Universidad del Bío-Bío
Facultad de Ciencias Empresariales
Departamento de Sistemas de Información

Profesor Guía: Pedro Campos Soto
Profesor Co-Guía: Cristhian Aguilera Carrasco

UNIVERSIDAD DEL BÍO-BÍO

Comparación de algoritmos de detección de defectos en paneles de madera utilizando imágenes multiespectrales

Memoria para optar al título de
Ingeniera Civil en Informática

Olivia Andrea Coñuepán Jara

Abril de 2016
Concepción - Chile

Resumen

Este proyecto se presenta para dar conformidad a los requisitos exigidos por la Universidad del Bío-Bío en el proceso de titulación para la carrera de Ingeniería Civil en Informática. El proyecto titulado "Comparación de algoritmos de detección de defectos en paneles de madera utilizando imágenes multiespectrales" abarca distintos métodos de análisis y procesamiento de imágenes digitales con el fin de detectar anomalías en muestras de paneles melamínicos a través de imágenes en distintas bandas espectrales.

La inspección visual constituye una parte importante del control de calidad en la industria. El control de calidad está diseñado para asegurar que los productos defectuosos no se les permita llegar al cliente. Por esta razón esta actividad constituye un elemento esencial de retroalimentación de información para toda la empresa, influyendo en el diseño, planificación de procesos y logística, así como también en la fabricación.

Este trabajo ha sido fuertemente confiado a inspectores humanos lo que conlleva un riesgo relativamente elevado ya que las decisiones tomadas por los operarios se ven afectadas por factores psicológicos como la fatiga o los hábitos adquiridos. Esto ha llevado al desarrollo de equipo automatizado rápido y especializado para facilitar los subjetivos procesos de control de calidad.

Existen herramientas robustas que permiten el control de calidad en la industria de la madera, sin embargo, esta gran capacidad es restringida por falta de adecuación a problemáticas concretas, como es el caso de superficies melaminicas.

El presente trabajo consiste en el estudio de algoritmos para la detección de defectos en paneles de madera a través de imágenes digitales los cuales son validados mediante el desarrollo de una aplicación en lenguaje C# y el uso de técnicas de librerías de OpenCV, estos métodos son encontrados a partir de la revisión de la literatura científica en las áreas de análisis de imágenes, aprendizaje automático y procesamiento de imágenes.

Agradecimientos

La realización de este proyecto fue gracias al financiamiento del Fondo de Fomento al Desarrollo Científico y Tecnológico del gobierno de Chile, en el marco del proyecto denominado "Métodos Multiespectrales para la detección de defectos y control automático de calidad en la industria de la Madera" FONDEF ID14i10364, dirigido por el Dr. Cristhian Aguilera Carrasco docente de la Universidad del Bío-Bío.

Índice General

Índice de Figuras

Índice de Tablas

Glosario de términos y abreviaturas

1. **LBP**: (en inglés, *Local Binary Pattern)*
2. **LBP-Liu:** Variación del algoritmo LBP
3. **SIFT**: (en inglés, *Scale Invariant Feature Transform*)
4. **VI**: Cámara espectro visible
5. **UV**: Cámara ultravioleta
6. **NIR**: Cámara Infrarrojo cercano
7. **LWIR**: Cámara Infrarrojo Lejano
8. **TIFF:** *Tagged Image File Format.* Es un formato de archivo informático para imágenes.
9. **RGB**: Modelo de color Red/Green/Blue, Rojo, Verde, Azul
10. **HSV:** Modelo de color *Hue Saturation Value*
11. **ROI**: Región de interés(en inglés, *Region of Interest*)
12. **EO**: *Energy Orientation,*
13. **Brillo**: Indica si un área está más o menos iluminada
14. **Tono**: Indica si un área parece similar al rojo, amarillo, verde o azul o a una proporción de ellos.
15. **Luminosidad**: Brillo de una zona respecto a otra zona blanca en la imagen.
16. **Ruido**: *Píxeles* aislados que toman un nivel de gris diferente al de sus vecinos.
17. **KeyPoints:** Puntos clave detectados por el algoritmo SIFT
18. **Descriptores:** Conjunto de métricas calculadas en el procesamiento de imágenes para cuantificar la textura percibida.
19. **SVM**: (en inglés, *Support Vector Machine)*

CAPÍTULO I: GENERALIDADES

1.1. Motivación

En los últimos años el campo de Visión Artificial ha experimentado un aumento en sus investigaciones y aplicaciones, abarcando diferentes áreas. En la industria de fabricación de tableros, el control de calidad consiste en detectar las distintas anomalías presentes en la superficie, esto es realizado mediante la inspección visual que en la mayoría de los casos se realiza en forma manual. Esto suele ser subjetivo debido a factores como la fatiga visual, estado de ánimo, cansancio, aburrimiento, etc. Incidiendo en la clasificación del panel, y a posibles rechazos en las etapas posteriores o en el usuario final, lo que implica costos económicos para el fabricante.

Los defectos en paneles de melamina, presentan una mayor dificultad en su detección debido al brillo y al color, esto principalmente por las capacidades del operador humano que debe realizar un trabajo monótono y repetitivo, disminuyendo la exactitud de visión.

Esta necesidad de una herramienta robusta y objetiva que tenga las capacidades de cumplir la función de control de calidad y una alta repetitividad motiva la realización de este proyecto, el cual tiene como objetivo aportar en la detección de defectos en la superficie para la futura implementación de un sistema completo de visión artificial, según se detalla a continuación.

1.2. Objetivos del proyecto

Objetivo General:

Estudiar, implementar y evaluar algoritmos de detección de defectos en paneles de madera utilizando imágenes multiespectrales.

Objetivos Específicos:

1. Estudiar algoritmos de detección de defectos en paneles de madera utilizando imágenes multiespectrales existentes en la literatura del área.
2. Implementar y probar los algoritmos estudiados en lenguaje C#.
3. Evaluar el desempeño de los algoritmos implementados.

1.3. Limitaciones y alcances

Si bien este trabajo aporta a la detección de defectos mediante distintas combinaciones de técnicas y el estudio de algoritmos de detección de defectos, no se utilizan los descriptores para especificar un defecto determinado o dar a entender una diferenciación entre los tipos de defectos. En las conclusiones quedan las directrices para los trabajos futuros y continuación de este proyecto.

1.4. Herramientas y tecnologías utilizadas

Para el desarrollo del proyecto se han utilizado diversas tecnologías y herramientas. Por un lado es necesario realizar un tratamiento a las imágenes, para esto se utilizan las librerías OpenCV[1] a través de un *wrapper* o envoltorio que permite trabaja con el lenguaje C#.

Para la evaluación de los algoritmos se cuenta con un conjunto de imágenes de distintas bandas espectrales proporcionadas por el Laboratorio CIM de la Universidad del Bío-Bío.

[1] Open source computer visión library OPENCV, disponible en http://www.opencv.org

1.5. Presentación de capítulos

Capítulo I: Generalidades

Este capítulo permite al lector introducirse al tema central del proyecto, donde se entrega la justificación de lo que se realizará y los objetivos que se deben lograr.

Capítulo II: Marco Teórico

Este capítulo presenta al lector conocimiento teórico generales sobre imágenes multiespectrales y el procesamiento al que son sometidas, que sirve de base para el desarrollo de este trabajo.

Capítulo III: Estudio del estado del arte

Este capítulo describe investigaciones recientes en cuanto a la detección de defectos en texturas y los algoritmos empleados, junto al control de calidad por medio de imágenes multiespectrales.

Capítulo IV: Desarrollo

En este capítulo se describe el trabajo desarrollado, definiendo en él la metodología de detección propuesta. Se detallan las etapas del procesamiento digital de imágenes, teniendo en cuenta las etapas de pre-procesamiento, segmentación, extracción de características de textura, y evaluación de los algoritmos que permiten dar solución al problema planteado.

Capítulo V: Experimentos

En este capítulo se muestran los resultados obtenidos de acuerdo al procesamiento de las imágenes en diferentes escenarios. También se grafica los resultados de las diversas pruebas para mostrar el algoritmo con mejor resultado.

Finalmente se exponen las conclusiones de este trabajo, comparando los métodos estudiados y destacando sus fortalezas, debilidades y dominios de aplicación. Además se presentan futuras líneas de investigación que puedan ser seguidas a partir de lo presentado.

CAPÍTULO II: MARCO TEÓRICO

2.1. Introducción a la visión artificial

Para comenzar se diferenciarán los términos visión artificial y visión por computador los cuales frecuentemente están siendo utilizados indistintamente[1] Visión por computador es el conjunto de técnicas que permite obtener información útil de una o varias imágenes por medio de un dispositivo electrónico programable, esto es, extraer las características más representativas para el análisis de la imagen, lo que constituye el objetivo de la visión por computador [26].

En cuanto a la visión artificial, es definida como el campo de la inteligencia artificial que con la utilización de técnicas adecuadas, permite adquirir, procesar y analizar cualquier tipo de información obtenida a través de imágenes digitales [26]. Es decir, la visión artificial es la aplicación de la visión por computador a la industria con el fin de controlar un sistema automático utilizando información visual.

Los sistemas de visión artificial cumplen tareas de inspección con un alto nivel de repetitividad y flexibilidad, nunca se cansan ni se distraen, pueden ser puestos en funcionamiento en ambientes donde los operadores humanos no podrían realizar labores bajo condiciones de seguridad. Para un inspector humano, los ojos proporcionan información del ambiente que lo rodea, el cerebro interpreta la información de acuerdo a experiencias previas con objetos similares y basándose en esta interpretación toma decisiones y ejecuta acciones. De manera similar, los sistemas de visión artificial ven al objeto, lo interpretan y toman decisiones.

2.1.1. Imagen Digital

La imagen es un elemento constitutivo de todo proceso de transmisión de información, es definida como una función bidimensional $f\ (x, y\ )$ donde x e y son coordenadas en el plano y la amplitud f es llamada intensidad o nivel de gris en ese punto [libro imagen]. Al ser digital, procedente directamente de una cámara digital, es considerada como una matriz bidimensional discretizada en niveles de grises.

Cada celda de la matriz es un elemento propio de la imagen digital, llamado ***píxel***, cada uno de los cuales tiene un valor y una posición particular. El valor es relativo a alguna propiedad del punto que representa, como por ejemplo su brillo o su matiz.

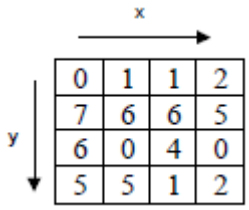


Figura 1: Imagen de 16 píxeles, origen: [4]

El ***píxel*** (abreviación de *Picture element*) es la unidad mínima de información de una imagen, se compone de tres registros de color, mediante la combinación de cierta cantidad de rojo, verde y azul, el *píxel* adopta un color particular.

Figura 2: Representación de un histograma, origen: [19]Figura 3: Imagen de 16 píxeles, origen: [4]

La **resolución de una imagen** es el número de *píxeles* que contiene una imagen. Expresada como 640x480, 800x600, por ejemplo, lo cual es un atributo de calidad de la imagen.

El **Histograma** corresponde a la distribución estadística de los niveles de intensidad de todos los *píxeles* de una imagen monocroma, esto es, un gráfico de barras que representa la frecuencia de cada valor de intensidad (o "nivel de gris") en la imagen.

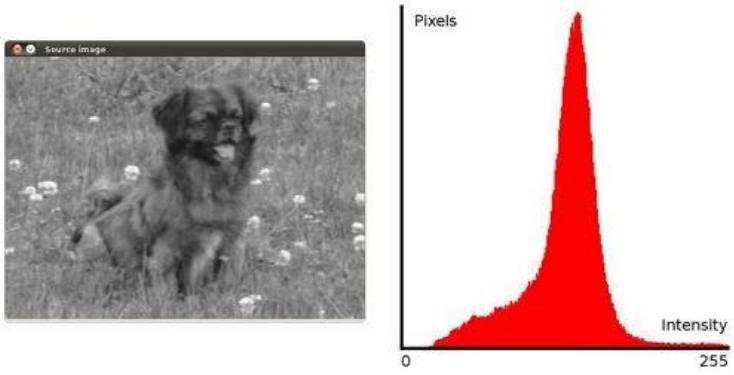


Figura 4: Representación de un histograma, origen: [19]

Figura 5: Etapas del procesado de imagen, adaptación de [25]Figura 6: Representación de un histograma, origen: [19]

2.1.2. Multiespectral

Imagen multiespectral se refiere a la imagen construida con bandas de ondas simples, esto es, la imagen formada con luz visible, rayos infrarrojos, radiación ultravioleta, ondas milimétricas o rayos x[2]. La imagen multiespectral contiene gran cantidad de información sin embargo requiere el desarrollo de nuevas herramientas o la adaptación de las actuales abriendo nuevos campos de investigaciones en torno al procesamiento de video e imágenes. El análisis multiespectral ha sido ampliamente estudiado en la teledetección donde imágenes satelitales son comúnmente registradas y fusionadas. Además los avances tecnológicos han llevado a utilizarlas en aplicaciones como vigilancia., análisis de imágenes médicas o exploración geológica. En muchos casos se basa en la extracción de información utilizando la detección de bordes, lo cual constituye el proceso más difícil en el procesamiento de imágenes[3][2][3].

Las bandas o segmentos del espectro electromagnético utilizado en este trabajo son las siguientes:

2.1.2.1. Luz visible

Ocupa la banda entre 0,4 μm y 0,7 μm y su denominación se debe a que es la única radiación que puede captar el ojo humano. Dentro del visible se pueden distinguir tres bandas espectrales correspondientes a los colores básicos: azul, verde y rojo.

2.1.2.2. Infrarrojos

La radiación infrarroja, radiación térmica o radiación IR es un tipo de radiación electromagnética de mayor longitud de onda que la luz visible, abarca desde, aproximadamente, 0,75 μm hasta 1000 μm, es emitida por cualquier cuerpo cuya temperatura sea mayor que el cero absoluto (0° Kelvin). Dentro de este rango se pueden diferenciar distintas regiones:

- Infrarrojo cercano(NIR) 0,75 a 1,4 μm
- Infrarrojo mediano 01,4 q 8 μm
- Infrarrojo lejano (LWIR)8 a 15 μm

2.1.2.3. Ultravioleta

La radiación ultravioleta UV está comprendida entre 0,15 y 0,4 μm. Su longitud de onda comienza detrás del espectro visible que las personas observamos como color violeta. Es utilizada en algunos contextos para detectar fluidos en superficies y esterilizar productos ya que matan virus y bacterias. En cuanto a los rayos del sol, la mayor parte de la radiación UV es absorbida por la atmósfera. Sin embargo, debido al agujero de la capa de ozono, los rayos ultravioletas llegan cada vez en mayor cantidad a la superficie terrestre [24].

