones dentro de la imagen. Esta representación puede adoptar la forma de los límites entre las regiones o información acerca de qué píxel pertenece a qué región.

La segmentación se define como el proceso de dividir una imagen en un conjunto de regiones no superpuestas cuya unión es la imagen completa. Estas regiones deberían idealmente corresponder a los objetos y sus partes significativas, y de fondo. La mayoría de los algoritmos de segmentación de imágenes se basan en una de las dos propiedades básicas que se pueden extraer de pixel valores-discontinuidad y la similitud, o una combinación de ellos.

La segmentación de imágenes no triviales es un grave problema, aún más difícil por la iluminación no uniforme, sombras, la superposición entre los objetos, falta de contraste entre los objetos y el fondo, y así sucesivamente que ha sido abordado desde muchos ángulos diferentes, con un éxito limitado a este fecha. Muchas de las técnicas de segmentación de imágenes y algoritmos se han propuesto e implementado durante los últimos 40 años y, sin embargo, a excepción de las escenas relativamente "fáciles", el problema de la segmentación sigue sin resolverse [7].

6.4 Formación de una imagen

El proceso para formar una imagen es el siguiente: La luz que se refleja en los objetos llega a la cámara, que posee sensores como los de la retina de nuestros ojos (bastones y conos). Las cámaras poseen arreglos de sensores (e.g.1000x1000) que son capaces de discretizar la luz que están recibiendo. Discretización es el proceso de convertir una señal analógica a una señal digital con el objetivo que ocupe menos espacio de memoria. Este proceso que lo aplica el sensor de la cámara en dos dimensiones (espacio y amplitud) realiza dos subprocesos: muestreo (sampling) que se encarga de tomar muestras finitas en el

dominio del tiempo y la cuantización que toma aproximaciones en los valores de grises (hace uso de redondeos) [3]. Una buena cámara tiene una gran cantidad de capacidad de muestreo y busca que los valores de cuantización sean lo más pequeños posibles para no notar que está discretizado.

RGB (Red Green Blue) es el modelo utilizado en las cámaras para representar los colores [3]. El mejor de los casos y el más costoso es que la cámara tome tres distintas imágenes en cada uno de los colores (rojo, verde y azul) y luego los sobreponga, pero en cámaras de menor costo lo que se hace es tener una sola imagen alternando los colores RGB en los píxeles.

Para el caso de los videos lo que tenemos es por ejemplo 30 imágenes RGB que se reproducen por segundo lo que representa 30 fps. El valor estándar utilizado para representar la escala de grises es de 8 bits (2^8=256). Al tener imágenes compuestas de 3 colores, hablamos de que la imagen está compuesta de 24bits [3].

A veces los sensores tienen problemas cuando se presenta una región muy oscura y una muy clara juntas, el sensor tiene una cantidad de niveles finito (256) y puede que lo que queramos capturar salga de ese rango. Cuando el valor es por encima de 255 se produce saturación y cuando es menor a 0 se produce ruido.

6.5 Operaciones por medio de mapa de bits

Las imágenes en escala de grises pueden ser transformadas en una secuencia de imágenes binarias por medio de su plano de bits. Los planos de bits o mapas de bits se las suele definir por su altura y anchura (en píxeles) y por su profundidad de color (en bits por píxel), que determina el número de colores distintos que se pueden almacenar en cada punto individual, y por lo tanto, en gran medida, la calidad del color de la imagen [10].

Con la transformación de los mapas de bits o planos de bits podemos obtener la realización de operaciones básicas matemáticas, algunos ejemplos de cómo sería esto son [10]:

Suma o adición

Figura 6.2. Operación de suma

La operación de suma se realiza aplicándole a la imagen original una adición de un valor numérico definido (para el ejemplo es 20) que se hace punto a punto, tomando en cuenta que la imagen es considerada como una matriz con valores que van desde 0 a 255, esto se realiza igual que una suma en una matriz y el resultado es la imagen de la derecha en comparación con la original que se encuentra a la izquierda.

Resta o substracción

Figura 6.3. Operación de resta

La operación de resta se efectúa de la misma manera que la suma ya que aplica la misma teoría, sin embargo el resultado en este caso es una imagen con valores

más cercanos a 0 o al negro, mientras que en la suma son valores cercanos al blanco que es 255.

Multiplicación

Figura 6.4.Operación de multiplicación

La operación de multiplicación funciona igual que las anteriores ya que lo que hace es aumentar el número de los valores en la matriz, por ende la imagen se va a esclarecer, funcionando así como un filtro para agregar más brillo a una imagen.

Complemento

Figura 6.5.Operación de complemento

El complemento de una imagen equivale a cambiar la polaridad de esta, es decir, que dentro de la imagen los valores oscuros pasan a ser claros y viceversa. Al complemento también se le conoce como un filtro denominado negativo, que realiza precisamente el efecto de invertir los tonos en una imagen.

7. Video

Una imagen “fija” es una distribución espacial de intensidad que es constante con respecto al tiempo, por otro lado el vídeo es un patrón de intensidad espacial que cambia con respecto al tiempo, otro significado común para video es el definirlo como un conjunto o secuencia de imágenes con alto grado de relación entre sí, por tanto el video se puede considerar como una secuencia de imágenes fijas [11].

7.2 Vídeo Analógico

En una señal de video la amplitud del voltaje o de la corriente cambia respecto al tiempo (como en una señal de audio), las variaciones en la señal de video corresponden a información visual, producida por una cámara que convierte la luz en señales eléctricas [9].

El video tradicionalmente ha sido capturado, almacenado y transmitido en forma analógica, el término señal de video análoga hace referencia a una señal eléctrica unidimensional de tiempo que es obtenida por el muestreo del patrón de intensidad en las coordenadas horizontal, vertical y temporal y convertido en una representación de tipo eléctrica, este proceso de muestreo es conocido como barrido [11].

El barrido de una imagen comienza en la esquina superior izquierda y continua horizontalmente, cuando alcanza el final de la línea regresa al principio solo que ahora salta a la siguiente línea para comenzar de nuevo; una vez que alcanza el final de la última línea se ha completado un cuadro y el barrido regresa de nuevo a la esquina superior izquierda para comenzar a trazar el siguiente cuadro, durante el trazo son insertados los pulsos de sincronización y de borrado en la señal.

Los métodos comúnmente utilizados generan un barrido progresivo o entrelazado; en el progresivo un cuadro es formado por un simple barrido continuo, en el método entrelazado un cuadro es formado por dos barridos sucesivos denominados campos, en el primer campo las líneas impares son barridas

formando la mitad de un cuadro, luego las líneas pares son barridas para formar el segundo campo, cuando se entrelazan las líneas los dos campos forman un cuadro.

Aunque parezca que las imágenes reproducidas por una señal de video se mueven, en realidad se muestran una serie sucesiva de imágenes fijas con suficiente rapidez para dar la ilusión de movimiento al espectador.

Líneas por segundo: el número de líneas barridas para una imagen completa debe ser grande, con la finalidad de incluir el mayor número de elementos de imagen y por tanto más detalle, existen diferentes estándares con diferente resolución, en uno de ellos la cantidad de líneas posibles por cuadro es 525 para formar la imagen completa.

Cuadros por segundo: como se mencionó el barrido horizontal se realiza en forma lenta hacia abajo, mientras se realiza el barrido horizontal también se efectúa un barrido vertical, este movimiento es necesario para no barrer las líneas una encima de la otra; el barrido horizontal produce líneas de izquierda a derecha mientras que el vertical distribuye las líneas para completar el cuadro de abajo hacia arriba.

Un cuadro está formado por 720 píxeles por línea y posee 525 líneas. Estas 525 líneas se exploran en 1/29.97 segundos, para crear la ilusión de movimiento deben mostrarse las imágenes completas suficientes durante cada segundo, este efecto se logra si se tiene una tasa de repetición de imágenes mayor que 16 por segundo. La tasa de repetición de 24 imágenes por segundo empleada en las películas de cine es suficiente para producir la ilusión de movimiento en la pantalla.

Para eliminar posibles efectos de parpadeo en el sistema cada cuadro se repite dos veces, esto se logra entrelazando las líneas de barrido horizontales en los dos campos. La tasa de repetición de los campos es 60 por segundo porque se barren dos campos durante un periodo de cuadro de 1/30 seg de esta manera se

muestran 60 vistas de la imagen durante un segundo, esta tasa de repetición es suficientemente rápida para eliminar cualquier efecto de parpadeo.

La frecuencia de barrido vertical es de 60 Hz, y es la rapidez del trazo de vídeo en que completa sus ciclos de movimiento vertical de arriba abajo y luego de regreso hacia arriba. El número de líneas de barrido horizontales en un campo es la mitad del total de 525 líneas para un cuadro completo, como un campo contiene líneas alternadas se obtienen 262.5 líneas horizontales para cada campo vertical; y el tiempo para un campo es de 1/60 seg, entonces el número de líneas por segundo es:

262.5 x 60 Hz = 15750 Hz

O bien si se consideran 525 líneas para un par sucesivo de campos se puede multiplicar la rapidez de cuadros de 30 por 525 que da el mismo resultado de 15750 líneas barridas en 1 segundo. Esta frecuencia de 15750 Hz es la velocidad a la cual el trazo de barrido completa sus ciclos de movimiento horizontal de izquierda a derecha y de regreso a la izquierda. El tiempo para cada línea de barrido horizontal es 1 / 15750 segundos, lo cual en términos de microsegundos es:

Tiempo H = 1 / 15750 = 63.5 μseg

El tiempo en μs indica que la señal de video para los elementos de imagen dentro de una línea horizontal puede tener altas frecuencias del orden de MegaHertz, si hubiera más líneas el tiempo de barrido sería menor y se obtendrían frecuencias de video mayores, por lo tanto la mayor frecuencia de video está limitada a cerca de 4.2 MHz [9].

7.3 Análisis de la señal de video

Las tres partes principales de la señal de video compuesto son:

1) Señal de cámara correspondiente a las variaciones de luz en la escena.

2) Pulsos de sincronización o sincronía para el barrido.

3) Pulsos de borrado que hacen invisibles los retornos.

La señal de cámara se combina con el pulso de borrado luego se añade la sincronización para producir la señal de video compuesta de la Figura 7.1.

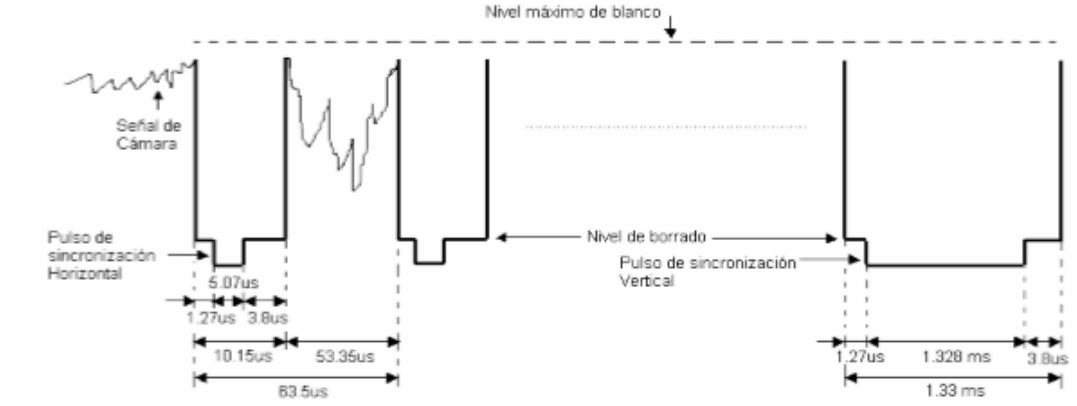


Figura 7.1.- Señal de video compuesto analógica

En la Figura 7.1 se muestran los valores sucesivos de amplitud de voltaje para el barrido de dos líneas horizontales en la imagen, cuando aumenta el tiempo en la dirección horizontal las amplitudes varían para los matices de blanco, gris y negro en la imagen empezando en el extremo izquierdo. La señal está en un nivel de blanco para la primera línea en la segunda estas variaciones son más marcadas hacia el gris y negro, esté proceso continua trazando todas las líneas hasta el final del cuadro donde se genera el pulso de sincronización vertical.

El estándar de la industria para la distribución de video de las señales de entrada y salida de un equipo es una señal como la mostrada en la Figura 7.1 con amplitud pico a pico de 1V, la lista del equipo que la utiliza incluye cámaras, camcorders, videocaseteras, equipos de edición, monitores receptores de TV [9].

7.3.1 Sincronización

Los pulsos de sincronización se añaden como parte de la señal de video completa, un pulso de sincronización horizontal al final de cada línea determina el inicio del

retorno horizontal, el retorno horizontal de barrido comienza en el lado derecho de la imagen.

La sincronización vertical al final de cada campo determina el inicio del retorno vertical. Si la sincronización de campo vertical no estuviera presente, la imagen reproducida no se mantiene fija, en el sentido vertical se movería de arriba hacia abajo en la pantalla del tubo de imagen. Sin los pulsos horizontales la imagen no se detiene en el sentido horizontal, se desliza a la izquierda o a la derecha y después se separa en segmentos diagonales.

La frecuencia de barrido horizontal es 15750 Hz por lo tanto la frecuencia de sincronía horizontal también es 15750 Hz, la velocidad de repetición de los cuadros es 30 seg, pero la frecuencia de barrido vertical de campos es 60 Hz, debido a esto la frecuencia de los pulsos de sincronización vertical también es 60 Hz [7].

7.3.2 Borrado

El término borrado significa pasar a negro, es decir, ennegrecer la pantalla como parte de la señal de video, el voltaje de borrado se encuentra en el nivel de negro, la señal de video compuesta contiene los pulsos de borrado para hacer invisibles las líneas de retorno cambiando la amplitud de la señal a negro, cuando se producen los retornos toda la información de la imagen queda suprimida durante el tiempo de borrado.

El tiempo necesario para el borrado horizontal es aproximadamente 16% del tiempo de cada línea horizontal (H), el tiempo H total es 63.5μs que incluye el trazo y el retorno, el tiempo de borrado para cada línea es entonces 63.5μs x 0.16 = 10.15 μs, este tiempo de borrado H significa que el retorno de derecha a izquierda debe hacerse en 10.15 μs antes del inicio de la información visible de la nueva línea, durante el barrido de izquierda a derecha.

El tiempo para el borrado vertical (V) se aproxima a 8% del tiempo de cada campo vertical (V), el tiempo total vertical es 1/60 seg que incluye el trazo hacia abajo y el retorno hacia arriba entonces el tiempo de barrido de cada campo es 1/60 x 0.08 =

0.0013 seg (1.3 ms) en este tiempo el retorno vertical debe completarse de abajo hacia arriba de la imagen [9].

7.3.3 Adición del color

El análisis descrito anteriormente es para una señal en tonos de gris (Y), para una señal de video en color se agregan la señal de crominancia (modulada en cuadratura) de 3.58 MHz junto con una señal de ráfaga de sincronización de color, el intentar describir a fondo la realización de este proceso queda fuera de este estudio si se desea más información puede consultar las referencias señaladas [7].

7.4 Formatos de video NTSC, PAL, SECAM

Existen diferentes formatos de vídeo compuesto tales como NTSC (comité nacional de sistemas de televisión por sus siglas en inglés), PAL (Alternación línea Fase) y SECAM (Systeme Electronique Color Avec Memorie) usados en diferentes países alrededor del mundo. El estándar de video compuesto NTSC definido en 1952 es usada principalmente en Norteamérica y Japón, la señal NTSC es una señal de video entrelazado con 262.5 líneas por campo (525 por cuadro), 60 campos por segundo, con una resolución horizontal de 720 elementos de imagen (también denominados píxeles aunque estos no están cuantizados por lo tanto no son digitales).

PAL y SECAM desarrollados en la década de los 60 son comúnmente utilizados en Europa, también es video entrelazado pero tiene una resolución temporal diferente en comparación con NTSC, además de tratar al color de una manera distinta. Ambos PAL y SECAM tienen 625 líneas por cuadro resolución horizontal de 720 y 50 campos por segundo, por lo tanto tienen una resolución vertical mayor que NTSC, estos parámetros generan una frecuencia de barrido de 15625 Hz; una diferencia entre PAL y SECAM es en la manera de representación del color, ambos representan mejor la información que NTSC [7].

7.5 Formatos de video analógico

Los formatos de señal compuesta usualmente generan errores en el rendimiento del color, tales como errores en la saturación del color, debido a las imprecisiones en la separación de la señal de color. Existen varios estándares de señales de video analógico los cuales difieren en la forma de representar el color, los más importantes son [9]:

• Vídeo compuesto (PAL, SECAM, NTSC) descritos anteriormente.

• Vídeo componente: La señal de video es dividida en tres partes, pueden ser señales RGB, luminancia-crominancia o una transformación de ellos, este formato permite la mejor representación del color porque utiliza componentes separados.

• S-video: Es un compromiso entre el video compuesto y el video de componentes, donde se representa el vídeo con dos señales una de luminancia y una señal compuesta de crominancía que puede estar basada en la representación (I,Q) o (U,V) de los sistemas NTSC, PAL y SECAM, S-video es comúnmente usado en grabadoras de videocasete y camcorders para obtener mejor calidad que la del vídeo compuesto.

7.6 Video digital

El video digital es simplemente un medio alternativo de representar la información contenida en una forma de onda de video analógica, utilizando las herramientas que nos proporciona la tecnología digital.

El proceso de digitalización de vídeo involucra tres operaciones básicas: filtrado, muestreo y cuantización; si la frecuencia de muestreo no es el doble que la frecuencia máxima de la señal analógica ocurrirán efectos de aliasing, por lo tanto

la operación de filtrado es usada para limitar el ancho de banda de la señal de entrada; la señal analógica filtrada es muestreada en un número específico de veces para generar una señal en tiempo discreto la mínima razón de muestreo es conocida como la razón de Nyquist. Por ejemplo para un sistema PAL el intervalo es de 10 – 11 MHz con lo cual la frecuencia de muestreo normalmente es de 13.5 MHz.

Las muestras resultantes del proceso de muestreo tienen amplitudes continuas, por lo tanto se requerirá precisión infinita para representarlos; la operación de cuantización es usada para mapear tales valores hacía un conjunto finito de amplitudes discretas que pueden ser representados por un número finito de bits. Cada muestra es definida como un elemento de imagen y es organizado en un arreglo bidimensional para formar una imagen digital fija o cuadro digital; por lo tanto el video digital consiste de una secuencia de imágenes digitales fijas (bitmaps).

Para producir imágenes en movimiento es necesario proporcionar un mecanismo donde el valor de cada píxel pueda actualizarse periódicamente esto da como resultado una matriz tridimensional donde dos de los ejes son espaciales y el tercero es temporal esto se muestra en la Figura 7.2, en ella se aprecia como el video es una sucesión de imágenes bidimensionales que ocurren en instantes de tiempo discreto [12].

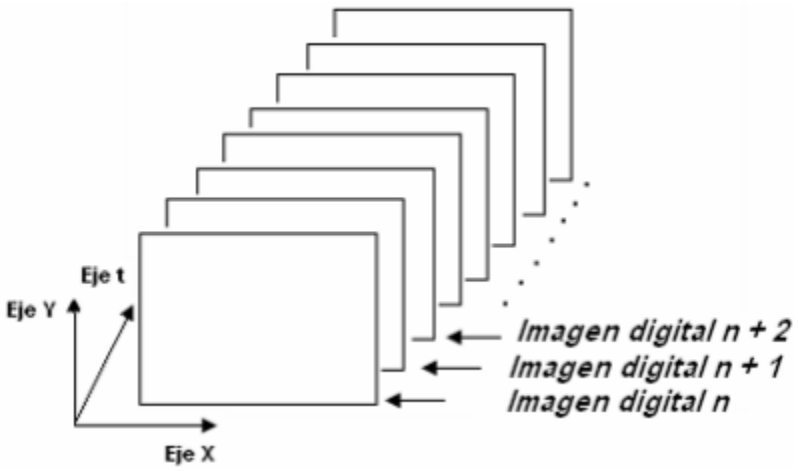


Figura 7.2 . Video digital

Si la señal de video digital se encuentra en color entonces está será representada como en la figura anterior solo que cada imagen tendrá asociada sus respectivos componentes para el sistema de color que utilicen (RGB, YCrCb, etc). En el video digital no existe la necesidad de utilizar pulsos de sincronía y borrado por lo que la computadora conoce exactamente donde empieza una nueva línea, su tamaño y el número de píxeles, por lo tanto estos pulsos son removidos en la conversión A/D. Considerando como ejemplo el estándar Europeo de TV que difiere para el Norteamericano, pero el número de sus muestras por línea (píxeles) es 720 para ambos, dado que en el video digital sólo interesan las partes activas de la imagen y el número de líneas activas por cuadro es de 625, el número total de píxeles por segundo llega a ser igual a 720 x 625 x 25 = 11, 250, 000 píxeles/s.