2.1.3. Procesado de imagen

Se entiende por procesamiento la realización de operaciones directamente sobre el valor de los *píxeles* que forman la imagen de[4]. De acuerdo a la literatura, los pasos que se deben seguir para un sistema de visión artificial se resumen en el siguiente esquema:

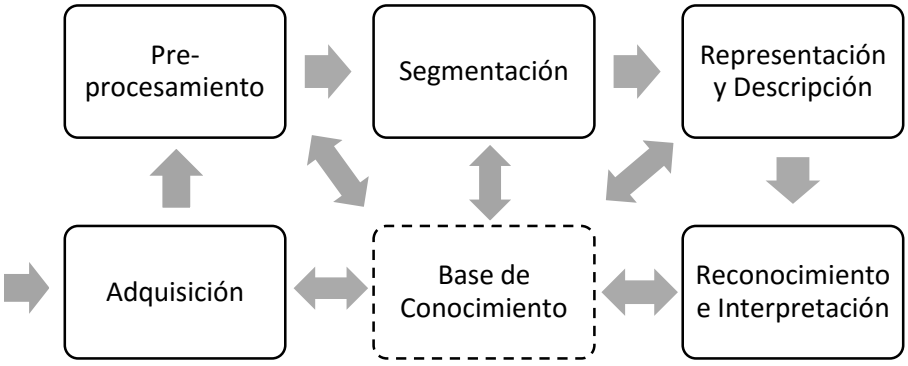


Figura 7: Etapas del procesado de imagen, adaptación de [25]

Para la primera etapa es necesario un dispositivo tal como cámaras digitales, cámaras web, cámaras filmadores, entre otros.

Figura 8: Relación entre propiedades, origen: [4]Figura 9: Etapas del procesado de imagen, adaptación de [25]

El pre-procesamiento consiste en realizar una serie de métodos para mejorar la imagen con el fin de aumentar las posibilidades de éxito de detección en etapas posteriores. Para esto se aplican técnicas de mejoramiento del contraste, operaciones morfológicas, aplicación de filtros y umbralización. Luego la segmentación consiste en dividir la imagen en sus partes constituyentes, destacando el objeto en estudio. Esto permite decidir si los *píxeles* son un contorno o la región completa. De esta forma la descripción se encarga de extraer los rasgos con información cuantitativa de interés para poder diferenciar una clase de objeto de otra. Luego la etapa de reconocimiento es el proceso que asigna una etiqueta a un objeto basándose en la información proporcionada por sus descriptores. La interpretación implica asignar significado a un conjunto de objetos reconocidos. Finalmente la base de conocimiento sirve de almacenamiento de información: Ya sean los datos de las imágenes en cada paso, o heurísticas que ayudarán a obtener mejores imágenes en subsecuentes procesamientos.

Los algoritmos que se describen en esta sección sirvieron como referencia para aplicar secuencias y / o combinaciones en las metodologías desarrolladas. Todos ellos fueron probados con todas las imágenes de las diferentes bandas espectrales.

2.1.3.1. Modelo de color HSV

El modelo HSV (***H**ue, **S**aturation, **V**alue* – Matiz, Saturación, Valor*)* corresponde a la manera más cercana en que los humanos percibimos el color, lo describe en términos de su tono (H), saturación(S) y su brillo (V). Resulta útil para sistemas automáticos de detección del grado de maduración de frutas y vegetales, sistemas para comparar muestras de color o inspeccionar la calidad de productos coloreados[5].

- El matiz (**H**) de un color se refiere a que color puro se asemeja. Son descritos por un número que especifica la posición del correspondiente color puro en el círculo cromático, como una fracción entre 0 y 1. Valor 0 se refiere al rojo; 1/6 es amarillo; 1/3 es verde; y así sucesivamente alrededor del círculo.
- La saturación(**S**) de un color describe que tan blanco es el color. Un rojo puro está totalmente saturado, con una saturación de 1.
- El valor o brillo (**V**) describe que tan oscuro es el color, se refiere a la cantidad de luz percibida. Una valor 0 es negro. Por lo tanto, a medida que a un color se le agrega más negro, se intensifica su oscuridad y se obtiene un valor más bajo y a medida que a un color se le agrega más blanco se intensifica la claridad del mismo por lo que se obtienen valores más altos.

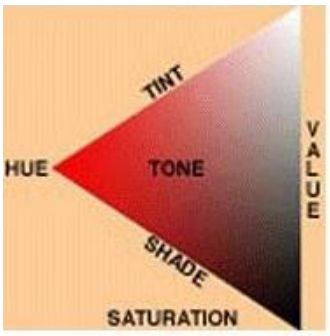


Figura 10: Relación entre propiedades, origen: [4]

Figura 11: Filtro espacial, origen: [23]Figura 12: Relación entre propiedades, origen: [4]

2.1.3.2. Conversión a escala de grises

Las imágenes son convertidas a escala de grises para poder manipularlas, así los datos consisten en un solo canal que representa la intensidad, brillo o densidad de la imagen. Cada *píxel* es representado con un único valor entre [0-255], donde el valor mínimo representa el brillo mínimo (negro) y el valor máximo del rango representa el brillo máximo (blanco). Al trabajar con el modelo HSV, se toman los valores contenidos en el canal V de este modelo. De esta manera el *píxel* queda representado por un único valor en la escala de grises.

2.1.3.3. Umbralización

La elección del umbral es el paso más importante para desarrollar la segmentación de la imagen de manera satisfactoria [4]. La elección del valor umbral puede ser manual o automático. Al trabajar con imágenes del mundo real, utilizar valores arbitrarios de umbral no da buenos resultados debido a la iluminación inadecuada, histogramas planos o presencia de ruido.

Método Otsu: este tipo de umbralización automática es uno de los mejores métodos de selección de umbral para imágenes del mundo real [1]. Consiste en seleccionar el umbral óptimo maximizando la varianza entre clases mediante una búsqueda exhaustiva[6]. Dentro de sus ventajas está la buena respuesta del método frente a la mayoría en situaciones del mundo real (ejemplo: imágenes ruidosas, con histogramas planos o mal iluminadas) y no precisa de supervisión humana, pre-procesamiento de la imagen ni otro tipo de información acerca de la misma. Sin embargo a medida que el número de clases en la imagen aumenta, el método necesita mucho más tiempo para seleccionar un umbral multinivel adecuado. También muestra buenos resultados en aplicaciones de detección de defectos, cuando estos varían de pequeños a grandes defectos[7].

2.1.3.4. Binarización

Una vez que la imagen es convertida en escala de grises, es posible transformarla en una representación de dos tonos de color: blanco y negro. Esta tarea se lleva a cabo comparando los niveles de gris presentes en la imagen con un valor predeterminado (umbral). Si el nivel de gris de la imagen es menor que la el umbral predeterminado, se le asigna al *píxel* de la imagen binarizada el valor 0(negro), y si es mayor, se le asigna un 1(blanco). Este procesamiento simplifica la representación de cada *píxel*, pasando de 256 posibles valores a tan solo 2. Formalmente este método es definido como: Escoger un valor umbral *T(x, y)* para cada *píxel I*(x, y) y establecer el valor de cada *píxel* como THRES_BINARY:

$$dst(x,y) = \begin{cases} 0, & si\ I(x,y) < T(x,y) \\ 255, & en\ cualquier\ otro\ caso \end{cases}$$

2.1.3.5. Filtros Digitales

La técnica de filtrado permite resaltar o atenuar características de la imagen, constituye uno de los principales modos de operar en el procesamiento de imágenes digitales. Los filtros son aplicados a las imágenes para distintos objetivos [23]:

- Suavizar la imagen: reducir las variaciones de intensidad entre *píxeles* vecinos.
- Eliminar ruido: modificar aquellos *píxeles* cuyo nivel de intensidad es muy diferente al de sus vecinos.
- Realzar la imagen: consiste en aumentar las variaciones de intensidad, donde son producidas.
- Detectar bordes: detectar aquellos *píxeles* donde se produce un cambio brusco en la función de intensidad.

Los filtros pueden ser clasificados según el dominio del espacio (lineal y no lineal) y en el dominio de la frecuencia.

Filtros espaciales

Se refiere a que las operaciones espaciales de filtrado son definidas en un entorno de vecindad del punto a transformar*(x, y*), esto es, el trabajo directo sobre los *píxeles* de la imagen. La siguiente figura muestra lo anterior.

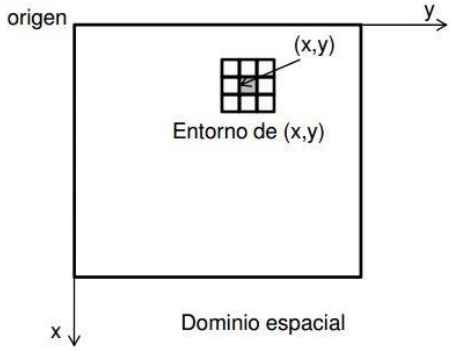


Figura 13: Filtro espacial, origen: [23]

Figura 14: Obtención de la mediana , origen: [8]*Figura 15: Filtro espacial[2], origen: [23]*

Pueden ser lineales (basados en máscaras de convolución[2]) y no lineales.

Filtro de la Mediana: Es del tipo no lineal y se basa en sustituir cada *píxel* de la imagen por el valor de la mediana del conjunto formado por el mismo y sus ocho *píxeles* vecinos. Frecuentemente es utilizado para eliminar el ruido en la imagen, ya que fuerza a que *píxeles* con niveles de gris diferente a los demás se asemejen más a sus vecinos.

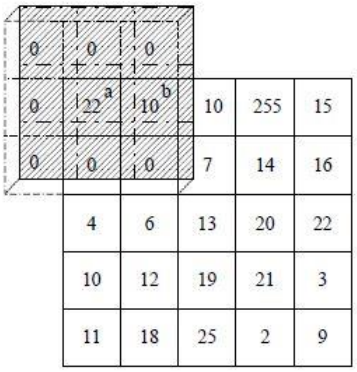


Figura 16: Obtención de la mediana , origen: [8]

Figura 17: Máscaras operador Sobel, origen: [27]*Figura 18: Obtención de la mediana , origen:* [8]

[2] Convolución: Se define como la integral del producto de dos funciones después de desplazar una de ellas una distancia *t*. Fuente: [9]

En la figura 3, se observa la vecindad de a= {0,0,0,0,22,10,0,0,0}, ordenando de menor a mayor queda vecindad de a= {0,0,0,0,0,0,10,22}, donde la mediana es **0**. Al usar filtros de reducción de ruido la nitidez puede verse afectada, por lo tanto es conveniente utilizar filtros de realce espacial para acentuar los detalles de la imagen[8]. Es efectivo cuando el objetivo es reducir el ruido y al mismo tiempo preservar los bordes.

Filtros en el dominio de la frecuencia

El sistema matricial de coordenadas de una imagen es lo que se denomina dominio espacial. Sin embargo, la misma imagen puede ser considerada como una función no periódica y definirse en otro espacio bidimensional, cuyos ejes vengan determinados por la amplitud y la frecuencia para cada dirección de la imagen. Este nuevo espacio de referencia para la descripción de la imagen se conoce como dominio de la frecuencia [9]. Permite seleccionar no sólo la dirección de filtrado, sino también los intervalos de frecuencia que requieran ser eliminados.

Los filtros definidos en el dominio espacial en realidad tienen su repercusión en el de la frecuencia. De ello deriva la terminología empleada cuando se habla de filtros de paso alto o bajo. Se refiere a que retienen bajas o altas frecuencias, explicando precisamente el efecto que causan en el espacio frecuencial de la imagen.

Filtro Gaussian Smoothing: Es del tipo lineal y de paso-bajo. Es utilizado para imágenes con falta de definición y también para quitar detalles y ruidos. La salida gaussiana entrega una media ponderada de la vecindad de cada *píxel*, el cual tiende más hacia el valor de los *píxeles* centrales. Como resultado entrega una imagen difuminada que emborrona menos los bordes, en comparación a otros filtros [9]

2.1.3.6. Realce de bordes

Consiste en enfatizar o resaltar aquellos *píxeles* que tienen un valor de gris diferente al de sus vecinos. Sin embargo al contener ruido su efecto se multiplicará, por lo que es recomendable eliminar primero el ruido[8].

2.1.3.7. Detección de Contornos

En una imagen los contornos corresponden a los límites de los objetos presentes. Para hallar estos contornos se buscan los lugares en la imagen en los que la intensidad del *píxel* cambia rápidamente, generalmente se utilizan algunos de los siguientes criterios:

- Lugares donde el gradiente (primera derivada) de la intensidad es de magnitud mayor que la de un umbral predefinido.
- Lugares donde el laplaciano (segunda derivada) de la intensidad tiene un cruce por cero.

Operador Sobel: es un método basado en el gradiente donde la derivada es calculada en forma separada para los ejes x e y. Los bordes horizontales y verticales son detectados en forma separada, por medio de la convolución de dos máscaras de 3x3 con la imagen en escala de grises. Sobel no solo calcula la magnitud de los bordes, sino que también su dirección.

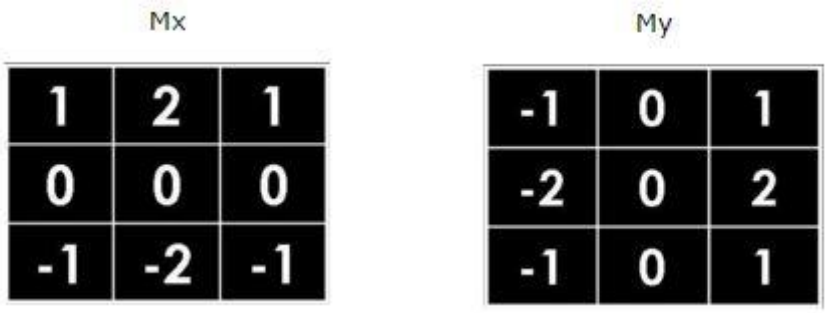


Figura 19: Máscaras operador Sobel, origen: [27]

Figura 20: Ecualización de histograma, origen: [19]Figura 21: Máscaras operador Sobel, origen: [27]

2.1.3.8. Ecualización del histograma

Esta técnica transforma o modifica el histograma de una imagen con el fin de realzar su contraste expandiendo la distribución de los niveles de gris de forma uniforme.

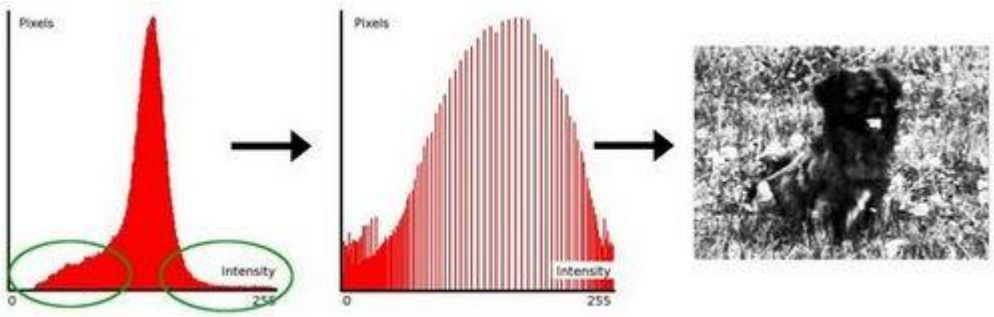


Figura 22: Ecualización de histograma, origen: [19]

Figura 23: Clasificador SVM, origen: [31]Figura 24: Ecualización de histograma, origen: [19]

2.1.4. Segmentación

El proceso de segmentación de una imagen depende del problema que se desee resolver y determina el eventual éxito o fracaso del análisis y reconocimiento, por este motivo es considerada una de las etapas más difíciles. La segmentación busca destacar el objeto en estudio sobre el fondo de la imagen, desechando de esta forma la información sin utilidad.

Los algoritmos de segmentación de imágenes se basan en dos propiedades básicas de los niveles de gris de la imagen: **discontinuidad** que intenta dividir la imagen basándose en los cambios bruscos del nivel de gris con el fin de detectar puntos, líneas y bordes en la imagen**.** Y la **similitud** que se basa en técnicas de umbrales.

Las técnicas de segmentación actuales son divididas en dos grandes enfoques:

1. Basadas en regiones: intentan encontrar particiones de los píxeles de la imagen en conjuntos correspondientes a las propiedades de imagen coherente como brillo, color y textura.
2. Basadas en contornos: Por lo general comienzan con una primera etapa de detección de bordes, seguido de un proceso de vinculación que busca explorar continuidad curvilínea.

Estos enfoques por separados, son insuficientes para segmentar y describir la mayoría de los problemas reales. Por ello, se suele utilizar combinaciones de varios métodos propuestos al igual que realizar modificaciones para ajustar estos métodos al problema particular que se trate.

2.1.5. Extracción de características

Finalmente, después del procesado se obtienen características más significativas, sin embargo, dado que los paneles en algunas imágenes presentan inclinaciones, es necesario encontrar características que sean invariantes a la rotación y a la escala, esto es, que al rotar una imagen o escalarla, el valor numérico de las características sea similar.

Para lograr esto se pueden tomar dos enfoques teniendo en cuenta que las características buscadas deben ser lo más similar posible para objetos de la misma clase y lo más diferente posible para objetos de distintas clases[10]:

1. Transformar la imagen: rotar y escalar la imagen para hacerla lo más similar posible a un patrón estándar, luego se extraen características numéricas de ella.

2. Trabajar con la imagen tal cual y utilizar características en las que se obtengan valores numéricos similares para una misma imagen rotada y escalada.

2.1.6. Clasificación

Una vez extraídas las características de las imágenes, se debe encontrar un sistema que dado un conjunto de características de la imagen indique a qué imagen de las almacenadas se parece más o corresponde. Esto es lo que se conoce como clasificador. Este sistema de aprendizaje trata de etiquetar (clasificar) una serie de vectores utilizando una entre varias categorías (clases). La base de conocimiento del sistema está formada por ejemplos etiquetados anteriormente.

Support Vector Machine (SVM)

Originalmente, máquinas de soporte vectorial (SVM) era una técnica para la construcción de un clasificador óptimo binario (2-clases). Más tarde, la técnica se extendió a los problemas de regresión y de agrupamiento [19]. Actualmente SVM es definido como un conjunto de métodos de aprendizaje supervisado utilizados para clasificación y regresión. Visualiza los datos de

entrada como dos conjuntos de vectores en un espacio n-dimensional, un SVM construirá un hiperplano que separa ese espacio y que maximiza el margen entre los dos conjuntos de datos. Para calcular el margen, se construyen dos hiperplanos paralelos, uno a cada lado del hiperplano de separación, los cuales son "empujados" contra los dos conjuntos de datos [19]. Y cuanto mayor sea el margen, más bajo es el error de generalización del clasificador.

Dentro de sus fortalezas se encuentran [30]:

- El entrenamiento es relativamente fácil.
- No hay óptimo local, como en las redes neuronales.
- Se escalan relativamente bien para datos en espacios dimensionales altos.

El hiperplano de separación de clases se obtiene a partir de la solución de un problema de optimización: distancia máxima entre los hiperplanos que contienen los vectores de soporte de ambas clases (margen máximo).

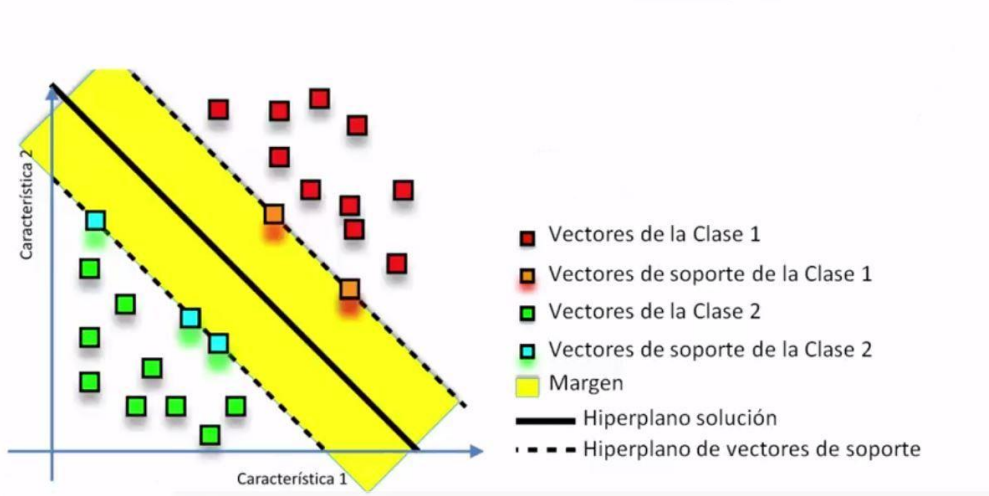


Figura 25: Clasificador SVM, origen: [31]

Figura 26: Filtros Gaussianos, origen
[19]Figura 27: Clasificador SVM, origen: [31]

2.2. Métricas de evaluación

A pesar de la importancia de la evaluación cuantitativa del rendimiento de los algoritmos de detección, esta se encuentra ausente en la mayoría de los trabajos publicados. Se debe principalmente a que la calidad de un detector de contorno puede depender de la tarea específica para la que se realiza y no existe un acuerdo general acerca de cómo la evaluación del desempeño debe llevarse a cabo ya que los enfoques propuestos sufren diversas deficiencias[11].

Existen autores que realizan una medición de diversas propiedades de los contornos detectados, tales como la coherencia, la continuidad, la suavidad y buena continuación. Este enfoque tiene la ventaja de ser objetivo, pero la coherencia entre la imagen de entrada y contornos detectados no se toma en cuenta [11]. Además esta medición se realiza con experimentos donde el observador humano debe juzgar la calidad de las salidas de los detectores, siendo este enfoque costoso y no automático. Estas dificultades ha llevado a la mayoría de los autores a trabajar con el enfoque de comparar un conjunto de *píxeles* del contorno detectado (DC) con un conjunto de *píxeles* reales GT (*Ground Truth)* los que especifican la salida deseada para una imagen de entrada dada. Debido a lo expuesto es necesario además utilizar métodos robustos en cuanto a precisión y objetividad, que no dependa de un operador humano para conocer su exactitud.

2.2.1. Análisis Cuantitativo

Para medir el rendimiento de los algoritmos desarrollados aplicados en las imágenes en diferentes espectros es necesario controlar el tiempo de ejecución de estos en milisegundos. En el caso de SIFT el tiempo en detectar los puntos claves [28] y en LBP-Liu el tiempo en calcular descriptores de intensidad central, de la vecindad, diferencia angular y diferencia radial para cada pixel de acuerdo a un radio dado [12].

Recall y *Precision*

Con el fin de interpretar los resultados del clasificador SVM se utilizará una herramienta básica en la evaluación de credibilidad: Matriz de Confusión.

Esta matriz simple consta de filas que se refieren a las condiciones verdaderas que arroja el algoritmo y las columnas se refieren a las condiciones predichas que deberían cumplirse.

La tabla adjunta muestra las posibles salidas. Si el algoritmo detecta defectos, se clasificará como *Positive*, sino se dirá que es *Negative.* La detección podría ser correcta (*true*) o incorrecta (*false*), por lo tanto la matriz contiene cuatro posibles salidas de combinaciones entre *positive/negative* y *true/false*.

	Positive	Negative
True	True Positive (TP)	True Negative (TN)
False	False Positive (FP)	False Negative (FN)

Tabla 2: matriz de confusión. Origen: [29]

De la tabla anterior es posible obtener valores con las siguientes definiciones:

$$Recall = \frac{TP}{P}, donde\ P = TP + FN$$

Dice qué proporción de muestras que realmente tenía defectos fueron detectadas por el algoritmo. Y

$$Precision = \frac{TP}{(TP + FP)}$$

Indica el porcentaje de predicciones positivas que son correctas.

Estos dos indicadores están relacionados inversamente, si *Recall* disminuye *Precision* aumenta y viceversa, lo cual permite elegir un algoritmo apropiado dependiendo del requerimiento: una gran precisión a costa de exhaustividad (*recall)* o gran exhaustividad (*recall)* con baja precisión. Para mantener un equilibrio entre estos indicadores se obtiene una media armónica también llamada *F1 Score*.

$$F1\ Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

CAPÍTULO III: ALGORITMOS DE DETECCIÓN Y CONTROL DE CALIDAD

3.1. Detección de defectos en texturas

A través del tiempo, distintas metodologías se han creado para obtener mayor información de las imágenes, como también nuevas técnicas de mejora de la imagen y nuevos algoritmos de procesamiento digital están en constante evolución, así como también la combinación de estas técnicas.

En sistemas de inspección visual, detectar defectos en las superficies se basa en la detección de algún cambio significativo en las características de la superficie. El espacio de características puede ser muy grande y complejo, y usar un número excesivo de características puede incrementar la complejidad y degradar el rendimiento[13].

La detección de defectos en texturas puede ser definida como el proceso de determinar la ubicación y/o la extensión de una colección de *píxeles* en la imagen texturizada con una desviación marcada en sus valores de intensidad o arreglo espacial con respecto a la textura de fondo[14].

En el análisis, el objetivo es describir una textura determinada con algún (preferentemente pequeño) conjunto de descriptores. Sin embargo, la textura tiene una propiedad de imagen no local. Los métodos básicos para el reconocimiento basado en la textura son combinaciones de dominio de frecuencia y espaciales.

En [15] se describe una segmentación de textura combinando técnicas multi-banda y Filtros de Gabor detectando así de manera exitosa formas en texturas con fondos desordenados.

Otra forma de detectar texturas es utilizando descriptores locales invariantes a escala[16], donde los vectores resultantes son conocidos como SIFT *keys* y son utilizados en un enfoque vecino más cercano para identificar posibles objetos en una imagen.

Por otro lado en [10] se analiza una imagen que no contiene una textura bien marcada. Esto es realizado por medio del tratamiento de contornos y análisis de patrones en forma simultánea.

Los detectores de contorno se dividen en operadores locales y globales. Las primeras se basan principalmente en el análisis diferencial, enfoques estadísticos, la congruencia de fase, filtros orden de rango, y combinaciones de los mismos. Estos últimos incluyen el cálculo de la prominencia de contorno, agrupación perceptual, etiquetado relajación y contornos activos[11].

3.2. Algoritmos de detección

3.2.1 Scale Invariant Feature Transformation (SIFT)

En 2004, David Lowe presenta un método para extraer de la imagen características distintivas invariantes, mediante la extracción de puntos claves y el cálculo de sus descriptores[17].

Hay principalmente cuatro etapas implicadas en el algoritmo SIFT:

1. Detección de máximos y mínimos espacio-escala: La imagen se procesa con filtros gaussianos a diferentes escalas, y luego se calcula la diferencia de los sucesivos puntos Gaussianos que se han encontrado en la imagen. Los máximos y mínimos de esta función proporcionan las características más estables, con lo que esto serán los puntos claves (*keypoints*).

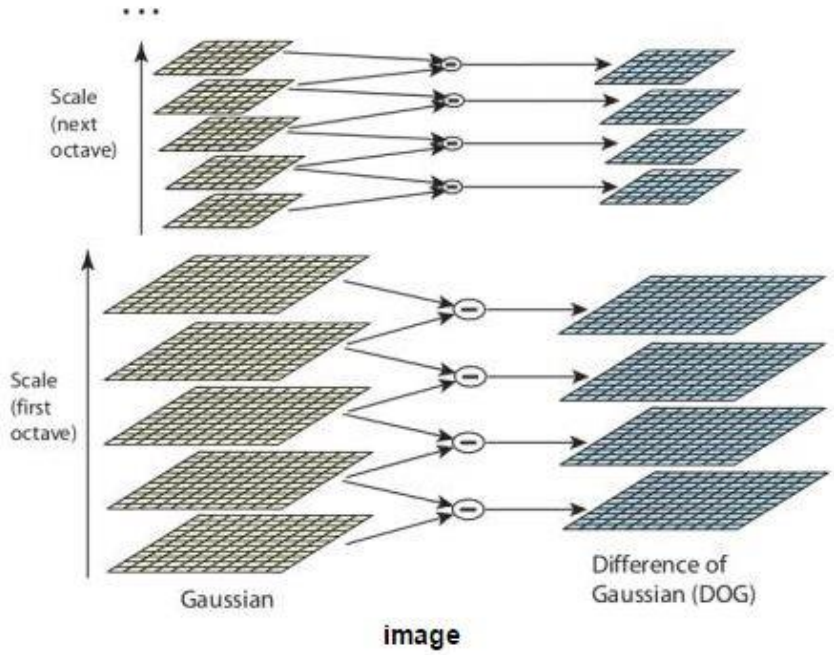


Figura 28: Filtros Gaussianos, origen [19]

Figura 29: Izquierda, gradiente de la imagen. Derecha, Descriptores, origen: [17]*Figura 30: Filtros Gaussianos, origen [19]*

2. Localización de los *Keypoints*: El paso anterior da como resultado demasiados candidatos *Keypoint*, así que se realiza un ajuste detallado de los datos más cercanos para la localización exacta, la escala y proporción de curvaturas principales. De esta forma se rechazan puntos con bajo contraste (y por lo tanto sensibles al ruido) o mal localizados.