La razón de bit total es calculada considerando los diferentes componentes de la imagen así como su profundidad de bits, suponiendo componentes RGB y una profundidad de 8 bits se tiene 11, 250, 000 x 3 x 8 = 270, 000, 000 bits/s ó 257.49 Mbits/s lo cual es una razón muy elevada para su ancho de banda, esta es la mayor desventaja del video digital en la actualidad tanto para su almacenamiento como para su transmisión [12].

7.6.1 Ventajas del video digital

Las aplicaciones del video digital se están volviendo cada vez más importantes, entre las principales se pueden citar, videotelefonía, videoconferencia, TV digital, TV digital de alta definición (HDTV) , aprendizaje a distancia, vigilancia, etc [8].

Las principales ventajas del video digital sobre el analógico son [11]:

• Fácil de editar, mejoras y creación de efectos especiales.

• Evita las fallas típicas del video analógico como por ejemplo aquellas causadas por la grabación repetida de las cintas, errores en la representación del color debido a las inexactitudes en la separación de señales de video compuesto.

• Fácil conversión de formatos por medio de Software, para video analógico la conversión necesita costosos equipos.

• Robustez al ruido y facilidad para los procesos de encriptación.

• Interactividad, fácil de indexar, buscar y recuperar para video analógico se requiere de tediosos escaneos.

Estas ventajas permiten un número de nuevas aplicaciones y servicios para ser introducidos, como por ejemplo la interactividad producida en la HDTV.

7.7 Formatos de vídeo digital

Debido al muestreo, una señal de video alcanza una gran tasa de datos, como se vio anteriormente la tasa generada por un sistema PAL es 257.49 Mbits/s, tal cantidad de datos imposibilitaría la aplicación del video digital en las aplicaciones de consumo (tales como camcorders digitales, TV digital). Una solución a este problema es reducir el número de componentes para las que el ojo humano tiene poca sensibilidad y a las técnicas de compresión.

Dependiendo de la aplicación, las imágenes tienen diferentes grados de resolución, por ejemplo en videoconferencia o video teléfono las imágenes son de tamaño pequeño con resoluciones bajas y por tanto requieren mucho menor ancho de banda para su transmisión. A su vez el intercambio de video digital entre diferentes industrias, redes y plataformas de Hardware requiere de formatos de video estandarizados.

Los patrones denominados de video digital definen la composición de las muestras de las componentes de imagen cuando tienen la forma YCrCb, los estudios llevados a cabo sobre percepción de detalles de imagen han demostrado que el ojo es considerablemente sensible a la luminancia, pero mucho menos a los colores, lo que permite suprimir muestras de las señales de diferencia de color sin pérdida apreciable de calidad para una visión promedio humana; dando lugar a una reducción considerable de la información de video [12].

Los patrones de submuestreo comúnmente utilizados para la crominancia son los siguientes [11]:

• Patrón 4:4:4: En este formato le corresponden igual número de muestras a cada componente, YCrCb, es decir, se tiene la misma relación de píxeles tanto horizontal como verticalmente.

• Patrón 4:2:2: En este formato existen dos muestras de Y por una muestra de Cr y Cb respectivamente en la dirección horizontal, manteniendo la misma relación entre componentes para la dirección vertical.

• Patrón 4:1:1: En este formato existen cuatro muestras de Y por una muestra de Cr y Cb respectivamente en la dirección horizontal, manteniendo la misma relación entre componentes para la dirección vertical.

• Patrón 4:2:0: En este formato existen dos muestras de Y por una muestra de CrCb en las direcciones horizontales y verticales, con lo cual se logra disminuir la cantidad de información la Crominancia con respecto a la Luminancia en un factor de 2:1.

El Comité Internacional Consultivo para la Radio (CCIR), en su recomendación 601 define el formato de televisión digital estándar (SDTV) para el intercambio y transmisión de video digital, como en los estándares analógicos CCIR-601 define dos sistemas el 525/60 y 625/50, los cuales utilizan el patrón 4:2:2, donde los componentes están cuantizados en ocho bits. En el sistema 525/60 la componente de luminancia de un cuadro tiene dimensiones activas de 720 píxeles x 480 líneas y las componentes de crominancia tienen dimensiones de 360 píxeles x 480 líneas.

En el sistema 625/50 los valores correspondientes son 720 píxeles x 576 líneas para luminancia y 360 píxeles x 576 líneas para crominancia, a pesar de las diferencias entre los dos sistemas se genera la misma cantidad de bits 165.89 Mbits/s [11].

7.7.1 Formatos SIF y QSIF

Para aplicaciones de almacenamiento existe un formato de baja resolución llamado “formato de entrada fuente” (SIF), este es un formato progresivo (4:2:0), este formato tiene 360 píxeles por línea para la luminancia; debido a que existen dos sistemas para SDTV (CCIR-601) tenemos por lo tanto dos formatos SIF. El primero tiene una componente de luminancia de 360 píxeles x 240 líneas, las componentes de crominancia tienen 176 píxeles x 120 líneas, y una razón de cuadros de 30/s.

Mientras que el segundo formato tiene una luminaria de 360 píxeles x 288 líneas, sus componentes de crominancia tienen una resolución de 176 píxeles x 144 líneas y una razón de cuadros de 25/s.

Una versión de baja resolución del SIF es el formato cuarto de SIF (QSIF por sus siglas en ingles), tiene la mitad de las dimensiones del SIF en ambas direcciones. De nuevo existen dos versiones, la primera tiene 176 píxeles x 120 líneas para su luminancia, 88 píxeles x 60 líneas de crominancia y una razón de cuadros de 30/s; mientras que la segunda tiene una luminancia de 176 píxeles x 144 líneas, y 88 píxeles x 72 líneas de crominancia a una razón de 25/s [11].

7.7.2 Formato CIF

El formato CIF fue definido para lograr coincidir con ambos formatos 525/60 y 625/50, en este formato la componente de luminancia tiene una resolución horizontal que es la mitad de ambos sistemas del CCIR-601, una resolución vertical que es la mitad del sistema 625/50 y una resolución temporal que es la mitad del sistema 525/60. La elección intermedia de una resolución vertical de un sistema y la resolución temporal del otro permiten nombrarlo formato intermedio común CIF, este formato CIF es progresivo (4:2:0) y utiliza una razón de cuadros de 30/s [11].

7.7.3 Familia de formatos CIF

Para aplicaciones tales como video sobre redes móviles o video telefonía es posible reducir la razón de cuadros, ésta puede variar entre 15, 10, 12.5, 8.3, 7.5 o

5 por segundo. Existe un número de miembros de mayor y menor resolución que pueden utilizar estas razones (pertenecientes al formato CIF).

Tabla 7.1 Formatos CIF

	Píxeles por Línea	*Líneas por Cuadro*	*Píxeles por Línea*	*Líneas por Cuadro*
SQCIF	*128*	*96*	*64*	*48*
QCIF	*176*	*144*	*88*	*72*
CIF	*352*	*288*	*176*	*144*
4CIF	*704*	*576*	*352*	*288*
16CIF	*1408*	*1152*	*704*	*576*
	Luminancia		***Crominancia***	

Todos estos son formatos progresivos y utilizan el patrón 4: 2: 0 [11].

7.7.4 Otros formatos

La razón de aspecto simplemente es la valor del cociente que resulta de dividir la resolución vertical entre la resolución horizontal, para las imágenes que forman el video. Alrededor del mundo se emplean diferentes razones dependiendo de la aplicación, por ejemplo la razón 3:2 es utilizada para películas de 35 mm; las razones (1.85 a 1, 2.39:1) son utilizadas en los formatos de cine, la razón 16:9 es utilizada para SDTV y HDTV, a su vez HDTV emplea dos tipos de resoluciones 1920 x 1080 y 1280 x 720 pudiendo manejar diferentes tipos de resolución temporal.

Para resoluciones de 1920 x 1080 se emplean las resoluciones temporales de 59.94, 29.97 y 23.97 cuadros/s siendo entrelazado la primera y las otras dos progresivas; en la resolución de 1280 x 720 se utilizan 59.94, 29.97, 23.97 cuadros/s siendo todas ellas progresivas [12].

RGB: Es conocido como un estándar para la representación de imágenes y videos por medio de los tres colores primarios pero también es conocido como un formato para la interpretación de videos digitales mediante el tratamiento de señales, por lo tanto, es el tratamiento de la señal de vídeo que trata por separado las señales de

los tres colores rojo, verde y azul. Al usarlo independientemente, proporciona mayor calidad y reproducción más fiel del color en el video [11].

MJPG: Motion JPEG es un nombre trivial para aquellos formatos multimedia donde cada fotograma o campo entrelazado de una secuencia de video digital es comprimida por separado como una imagen JPEG. Es frecuentemente usado en dispositivos portátiles tales como cámaras digitales. El Motion JPEG utiliza tecnología de codificación intracuadro, que es muy similar en tecnología a la parte I-frame de los estándares de codificación como el MPEG-1 y el MPEG-2, sin emplear la predicción intercuadro. La ausencia del uso de la predicción intercuadro conlleva a una pérdida en la capacidad de compresión, pero facilitando la edición de video, dado que se pueden realizar ediciones simples en cualquier cuadro cuando todos estos son I-frames. Los formatos de codificación tales como el MPEG-2 pueden ser también utilizados basándose meramente en este principio para proveer capacidades similares de compresión y de edición [12].

8. Aplicaciones del procesamiento digital de imágenes y videos y desarrollo de software

Existen en este campo un sin número de aplicaciones que se podrían realizar con ayuda del desarrollo de un software ya que en la actualidad esta es una herramienta muy poderosa, sin embargo hemos seleccionado una aplicación básica de imágenes con la cual los estudiantes podrán comprender como están formadas las imágenes así como entender un poco mejor el uso de Matlab®, ya que todas las guías se desarrollaron en Matlab® para ayudar a los estudiantes a desarrollar sus habilidades con el uso de esta poderosa herramienta la cual a lo largo de la carrera es realmente muy utilizada.

8.1 Desarrollo guía básica de imágenes

La guía básica sobre imágenes consiste en realizar en Matlab® un cambio en la profundidad de bits, una de las variables que influye en la calidad de una imagen. La profundidad de bits es determinada por la cantidad de bits utilizados para definir cada píxel, entonces entre mayor profundidad de bits, mayor será la cantidad de tonos (escala de grises o color) que puedan ser representados. Las imágenes digitales se pueden producir en blanco y negro (en forma bitonal), a escala de grises o a color.

Una imagen bitonal está representada por píxeles que constan de 1 bit cada uno, que pueden representar dos tonos (típicamente negro y blanco), utilizando los valores 0 para el negro y 1 para el blanco o viceversa [3].

Una imagen a escala de grises está compuesta por píxeles representados por múltiples bits de información, que típicamente varían entre 2 a 8 bits o más. Por ejemplo: En una imagen de 2 bits, existen cuatro combinaciones posibles: 00, 01, 10 y 11. Si "00" representa el negro, y "11" representa el blanco, entonces "01" es igual a gris oscuro y "10" es igual a gris claro. La profundidad de bits es dos, pero la cantidad de tonos que pueden representarse es 2^2 ó 4. A 8 bits, pueden asignarse 256 (2^8) tonos diferentes a cada píxel.

Una imagen a color está típicamente representada por una profundidad de bits entre 8 y 24 o superior a ésta. En una imagen de 24 bits, los bits por lo general están divididos en tres grupos: 8 para el rojo, 8 para el verde, y 8 para el azul. Para representar otros colores se utilizan combinaciones de esos bits. Una imagen de 24 bits ofrece 16,7 millones (2^{24}) de valores de color. Cada vez más, los escáneres están capturando 10 bits o más por canal de color y por lo general imprimen a 8 bits para compensar el "ruido" del escáner y para presentar una imagen que se acerque en el mayor grado posible a la percepción humana.

Profundidad de bits: De izquierda a derecha - imagen bitonal de 1 bit, a escala de grises de 8 bits, y a color de 24 bits.

Cálculos binarios para la cantidad de tonos representados por profundidades de bits comunes:

1 bit (2^1) = 2 tonos
2 bits (2^2) = 4 tonos
3 bits (2^3) = 8 tonos
4 bits (2^4) = 16 tonos
8 bits (2^8) = 256 tonos
16 bits (2^{16}) = 65.536 tonos
24 bits (2^{24}) = 16,7 millones de tonos

Ahora bien, Una escala de grises dentro del contexto de las artes gráficas y la educación artística es el sistema ordenado y gradual que cubre un rango limitado de valores de luminosidad entre el blanco, el gris y el negro [3].La escala de grises más simple es la de tres valores: blanco, gris y negro; seguida por la de cinco valores: blanco, gris claro, gris medio, gris oscuro y negro; luego conforme se incrementa el valor de la escala van creciendo los valores de grises que puede ir tomando.

Para qué sirve una escala de grises

La escala de grises es una herramienta de referencia que ayuda a:

- Familiarizarse con las graduaciones de grises entre el blanco y el negro. Son estas graduaciones las que se emplean en el dibujo y la pintura realista para dar la sensación de volumen.
- Percibir el valor de un color independientemente de su tono. Distintos colores tienen también diferentes valores de luminosidad. En un círculo cromático con tonos de colores saturados, por ejemplo, el amarillo tiene un valor luminoso alto mientras que su color complementario, el violeta, es oscuro.
- Medir los valores reales del modelo que se está representando. Debido a los efectos del contraste simultáneo cuando un tono claro y otro oscuro entran en contacto se producen ilusiones ópticas que modifican el grado de luminosidad con el que ambos son percibidos.

- Tener en cuenta el contraste de valores en el diseño de una composición. Explotar el uso del contraste de valores puede facilitar la lectura de una imagen y hacerla más interesante.

Tomando en cuenta lo anterior, hemos decidido utilizar esta aplicación de imágenes ya que consideramos que contiene todos los elementos que nos ayudarían a explicar de mejor forma como se forma una imagen y como es su interpretación real en una computadora ya que prácticamente en la actualidad todas las imágenes y fotos que se capturan son digitales. Con esto creemos que los estudiantes serán capaces de poder explicar cómo se forma una imagen y saber porque cada imagen muestra tan a detalle los colores sin importar cuál sea la combinación que lo genere. Como valor agregado a esto, tendrán la posibilidad de mejorar su dominio en Matlab® que como ya mencionamos anteriormente es fundamental para los próximos años de la carrera.

Análisis del desarrollo de la guía

Para el desarrollo en Matlab® de esta aplicación hemos desarrollado un código dentro de una interfaz gráfica o GUI como se le conoce más comúnmente, la cual le da al usuario la facilidad de realizar tareas o acciones que faciliten la visualización de los cambios que se están realizando. Como resultado del desarrollo del código obtuvimos lo siguiente:

Hemos desarrollado un código con el cual podemos hacer una representación de escala de grises de una imagen junto con su histograma para apreciar mejor las diferentes tonalidades de grises que puede contener en los diferentes niveles de escala de grises los cuales van, en el caso de la guía, de 1 a 8 presentando 8 todos los valores de grises de la imagen y 1 solo el blanco y negro.

Podemos ver la imagen original que se utiliza la cual posee todos sus valores de grises o dicho de otra forma es la imagen original.

Figura 8.1. Imagen original 8 bits

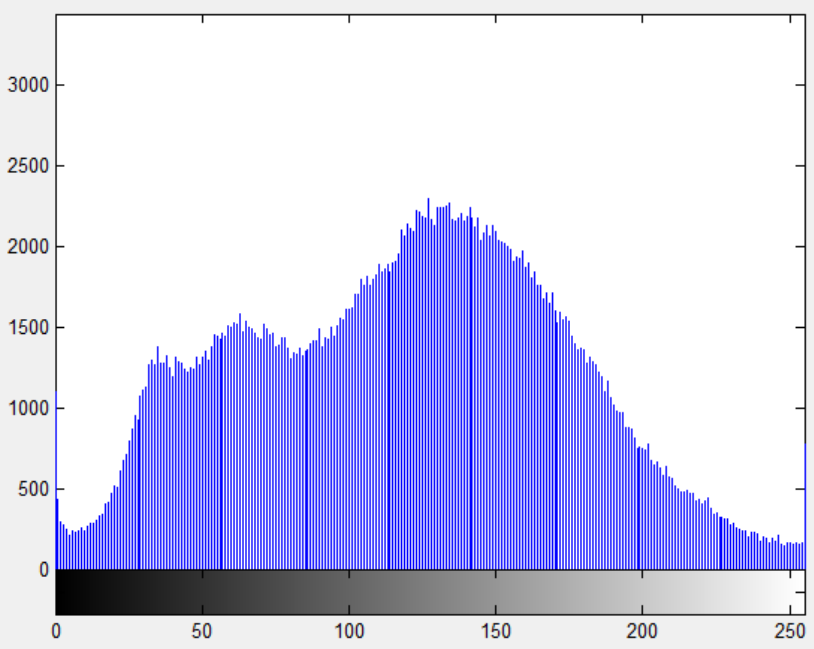


Figura 8.2 Histograma imagen original

Como podemos observar la imagen en blanco y negro se aprecia con todos sus valores de grises que es equivalente a un valor de 8 en nuestra escala y en su histograma podemos ver representados todos esos valores a lo largo de la imagen, sin embargo si comenzamos a reducir los valores se irán perdiendo tonos de grises. Para efectos demostrativos la siguiente representación es con valor de

3 en la escala ya que es un valor en donde se nota bastante la diferencia respecto al original.

Figura 8.3. Imagen reducida 3 bits

Si se compara con la imagen original podremos apreciar a simple vista la diferencia, y es que no solo en la imagen existe una diferencia, también podemos apreciarlo en su histograma el cual se ve drásticamente diferente.

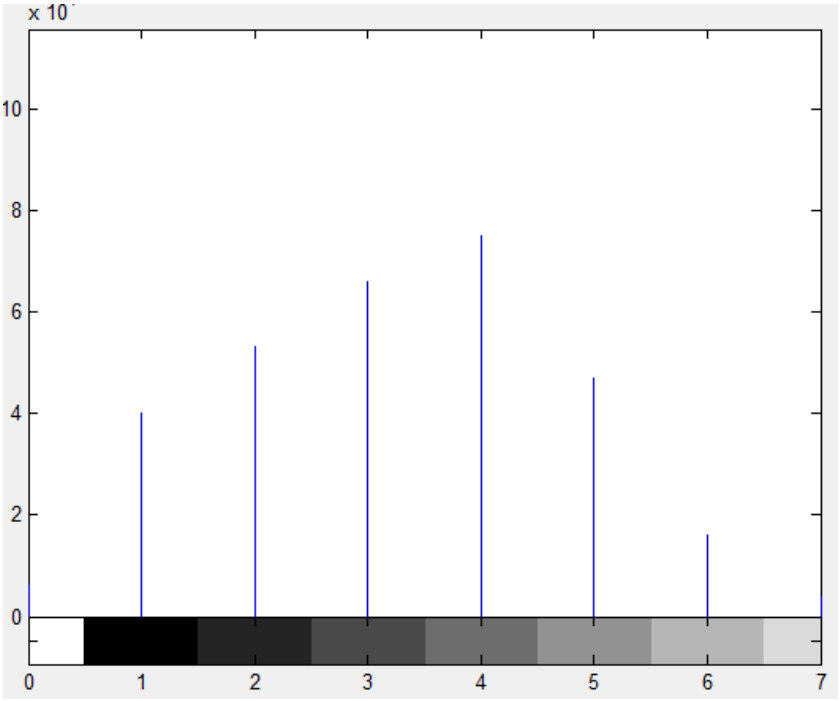


Figura 8.4. Histograma imagen 3 bits

Podemos ver que ambos histogramas son totalmente diferente y si reducimos al mínimo sus valores, es decir, solo blanco y negro tendríamos lo siguiente

Figura 8.5. Imagen reducida 1 bit

Y su histograma con solo dos valores

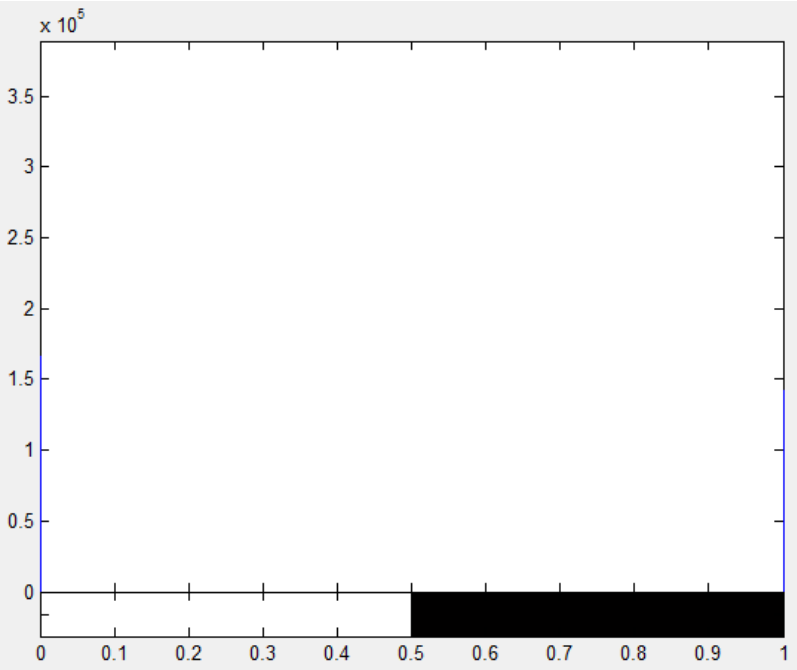


Figura 8.6. Histograma imagen 1 bit

La explicación vista desde el punto de vista de la física de lo que se miran en todas estas imágenes esta tan simple como que la computadora interpreta cada pixel como un valor numérico, entonces al reducir la escala de grises lo valores más cercanos los junta para formar un solo valor o un solo tono de gris el cual va a abarcar un área mayor dentro de la imagen con ese mismo valor. Así se van reduciendo y reduciendo los valores hasta que llega a totalmente negro o totalmente blanco.