3. Asignación de la orientación: Ahora una orientación se asigna a cada *keypoint* para lograr invariancia a la rotación de la imagen. Un vecindario se toma alrededor de la ubicación del *keypoint* dependiendo de la escala y la magnitud y la dirección del gradiente es calculada en esa región. Se crea un histograma de orientación con 36 divisiones que abarcan 360 grados. Cada muestra añadida al histograma es pesada por su magnitud del gradiente y por una máscara gaussiana circular con un σ 1,5 veces el valor que posea el *keypoint*. El punto más alto en el histograma se toma y cualquier punto por encima del 80% de este, también es considerada para calcular la orientación. Se crean puntos clave con la misma ubicación y escala, pero con diferentes direcciones. Esto contribuye a la estabilidad del método.

4. Descriptores de los *keypoints*: Luego se debe segmentar la vecindad del *keypoint* en 4x4 regiones de 4x4 *píxeles*. Una vez realizado esto se genera un histograma de orientación gradiente para cada región. Y se utiliza una ponderación gaussiana con un ancho σ = 4 *píxeles*. Como los histogramas de orientación de cada región están divididos en 8 barras, por cada vecindad del *keypoint* se puede construir un histograma tridimensional de 4×4×8 valores, formando así un vector con 128 valores lo cual constituye el descriptor.

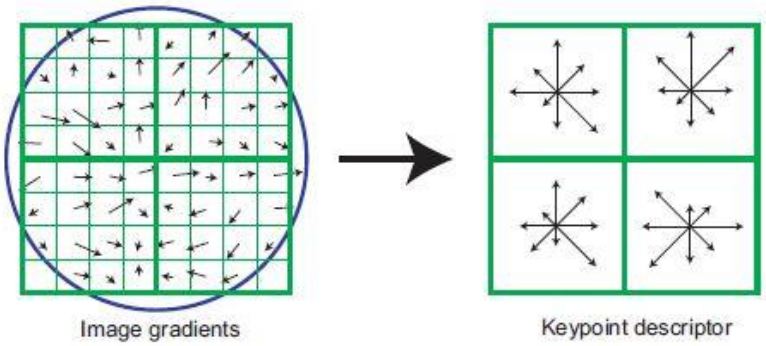


Figura 31: Izquierda, gradiente de la imagen. Derecha, Descriptores, origen: [17]

Figura 32: Izquierda, gradiente de la imagen. Derecha, Descriptores, origen: [17]

3.2.2 Local Binary Pattern (LBP)

En contraste con los descriptores de imagen globales que computan características directamente de toda la imagen, los descriptores locales representan las características en pequeños sectores de imagen, lo cual ha demostrado ser más eficaces en el mundo real[18][12].

El descriptor **LBP** *Local Binary Pattern* es un poderoso medio de descripción de la textura y entre sus propiedades en aplicaciones del mundo real son su poder discriminativo, simplicidad computacional y la tolerancia frente a los cambios monótonos en escala de grises.

Un operador de "patrones local" describe las relaciones entre un pixel y los pixeles vecinos en una vecindad de 3x3; todos los vecinos que tienen valores más altos o igual que el valor del pixel central se le asigna un valor de 1, y todos los que tienen un valor menor, un 0. A cada resultado del umbral se le asigna un peso de *2^n*, en donde *n* depende de la posición del vecino con respecto al *píxel* central, finalmente se realiza una suma de los diferentes pesos obteniendo la representación LBP del *píxel*, el cual puede ser utilizado para caracterizar la textura local. Así el histograma corresponde a $2^8 = 256$ etiquetas diferentes que se pueden utilizar como descriptores de textura para su posterior análisis. Dado que LBP es invariante a los cambios monótonos en escala de grises, se complementó con una medida ortogonal de contraste local.

La figura muestra cómo se calcula la medida de contraste C: El promedio de los niveles de gris debajo del valor del *píxel* central es restado del promedio de los niveles de gris por encima (o igual) del *píxel* central.

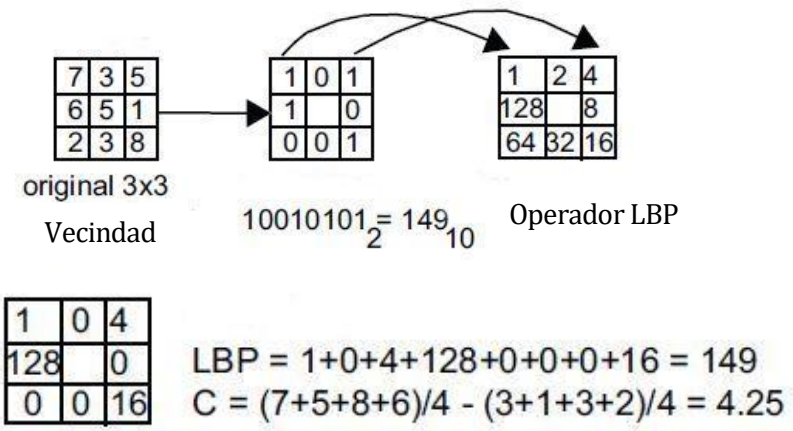


Figura 34: Algoritmo LBP, adaptación: [26]

Figura 35: Vecinos circulares de un píxel de resolución múltiple LBP, origen: [26]Figura 36: Algoritmo LBP, adaptación: [26]

Sin embargo presenta desventajas[12]:

1. Produce largos histogramas que son sensibles a la rotación de la imagen.
2. No puede detectar correctamente estructuras de textura a gran escala.
3. Es muy sensible al ruido. La mínima fluctuación por encima o por debajo del valor del *píxel* central se trata como equivalente a un gran contraste entre el *píxel* central y sus alrededores.

Esto ha llevado a desarrollar una variante del algoritmo, el **LBP Extendido**, el cual consiste en una vecindad circular del *píxel* para lograr un análisis multi-escala e invariante a la rotación. Esto se consigue con la interpolación de los valores de los *píxeles*, lo que permite utilizar cualquier radio y por lo tanto cualquier número de *píxeles* vecinos. En la figura se puede observar ejemplos para diferentes tamaños, donde P indica el número de puntos de muestreo y R el radio del círculo.

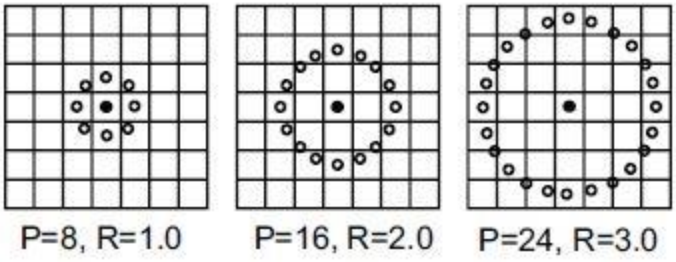


Figura 37: Vecinos circulares de un píxel de resolución múltiple LBP, origen: [26]

Figura 38: Enfoque LBP-Liu, origen: [12]*Figura 39: Vecinos circulares de un píxel de resolución múltiple LBP, origen: [26]*

De este operador se han desarrollado diversas nuevas extensiones, y en el presente trabajo se utilizará la desarrollada por Liu en 2012 para la clasificación de texturas[12]. Este enfoque extrae dos tipos de características diferentes y complementarias (intensidad del *píxel* y diferencias).

Las características basadas en la intensidad considera el *píxel* central (CI) y el de sus vecinos (NI); mientras que las características basadas en la diferencia, se calculan dos componentes: la diferencia radial (RD) y la diferencia angular (AD). Por lo tanto, dos descriptores basados en la intensidad CI-LBP y NI-LBP, y dos descriptores basados en diferencias RD-LBP y AD-LBP son combinados para formar histogramas que representen imágenes con textura.

El enfoque propuesto de este algoritmo es presentado en la figura 13:

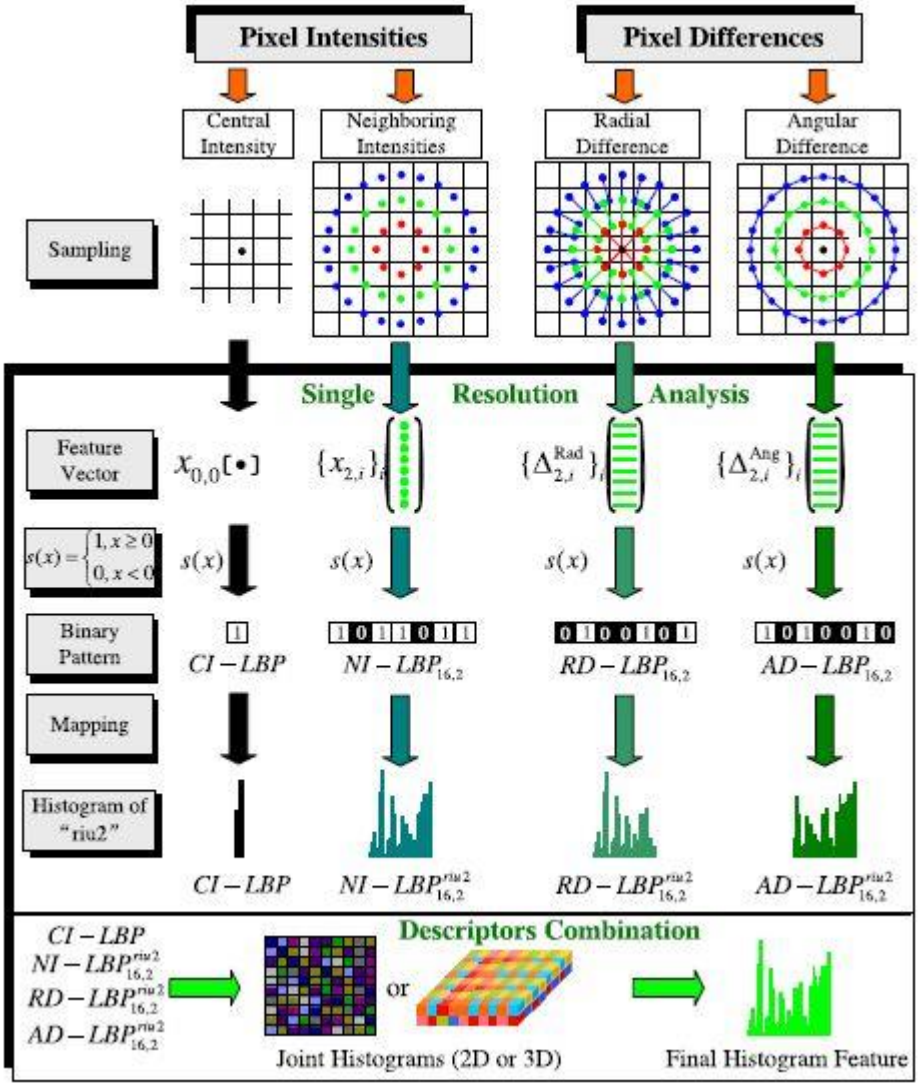


Figura 40: Enfoque LBP-Liu, origen: [12]

Figura 41: Imágenes con defectos tomadas con distintas cámaras. De izquierda a derecha: Visible, NIR, LWIR, UVFigura 42: Enfoque LBP-Liu, origen: [12]

3.3. Sistemas multiespectrales

Un sistema de visión artificial va más allá de la limitada capacidad humana, debido a que permite evaluar los procesos de forma objetiva a largo plazo o apreciar los acontecimientos fuera del espectro electromagnético visible. El uso del espectro infrarrojo cercano o ultravioleta hace posible explorar defectos en superficies que el ojo humano es incapaz de observar. Los sistemas multiespectrales proporcionan información sobre ciertos defectos que pueden ser percibidos sólo en determinadas longitudes de onda, permitiendo ser utilizados como una herramienta potente para desarrollar sistemas de visión por computador adaptados a objetivos particulares.

La figura muestra imágenes con defectos en la superficie tomadas con cámaras con distintas longitudes de ondas, donde es posible apreciar que en algunas el defecto se destaca más.

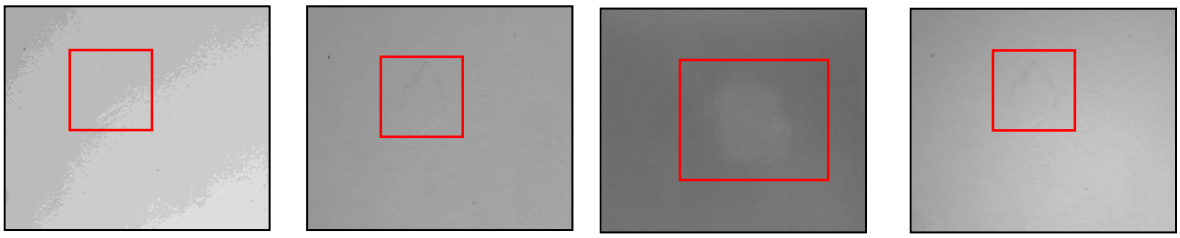


Figura 43: Imágenes con defectos tomadas con distintas cámaras. De izquierda a derecha: Visible, NIR, LWIR, UV

Figura 44: Método propuestoFigura 45: Imágenes con defectos tomadas con distintas cámaras. De izquierda a derecha: Visible, NIR, LWIR, UV

CAPÍTULO IV: DESARROLLO

4.1. Método propuesto

Como se ha mencionado anteriormente, el objetivo es buscar métodos eficaces que puedan detectar defectos en paneles de melamina. Distintas técnicas son implementadas en C# con el uso de librerías de OpenCV. El método propuesto para cumplir los objetivos es mostrado en el siguiente diagrama:

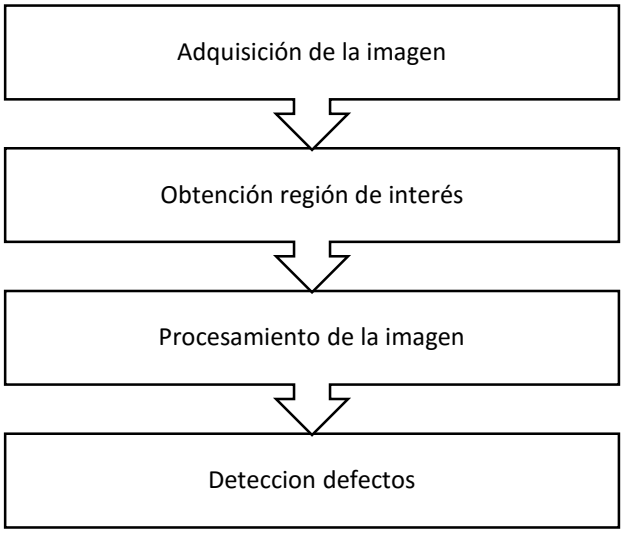


Figura 46: Método propuesto

Figura 47: Método propuesto

4.2. Adquisición de imágenes

Las imágenes fueron tomadas por distintas cámaras con las siguientes especificaciones:

Cámara	Resolución	Frame Rate
Basler acA645-100gm	659x494 px	100 fps
Basler acA2000-50gm NIR	2048x1088 px	50 fps
Jai RM-6740-GE UV	640x480 px	200fps

Tabla 3: Especificación de Cámaras

Muestras

Los paneles de madera están cubiertos con láminas de melamina blanca brillante, los cuales debido a errores de fabricación, presentan defectos en el acabado de la superficie. Los defectos en las muestras para el estudio son el **Papel Pegado** y el **Papel Corrido**, las siguientes imágenes muestran los defectos presentes en los paneles:

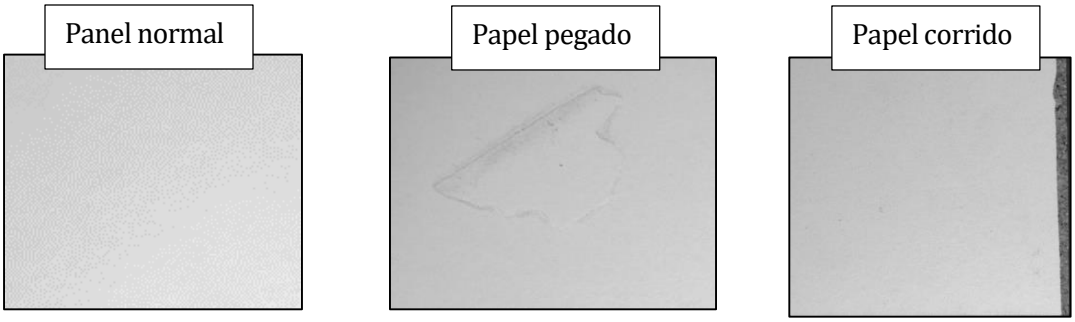
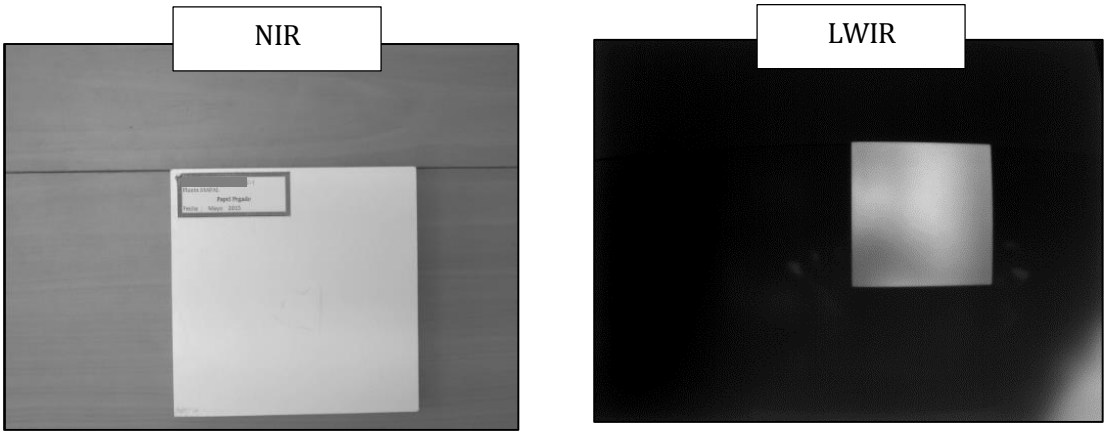


Figura 49: Ejemplo defectos

Figura 50: Ejemplo defectos

A continuación se muestran ejemplos de las imágenes utilizadas en el proyecto con papel pegado en las distintas bandas espectrales:



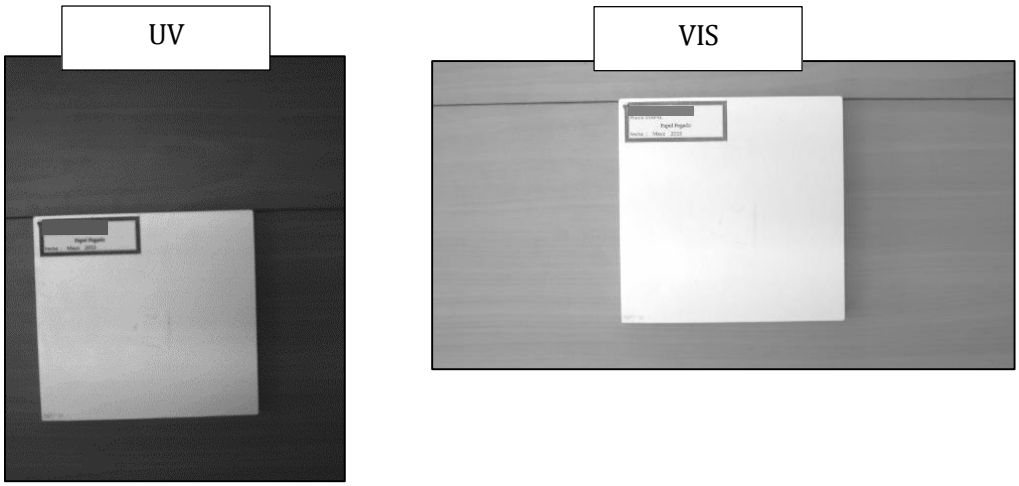


Figura 52: Ejemplo imágenes de muestra de paneles

Y el detalle de las imágenes disponibles queda descrito en la siguiente tabla:

Defecto	**Espectro**	**Tamaño**	**Cantidad imágenes**	**Formato**
Papel pegado	Visible	96x104	14	tiff
	NIR	216x234	14	tiff
	UV	96x104	14	tiff
	LWIR	174x174	14	png
Papel corrido	Visible	96x104	20	tiff
	NIR	216x234	20	tiff
	UV	96x104	20	tiff
	LWIR	174x174	20	png

Tabla 4: Resumen muestras

4.3. Obtención región de interés de la muestra

En este caso, todas las imágenes proporcionadas contienen un logo de etiquetado, por lo tanto no es posible aplicar directamente el algoritmo de detección, esto origina dos vías posibles para extraer un área de interés:

a) Extracción manual del área de interés
   Consiste en utilizar una herramienta de libre distribución para recortar el área con el defecto presente, extrayendo de esta manera el fondo negro y la etiqueta.

b) Extracción automática del área de interés
   Consiste en aplicar algoritmos que permitan, dada la imagen seleccionada, eliminar el fondo y el rectángulo que contiene la identificación del panel.

   Para el caso de extracción automática, en primer lugar se obtiene la representación en escala de grises de la imagen mediante la extracción del canal V (v*alue)* del modelo HSV, luego se procede a realizar una binarización. Esto es útil para el recorte de fondo, ya que se distinguen principalmente los rectángulos que componen la imagen, perdiendo muy poca información relevante y simplificando la representación de cada *píxel*.

Figura 55: Imagen binarizada de muestra de paneles

Luego, utilizando un algoritmo de detección de contornos, se identifica el borde externo del panel y el logo en la parte superior.

El algoritmo utilizado es el siguiente:

```
Para cada píxel en la imagen binarizada
{
Si el píxel pertenece a un contorno:
   Buscar en sus vecinos si también pertenecen a contorno
     Repetir la búsqueda hasta encontrar el píxel inicial
   Si no se encuentra, descartar los píxeles encontrados y seguir con el
   siguiente píxel de la imagen binarizada
Si no seguir con el siguiente píxel de la imagen binarizada
}
Para cada contorno cerrado encontrado:
{
   Descartar los objetos muy pequeños (ruido)
}
```

De esta forma es posible encontrar dos contornos, y utilizando sus dimensiones y coordenadas se crea un rectángulo que es copiado sobre la imagen original, eliminado de esta forma gran parte de información innecesaria para el posterior análisis.

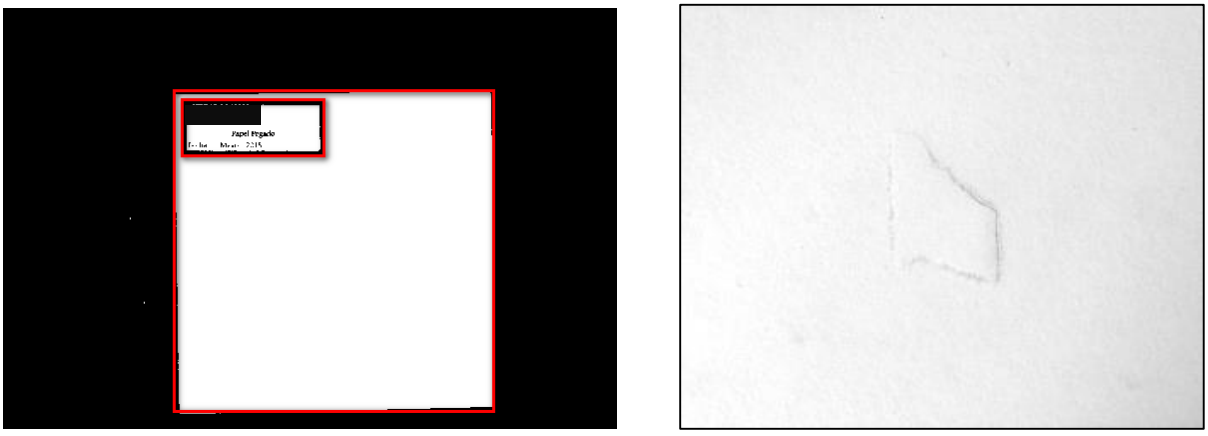


Figura 58: Izquierda: Imagen binarizada con detección de contornos, Derecha: ROI imagen original

4.4. Procesamiento de la imagen

A partir de la extracción del área de interés descrita en el punto anterior, se aplican diferentes algoritmos que permitan resaltar las características deseadas o disminuir el brillo de la muestra. Las siguientes muestras de papel pegado tienen algoritmos aplicados en forma separada sobre una imagen NIR.

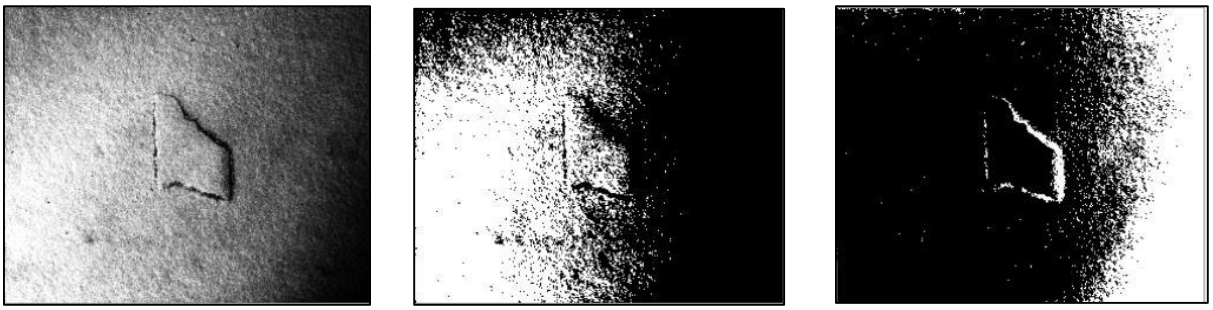


Figura 61: De izquierda a derecha: imagen ecualizada, método Otsu, umbralización manual con límites 31-232

Figura 64: De izquierda a derecha: Filtro Sobel, Mediana+Sobel, Suavizado gaussiano+Sobel

4.5. Detección de Defectos

En las imágenes procesadas se deben detectar cambios significativos que permitan establecer si son defectos. Para esto es necesario describir la textura y de acuerdo a la revisión de la literatura existen algoritmos que permiten identificar patrones de textura. En el punto 3.2 se describe la teoría de los algoritmos utilizados: SIFT y LBP-Liu. A continuación, se describe en pseudocódigo la implementación realizada de estos algoritmos en la aplicación desarrollada.

4.5.1 Algoritmos de detección

Algoritmo SIFT

Estructura:

• Detección de escala-espacio (DEE): Se emplea la función de diferencias gaussianas para identificar los puntos de interés que son invariantes a escala y orientación. Así busca en todas las escalas y localizaciones de la imagen.

• La localización del "punto clave" (KeyPoint): En cada localización candidata se adecua un modelo cuadrático detallado para determinar la localización y la escala. Los puntos claves se seleccionan basándose en su estabilidad.

• Asignación de orientación (AO): Una o más orientaciones se asignan a cada localización de los puntos claves, basándose en la dirección del gradiente de la imagen. Todas las futuras operaciones se ejecutan con los datos de la imagen que ha sido transformada de acuerdo con la orientación, escala y localización asignada para cada característica, de este modo se proporciona invariancia a estas transformaciones.

• Descriptor del "punto clave" (*KeyPoint*): Los gradientes de la imagen son medidos en la escala seleccionada en la región alrededor de cada punto clave. Estos son transformados a una representación que permite, para niveles significativos, distorsión de la forma local y cambios en la iluminación.