Dentro de esta misma guía se realizó un código para la misma representación solo que de imágenes a colores. Previamente hablamos sobre la representación en tres planos de rojo, azul y verde de la profundidad de bits de las imágenes, entonces tenemos la siguiente imagen

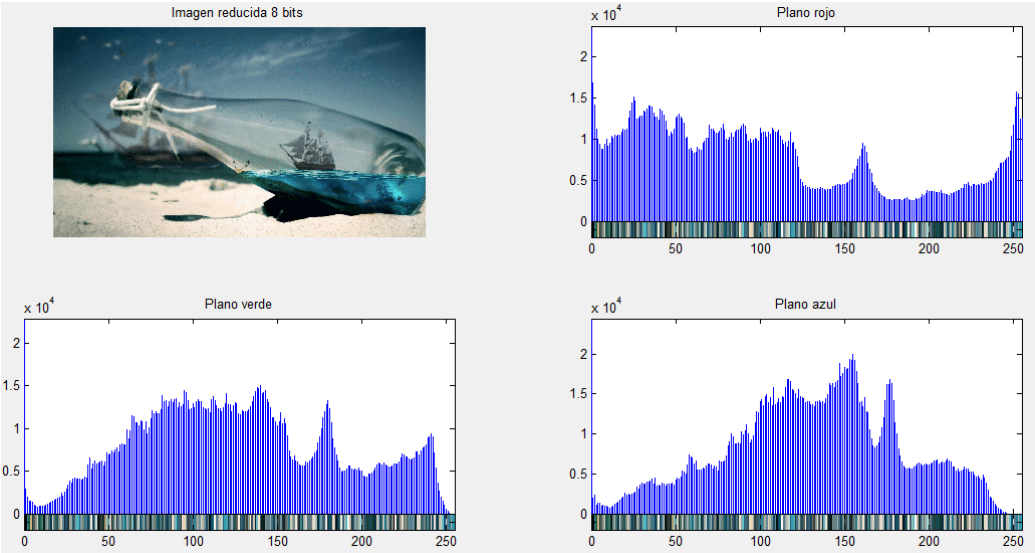


Figura 8.7. Imagen a color con histogramas de RGB

De esta imagen podemos ver que sus histogramas, al ser una imagen a color, están divido en tres planos que representan cada color por separado, y dentro de cada color, el rango de tonalidad que puede tomar cuando la imagen está en sus 8 bits originales en los tres planos.

Al reducir de bits la imagen (específicamente a 4) podemos apreciar que la imagen, además de verse borrosa y con un poco de ruido, sus histogramas cambian en los tres planos por igual, representando menos rango de colores por cada plano.

Al compararlo con la original, podremos notar el cambio tanto en el fondo que se mira más ruidoso y borroso, como el barco dentro de la botella que se mira más borroso, además de corroborar esto con los histogramas donde podemos ver que la representación de colores en cada plano es mucho menor.

La imagen es:

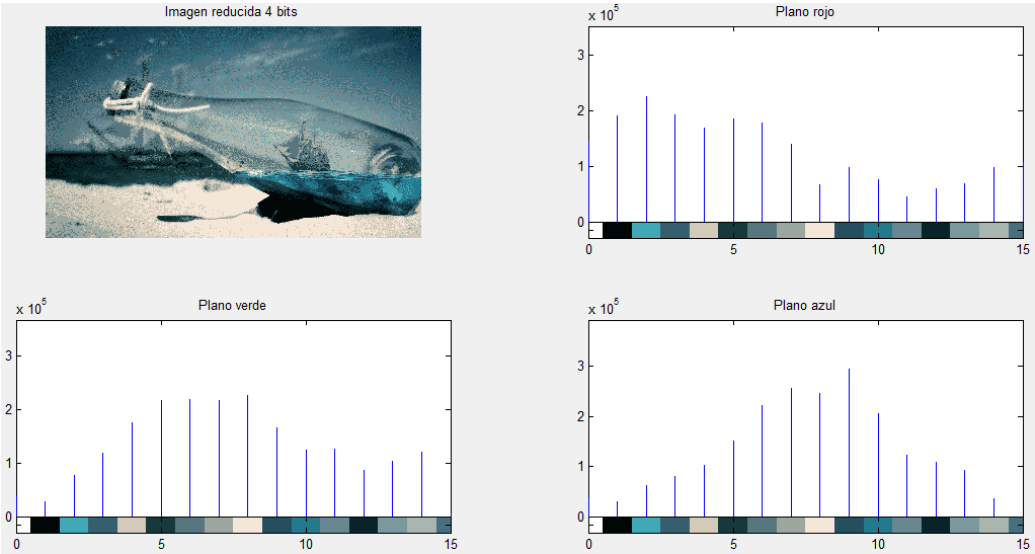


Figura 8.8. Imagen con histogramas reducida a 4 bits

Y por último, tenemos la presentación a un bit de la imagen, donde igual que en la imagen de blanco y negro, tenemos solo dos posibles colores que vendrían a ser nuestro 0 y 1 pero para cada plano del RGB, es decir, dos posibles colores para el plano verde, dos posibles colores para el plano rojo y dos posibles colores para el plano azul. La imagen quedaría así

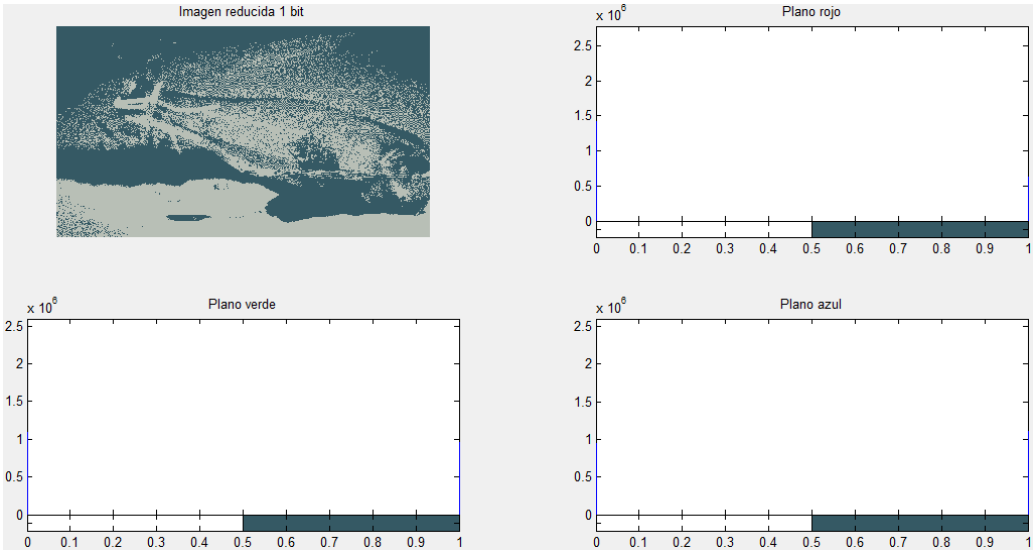


Figura 8.9. Imagen con histogramas reducida a 1 bit

Ambos códigos pueden ser encontrados en la sección de anexos así como resultado de la aplicación básica, la cual también podemos encontrar en anexos la primer guía de laboratorio titulada *"Introducción a imágenes digitales y escala de grises".*

8.2 Desarrollo guía avanzada de imágenes

La aplicación que seleccionamos para la guía avanzada de imágenes consiste en la implementación de filtros al estilo de la popular red social Instagram, en la cual podemos aplicarle filtros a las fotos o imágenes.

Lo primero que debemos saber es que un filtro es dispositivo que discrimina una determinada frecuencia o gama de frecuencias de una señal eléctrica que pasa a través de él, pudiendo modificar tanto su amplitud como su fase [3]. En una imagen las operaciones de filtrado se llevan a cabo directamente sobre los píxeles de esta. En este proceso se relaciona, para todos y cada uno de los puntos de la imagen, un conjunto de píxeles próximos al píxel objetivo con la finalidad de obtener una información útil, dependiente del tipo de filtro aplicado, que permita actuar sobre el píxel concreto en que se está llevando a cabo el proceso de filtrado para, de este modo, obtener mejoras sobre la imagen y datos que podrían ser utilizados en futuras acciones o procesos de trabajo sobre ella.

Los filtros suelen dividirse en lineales y no lineales. Los filtros lineales son aquellos basados en kernels y en mascaras de convolución y para estos existen diferentes tipos como el filtro pasa bajo o de suavizamiento, filtro pasa alto o de atenuación, realce de bordes por desplazamiento, realce de bordes mediante Laplace, realce de bordes con gradiente direccional y detección de bordes y filtros con contorno.

Suavizado de imágenes

Las operaciones de suavizado se utilizan para disminuir los efectos negativos que se pueden presentar en una imagen digital como consecuencia de un sistema de muestreo pobre o del canal de transmisión. Por ejemplo ruido. Promedio de vecinos Dada una imagen f(x, y) de tamaño NxN, el valor del nivel de gris de la imagen suavizada g(x, y) en el punto (x, y) se obtiene promediando los valores de nivel de gris de los puntos de f contenidos en una cierta vecindad de (x, y).

$$\boldsymbol{g(x,y) = \frac{1}{M} \sum_{(n,m)\in S} f(n,m)} \qquad (8.1)$$

donde x, y = 0, 1, . . . , N − 1. S es el conjunto de coordenadas de los puntos vecinos a (x, y), incluyendo el propio (x, y), y M es el número de puntos de la vecindad. Por ejemplo, imaginemos la subimagen y la máscara siguientes:

		⋮		
	$(x-1,y-1)$	$(x,y-1)$	$(x+1,y-1)$	
...	$(x-1,y)$	(x,y)	$(x+1,y)$	...
	$(x-1,y+1)$	$(x,y+1)$	$(x+1,y+1)$	
		⋮		

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Figura 8.10 Subimagen y mascara de una imagen en escala de grises

Y que queremos reemplazar el valor de f(x, y) por el promedio de los puntos en una región de tamaño 3x3 centrada en (x, y), es decir, queremos asignar el valor promedio a f(x, y):

g(x, y) = 1 9 {f(x − 1, y − 1) + f(x − 1, y) + f(x + 1, y − 1) + f(x−, y)+

+f(x, y) + f(x + 1, y) + f(x − 1, y − 1) + f(x, y + 1) + f(x + 1, y + 1)} *(8.2)*

Esta operación se puede realizar de forma general centrando la máscara en (x, y) y multiplicando cada punto debajo de la máscara por el correspondiente coeficiente de la máscara y sumando el resultado.

Contrastado de una imagen

Las técnicas de contrastación son útiles principalmente para resaltar los bordes en una imagen. A continuación se presentan métodos de contrastación en el dominio espacial y en el dominio de la frecuencia. Contrastación por diferenciación Ya hemos visto en la sección anterior que el promedio de puntos en una vecindad tiende a suavizar o difuminar (emborronar) los detalles de una imagen. Como el proceso de promediar es análogo a la integración, es natural esperar que la diferenciación tendrá el efecto opuesto y por lo tanto contrastará la imagen. El método más usado en la diferenciación de imágenes es el gradiente. Se define el gradiente de una imagen f(x, y) en el punto (x, y) como el vector de dos dimensiones:

$$\mathbf{G}\big(\mathbf{f}(\mathbf{x},\mathbf{y})\big) = \begin{pmatrix}\mathbf{Gx}\\ \mathbf{Gy}\end{pmatrix} = \begin{pmatrix}\frac{\partial f}{\partial x}\\ \frac{\partial f}{\partial y}\end{pmatrix} \tag{8.3}$$

Una importante propiedad es que el vector gradiente G apunta en la dirección de máximo cambio de f en el punto (x, y). Para la detección de bordes sólo nos interesa la magnitud, que llamaremos simplemente gradiente y que denotaremos por G(f(x, y))

$$\mathbf{G}\big(\mathbf{f}(\mathbf{x},\mathbf{y})\big) = \sqrt{\boldsymbol{G}_{\boldsymbol{x}}^{\mathbf{2}} + \boldsymbol{G}_{\boldsymbol{y}}^{\mathbf{2}}} \tag{8.4}$$

En una imagen digital las derivadas son aproximadas por diferencias. Una de las aproximaciones que se suele hacer es

$$G(f(x, y)) \text{ ' } |f(x, y) - f(x + 1, y)| + |f(x, y) - f(x, y + 1)| \tag{8.5}$$

Otra aproximación es el Operador de Roberts definido como

$$G(f(x, y)) \approx |f(x, y) - f(x + 1, y + 1)| + |f(x + 1, y) - f(x, y + 1)| \qquad (8.6)$$

En todas las aproximaciones el valor del gradiente es proporcional a la diferencia en los niveles de gris de puntos adyacentes. Por lo tanto, el gradiente tomará valores altos en los bordes de objetos con niveles de gris considerablemente diferentes, mientras que tomará valores bajos en aquellas zonas en los que no existen cambios bruscos de intensidad, y tomará el valor 0 en aquellas regiones en donde el nivel de gris es constante.

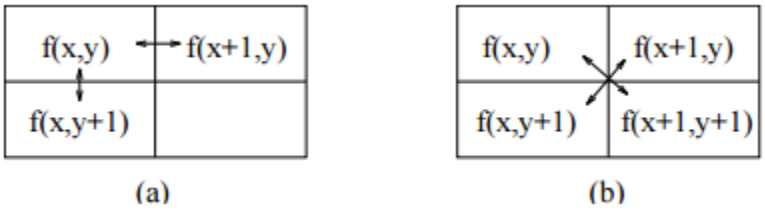


Figura 8.11. Procedimientos para obtener el gradiente de una imagen

Vamos a modificar la Transformada de Fourier de la imagen multiplicando por una cierta función H que hará atenuar las bajas o altas frecuencias según nos interese. Dada la imagen original, f(x, y), con Transformada de Fourier, F, vamos a aplicar la transformación:

$$G(u, v) = H(u, v) * F(u, v) \qquad (8.7)$$

Los filtros pasa-baja atenúan las componentes de medias-bajas frecuencias y dejan intactas las bajas en función de la frecuencia de corte que se elija. Se usan para eliminar ruido de alta frecuencia, o eliminar todo lo que no sean variaciones suaves de nivel de gris. Los filtros pasa-alta atenúan las componentes de baja frecuencia y dejan intactas las de medias-altas en función de la frecuencia de corte que se elija. Se usan para quedarnos con las propiedades de la imagen en los que los niveles de gris varían bruscamente, por bordes de la imagen.

Análisis del desarrollo de la guía

Al realizar esta aplicación dentro de una GUI buscábamos que la manipulación de los filtros y los efectos utilizados fuera sencilla ya que esto posee herramientas como los slider y un menú que nos ayudan a realizar los cambios y notar estos cambios de manera sencilla, sin embargo, lo importante en esta aplicación es la interpretación del código para los estudiantes ya que, en él es donde deberán realizar todos los cambios pertinentes para cumplir con las tareas asignadas en las guía.

Las tareas a realizar son sencillas tomando en cuenta que los estudiantes no poseen conocimientos avanzados en Matlab®, sin embargo, esto les ayudara a comprender mejor tanto sobre programación como el fenómeno físico del procesamiento digital de imágenes de una manera en la que se sientan motivados a estudiarlo. En la aplicación tomamos algunos filtros y los implementamos en Matlab. Podemos observar algunos efectos conocidos

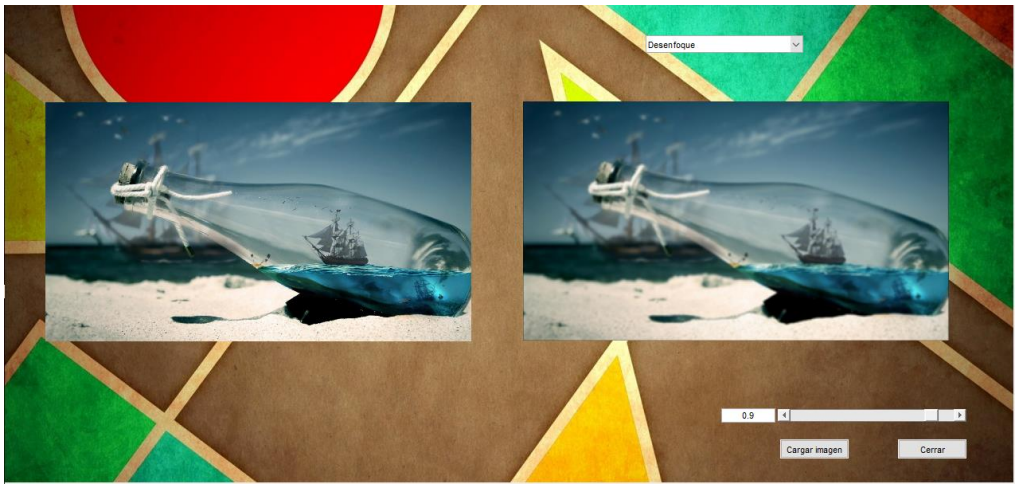


Figura 8.12. Filtro desenfoque

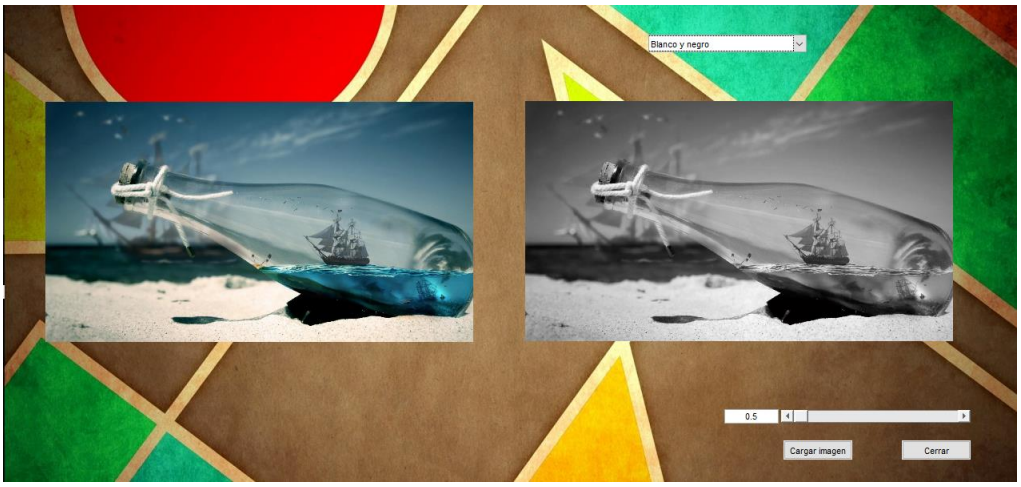


Figura 8.13. Filtro blanco y negro

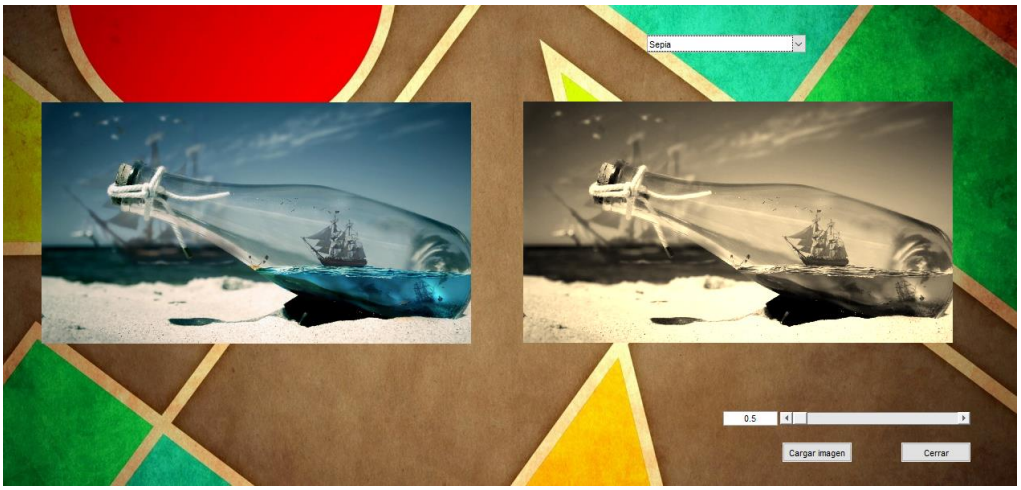


Figura 8.14. Filtro sepia

Como podemos observar en las imágenes, en la parte superior podemos ver el nombre del filtro que tienen en una barra donde podemos seleccionar el filtro que queramos y posteriormente del lado izquierdo podemos ver la imagen original y al lado la imagen con el filtro para apreciar la diferencia, luego debajo de esto tenemos un slider que podemos variar dependiendo del filtro para desenfocar o esclarecer la imagen y abajo tenemos un botón de cerrar y un botón de cargar imagen donde podemos seleccionar la imagen que queramos.