Parámetros de entrada:

• OctaveLayers: 3 , el número de octavas

• contrastThreshold: 0.04, umbral máximo, filtra extremos muy pequeños del espacio-escala de la diferencia gaussiana(DoG)

• edgeThreshold: 10.0 umbral borde, elimina extremos del espacio-escala de DoG cuya curvatura es demasiado pequeña(tales extremos producen tramas mal localizadas)

• sigma: 1.6 escala base

*/*Los valores asignados son los sugeridos descritos en* [17]**/*

Detector SIFT:

1. Se crea un detector Sf con (nOctaveLayers, contrastThreshold, edgeThreshold, sigma)
2. Se detectan los KeyPoints(KP) de la imagen: Sf(ImagenEntrada)

   Por cada KP detectado se almacena Angulo: orientación del KP
   Point: posición del KP
   Size: tamaño del KP

3. Se guardan los KeyPoints en un vector mKeyPoint.
4. Se calculan los descriptores Sf(imagenEntrada, mKeyPoint)

Algoritmo LBP-Liu

Parámetros de entrada: imagen,radio

Por cada *pixel* de la imagen

{

1. Función ptLBP: guarda en un vector los elementos que se encuentran alrededor del píxel en un radio determinado.
   Entrada: imagen, x,y, radio
   Salida: vectorP con elementos alrededor del pixel

2. Función ciLBP: calcula la intensidad del píxel
   Entrada:imagen, x, y, media del vectorP
   Salida:intensidad del píxel

3. Función descLBP: calcula intensidad de los pixeles vecinos
   Entrada: imagen,x,y, vectorP¨, media vectorP
   Salida: patrón binario diferencia vecindad

4. Función rdif: calcula la diferencia radial
   Entrada: imagen, x,y, vectorP, radio
   Salida:patrón binario diferencia radial

5. Función adif: calcula la diferencia angular
   Entrada: vectorP, radio, delta
   Salida:patrón binario diferencia angular

6. Función riu2: fusiona los patrones no uniformes
   Entrada: patrón binario
   Salida: patrón disminuido

}

Se concatena un histograma con las salidas de la función riu2 del pixel central (ciLBP), vecindad (descLBP), radial(rdif) y angular(adif).

/* Las fórmulas utilizadas dentro de cada función de LBP-Liu se encuentran descritas en el *paper* original [12]. */

4.5.2 Clasificación

Una vez extraídas las características de las imágenes, se utiliza una máquina de soporte vectorial (SVM) proporcionada por la librería EmguCV que permite evaluar los algoritmos LBP-Liu y SIFT como descriptores de defectos. Es importante destacar que no se realiza reconocimiento del tipo de defecto, solo su clasificación, correspondiendo a las clases "sin defecto" o "con defecto".

El procedimiento consta de dos fases:

1. Entrenamiento del clasificador:

    a) Se crean dos conjuntos de imágenes: con defecto y sin defecto (imágenes de entrenamiento).
    b) A cada conjunto se le aplica el algoritmo detector (LBP-Liu o *SIFT) y el histograma resultante de cada imagen es almacenado en una lista.
    c) La lista anterior es ingresada en SVM como muestras de entrenamiento para crear un modelo con dos clases.

2. Validación:

    a) Se crean dos conjuntos nuevos de imágenes: con defecto y sin defecto (imágenes de prueba).
    b) A cada conjunto se le aplica el algoritmo detector (LBP-Liu o SIFT) y el histograma resultante de cada imagen es almacenado en una lista_nueva.
    c) Se compara cada elemento de la lista_nueva con el modelo generado previamente.
    d) Se evalúa el resultado, si la muestra ingresada es de la clase con defecto o sin defecto.

* En el caso de SIFT, entrega una cantidad de puntos claves distintos en cada imagen, los cuales no son posibles de ingresar directamente a SVM. Esto requiere la utilización de una técnica denominada "*Bag of Visual Words*" [32], la cual crea un vocabulario de características y mediante el agrupamiento de estas es posible extraer histogramas que caractericen la imagen, los que posteriormente son entrenados y probados en SVM.

El procedimiento para "*Bag of Visual Words*" aplicado a SIFT es el siguiente:

1. Obtención de un conjunto de *bag of features*
   a) Se selecciona un conjunto de imágenes.
   b) Se aplica SIFT y se extraen los *keypoints* de todas las imágenes.
   c) Cada conjunto de *keypoints* extraído es convertido en un descriptor de la imagen.
   d) Se agrupa el conjunto de descriptores para una cantidad de *bag* definida(200)
   e) Se entrenan las 200 *bag* con los descriptores creados en c)
   f) Se obtiene un vocabulario de características (de tamaño 200).

2. Obtención de un Descriptor BOW para una imagen
   a) Se extraen los *keypoints* de una imagen dada.
   b) Se obtiene el descriptor SIFT para cada *keypoint*.
   c) Se compara el descriptor con el vocabulario creado en el paso 1.
   d) Se construye un histograma de la imagen.

También es necesario analizar el rendimiento de cada algoritmo de detección (SIFT y LBP-Liu). Para esto se utiliza el conjunto de imágenes de entrenamiento del clasificador. Y para hacer equitativo el rendimiento, se inicia el temporizador cuando una imagen comienza a ser intervenida por el algoritmo y termina cuando se obtiene un vector descriptor de la imagen, es decir, en el caso de SIFT además se incluye el tiempo de *Bag of Visual Words,* así es posible medir el tiempo de ejecución en entregar para cada imagen un vector de características de largo *n.*

De esta manera, es posible contar con las métricas que permitan comparar los algoritmos de acuerdo a su rendimiento y efectividad.

CAPÍTULO V: EXPERIMENTOS

5.1. Ambiente de pruebas

Los experimentos fueron realizados en un computador con las siguientes características:

- Procesador Intel Core i-3, 2.4 Ghz
- Memoria Ram 4GB
- Tarjeta gráfica integrada Intel HD Graphics4000

5.2. Experimentos realizados.

5.2.1 Procesamiento de la imagen

Para resaltar defectos de **papel pegado** en las imágenes, se aplicaron diferentes combinaciones por ejemplo el ecualizador de histograma en imágenes VIS y NIR

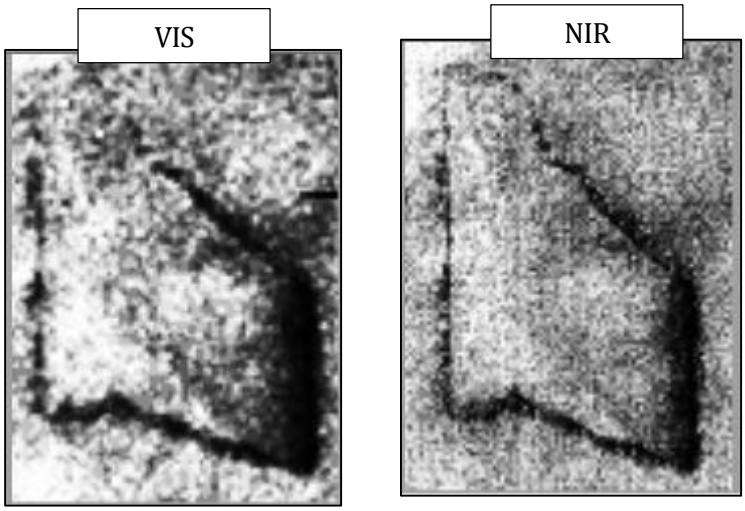


Figura 67: imágenes ecualizadas de muestras con papel pegado

A modo de experimentación también se decidió ponderar los componentes de ambas imágenes, para lo cual se redimensiona la imagen de mayor tamaño a la de menor por medio de interpolación lineal, la cual es menos rápida en comparación a la interpolación cúbica y la de

vecinos cercanos, pero se pierde menos información, ya que aplica un filtro de paso bajo con el fin de lograr un equilibrio entre la calidad visual y la eliminación de bordes [20].

Luego estas imágenes del mismo tamaño son ponderadas con un peso de 0.5 cada una.

Figura 70: Suma ponderada de imágenes ecualizadas VIS+NIR

También se probó la combinación de filtro de la mediana y luego filtro Sobel.

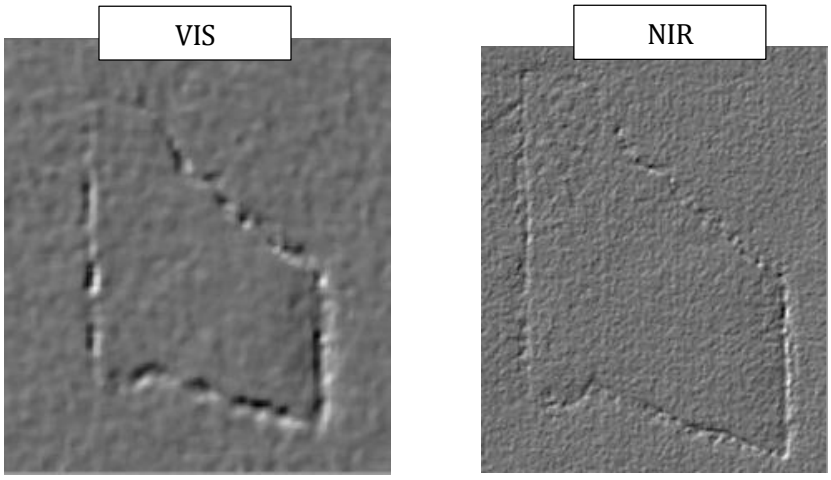


Figura 73: Filtro mediana + Sobel en muestras de papel pegado

En el caso de las muestras de **papel corrido**, igualmente se probó la combinación de Sobel, pero con suavizado Gaussiano previamente.

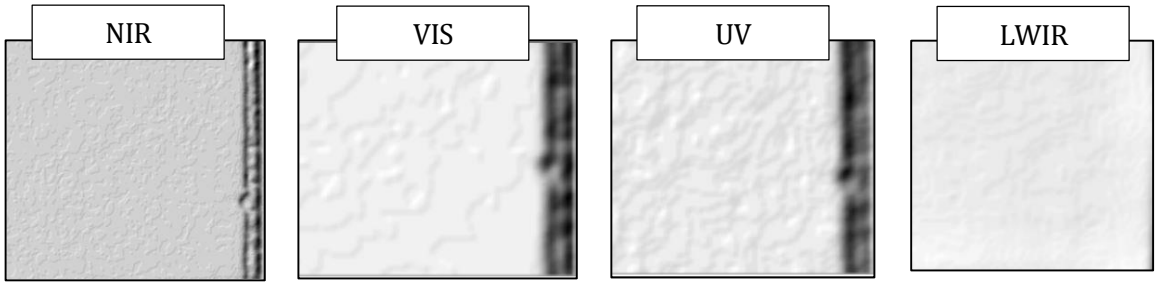


Figura 76: Suavizado Gaussiano + Filtro Sobel en muestras de papel corrido

5.2.2 Detección Defectos

5.2.1.1 Algoritmo SIFT

Aplicando el algoritmo de detección sobre el conjunto de imágenes de **papel corrido** procesada previamente con suavizado gaussiano y filtro Sobel, se obtienen *KeyPoints* como se muestra a continuación:

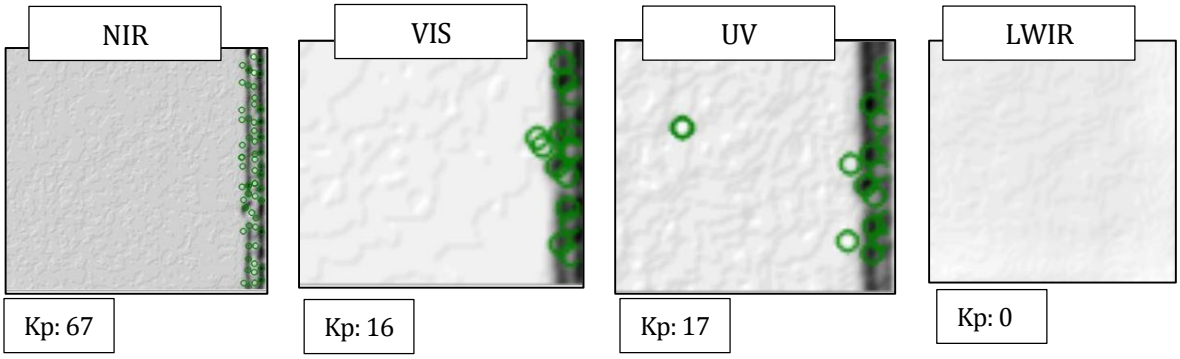


Figura 79: SIFT sobre papel corrido

También se realizaron diversas pruebas con muestras de **papel pegado** aplicando ecualizador de histograma+ filtro de la mediana. El tamaño de cada una es de 85x117.

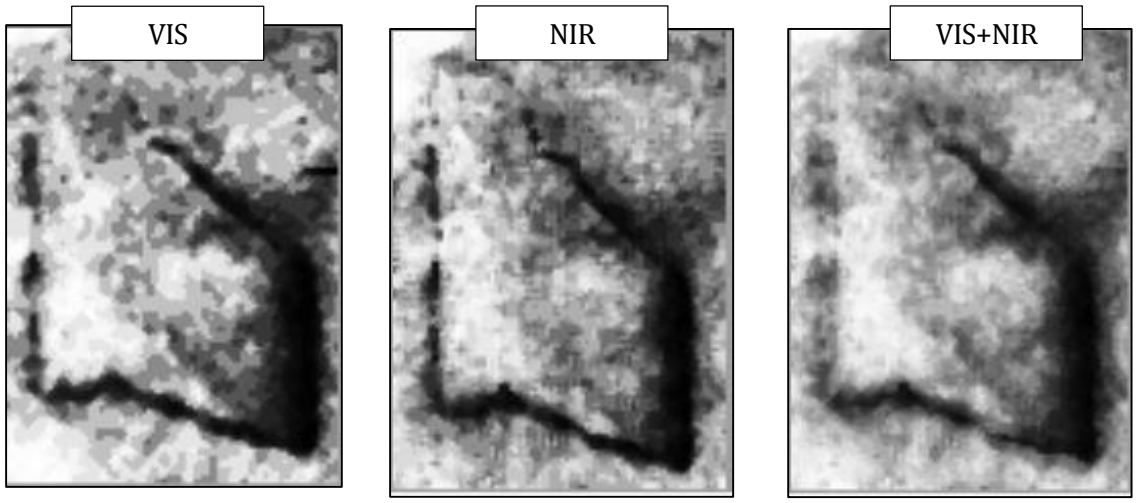


Figura 82: Imágenes ecualizadas y filtradas

Luego a estas imágenes se les aplicó el algoritmo SIFT

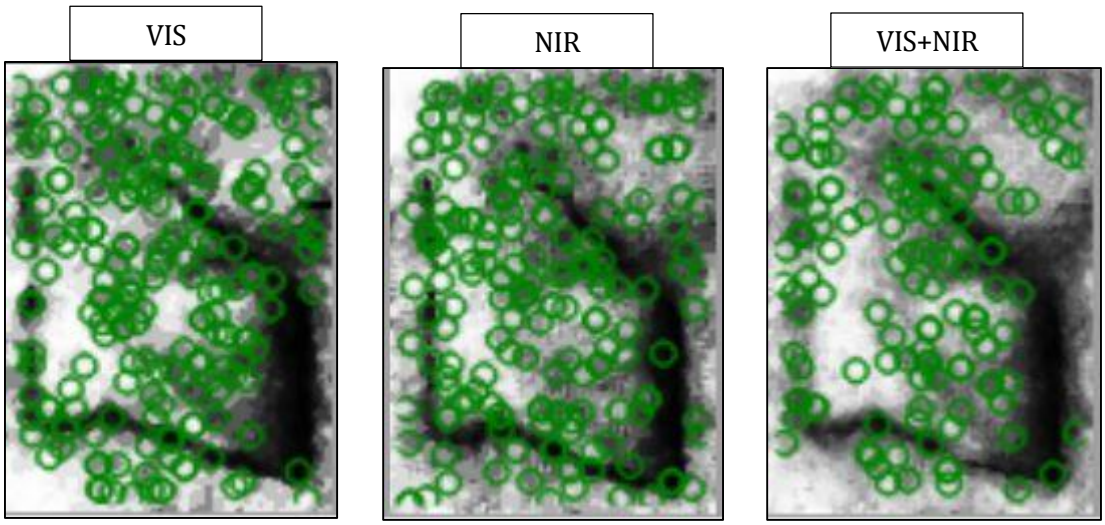


Figura 85: KeyPoints en muestras de papel pegado

En el caso de imagen VIS se detectaron 208 puntos clave, con la imagen NIR 184, y con la imagen sumada 143.

También se hizo la prueba de detección sobre las imágenes VIS con filtro de la mediana + sobel y suavizado gaussiano + sobel arrojando 85 y 76 *Keypoints* respectivamente.

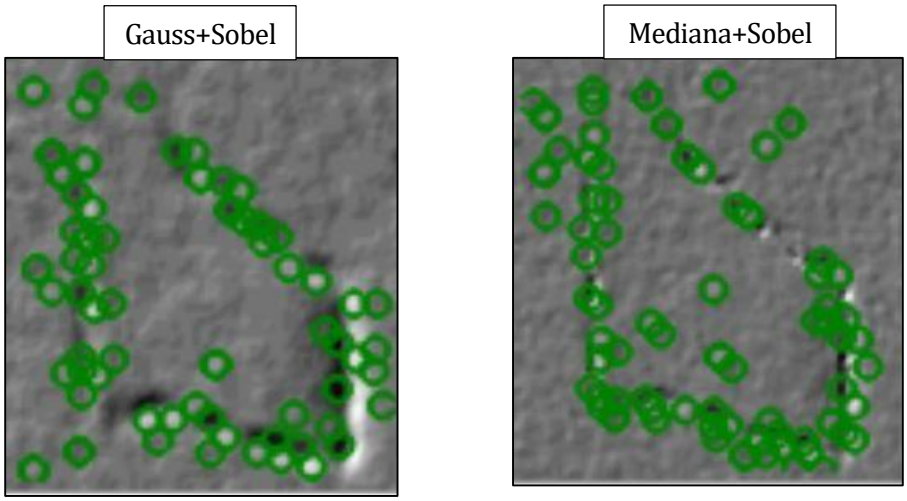


Figura 88: KeyPoints en muestras de papel pegado 2

De acuerdo a los experimentos realizados, la aplicación de suavizado gaussiano+ sobel como pre-procesamiento de la imagen para resaltar los defectos entrega un mejor efecto sobre ella, destacando el área de estudio y disminuyendo el ruido presente.

A partir de los experimentos realizados y para comparar resultados del funcionamiento de los algoritmos SIFT y LBP-Liu se utilizarán muestras en el espectro Visible e imágenes multiespectrales (VI+UV+NIR+LWIR), estas imágenes serán procesadas separadamente con:

- suavizado gaussiano+ sobel
- sobel
- ecualizador

5.2.1.2 Algoritmo LBP-Liu

En el caso de este algoritmo, a cada imagen de entrada se calcula los patrones locales de acuerdo a un radio dado y las salidas (histogramas) son almacenadas en vectores para su posterior evaluación en el clasificador SVM. Además se utilizan las clases *DenseHistogram* e *HistogramBox* de EmguCV para crear el histograma y mostrarlo gráficamente.

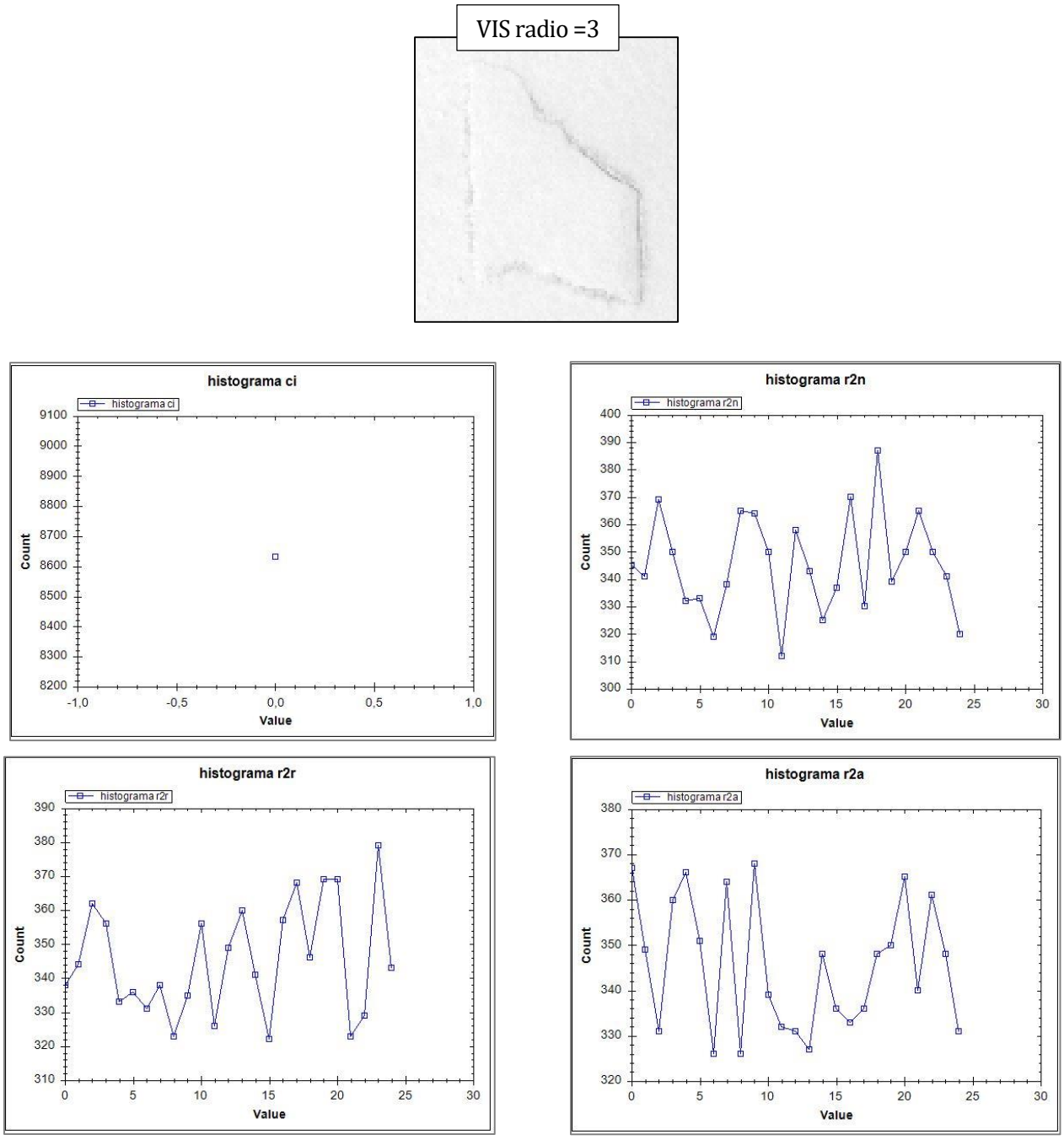


Figura 91: Histogramas LBP-Liu para una imagen VIS

También se realizaron experimentos sobre la imagen aplicando previamente suavizado gaussiano y filtro sobel para contrastar los histogramas resultantes y visualizar el funcionamiento del algoritmo.

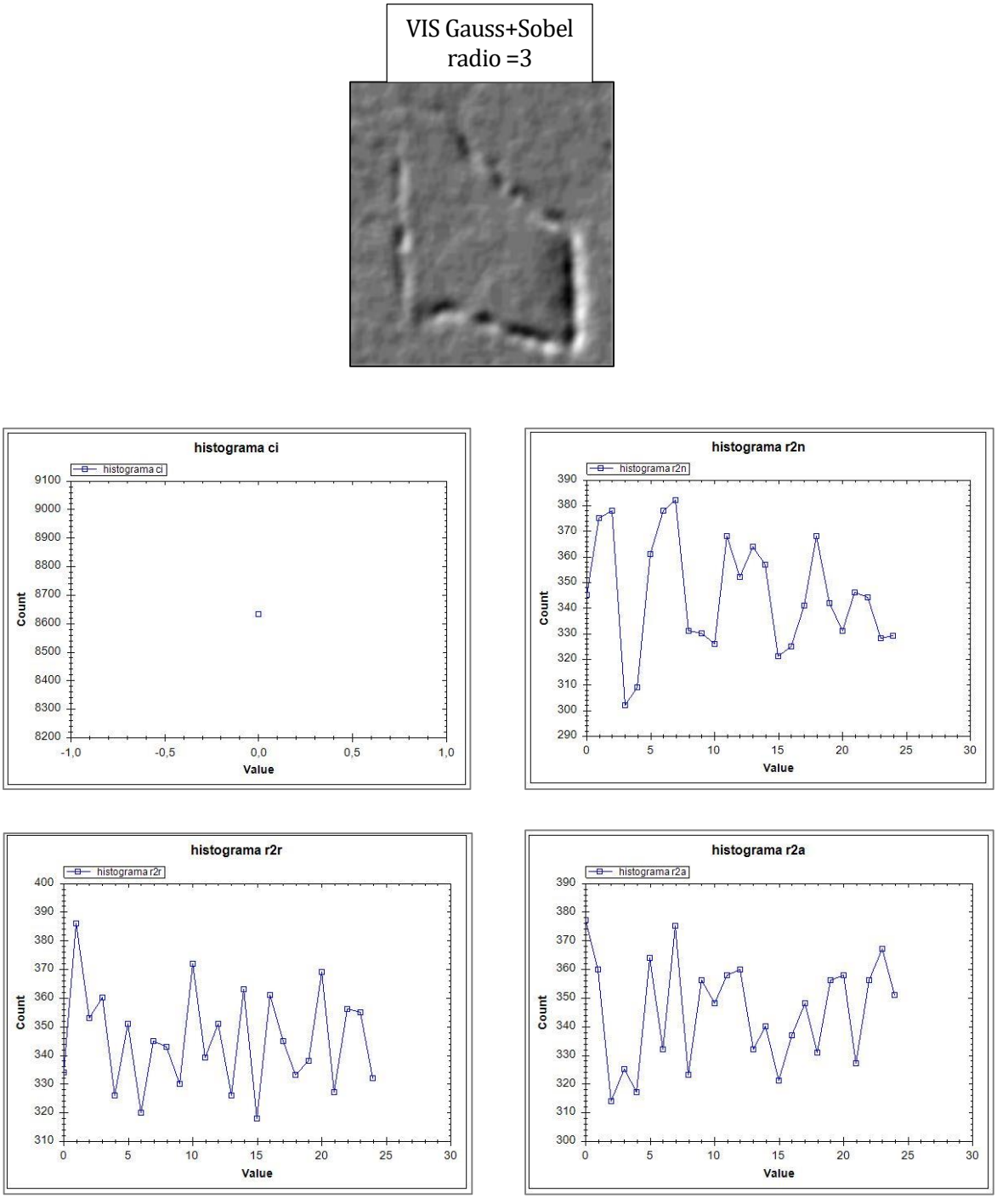


Figura 94: Histogramas LBP-Liu para una imagen VIS con filtro

5.2.3 Clasificación

Para crear un modelo SVM es necesario contar con una cantidad significativa de muestras de entrenamiento, por lo tanto se crea un conjunto de imágenes positivas (que contiene defecto) y un conjunto de imágenes negativas (sin defecto). Estos conjuntos de imágenes se crean seleccionando el área de defecto de la muestra y cambiando su ángulo o aplicando ruido. Donde no contiene defecto se aplica el mismo procedimiento y se crean las muestras negativas. De esta manera se cuenta con conjuntos de 65 muestras negativas y 65 positivas de **papel pegado** en el espectro VIS, UV, NIR y LWIR.

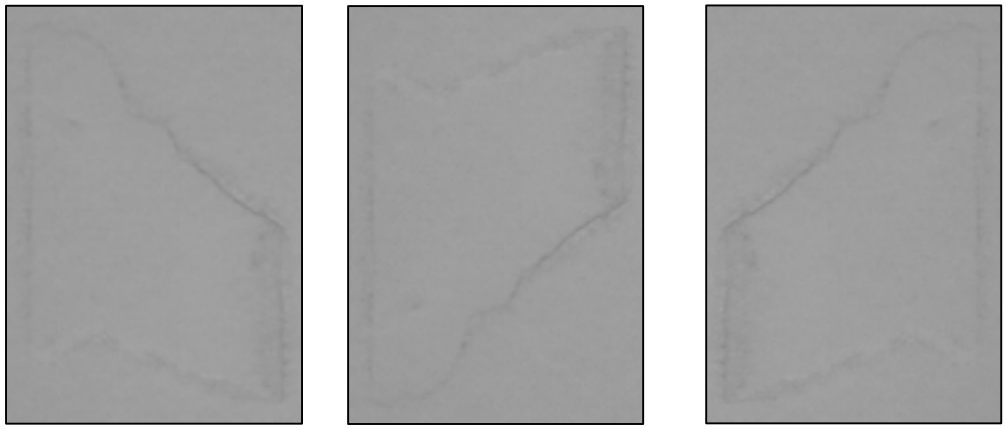


Figura 97: Muestras Positivas de papel pegado

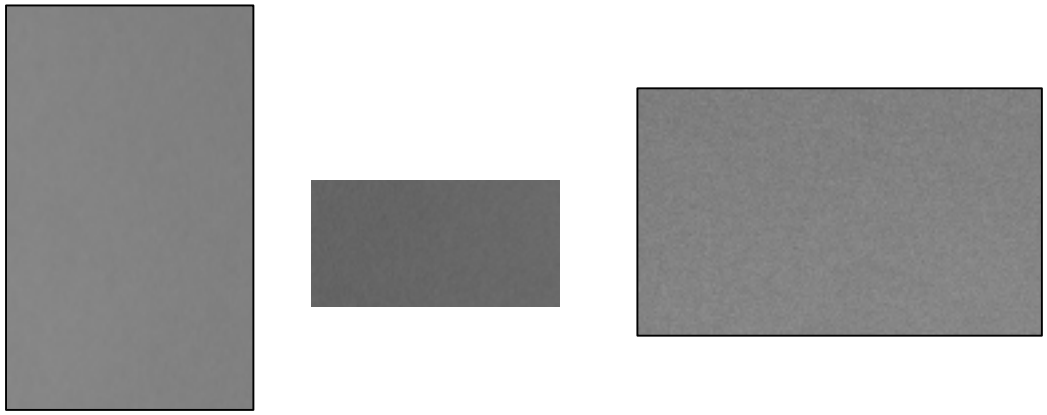


Figura 100: Muestras Negativas de papel pegado

Además es necesario contar con muestras de prueba, las cuales tienen que ser diferentes a las del entrenamiento previo, Por lo tanto se crean dos conjuntos de imágenes nuevas (50 positivas y 50 negativas).