En las imágenes pusimos el filtro de desenfoque que es solo la aplicación de un filtro pasa bajos, luego tenemos el filtro en blanco y negro que se le aplica comúnmente a las fotos y el sepia el cual obtuvimos de los valores por defecto que tiene definidos Microsoft para la distribución de cada color en él, dándole a las fotos una apariencia a una foto vieja o envejecida y estos son solo algunos de los filtros que hemos desarrollado para esta aplicación y en cada una de ellos se piden tareas diferentes a realizar.

Todo este conglomerado de filtros les mostrará a los estudiantes como los filtros transforman las señales desde el punto de vista de colores así como la nitidez de la imagen en algunos casos y además de esto la intención de esto es motivar a los estudiantes a que no crean que por ser algo popular y una idea innovadora sea imposible de realizar, tratamos de que los estudiantes sean capaces de comprender a la perfección el funcionamiento de los filtros de colores en las imágenes así como crear combinaciones de estos y modificarlos a su gusto.

Los resultados de la selección de esta aplicación contribuyo a realización de la guía de laboratorio numero 2 titulada *"filtros de Instagram"* la cual podemos encontrar en la sección de anexos.

8.3 Desarrollo guía básica de videos

En este documento ya hemos hablado un poco sobre lo que es video, tanto analógico como digital, por lo que hay que hablar un poco más sobre las aplicaciones que esto puede tener.

Para hablar sobre la detección de objetos en un video primero debemos tener claros que un video es el procesamiento, almacenamiento, transmisión de imágenes y reconstrucción por medios electrónicos digitales o analógicos de una secuencia de imágenes que representan escenas en movimiento [7]. Por lo tanto, haremos referencia a la segmentación de imágenes a partir de ahora como parte de un video ya que esto aplica a la cantidad por segundo de imágenes que son tomadas en un video.

El primer paso para ello lo constituye la segmentación de imágenes que se ocupa de descomponer una imagen en sus partes constituyentes, es decir, los objetos de interés y el fondo, basándose en ciertas características locales que nos permiten distinguir un objeto del fondo y objetos entre sí.

La mayoría de las imágenes están constituidas por regiones o zonas que tienen características homogéneas (nivel de gris, textura, momentos, etc.). Generalmente estas regiones corresponden a objetos de la imagen. La segmentación de una imagen consiste en la división o partición de la imagen en varias zonas o regiones homogéneas y disjuntas a partir de su contorno, su conectividad, o en términos de un conjunto de características de los píxeles de la imagen que permitan discriminar unas regiones de otras. Los tonos de gris, la textura, los momentos, la magnitud del gradiente, la dirección de los bordes, las modas de los tonos de gris en ventanas 3x3, 7x7 y 15x15, etc., son características a utilizar para la segmentación [7]. Distinguiremos entre segmentación completa, cuando las regiones disjuntas corresponden directamente a objetos de la imagen y segmentación parcial, cuando las regiones no se corresponden directamente con objetos de la imagen. Para conseguir la segmentación completa se necesita un nivel superior de conocimiento que utiliza un conocimiento específico del dominio de la escena. Este conocimiento de nivel superior puede ser, por ejemplo, que los objetos de la imagen corresponden a caracteres numéricos o letras de un alfabeto.

La operación de segmentación trata de distinguir si un píxel pertenece, o no, a un objeto de interés y, por lo tanto, produce una imagen binaria. Todavía no hay una teoría unificada de la segmentación de imágenes, solamente disponemos de un conjunto de algoritmos. Los algoritmos de segmentación de imágenes monocromáticas se basan en alguna de las tres propiedades siguientes:

a) Discontinuidad en los tonos de gris de los píxeles de un entorno, que permite detectar puntos aislados, líneas y aristas (bordes).
b) Similaridad en los tonos de gris de los píxeles de un entorno, que permite construir regiones por división y fusión, por crecimiento o por umbralización.

c) Conectividad de los píxeles desempeña un papel importante en la segmentación de imágenes. Recordemos que una región D se dice conexa o conectada si para cada par de píxeles de la región existe un camino formado por píxeles de D que los conecta. Un camino de píxeles es una secuencia de píxeles adyacentes (que pertenecen a su entorno inmediato).

Los métodos de segmentación se pueden agrupar en cuatro clases diferentes:

a) Métodos basados en píxeles, que a su vez pueden ser: - locales (basadas en las propiedades de los píxeles y su entorno) - globales (basadas en la información global obtenida, por ejemplo, con el histograma de la imagen).
b) Métodos basados en bordes.
c) Métodos basados en regiones, que utilizan las nociones de homogeneidad y proximidad geométrica, como las técnicas de crecimiento, fusión o división.
d) Métodos basados en modelos. Antes de pasar a estudiar cada uno de estos modelos vamos a ver técnicas para la detección de puntos, rectas, bordes y contornos, como herramientas previas.

Detección de puntos

Un punto aislado de una imagen tiene un tono de gris que difiere significativamente de los tonos de gris de sus píxeles vecinos, es decir, de los ocho píxeles de su entorno 3×3. Una máscara para detectar un punto aislado es la siguiente:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Figura 8.15. Mascara para detección de puntos

Aplicando esta máscara al píxel (i,j) obtenemos:

g(i,j) = -f(i-1,j+1)-f(i-1,j)-f(i+1,j+1)- f(i-1,j)+8f(i,j)-f(i+1,j)-f(i-1,j-1)-f(i,j-1)-f(i+1,j-1) (8.8)

Diremos que el píxel (i, j) es un punto aislado si

$$|g(i, j)| > T \qquad (8.9)$$

Donde T es el valor umbral fijado por el decisor. Dicho valor depende de la aplicación que estemos realizando. Sin embargo, esta máscara puede detectar, como puntos aislados, píxeles que forman parte de un borde. Por ello, es más conveniente utilizar el filtro no lineal siguiente:

$$R(i, j) = \min_{\substack{(r,s)\in N_8(i,j) \\ (r,s)\neq(i,j)}} \left| f(r, s) - f(i, j) \right| \qquad (8.10)$$

Diremos que el píxel (i,j) es un punto aislado si

$$|R(i,j)| > T \qquad (8.11)$$

Detección de rostros

La detección de caras consiste en determinar si en una imagen arbitraria hay alguna cara y, en caso se haberla, en qué posición se encuentra [13]. Un detector de caras debería ser capaz de encontrar todas las caras de una imagen. Un detector de caras debe devolver para cada cara detectada en la imagen la posición y tamaño de una caja de inclusión en la que se encuentren los ojos, la nariz y la boca. Además sería deseable que nos devolviera otros datos como el ángulo de rotación de la cara y un score que nos indique la fiabilidad de esa detección. Existen varios algoritmos para la detección de caras. Cada uno de estos algoritmos está basado alguna técnica conocida de reconocimiento de formas como redes neuronales, vecino más próximo, etc. Cada detector tiene características que lo hacen diferente al resto. Por ejemplo, el detector de Viola-Jones que se ha implementado en este proyecto es un detector muy rápido pero

no es capaz de calcular la orientación de la cara ni darnos una estimación de lo seguro que está de la detección.

Medidas del error de un detector A la hora de evaluar un detector de caras debemos fijarnos en 3 medidas fundamentales. Tasa de detección: Es el porcentaje de caras correctamente detectadas respecto del total de caras de la imagen. Tasa de falsos positivos: Indica el número de regiones que se han marcada como caras donde realmente no hay una cara [13].

Tasa de falsos negativos: Indica el número de caras que no han sido correctamente detectadas, es decir, que han sido clasificadas negativamente [13]. Es importante utilizar los 2 últimos conjuntamente para no inducir a error. Por ejemplo un detector que encontrara caras en todos los pixeles de la imagen tendría una tasa de falsos negativos de 0,0 pero en cambio su tasa de falsos positivos seria cercana a 1,0. En una misma imagen solo puede unas pocas caras, en cambio puede haber miles de regiones. Está muy desequilibrado el número de caras y no caras por tanto necesitamos que la tasa de falsos positivos sea varios órdenes de magnitud en comparación con la tasa de falsos negativos [13].

8.1.1 Características tipo Haar

Una de las ideas fundamentales del detector de caras propuesto por Viola y Jones eran las características tipo Haar. Según los autores están basadas en las funciones base de Haar presentadas por Papageorgiou et al. [14]. Estas características tipo Haar se definen sobre regiones rectangulares de una imagen en escala de grises. Una característica está formado por un número finito de rectángulos y su valor escalar consistirá la sumar de los pixeles de cada rectángulo sumados aplicando un cierto factor de peso. La fórmula para calcular el valor de una característica es

$$Característica = X\ 1 \leq i \leq N\ wi \qquad (8.12)$$

Suma rectángulo (ri) donde {r1, . . . , rN } son los rectángulos que forman la característica y wi el peso de cada uno.

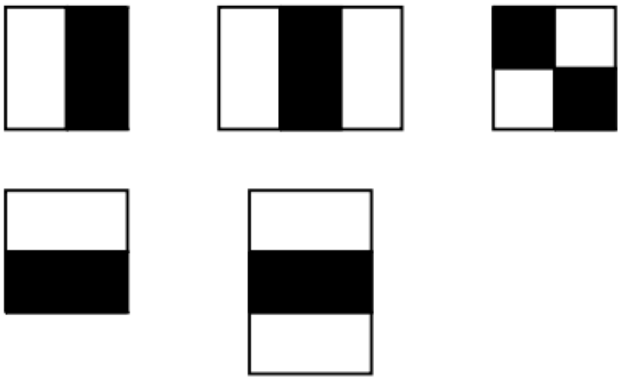


Figura 8.16. Características tipo Haar

Las características tipo Haar de 2, 3 y 4 rectángulos tal y como fueron definidas por Viola y Jones. Debajo se muestran las características rotadas Los autores limitaron el espacio de características permitiendo solo 2, 3 o 4 rectángulos. Además, estos rectángulos deben ser adyacentes y todos del mismo tamaño. Con estas limitaciones, las características se dividen en 3 tipos: las de bordes (2 rectángulos), la de líneas (3 rect.) y las de forma de X (4 rect.). Los pesos también se limitan a los valores 1 y −1. En la figura 8.16 vemos un ejemplo de cada uno de los 3 tipos de características. El valor de una característica se obtiene sumando todos los pixeles del rectángulo blanco y restándole todos los pixeles del rectángulo negro. Por ejemplo, la característica central de 3 rectángulos trataría de representar que en general la región de los ojos es más oscura que las regiones de alrededor. Cada característica (excepto la de 4 rectángulos) se puede rotar 90 grados para obtener nuevas características, también se puede invertir el peso de cada uno rectángulos. El número de características diferentes que se pueden generar para una imagen de 21x21 pixeles es de más de 90000. En su artículo, Viola y Jones solo emplean características cuadradas por lo que en una ventana de 24x24 consideran un total de 45396 características distintas.

Como hemos visto, se pueden definir más de 90000 características diferentes en una ventana de 21x21 pixeles modificando el tamaño de las características tipo Haar y su polaridad. Sumar uno a uno los pixeles de cada uno de los diferentes

rectángulos tiene un coste computacional proporcional al área del rectángulo pero los autores propusieron un método para calcular los valores en tiempo constante a partir de lo que ellos llamaron la imagen integral. La imagen integral es una representación alternativa para la imagen que se puede calcular de manera muy rápida. Un valor de la imagen integral ii(x, y) será igual al valor del pixel i(x, y) de la imagen sumado a todos los pixeles de la imagen que estén a la izquierda y arriba de la posición (x, y).

$$ii(x, y) = X \sum_{x' \leq x, y' \leq y} i(x', y') \qquad (8.13)$$

La imagen integral se puede calcular iterativamente comenzando por la posición (0, 0). En la figura 8.17 podemos ver una representación de la imagen integral de una fotografía. El pixel de abajo a la derecha contiene la suma de todas las intensidades de los pixeles de la imagen [14].

$$s(x, y) = s(x, y - 1) + i(x, y) \qquad (8.14)$$

$$ii(x, y) = ii(x - 1, y) + s(x, y) \qquad (8.15)$$

Gracias a esta representación podemos calcular la suma de los pixeles de un rectángulo de cualquier tamaño utilizando únicamente 4 accesos a memoria. Para calcular el valor de un rectángulo A cuya esquina superior izquierda está en (x, y) y cuyo tamaño es (sx, sy) únicamente necesitamos acceder al valor de la imagen integral de 4 esquinas. En la figura 8.17 se puede ver el esquema de la siguiente fórmula.

$$v(x, y, sx, sy) = ii(x+sx, y+sy) - ii(x-1, y+sy) - ii(x+sx, y-1) + ii(x-1, y-1) \qquad (8.16)$$

Además, como las características tipo Haar consisten en 2, 3 o 4 rectángulos contiguos se pueden evitar accesos a memoria calculando el valor de todos los rectángulos conjuntamente. La única desventaja de la imagen integral es que ocupa hasta 4 veces más memoria que la imagen original.

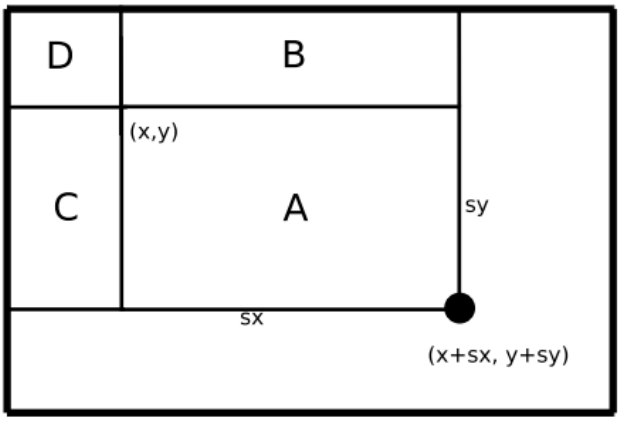


Figura 8.17 Imagen integral de una fotografía

Al ser una suma de los pixeles de la imagen, no puede ser definida como matriz de bytes tal y cómo se hace con las imágenes así que debemos utilizar una matriz de enteros que en la mayor parte de sistemas ocupan 4 bytes.

8.1.2 Características tipo Haar en la detección de caras

Una vez definidas las características de tipo Haar y ver que tenemos un mecanismo eficiente para calcular su valor pasamos a comprobar cómo se pueden aplicar a la detección de caras. Si comparamos las figuras 8.16 y 8.18 vemos cómo las características adquieren un cierto significado. Podemos utilizarlas para modelar las diferencias de intensidad de los pixeles de las distintas zonas de la cara. Viola y Jones comprobaron que no hace falta calcular el valor de todas las características para una misma imagen si no que con una pequeña parte ya podemos discernir con poco error si una región es o no una cara. En la figura 8.18 vemos los 3 tipos de características aplicadas sobre una cara. La primera característica trata de representar que la región de los ojos y cejas es más oscura que la región de debajo [14].

Figura 8.18. Características aplicadas en un rostro

Viola y Jones propusieron entrenar su detector de caras con un algoritmo de boosting que se encargara de seleccionar de entre las miles de posibles características las que mejor separaban el espacio entre las caras y no caras. Un algoritmo de boosting es aquél que se basa en la unión de varios clasificadores sencillos (en nuestro caso basados las características tipo Haar) para crear un clasificador que tiene una menor tasa de error que los clasificadores individuales que lo forman. El algoritmo de boosting utilizado por Viola y Jones es el Adaboost que pasamos a explicar a continuación [13].

Análisis de la guía

El desarrollo de la aplicación de video se desarrollaron dos códigos, uno con el cual los estudiantes puedan desarrollar sus habilidades en Matlab® y aprender un poco sobre cómo funciona el procesamiento digital de video mediante ejemplos básicos de detección de objetos según su color, además de cómo utilizar su cámara web y como inicializarla, entre otras cosas. Luego hay una aplicación que decidimos dejar de manera demostrativa la cual funciona con la detección de rostros que creemos esto ayudará a que los estudiantes despierten el interés en esta área ya que hoy en día la detección de rostro es una de las aplicaciones más utilizadas en cámaras alrededor del mundo, utilizando las herramientas que Matlab® nos da para esta tarea ya que esto es un proceso complicado que involucra algoritmos complejos y de inteligencia artificial lo cual no es competente con el desarrollo de las aplicaciones que realizamos por lo que nos limitamos a mostrar cómo se puede realizar estas funciones mediante la utilización del algoritmo de Viola-Jones del cual hablamos anteriormente.

La primera aplicación básica es la detección de objetos por medio de colores, con esto en un video transmitiéndose en tiempo real con la computadora que está capturando imágenes, podemos detectar objetos de un determinado color (verde, rojo o azul) para ser mostrado en forma de stream en nuestra pantalla de la siguiente manera

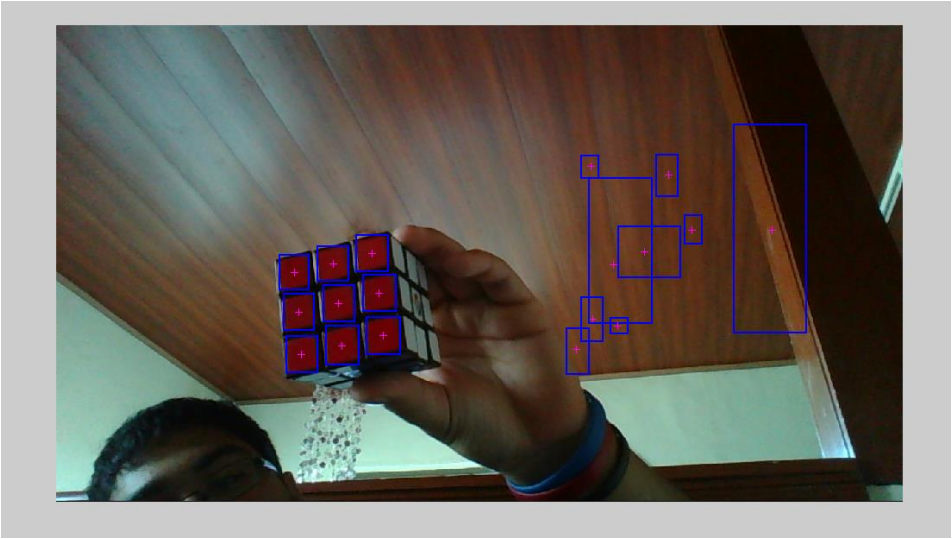


Figura 8.19. Detección del color rojo

El caso de esta aplicación es que se realiza paso por paso en el código lo cual ayuda a un mejor entendimiento por parte de los estudiantes, no solo de las funciones en Matlab® que se utilizan, sino también de que va realizando la aplicación y como lo va haciendo.

Se realizó otra aplicación que va dentro de la misma guía de laboratorio la cual ponemos como una demostración o un complemento, ya que a diferencia de esta primera aplicación, esta posee un comando en Matlab® que realiza todo el trabajo de la detección de rostros, sin embargo, consideramos fundamental que los estudiantes sepan un poco más sobre esto y se interesen en ello ya que esto es el futuro del procesamiento digital de videos, además de agregar elementos de Matlab® sobre las GUI que podrían ser útiles para los estudiantes.

Esta aplicación, como se mencionó anteriormente consta básicamente de detectar una cara o un rostro con una cámara web en tiempo real, esto se realiza en Matlab® mediante el algoritmo de Viola-Jones para la detección de rostros, el cual ya posee una función que realiza esta tarea.

Los resultados de la selección de esta aplicación contribuyo a realización de la guía de laboratorio número 3 con el nombre *"Detección de objetos con cámara web"* la cual podemos encontrar en la sección de anexos.

8.4 Pilotaje de la aplicación de las guías de laboratorio a un grupo de estudiantes.

Estas fueron elaboradas basándonos en ejemplos de diferentes guías y prácticas de laboratorio que fueron facilitadas por nuestro tutor, además de la experiencia que tuvimos como estudiantes realizando durante toda la carrera una incontable cantidad de estas. Las redactamos y realizamos modificaciones a los ejemplos que nos facilitaron basándonos en nuestra experiencia como estudiantes y lo que creemos nosotros podría ser más fácil de entender y de aceptar por los estudiantes, haciendo mención de aplicaciones populares hoy en día y tratando de incentivar un poco ese estado de curiosidad que poseemos los seres humanos y además tratar de hacer más entretenido el aprendizaje.

Luego del desarrollo del software y las guías de laboratorio, invitamos a un grupo de estudiantes de tercer año de la carrera a realizar el experimento de ver si podían realizar el trabajo dentro de estas. Por razones de tiempo, solo 5 estudiantes decidieron ayudarnos con la tarea de realizar las guías para luego, con esa información además de comprobar que hicimos un trabajo entendible, modificar observaciones y escuchar opiniones sobre que se podría modificar y hacer diferente para que sean más provechosas.

Luego de aplicar las guías a estos estudiantes les facilitamos una encuesta para valorar las guías no solo de manera técnica, sino también conocer que tan

importantes o conocer que tanto impacto tuvieron en ellos estas (esta encuesta puede ser encontrada en anexos).

Las guías le ayudaron a comprender mejor los filtros digitales y sus aplicaciones

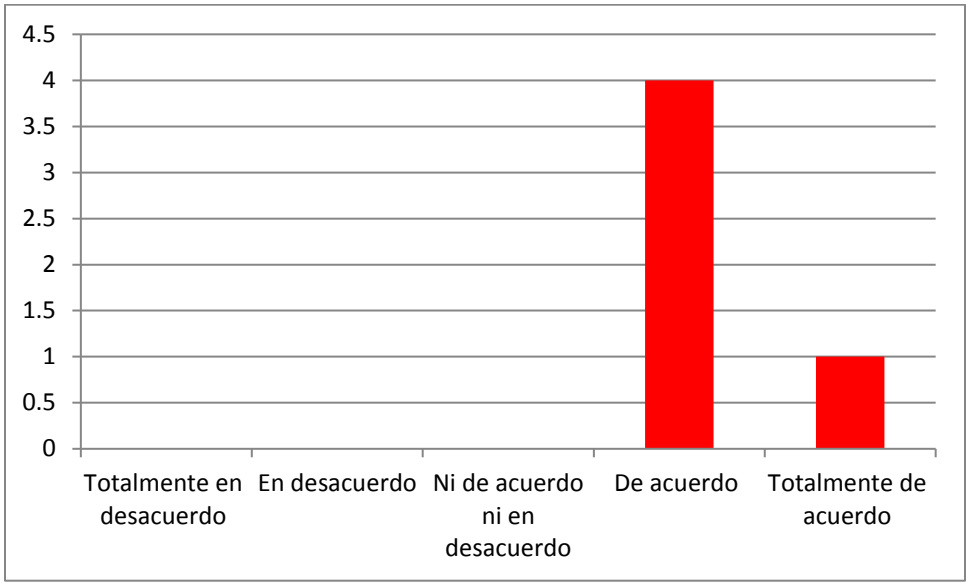


Figura 8.20. Gráfico de pregunta sobre filtros

Como podemos observar el grafico muestra todas las opciones posibles y la cantidad de personas que escogieron la misma respuesta. El grafico nos muestra como información relevante que los estudiantes que realizaron las guías están meramente de acuerdo con que hayan aprendido y comprendido mejor el uso de filtros digitales y las aplicaciones que estos poseen ya que hoy en día prácticamente en cualquier foto o imagen hay un filtro que la modifica ya sea en sus tonalidades de pixeles o le agrega elementos a estas.

Nosotros verificando el plan de clases de la asignatura pudimos observar que se recibe una temática sobre filtros, pero esta es una aplicación más moderna por lo tanto los estudiantes tienen la oportunidad de reafirmar sus conocimientos sobre filtros y ver una aplicación real de estos.

¿Cree usted que las guías de laboratorio cumplen con los objetivos planteados al inicio de estas?

Para esta pregunta tomando en cuenta que las respuestas eran sí o no, obtuvimos un total y rotundo si, por lo que creemos las guías cumplen con los objetivos planteados los cuales podemos leer en la sección de anexos en cada una de estas. Estos objetivos fueron realizados basándonos en ejemplos de guías de laboratorio y a la hora de redactar la guía fue pensando en el cumplimiento de los objetivos.

¿Considera que estas guías le ayudaron en el proceso de enseñanza-aprendizaje en el área de procesamiento digital de señales?

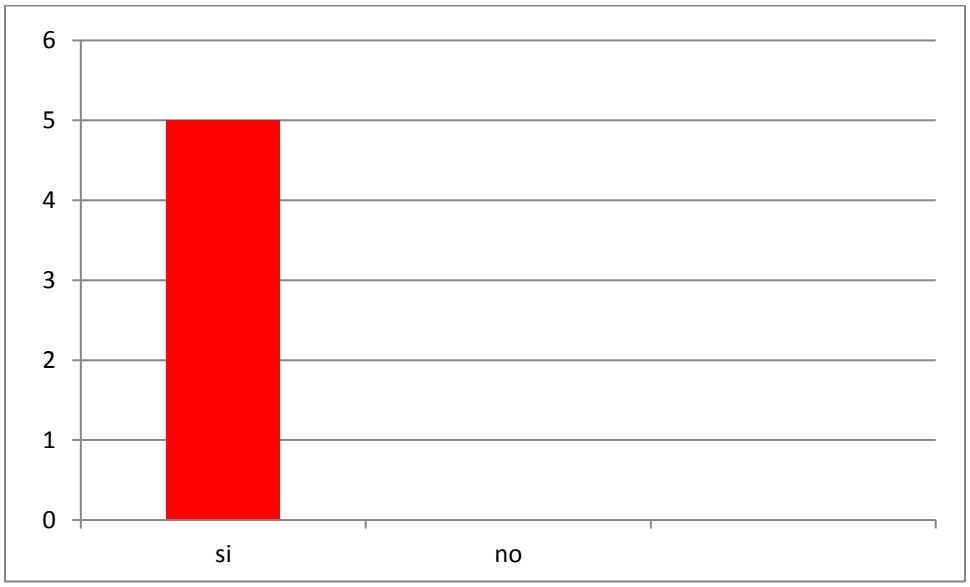


*Figura 8.2*1. Gráfico de pregunta sobre ayuda en DSP

Según la información brindada por los estudiantes que realizaron la encuesta podemos ver que las guías si tienen impacto en el proceso de enseñanza-aprendizaje de los estudiantes, que luego de aplicar todo el proceso junto con las encuestas y en la retroalimentación que nos dieron, nos informaron que en las guías se abarca contenido nuevo y llamativo y que podrían, en caso de ser aplicadas en los cursos de clase, ayudar un poco en la percepción de lo que es la ingeniería electrónica.

9. Conclusión

Con la teoría estudiada y analizada sobre el procesamiento digitales de imágenes y videos, desde la comprensión completa de conceptos como pixeles, formación de imágenes, barrido de un video, etc. desarrollamos aplicando parte de esos conocimientos software para el procesamiento digital de imágenes con la herramienta computacional Matlab® para imágenes, el cual posee un software básico que analiza los niveles de escala de grises de las imágenes y software avanzado en el cual se emplean filtros a imágenes al estilo de Instagram.

Para la parte de video se desarrollaron software que ayudaran a la comprensión básica de un video la cual es bastante complicada, con ayuda de una aplicación en Matlab con la cual se pueden detectar en un video objetos de colores, y otro donde se realiza detección de rostros la cual en la actualidad es una aplicación de esta área muy utilizada, buscando despertar el interés por los estudiantes en estas áreas. Posteriormente al desarrollo del software, se desarrollaron guías basándonos en metodologías y ejemplos empleados en guías anteriores que utilizamos como ejemplos de clases que ya habíamos cursado y ejemplos que el profesor nos facilitó.

Posteriormente, estas guías se le aplicaron a un número limitado de estudiantes para que nos dieran su retroalimentación para realizar unos pequeños ajustes, además de medir el impacto (a través de las encuestas) que estas tuvieron en el proceso de enseñanza-aprendizaje de ellos.

10. Recomendaciones

1. Validar que las guías realmente están aportando aprendizaje valioso y cumpliendo con los objetivos en los estudiantes mediante la aplicación de estas a los grupos de las clases de señales y sistemas tanto de la carrera de ingeniería electrónica como a la carrera de ingeniería en telecomunicaciones del instituto superior de estudios de la UNI.
2. Continuar con el desarrollo de guías para esta área ya que es un área que se desarrolla constantemente, por lo tanto realizar una guía intermedia sobre video digital que realice aplicaciones como: Remover objetos en un video, aplicación de efectos especiales, conversión de formatos y mejora de calidad en video digital.

11. Bibliografía

[1] G. E. Moore, *"Cramming more components onto integrated circuits"* artículo en inglés en la revista Electronics, volumen 38, Abr. 1965.

[2] Proakis, J.G. y D.G. Manolakis. "*Digital Signal Processing: Principles, algorithms and applications"*. Prentice-Hall, Inc. 1996.

[3] Gonzalez & Woods, "Digital Image Processing" Third Edition 2008.

[4] Mohammed Ghanbari, "Standar Codecs : image compression to advanced video coding", IEEE Telecommunications serie 49, Londres, Inglaterra, 2002

[5] Tomas Ang, Manual de fotografía, Editorial Georgiona Garner, ediciones Omega, Londres 2002

[6] Riwes Cruz Jesús Hernández, Compresión de imágenes utilizando la transformada Coseno en un DSP, Tesis México D.F, 2004.

[7] Oge marques, "Practical image and video processing using Matlab", Florida Atlatic university, Florida EEUU, 2011

[8] Pajares Gonzalo De la Cruz Jesús, "Visión por computador (imágenes digitales y aplicaciones)", Grupo Editorial Alfaomega, México D.F, 2004

[9] William B. Pennebaker, Joan L. Mitchell, Chapman & Hall, "JPEG STILL DATA COMPRESIÓN STANDARD", Digital Multimedia Standards, NuevaYork, N.Y, 1993

[10] Haris Papasaika-Hanusch. "*Digital image processing using matlab"* [online] Suiza: Institute of Geodesy and photogrammetry, ETH Zurich. Disponible en: https://sites.google.com/a/ci2s.com.ar/wiki/ci/wavelets/image-processing

[11] Mohammed Ebrahim Al – Mualla, C. Nishan Canagarajah and David R.Bull "Video Coding for Mobile Communications, Efficiency Complexity and Resillence", Academic Press, San Diego Cal, 2001

[12] Grob, Herdon, "Televisión práctica y sistemas de video", Grupo editorial Alfaomega 6a Edición, México D.F, 1992

[13] Philippe Valentin Giffard, "Viola-Jones Object Detection Framework" Publicado por tort, 2011

[14] Paul Viola and Michael Jones, “Fast and robust classification using asymmetric adaboost and a detector cascade”. In Advances in Neural Information Processing System 14, pages 1311–1318. MIT Press, 2002.

12. ANEXOS

12.1 GUÍA DE LABORATORIO 1

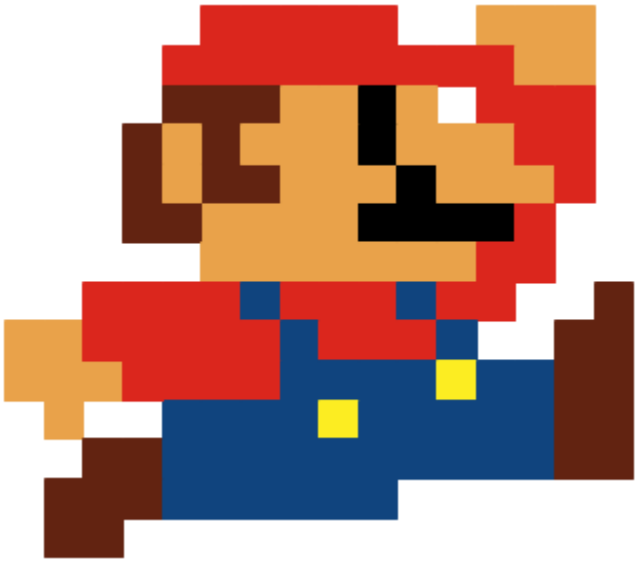
Universidad Nacional de Ingeniería

Departamento de sistemas digitales y telecomunicaciones

Señales y sistemas

Guía #1 Procesamiento digital de imágenes y videos

Introducción a imágenes digitales y escala de grises



I. Objetivos

- Utilizar MATLAB como herramienta de procesamiento digital de imágenes

Específicos

- Aprender a modificar imágenes utilizando MATLAB
- Comprender cómo se forman las imágenes en escala de grises y a color

- Adquirir conocimientos básicos sobre procesamiento digital de imágenes

Requerimientos mínimos

- Computadora
- MATLAB versión 2012a en adelante

II. Introducción a las imágenes digitales

La imagen digital es la representación bidimensional de una imagen empleando bits, unidad mínima de información compuesta por dígitos binarios (1 y 0), que se emplea a instancias de la informática y cualquier dispositivo de tipo digital. Estas imágenes pueden guardarse en dispositivos de almacenamiento masivo en formatos como JPEG, GIF, BMP, PNG, etc., también se pueden modificar mediante software como veremos en la realización de la guía.

Pueden clasificarse en dos grupos:

- Imágenes de mapa de bits que son formadas por píxeles.
- Imágenes vectoriales que son formadas por objetos geométricos independientes (segmentos, polígonos, arcos, etc.)

En el transcurso de la guía vamos a centrarnos en las imágenes de mapa de bits ya que son las captadas por nuestras cámaras y teléfonos celulares.

III. Trabajo previo

1) ¿Qué es un píxel?
2) Busque ejemplos de imágenes de mapa de bits y vectoriales
3) ¿Qué elementos influyen en la calidad de una imagen?

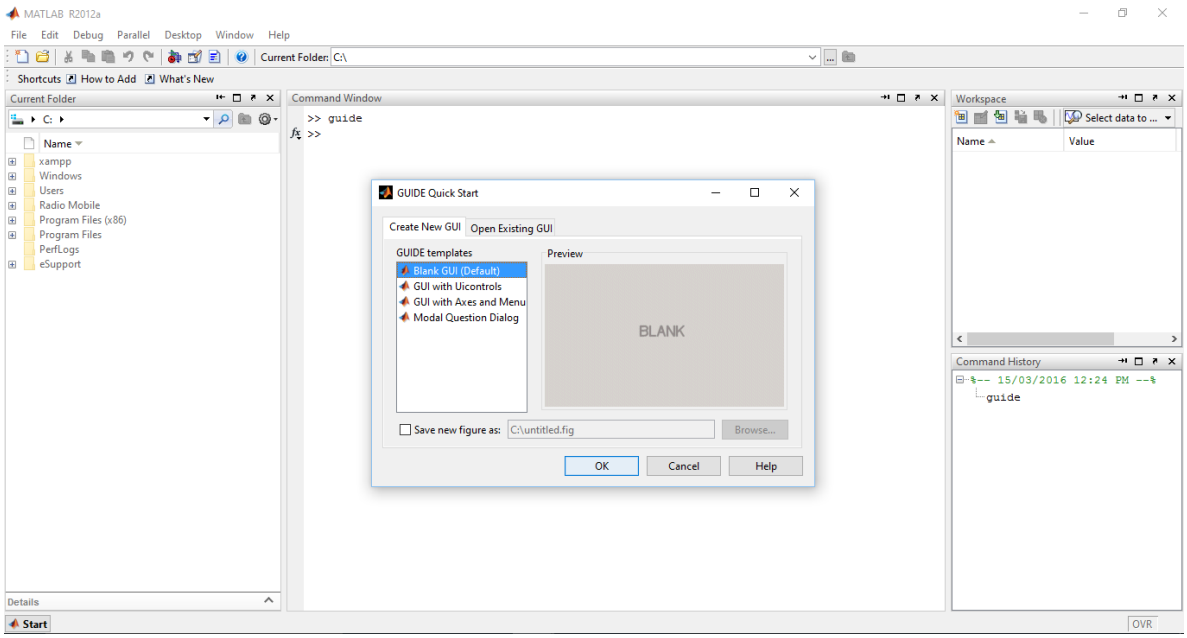
4) ¿Cuál es la cantidad de bits estándar en una imagen y su equivalencia en niveles de gris?

5) ¿Qué relación existe entre RGB y CMYK?

6) ¿Puedo conseguir una imagen en blanco y negro a partir de una imagen a color? ¿Y el proceso inverso?

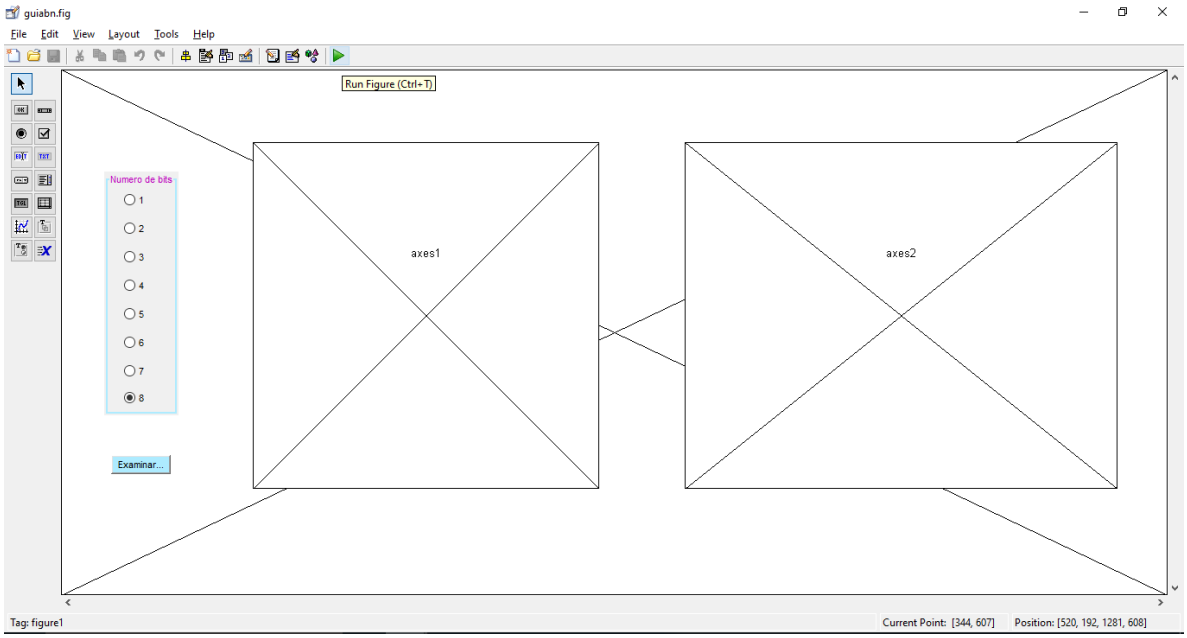
IV. Desarrollo

Para el desarrollo de la siguiente guía de laboratorio abriremos nuestro MATLAB y a continuación utilizaremos el comando "*guide*" el cual nos permite crear una interfaz gráfica o abrirla en nuestro caso. MATLAB nos mostrara una pantalla como la siguiente



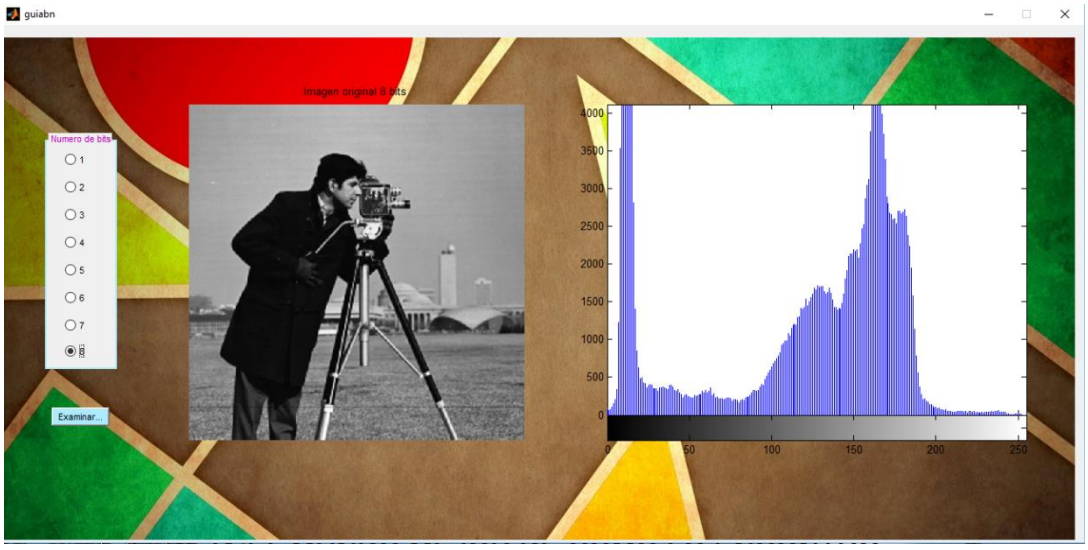
Con esa ventana en la opción de abrir una GUI existente, hacemos click en la opción buscar y abrimos el archivo guiabn.fig que el docente les facilitará antes de la práctica de laboratorio.

Se abrirá una GUI que fue diseñada para modificar una variable que afecta la calidad de las imágenes como es la cantidad de niveles de grises. Para poder realizar la prueba del programa se debe correr desde la interfaz gráfica.



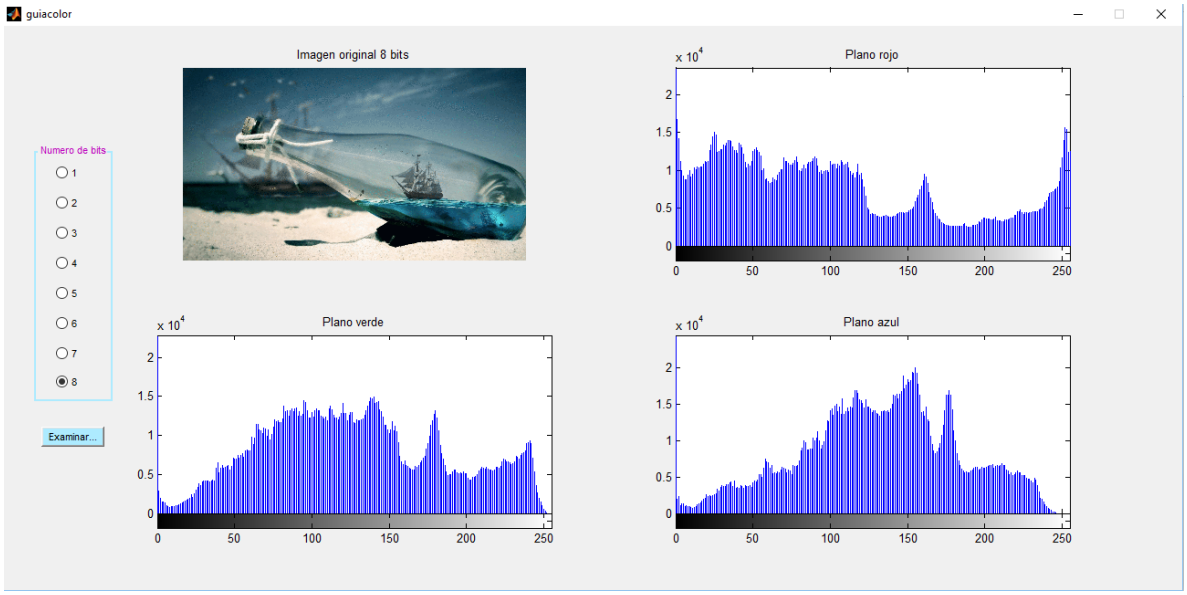
Nos aparecerá una ventana en la cual podremos tocar el botón examinar para cargar una imagen en blanco y negro. Al hacerlo aparecerá la imagen junto a su histograma.

1) Empiece a disminuir el número de bits desde 7 hasta llegar a 1.



2) Reporte sus resultados: ¿En qué momento notó que la imagen empezaba a disminuir su calidad? ¿Qué cantidad de bits considera la mínimamente aceptable para conservar los detalles de la imagen? ¿Qué utilidad tendría reducir una imagen a 1 bit de escala de grises?

Ahora vamos a realizar el mismo procedimiento con el archivo guiacolor.fig facilitado por el docente y abriremos una imagen a color como la siguiente



1) Empiece a disminuir el número de bits desde 7 hasta llegar a 1.
2) Reporte sus resultados: ¿En qué momento comenzó a ver diferencias? ¿Es más brusco el cambio que con imágenes en blanco y negro? ¿Por qué se necesitan tres planos para representar una imagen a color? ¿Cuántos bits tiene realmente una imagen a color?

12.1GUÍA DE LABORATORIO 2

Universidad Nacional de Ingeniería

Departamento de sistemas digitales y telecomunicaciones

Señales y sistemas

Guía #2 Procesamiento digital de imágenes y videos

Filtros de Instagram

I. Objetivos

- Entender cómo funcionan los filtros en imágenes
- Aplicar conocimientos sobre filtros en imágenes
- Presentar de una forma visual el funcionamiento de los filtros

Requerimientos mínimos

- Computadora
- MATLAB versión 2012a en adelante

II. Introducción a los filtros de imágenes

Los filtros de imágenes son operaciones que se aplican a los píxeles de una imagen digital para optimizarla, enfatizar cierta información o conseguir un efecto especial en ella. Este proceso puede llevarse a cabo sobre los dominios de frecuencia y/o espacio.

Los principales objetivos que se persiguen con la aplicación de filtros son:

- Suavizar la imagen.
- Reducir las variaciones entre píxeles vecinos.
- Eliminar ruido: aquellos píxeles cuyo nivel es muy diferente al de sus vecinos y que se pudieron haber originado en la transmisión o recepción de la imagen.
- Realzar bordes.
- Detectar bordes: detectar los píxeles donde se produce un cambio brusco de intensidad.

III. Trabajo previo

7) ¿Cuáles son los tipos de filtros que existen?

8) ¿Qué ventajas conlleva el uso de filtros en imágenes?

9) ¿Qué consecuencias podrían derivar del uso excesivo de filtros en una imagen?

10) ¿Qué impacto ha tenido el uso de filtros en la actualidad?

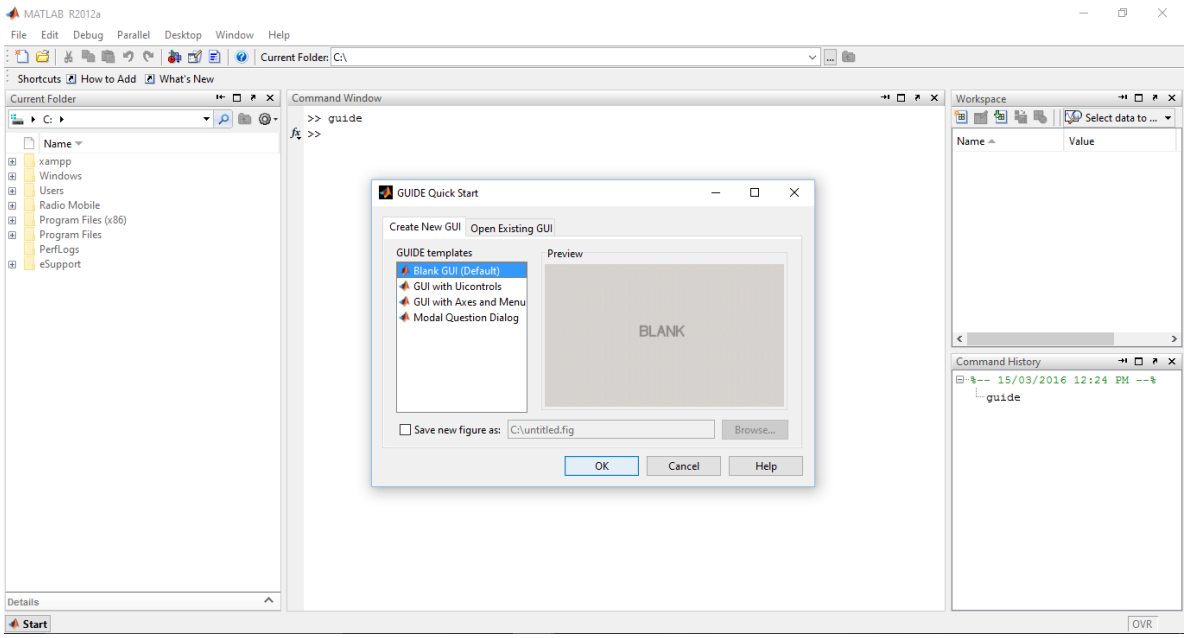
11) ¿Cómo podemos detectar objetos haciendo uso de filtros y en qué áreas nos sería de utilidad?

12) Investigue el ruido aditivo gaussiano.

13) Investigue sobre los valores de intensidad en una imagen.

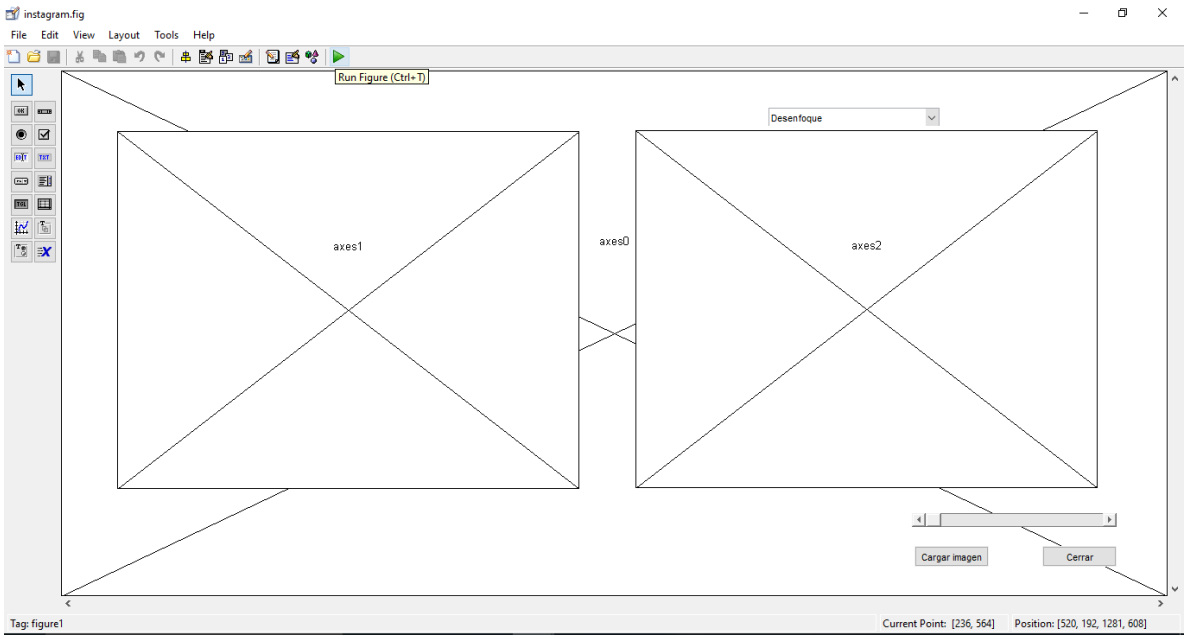
IV. Desarrollo

Para el desarrollo de la siguiente guía de laboratorio abriremos nuestro MATLAB y a continuación utilizaremos el comando "*guide*" el cual nos permite crear una interfaz gráfica o abrirla (nuestro caso). MATLAB nos mostrará una pantalla como la siguiente



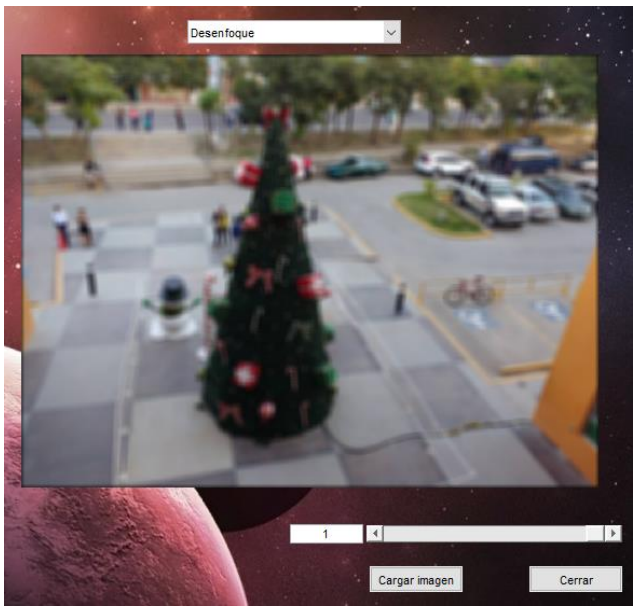
Con esa ventana en la opción de abrir una GUI existente, hacemos click en la opción buscar y abrimos el archivo instagram.fig que el docente les facilitará antes de la práctica de laboratorio.

Se abrirá una GUI que fue diseñada para pasar una imagen por distintos tipos de filtros y observar los resultados en tiempo real. Para poder realizar la prueba del programa se debe correr desde la interfaz gráfica.



Ahora cargaremos una imagen a color y procederemos a observar de qué manera inciden los filtros sobre la misma.

1) Seleccionamos en el menú “Desenfoque” luego de haber cargado la imagen



Como podemos observar en el código se utiliza un filtro paso bajo gaussiano que sigue la siguiente formula `h = fspecial('gaussian', hsize, sigma)`, esta convolución se utiliza para darle un suavizado a las imágenes. Lo que hace básicamente es que toma el valor de cada píxel y lo promedia con el de sus vecinos más cercanos, esto se logra con sigma que es la desviación típica, cuanto mayor sea sigma mayor será el suavizado porque se toman en cuenta los píxeles más lejanos.

```
82 -        case 'Desenfoque'
83 -
84 -            handles.slider_value=round((handles.slider_value)*12+1);
85 -            h = fspecial('gaussian',handles.slider_value,handles.slider_value);
86 -            handles.Image=imfilter(handles.a,h,'conv');
87 -            imshow(handles.Image,'Parent',handles.axes2);
```

La imagen anterior tiene un sigma de 13, disminuya y aumente el valor que aparece sombreado en el código. Ejecútelo y mire las diferencias

¿Existe un valor máximo? ¿Con que valor de sigma se realiza un suavizado de la imagen que sirva para retocar la imagen pero sin distorsionarla demasiado?

2) Seleccionamos en el menú el filtro “Negativo”

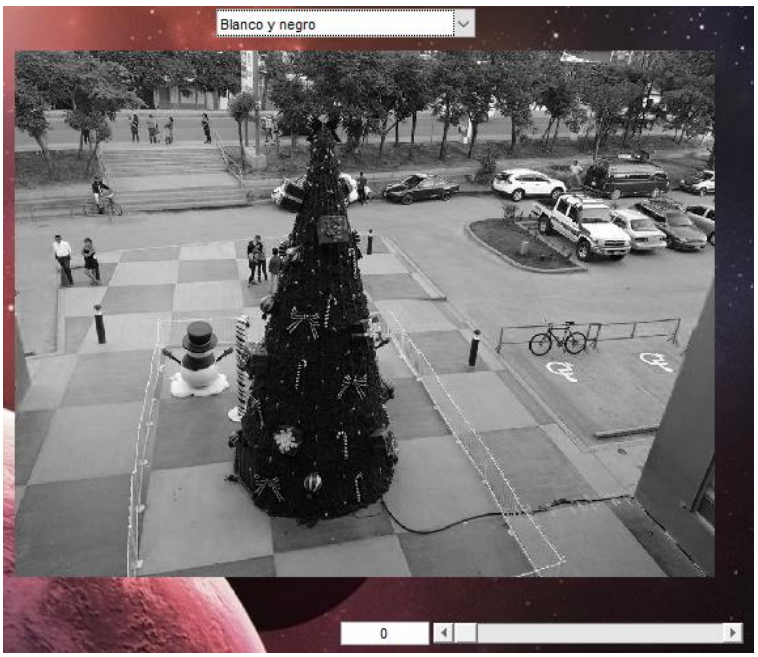
Para comprender el funcionamiento de este filtro necesitamos saber que una imagen está formada por píxeles y cada píxel toma un valor entre 0 y 255. El negativo es un efecto que se logra al restarle a cada píxel 255. Por ejemplo si tenemos un píxel con valor 255 (blanco) y le restamos 255 nos da 0(negro) el cual es el color inverso.

```
89 -        case 'Negativo'
90
91 -            handles.slider_value=round((1-handles.slider_value)*255);
92 -            imshow(handles.slider_value-handles.a,'Parent',handles.axes2);
```

¿Qué pasa con las partes de la imagen que tienen píxeles de color negro? ¿Puede tomar valores negativos o que pasa en estos casos?

Mueva el slider, diga ¿qué ocurre y a qué se debe esto? Luego diríjase al código y cambie el signo - por un signo +. Anote los resultados y explique el por qué se da esto.

3) Seleccionamos en el menú el filtro “Blanco y negro”



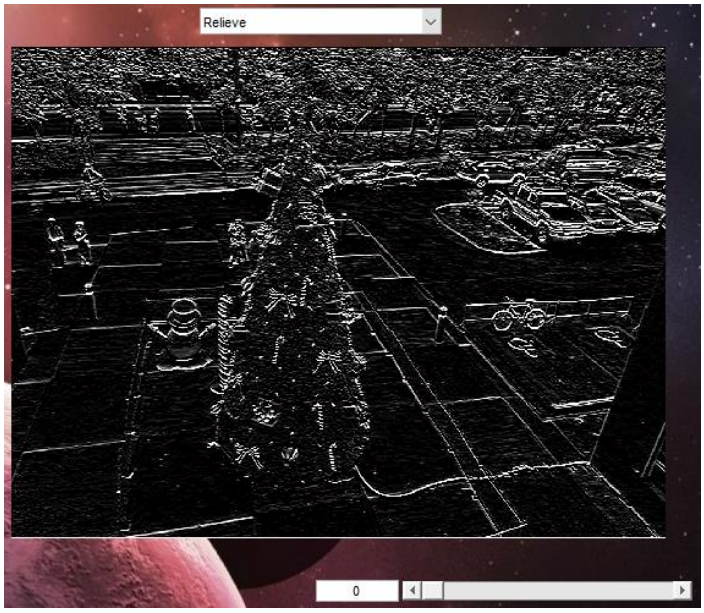
Este filtro lo que hace es dividir la imagen en sus tres planos RGB y luego lo que hace es tomar el valor de cada plano y multiplicarle el valor que le vayamos dando al slider.

```
94 -        case 'Blanco y negro'
95
96 -            planorojo = handles.a(:, :, 1);
97 -            planoverde = handles.a(:, :, 2);
98 -            planoazul = handles.a(:, :, 3);
99
100 -           grayImage = handles.slider_value*double(planorojo) + ...
101                         handles.slider_value*double(planoverde) + ...
102                         handles.slider_value*double(planoazul);
103 -           grayImage = uint8(grayImage);
104 -           imshow(grayImage,'Parent',handles.axes2);
```

Como podrá observar cuando seleccione este filtro, aparecerá automáticamente la versión en blanco y negro de la imagen. Ahora lo que tiene que hacer es comenzar a mover el slider hasta que obtenga una imagen lo más parecida a esta.

Diga ¿En qué valor del slider esto ocurre? ¿La imagen obtenida es exactamente igual a la que apareció cuando seleccionó el filtro? ¿A qué se debe esto? Explique por qué al momento de hacer la sumatoria de los planos la imagen resultante no es a color sino que es en blanco y negro.

4) Seleccionamos en el menú el filtro “Relieve”



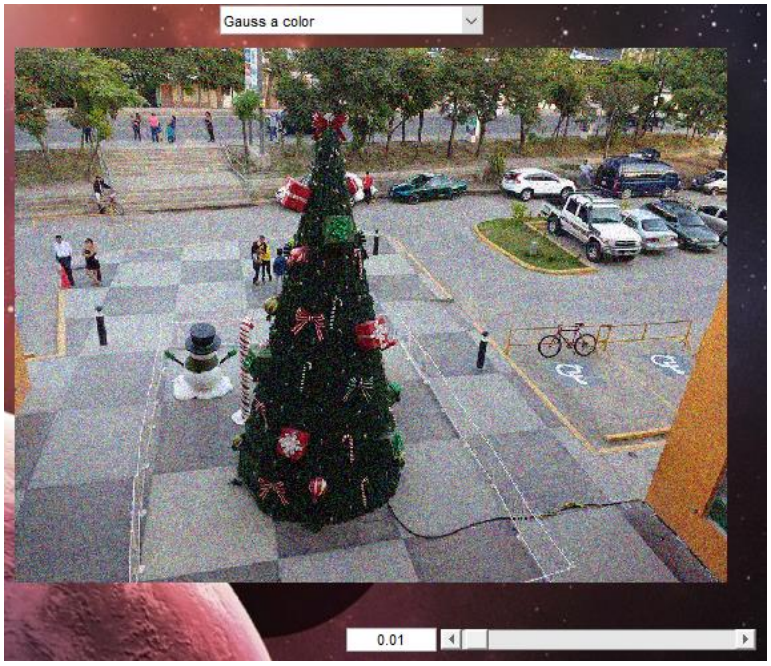
Este filtro hace uso del operador prewitt que calcula el gradiente de intensidad de una imagen en cada píxel, muestra cómo se da el cambio entre cada píxel, si este cambio es abrupto o no y las probabilidades de que este represente un borde en la imagen. Este al igual que el operador sobel es muy utilizado en la detección de bordes.

```
106 -       case 'Relieve'
107 -           sf=fspecial('prewitt');
108 -           sc=sf;
109 -           im=rgb2gray(handles.a);
110 -           b1=imfilter(im,sf);
111 -           b2=imfilter(im,sc);
112 -           relieve=imadd(b1,b2);
113 -           dibujo=imadjust(relieve,[],[handles.slider_value 1-handles.slider_value]);
114 -           imshow(relieve,'Parent',handles.axes2);
```

Ponga el slider en el valor de 0 y el valor de 1. ¿Qué relación tienen? ¿Se utiliza para el mismo propósito?

Con lo aprendido en el primer filtro “Desenfoque”, agréguele un ligero suavizado con el filtro gaussiano antes de pasarlo por el operador prewitt y note las diferencias con lo obtenido sin realizar el suavizado. ¿Vale la pena realizar este pre procesamiento?

5) Seleccionamos en el menú el filtro “Gauss a color”



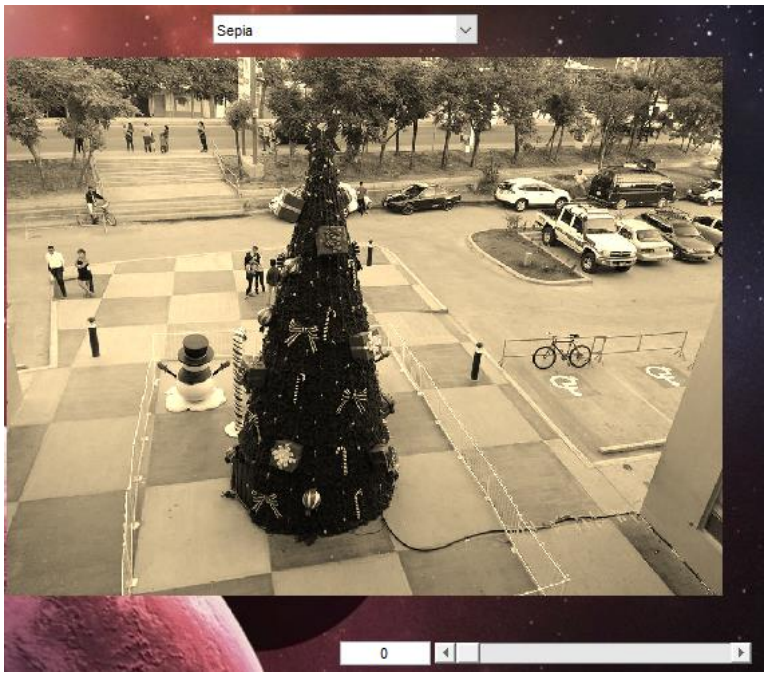
Este filtro lo que hace es agregarle ruido blanco gaussiano a la imagen y lo que hacemos con el slider es aumentar la varianza o sigma cuadrado.

```
117 -       case 'Gauss a color'
118 
119 -          sp =  imnoise(handles.a,'gaussian',handles.slider_value);
120 -          imshow(sp,'Parent',handles.axes2);
```

El ejercicio que vamos a hacer es el siguiente, tome una captura de pantalla de la imagen con varianzas de estos tres valores: 0.01, 0.1 y 0.2

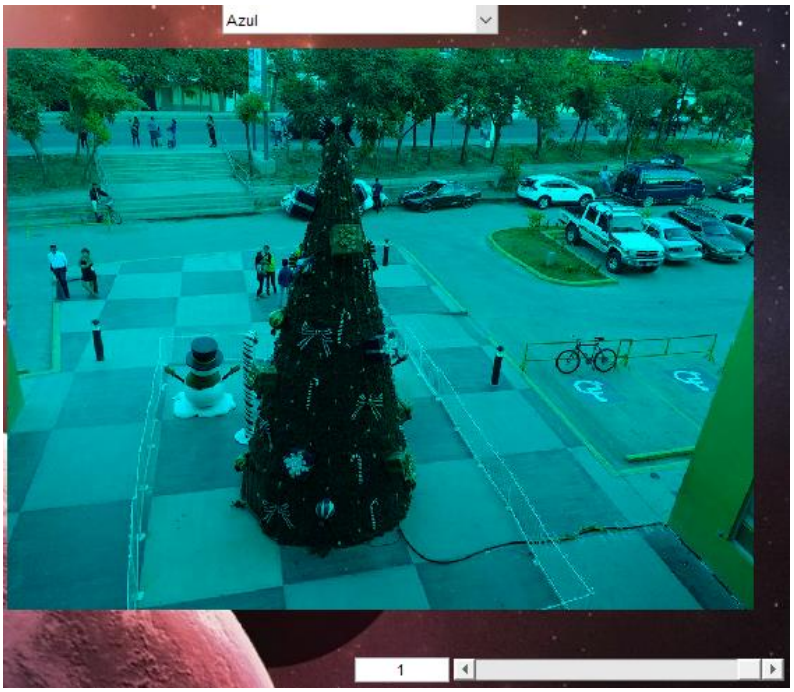
Luego cree un filtro que elimine el ruido de estas imágenes y anexe el script (m-file) en su informe.

6) Seleccionamos en el menú el filtro “Sepia”



El filtro sepia es muy popular, se obtiene un color anaranjado oscuro con una leve saturación. Investigue las coordenadas RGB del color sepia y explique cómo se aplicó esto en la elaboración del código del filtro.

7) Seleccionamos en el menú el filtro “Azul”



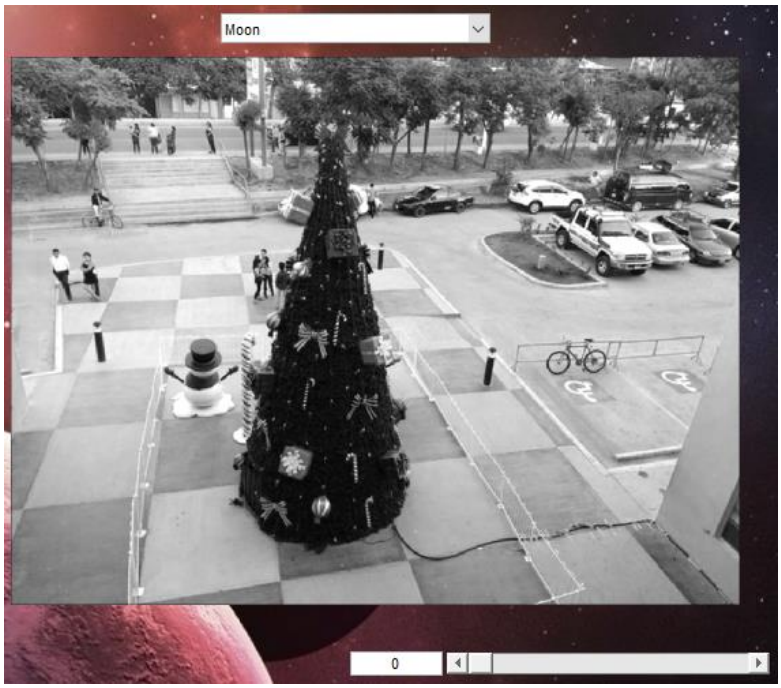
Como se puede apreciar el filtro Azul lo que hace es resaltar las tonalidades azules. Explique cómo es posible esto si el código lo que hace es ir eliminando el color rojo de la imagen.

```
case 'Azul'

    r = handles.a(:, :, 1);
    g = handles.a(:, :, 2);
    b = handles.a(:, :, 3);
    p = cat(3,r*(1-handles.slider_value),g,b);
    imshow(p,'Parent',handles.axes2);
```

Luego edite el código de forma que también se vaya eliminando poco a poco el color verde de la imagen y adjunte sus resultados. Con los conocimientos adquiridos haga lo mismo para resaltar las tonalidades verdes y luego las azules, adjunte el código y las imágenes resultantes.

8) Seleccionamos en el menú el filtro “Moon”



Busque en internet o Instagram el filtro Moon y luego compare los resultados obtenidos.

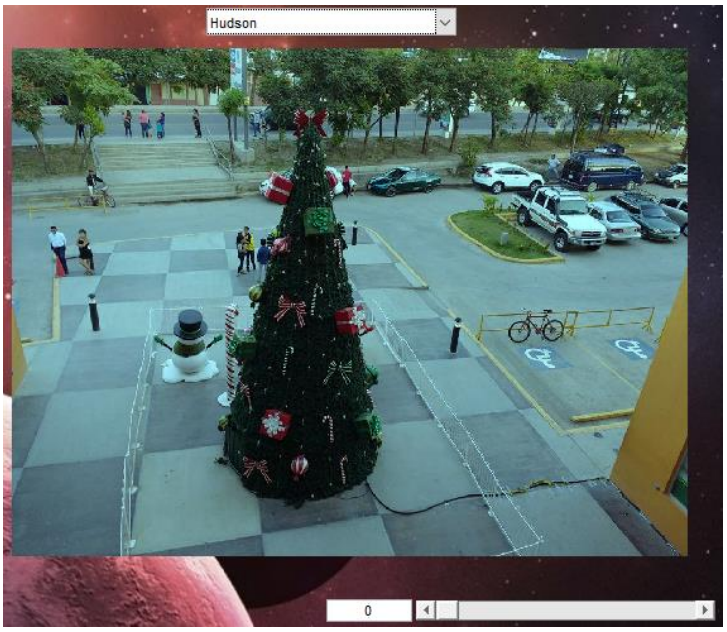
```
case 'Moon'

    r = handles.a(:, :, 1);
    g = handles.a(:, :, 2);
    b = handles.a(:, :, 3);
    % Hace la suma de los planos
    moon = 0.6*double(r) + ...
              (0.2)*double(g) + ...
              (0.2)*double(b);
    moon = uint8(moon);
    fm = imadjust(moon,[0.00; 0.80],[0.00; 1.00], 1.00);
    handles.slider_value=round((handles.slider_value)*12+3);
    fmh= fspecial('gaussian',3,3);
    dsf=imfilter(fm,fmh,'conv');
    imshow(dsf,'Parent',handles.axes2);
```

Para crear el efecto moon, como se puede apreciar en la imagen lo primero que hacemos es convertir la imagen a blanco y negro. Esto se logra separando los planos y al momento de hacer la suma de los mismos, se guarda en una variable de 8 bits porque como ya aprendimos las imágenes en blanco y negro solo toman valores de 0 a 255.

Teniendo la imagen en blanco y negro, procedemos a ajustar la intensidad. Los valores que podemos variar son los sombreados en azul, estos toman valores en un rango de 0 a 1.

9) Seleccionamos en el menú el filtro “Hudson”



Para la reproducción de este filtro lo que se ha hecho es tomar ejemplos de Instagram y tratar de simular en Matlab este efecto

```
case 'Hudson'

    r = handles.a(:, :, 1);
    g = handles.a(:, :, 2);
    b = handles.a(:, :, 3);
    hudson = cat(3,r*0.72,g,b);
    handles.slider_value=round((handles.slider_value)*12+3);
    h = fspecial('gaussian',handles.slider_value,handles.slider_value);
dh=imfilter(hudson,h,'conv');
    imshow(dh,'Parent',handles.axes2);
```

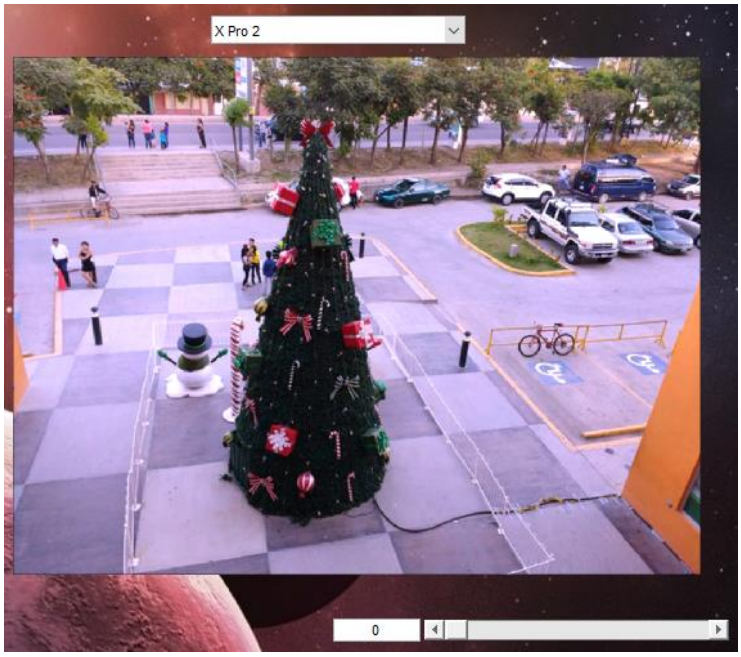
Para ello lo que hacemos es dividir la imagen en sus 3 planos RGB y al momento de volverlos a unir reducimos los valores originales del plano rojo a 72%. Realice lo mismo con los planos azul y verde y muestre los resultados.

10) Con lo aprendido en los filtros Moon y Hudson lo que necesitamos es crear el filtro de Instagram “XPro2”

Se les dará el código a utilizar, solamente necesitan modificar los valores sombreados y buscar que el efecto sea lo más parecido al original.

```
case 'X Pro 2'

      r = handles.a(:, :, 1);
      g = handles.a(:, :, 2);
      b = handles.a(:, :, 3);
      hudson = cat(3,r,g,b);
      r = hudson(:, :, 1);
      g = hudson(:, :, 2);
      b = hudson(:, :, 3);
      xpro = cat(3,r,g,b);
      handles.slider_value=round((handles.slider_value)*12+3);
      bx = imadjust(xpro,[0.00; 1.00],[0.00; 1.00], 1.00);
      h = fspecial('gaussian',handles.slider_value,handles.slider_value);
      dh=imfilter(bx,h,'conv');
      imshow(dh,'Parent',handles.axes2);
```



Al final, si ha realizado los cambios correctos le quedará un resultado similar al de la imagen.

RESULTADOS A REPORTAR

1) Respuesta a cada una de las preguntas de los incisos del desarrollo.
2) Los scripts desarrollados para el laboratorio.

12.1 GUÍA DE LABORATORIO 3

Universidad Nacional de Ingeniería

Departamento de sistemas digitales y telecomunicaciones

Señales y sistemas

Guía #3 procesamiento digital de imágenes y videos

Detección de objetos con cámara web

I. Objetivos

- Utilizar MATLAB como herramienta de procesamiento digital de video

Específicos

- Aprender a utilizar MATLAB para la simulación de video
- Aplicar y reforzar conocimientos previos sobre procesamiento digital de imágenes
- Observar el funcionamiento del video con detección de objetos

Requerimientos mínimos

- Computadora con cámara web
- MATLAB versión 2012a en adelante

II. Introducción a video

Una imagen “fija” es una distribución espacial de intensidad que es constante con respecto al tiempo, por otro lado el vídeo es un patrón de intensidad espacial que cambia con respecto al tiempo, otro significado común para video es el definirlo como un conjunto o secuencia de imágenes con alto grado de relación entre sí, por tanto el video se puede considerar como una secuencia de imágenes fijas.

.En la actualidad existen dos tipos de videos los cuales son el video analógico y el video digital, ambos se basan en el mismo principio pero con funcionamientos diferentes.

Video analógico

En una señal de video analógico la amplitud del voltaje o de la corriente cambia respecto al tiempo como generalmente sucede en una señal de audio, las variaciones en la señal de video corresponden a la información visual, producida por una cámara que convierte la luz en señales eléctricas.

El video tradicionalmente ha sido capturado, almacenado y transmitido en forma analógica, el término señal de video análoga hace referencia a una señal eléctrica unidimensional de tiempo que es obtenida por el muestreo del patrón de intensidad en las coordenadas horizontal, vertical y temporal y convertido en una representación de tipo eléctrica, este proceso de muestreo es conocido como barrido.

El barrido de una imagen comienza desde la esquina superior izquierda y continua horizontalmente, cuando este alcanza el final de la línea regresa al

principio solo que esta vez salta a la siguiente línea para comenzar de nuevo; una vez que alcanza el final de la última línea se ha completado un cuadro y el barrido regresa de nuevo a la esquina superior izquierda para comenzar a trazar el siguiente cuadro, durante el trazo son insertados los pulsos de sincronización y de borrado en la señal.

Aunque parezca que las imágenes reproducidas por una señal de video se mueven, en realidad se muestran una serie sucesiva de imágenes fijas con suficiente rapidez para dar la ilusión de movimiento al espectador.

Líneas por segundo: el número de líneas barridas para una imagen completa debe ser grande, con la finalidad de incluir el mayor número de elementos de imagen y por tanto más detalle.

Cuadros por segundo: como se mencionó el barrido horizontal se realiza en forma lenta hacia abajo, mientras se realiza el barrido horizontal también se efectúa un barrido vertical, este movimiento es necesario para no barrer las líneas una encima de la otra; el barrido horizontal produce líneas de izquierda a derecha mientras que el vertical distribuye las líneas para completar el cuadro de abajo hacia arriba.

Un cuadro está formado por 720 píxeles por línea y posee 525 líneas. Estas 525 líneas se exploran en 1/29.97 segundos, para crear la ilusión de movimiento deben mostrarse las imágenes completas suficientes durante cada segundo, este efecto se logra si se tiene una tasa de repetición de imágenes mayor que 16 por segundo. La tasa de repetición de 24 imágenes por segundo empleada en las películas de cine es suficiente para producir la ilusión de movimiento en la pantalla.

La frecuencia de barrido vertical es de 60 Hz, y es la rapidez del trazo de vídeo en que completa sus ciclos de movimiento vertical de arriba abajo y luego de regreso hacia arriba. El número de líneas de barrido horizontales en un campo es la mitad del total de 525 líneas para un cuadro completo, como un campo contiene líneas alternadas se obtienen 262.5 líneas horizontales para cada campo vertical; y el tiempo para un campo es de 1/60 seg, entonces el número de líneas por segundo es:

262.5 x 60 Hz = 15750 Hz

O bien si se consideran 525 líneas para un par sucesivo de campos se puede multiplicar la rapidez de cuadros de 30 por 525 que da el mismo resultado de 15750 líneas barridas en 1 segundo. Esta frecuencia de 15750 Hz es la velocidad a la cual el trazo de barrido completa sus ciclos de movimiento horizontal de izquierda a derecha y de regreso a la izquierda. El tiempo para cada línea de barrido horizontal es 1 / 15750 segundos, lo cual en términos de microsegundos es:

Tiempo H = 1 / 15750 = 63.5 µseg

El tiempo en µs indica que la señal de video para los elementos de imagen dentro de una línea horizontal puede tener altas frecuencias del orden de MegaHertz, si hubiera más líneas el tiempo de barrido sería menor y se obtendrían frecuencias de video mayores, por lo tanto la mayor frecuencia de video está limitada a cerca de 4.2 MHz

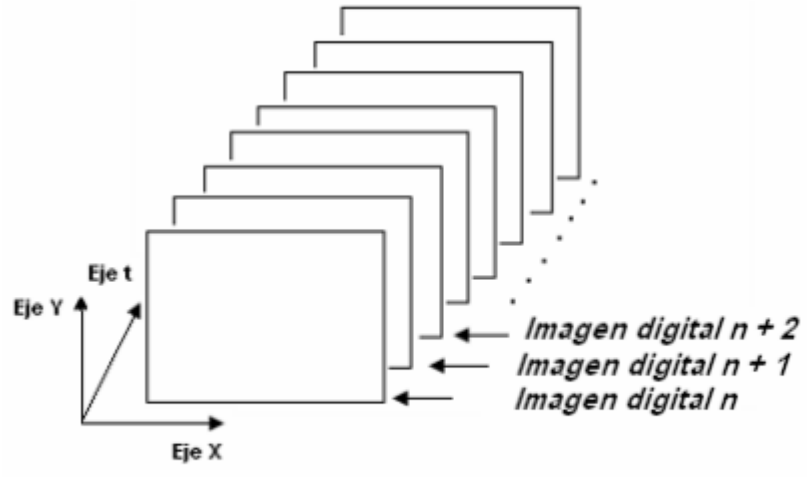
Video digital

El video digital no es más que un medio alternativo de representar la información contenida en una forma de onda de video analógica, utilizando las herramientas que nos proporciona la tecnología digital.

El proceso de digitalización de vídeo involucra tres operaciones básicas: filtrado, muestreo y cuantización; si la frecuencia de muestreo no es el doble que la frecuencia máxima de la señal analógica ocurrirán efectos de aliasing, por lo tanto la operación de filtrado es usada para limitar el ancho de banda de la señal de entrada; la señal analógica filtrada es muestreada en un número específico de veces para generar una señal en tiempo discreto la mínima razón de muestreo es conocida como la razón de Nyquist.

Las muestras resultantes del proceso de muestreo tienen amplitudes continuas, por lo tanto se requerirá precisión infinita para representarlos; la operación de cuantización es usada para mapear tales valores hacía un conjunto finito de amplitudes discretas que pueden ser representados por un número finito de bits. Cada muestra es definida como un elemento de imagen y es organizado en un arreglo bidimensional para formar una imagen digital fija o cuadro digital; por lo tanto el video digital consiste de una secuencia de imágenes digitales fijas (bitmaps).

Para producir imágenes en movimiento es necesario proporcionar un mecanismo donde el valor de cada píxel pueda actualizarse periódicamente y esto da como resultado una matriz tridimensional donde dos de los ejes son espaciales y el tercero es temporal, en ella se aprecia como el video es una sucesión de imágenes bidimensionales que ocurren en instantes de tiempo discreto



Si la señal de video digital se encuentra en color entonces está será representada como en la figura anterior solo que cada imagen tendrá asociada sus respectivos componentes para el sistema de color que utilizen (RGB, YCrCb, etc). En el video digital no existe la necesidad de utilizar pulsos de sincronía y borrado por lo que la computadora conoce exactamente donde empieza una nueva línea, su tamaño y el número de píxeles, por lo tanto estos pulsos son removidos en la conversión A/D. Considerando como ejemplo el estándar Europeo de TV que difiere para el Norteamericano, pero el número de sus muestras por línea (píxeles) es 720 para ambos, dado que en el video digital sólo interesan las partes activas de la imagen y el número de líneas activas por cuadro es de 625, el número total de píxeles por segundo llega a ser igual a 720 x 625 x 25 = 11, 250, 000 píxeles/s.

La razón de bit total es calculada considerando los diferentes componentes de la imagen así como su profundidad de bits, suponiendo componentes RGB y una profundidad de 8 bits se tiene 11, 250, 000 x 3 x 8 = 270, 000, 000 bits/s ó 257.49 Mbits/s lo cual es una razón muy elevada para su ancho de banda, esta es la mayor desventaja del video digital en la actualidad tanto para su almacenamiento como para su transmisión.

III. Trabajo previo

14) ¿Qué es un frame? Explique en qué se diferencia este con un fotograma.

15) Mencione que formatos de cámara web existen. Explique en que consiste cada uno de ellos.

16) De un ejemplo de conversión matricial de YUV a RGB y explique cómo funciona esto.

17) ¿Qué es thresholding (umbralización)? Explique para que funciona en un video.

18) Sobre el algoritmo Viola-Jones sobre el cual se habla más abajo, ¿En qué consisten las características tipo Haar que posee? Explique

IV. Desarrollo

La siguiente guía de laboratorio consta de dos partes, una de ellas será la detección de objetos de determinados colores como rojo, verde y azul (RGB) con ayuda de la cámara web de nuestra computadora.

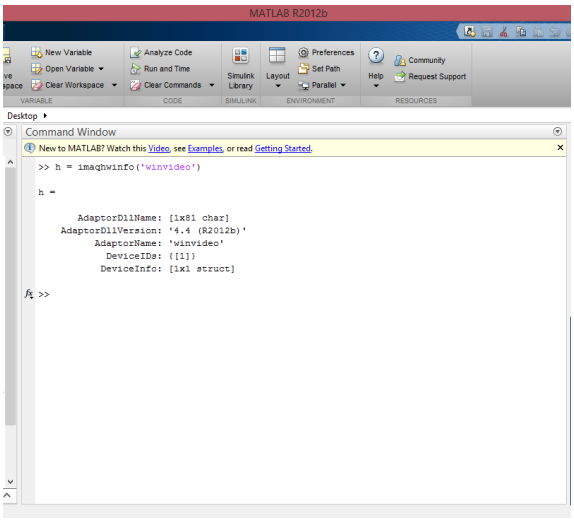
Para esta parte de la guía requeriremos tener cerca objetos de color verde rojo y azul para comprobar que se esté realizando la detección correctamente.

A continuación proseguiremos a abrir el archivo "detectar.m" el cual contiene el código con el que podremos hacer la detección de objetos color rojo.

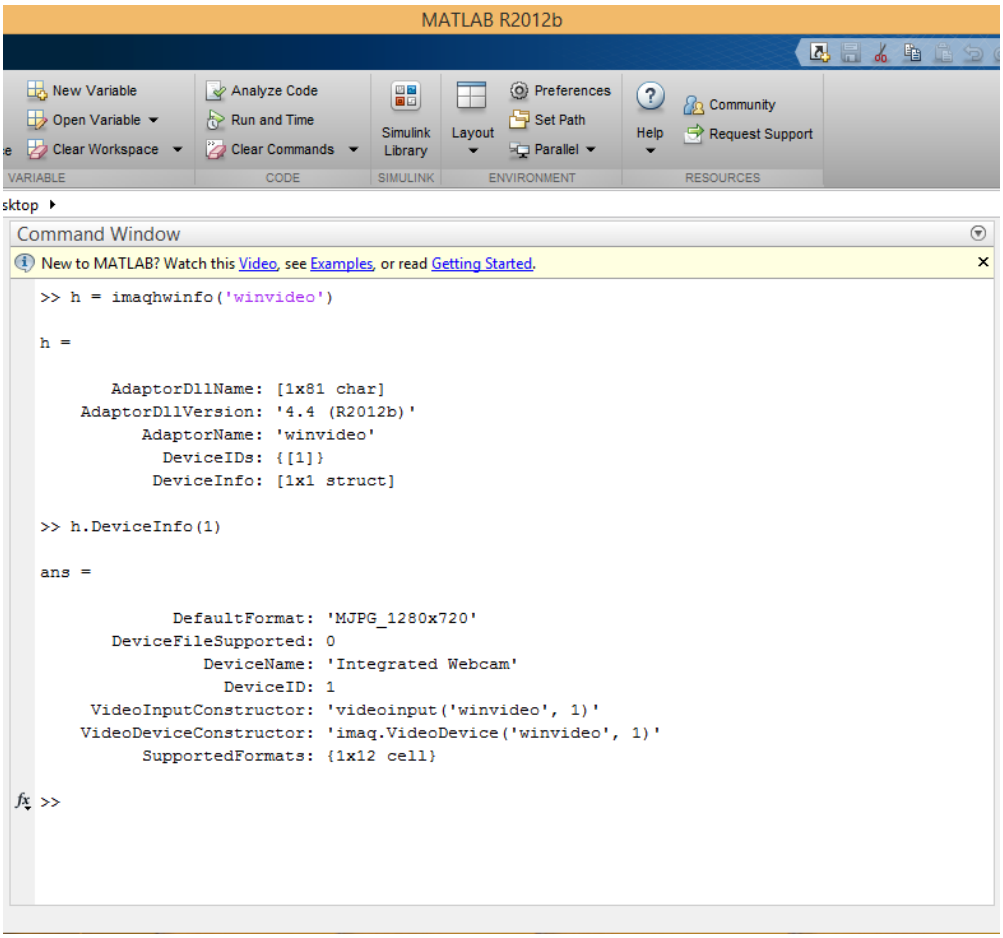
Recomendación importante: Investigar sobre las funciones utilizadas en el código para tener una mejor comprensión de estas

Lo primero a tomar en cuenta, al abrir el archivo, es que hay que configurar el formato de la cámara de nuestra computadora en el código, ya que todas las cámaras poseen un tipo de formato diferente.

Para hacer esto, debemos poner una serie de comandos en matlab los cuales nos darán esta información. En la línea de comandos de matlab proseguiremos a guardar en una variable el comando "*imaqhwinfo('winvideo')*" la cual nos mostrara lo siguiente



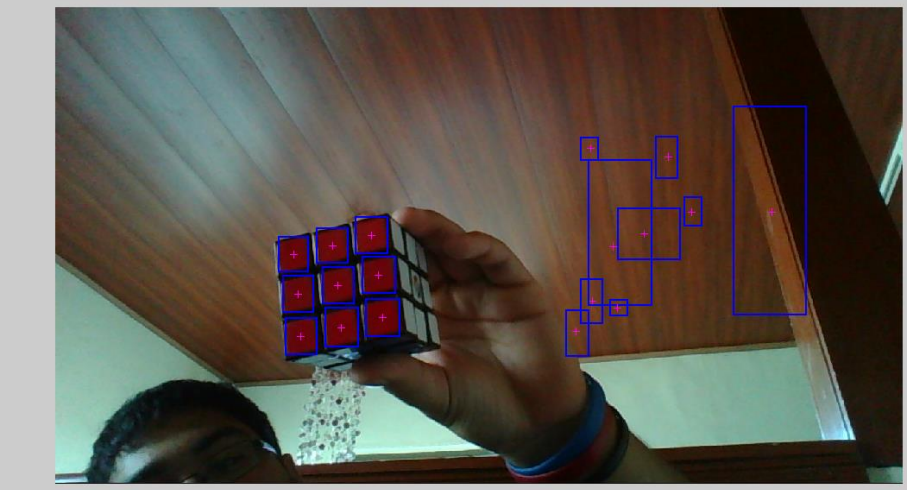
Ahora habiendo hecho eso escribiremos el nombre de nuestra variable seguida de *.DeviceInfo(1)* y ahora si nos mostrara la informacion completa de nuestra camara, se vera algo parecido a lo siguiente



Ahora con el “DefaultFormat” vamos a la línea #3 del código y cambiamos el formato por el que nos haya proporcionado matlab.

```
3 -     vid  = videoinput('winvideo',1,'MJPG_1280X720');
4 -     set(vid,'FramesPerTrigger',inf);
5 -     set(vid,'ReturnedColorspace','rgb');
```

Habiendo realizado estos sencillos pasos ahora procedemos a correr el código el cual es capaz de detectar objetos de color rojo como se muestra en la imagen



NOTA: Se les recomienda estar en espacios con luz a la hora de probar el código.

Habiendo realizado las pruebas pertinentes o que consideren necesarias el siguiente paso es dar solución a las siguientes preguntas acerca de este tema para una mejor comprensión e interpretación de su funcionamiento.

1) Sobre la línea 6 del código tenemos el comando `vid.FrameGrabInterval` que está definido en 5, cambie el valor a 15, 25, 35 y conteste ¿Qué fenómeno observó? ¿A qué se debe eso?

```
5 -     set(vid,'ReturnedColorspace','rgb');
6 -     vid.FrameGrabInterval = 5; %Cantidad de fr
7 -     start(vid); %se inicializa el objeto video
```

2) Luego de regresar el valor de la función a su original, en la línea 10 del código realizaremos un cambio del valor 1 por el valor 2 y observaremos el cambio y luego por el valor 3 y observaremos el cambio. Explique que manipula dentro del código este valor.

```
 9 -         data = getsnapshot(vid);     %obtencion de la im
10 -         diff_im=imsubtract(data(:,:,1),rgb2gray(data));
11 -         diff_im=medfilt2(diff_im,[3,3]); % Aplicacion d
```

Hint: tener objetos de otro color cerca

3) Regresamos al valor por defecto y esta vez sobre la línea 12 tenemos un diferencial de imagen con valor 0.18 el cual su rango varía entre 0 y 1, ahora vamos a modificar con valores cada vez más pequeños de 0.14, 0.10, 0.05. ¿Qué puede observar al variar los valores? Explique según lo que puede observar para que funciona esta parte del código.

```
11 -        diff_im=medfilt2(diff_im,[3,3]);
12 -        diff_im=im2bw(diff_im,0.18);   %co
13 -        diff_im=bwareaopen(diff_im,300);
```

4) Para este ejercicio vamos a borrar o comentar(preferiblemente comentar) una parte del código, desde el hold on hasta el pause

```
17 -        hold on
18 -        for(object = 1:length(stats))
19 -            bb = stats(object).BoundingBox;
20 -            bc = stats(object).Centroid;
21 -            rectangle('Position',bb,'EdgeColor','b','LineWidth',2)
22 -            plot(bc(1),bc(2),'-m+')
23 -        end
24 -        hold off
25 -        pause(1);
```

Luego de comentar todo lo mostrado en la imagen vamos a correr el programa y observaremos que sucede ¿Qué pudo apreciar al realizar esta acción? Explique la función de la condición ***for*** dentro del código.

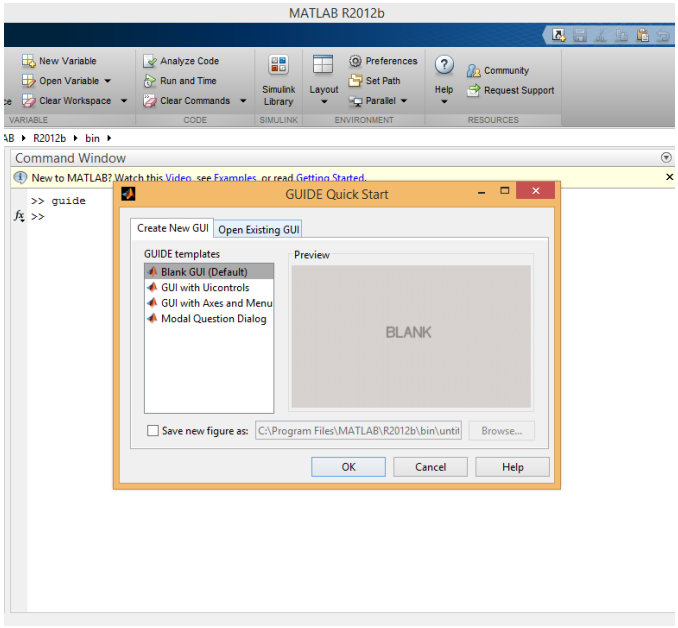
5) Regresamos el código a la normalidad y habiendo realizado el ejercicio anterior podemos notar que la variable stats que fue declarada anteriormente se manda a llamar dentro del ***for***. Determine cuál es la función que cumple esta variable dentro del ***for*** y explique cuál es el objetivo que cumple de esta.

Parte demostrativa (Detección de rostro)

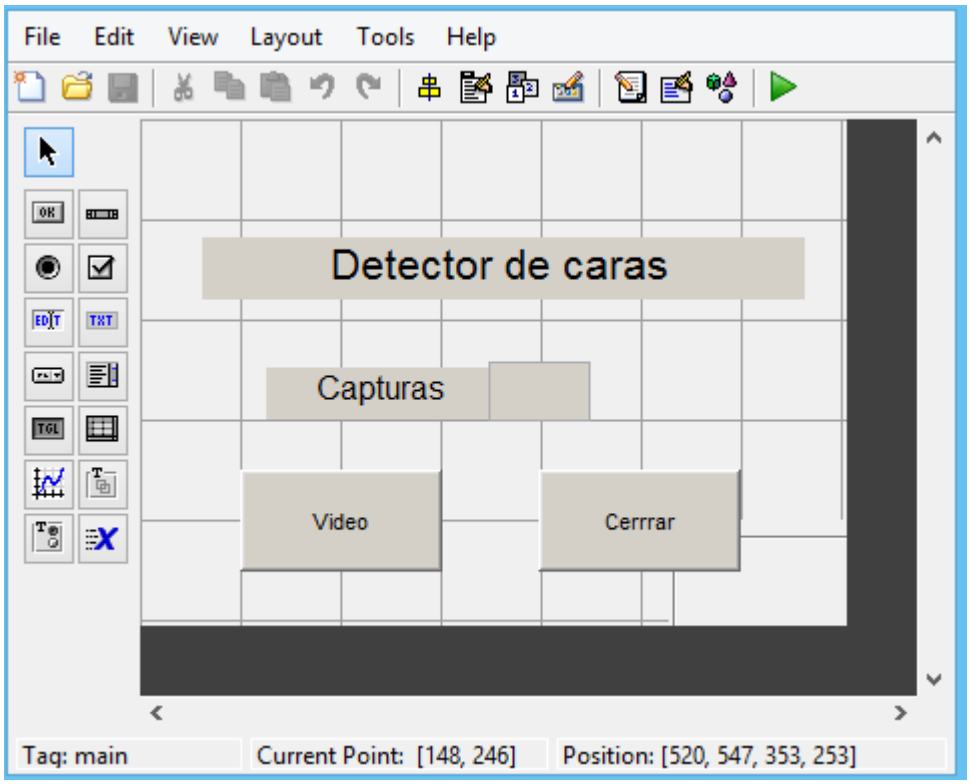
Hoy en día la detección de rosto es una herramienta utilizada en muchos ámbitos de la vida que prácticamente lo vemos a diario por ende debe considerarse como algo importante. Si tomamos ejemplos de esto podemos decir que en la seguridad de los aeropuertos se utilizan patrones de reconocimiento facial para determinar si la persona que está en frente es realmente la que está en el pasaporte o la NSA y la CIA utilizan detección de rostro para encontrar a los terroristas más buscados dentro de los videos que revisan a diario o incluso en lo más práctico y conocido, con la aplicación snapchat la utilización de todos los filtros para ponerse caras divertidas de perro o conejos y tomarse fotos videos y subirlas sacando la lengua, todo eso se realiza mediante la utilización de un algoritmo de detección de rostro el cual le permite a nuestros smartphones reconocer nuestros rasgos faciales y detectar nuestra cara. Con esta información es bastante claro que la detección de rostro hoy en día la encontramos en todos lados y ni siquiera lo notamos.

Para la detección de rostro existen muchos algoritmos capaces de realizar esta tarea, sin embargo el que matlab utiliza por defecto es conocido como viola-jones que también es considerado uno de los mejores algoritmos por su costo computacional tan bajo y su toque de inteligencia artificial por medio de redes neuronales que le permiten reconocer cuando se está captando una cara para realizar la detección o no. En términos generales el viola-jones consta de dos partes principales: clasificador en cascada, que este es capaz de garantizar una discriminación rápida y un entrenador de clasificadores basado en *Adaboost*. Viola Jones tiene una probabilidad de verdaderos positivos del 99,9% y una probabilidad de falso positivos del 3,33%, y a diferencia de otros algoritmos utilizados en métodos de caracteres invariantes procesa sólo la información presente en una imagen en escala de grises. No utiliza directamente la imagen sino que utiliza una representación de la imagen llamada imagen integral. Para determinar si en una imagen se encuentra una cara o no, el algoritmo divide la imagen integral en subregiones de tamaños diferentes y utiliza una serie de clasificadores (clasificadores en cascada), cada una con un conjunto de características visuales. En cada clasificador se determina si la subregión es una cara o no.

Habiendo dicho en términos generales como funciona esto procederemos a matlab y pondremos el comando “*guide*” el cual nos mostrara una ventana para crear una nueva GUI



Tocamos la opción de abrir GUI existente y luego damos click en el botón de buscar y abrimos el archivo *Drosto.fig* y luego nos mostrara una ventaja como esta



Ahora antes de correrlo en la segunda barra de opciones iremos a la opción editor (3 espacios al lado del botón para emular la GUI) la cual nos mostrara parte del código. Estando aquí iremos a la línea 85 y haremos lo

mismo que en el código anterior para modificar el formato de nuestra cámara

```
84      % imagen  la resolucion 640x480, para mejor desempeño
85 -    obj =imaq.VideoDevice('winvideo', 1, 'YUY2_640x480','ROI', [1 1 640 480]);
86 -    set(obj,'ReturnedColorSpace', 'rgb');
```

Pero además tendremos que poner los valores del tamaño de la cámara en la sección del lado

```
84      % imagen  la resolucion 640x480, para mejor desempeño
85 -    obj =imaq.VideoDevice('winvideo', 1, 'YUY2_640x480','ROI', [1 1 640 480]);
86 -    set(obj,'ReturnedColorSpace', 'rgb');
```

Estos son los mismos valores que nos da por defecto la cámara, es decir, si el formato es MJPG_1280x720 entonces deberemos dejar [1 1 1280 720] y así dependiendo de nuestra cámara.

Habiendo realizado estas modificaciones podremos compilar el código sin problemas y podremos observar el trabajo que realiza el programa de detección de rostro tomando en cuenta lo antes mencionado sobre el algoritmo que utiliza el detector de matlab y lo antes mencionado es una aplicación bastante útil y es bueno saber más sobre ella.

RESULTADOS A REPORTAR

1. Respuesta a cada una de las preguntas de los incisos del desarrollo.
2. Los Scripts desarrollado para el laboratorio (si desarrollo alguno).

12.2 ANEXOS 2: ENCUESTA

Nombre:_____

Edad:_____

de carnet:_____

¿Ya curso la asignatura de señales y sistemas? Si su respuesta es no ir a la pregunta 3.

1. Si___
2. No___

¿Qué tanto conocía sobre imágenes, video y su procesamiento?

Mucho_____ Poco_____ Muy Poco_____ Nada_____

Las guías le ayudaron a comprender mejor los filtros digitales y sus aplicaciones

Totalmente en desacuerdo	**En desacuerdo**	**Ni de acuerdo ni en desacuerdo**	**De acuerdo**	**Totalmente de acuerdo**

¿Cree usted que las guías de laboratorio cumplen con los objetivos planteados al inicio de estas?

3. Si___
4. No___

¿Considera que estas guías le ayudaron en el proceso de enseñanza-aprendizaje en el área de procesamiento digital de señales?

1. Si___
2. No___

3. Algunas cosas___

Cree usted que el trabajo previo que se manda a realizar tiene relación con el contenido utilizado en las guías

1. Si___
2. No___
3. Algunas cosas___

¿Considera que el lenguaje utilizado en el desarrollo de las guías es el correcto?

1. Si___
2. No___

¿Cree usted que las guías están redactadas de forma entendible?

1. Si___
2. No___