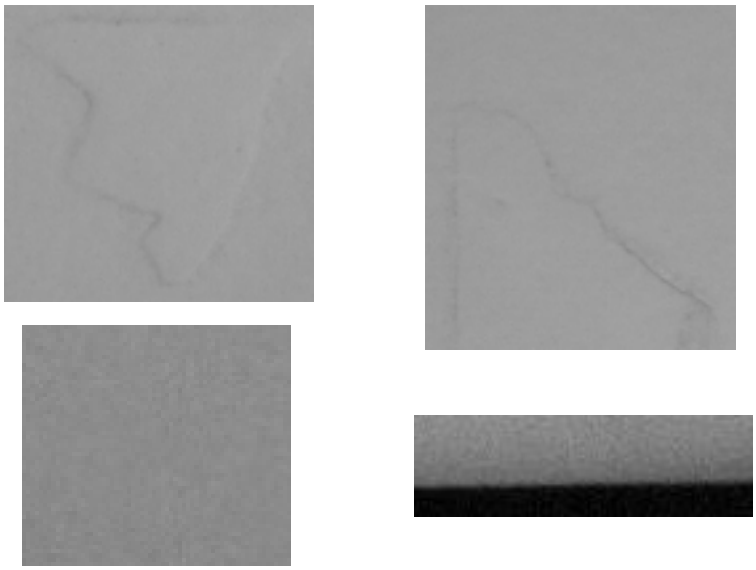


Figura 103: Imágenes de prueba SVM. Arriba con defecto, abajo muestras sin defectos.

Luego de entrenar el clasificador con las 130 muestras y para evaluar la efectividad del descriptor LBP-Liu, se prueban 2 conjuntos de 100 imágenes, uno en el espectro visible y otro con la combinación de las bandas visible(VIS), ultravioleta(UV), infrarojo cercano(NIR) e infrarojo lejano(LWIR). El resultado se registra en una matriz de confusión para el posterior cálculo de *Precision, Recall* y *F1 Score*. Este procedimiento se efectúa con imágenes filtradas (suavizado gaussiano + sobel, sobel, ecualizador) y no filtradas.

En el caso de SIFT entrega puntos claves que no es posible ingresar directamente a SVM, así que se crea un modelo "*Bag of Visual Words*" con 100 muestras de entrenamiento, luego se prueba un conjunto diferente de 100 imágenes. Esto es realizado solamente con imágenes filtradas (suavizado gaussiano + sobel y sobel), ya que al no existir este pre-procesamiento el algoritmo no detecta *keypoints* en la mayoría de los casos.

5.3. Resultados

5.3.1 Tiempos de ejecución

A continuación se resume el tiempo promedio en que cada algoritmo se ejecuta sobre un conjunto de 130 imágenes de muestras de **papel pegado** procesadas previamente con filtros. Los conjuntos pertenecen al espectro visible e imágenes multiespectrales (VI+UV+NIR+LWIR).

Tipo de Imagen	Promedio de tSobel	Promedio de tgauss+Sobel	Promedio de tEcualiz
Visible	501,49	645,88	966,13
Multiespectral	639,76	851,78	1009,12

Tabla 5: Resumen tiempo SIFT papel pegado

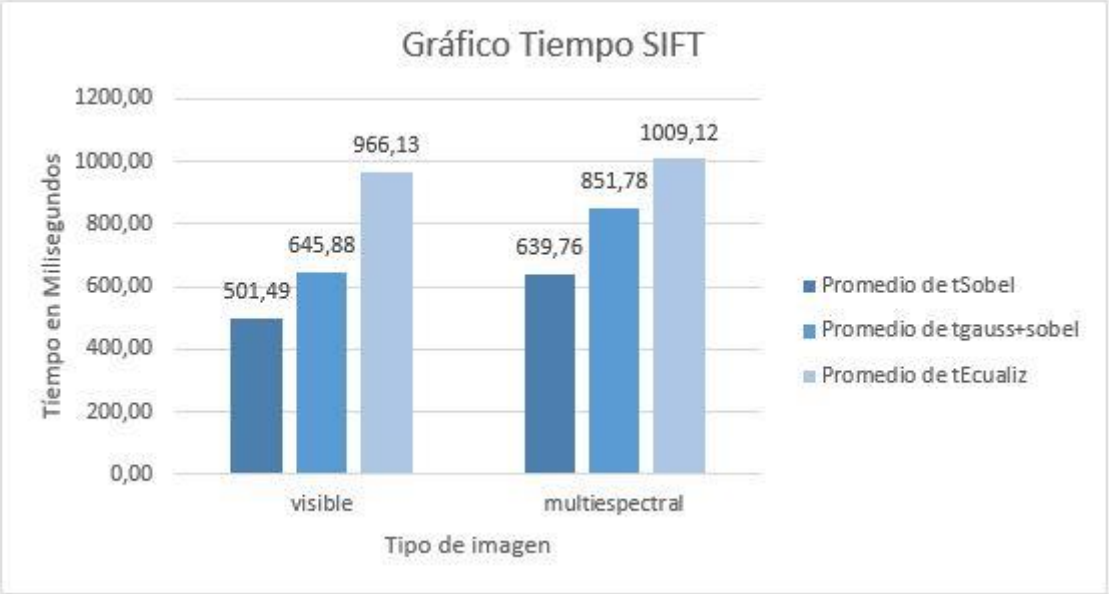


Figura 106: Gráfico Tiempo SIFT Papel pegado

El resultado anterior muestra que el algoritmo SIFT tiene mayores tiempos de ejecución sobre las imágenes multiespectrales. Cabe recordar que el tiempo se tomó desde que el algoritmo toma la imagen y entrega un descriptor de tamaño *n.* Por este motivo es que se incluye la construcción del *Bag of Visual Words* dentro de su tiempo de ejecución.

Para el caso de LBP-Liu, los tiempos de ejecución son menores que los de SIFT, en una proporción aproximada de 2:1 milisegundos en el espectro visible y 3:1 en imágenes multiespectrales. Y se resume a continuación:

Tipo de Imagen	Promedio de tSobel	Promedio de Sin procesamiento	Promedio de tEcualiz
Visible	212,71	196,08	203,26
Multiespectral	222,36	208,75	237,68

Tabla 6: Resumen tiempo LBP-Liu papel pegado

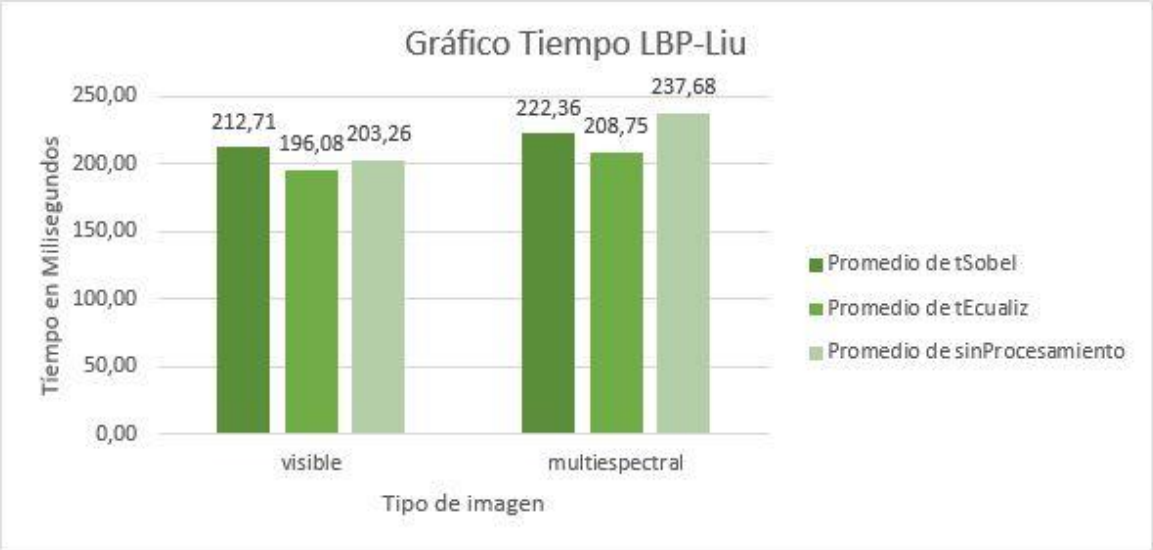


Figura 109: Gráfico Tiempo LBP-Liu Papel pegado

De los resultados anteriores es posible concluir que bajo las mismas condiciones, el algoritmo LBP-Liu tiene menores tiempos de ejecución como algoritmo extracto de características en cada uno de los conjuntos de imágenes. En relación a los espectros, ambos algoritmos presentan mejores tiempos tratándose de imágenes en una banda.

5.3.2 Matriz de confusión

A continuación se presentan los resultados de la clasificación descrita en la sección 5.2.3 utilizando imágenes con **papel pegado** en el espectro visible (VI) e imágenes multiespectrales combinadas con las cuatro bandas(VI+UV+NIR+LWIR) de peso 0,25 cada una. Además se calculan las métricas *Recall, Precision* y *F1 Score* cuya formulación matemática está definida en la sección 2.2.1.

LBP-Liu

En este caso se realizaron experimentos con procesamiento (Sobel, Ecualizador) y sin procesamiento.

Imagen en espectro Visible, Cantidad muestras = 100

Sin procesamiento		**Clasificación**	
		Con Defectos	**Sin Defectos**
Predicción	**True**	49	48
	False	2	1

Tabla 7: matriz de confusión LBP-Liu (espectro visible sin procesamiento)

Recall	98 %
Precision	96,08 %
F1 Score	97,03%

Sobel		**Clasificación**	
		Con Defectos	**Sin Defectos**
Predicción	**True**	40	45
	False	5	10

Tabla 8: matriz de confusión LBP-Liu (espectro visible y Sobel)

Recall	80 %
Precision	88,8 %
F1 Score	84,2%

Ecualizador		**Clasificación**	
		Con Defectos	**Sin Defectos**
Predicción	**True**	39	45
	False	5	11

Tabla 9: matriz de confusión LBP-Liu (espectro visible y Ecualizador)

Recall	78 %
Precision	88,6 %
F1 Score	82,9%

Imagen combinada con 4 bandas espectrales de peso 0.25 cada una: VIS, UV, NIR, LWIR
Cantidad muestras = 100

Sin procesamiento		Clasificación	
		Con Defectos	**Sin Defectos**
Predicción	**True**	44	47
	False	3	6

Recall	88 %
Precision	93,6 %
F1 Score	90,7%

Tabla 10: matriz de confusión LBP-Liu (multiespectral sin procesamiento)

Sobel		Clasificación	
		Con Defectos	**Sin Defectos**
Predicción	**True**	38	46
	False	4	12

Recall	76 %
Precision	90,4 %
F1 Score	82,6%

Tabla 11: matriz de confusión LBP-Liu (multiespectral y Sobel)

Ecualizador		Clasificación	
		Con Defectos	**Sin Defectos**
Predicción	**True**	43	46
	False	4	7

Recall	86 %
Precision	91,4 %
F1 Score	88,6%

Tabla 12: matriz de confusión LBP-Liu (multiespectral y Ecualizador)

SIFT

De acuerdo a diversos experimentos, el algoritmo SIFT no detecta *keypoints* sin procesamiento previo por este motivo se presentan estos resultados: con suavizado gaussiano+Sobel, Sobel y Ecualizador.

Imagen en espectro Visible
Cantidad muestras = 100

Sobel		Clasificación	
		Con Defectos	**Sin Defectos**
Predicción	**True**	42	48
	False	2	8

Recall 84 %
Precision 95,4 %
F1 Score 89,3%

Tabla 13: matriz de confusión SIFT (visible y Sobel)

sGaussiano+Sobel		Clasificación	
		Con Defectos	**Sin Defectos**
Predicción	**True**	47	47
	False	3	3

Recall 94 %
Precision 94 %
F1 Score 94%

Tabla 14: matriz de confusión SIFT (visible y gaussiano+Sobel)

Ecualizador		Clasificación	
		Con Defectos	**Sin Defectos**
Predicción	**True**	49	48
	False	2	1

Recall 98 %
Precision 96,08 %
F1 Score 97,03%

Tabla 15: matriz de confusión SIFT (visible y ecualizador)

Imagen combinada con 4 bandas espectrales de peso 0.25 cada una: VIS, UV, NIR, LWIR
Cantidad muestras = 100

Sobel		Clasificación	
		Con Defectos	Sin Defectos
Predicción	True	48	41
	False	9	2

Tabla 16: matriz de confusión SIFT (multiespectral y Sobel)

Recall	96 %
Precision	84,2 %
F1 Score	89,7%

sGaussiano+Sobel		Clasificación	
		Con Defectos	Sin Defectos
Predicción	True	45	44
	False	6	5

Tabla 17: matriz de confusión SIFT (multiespectral y gaussiano+Sobel)

Recall	90 %
Precision	88,2 %
F1 Score	89,1%

Ecualizador		Clasificación	
		Con Defectos	Sin Defectos
Predicción	True	50	49
	False	1	0

Tabla 18: matriz de confusión SIFT (multiespectral y ecualizador)

Recall	100 %
Precision	98 %
F1 Score	99%

5.4. Comentarios

Siguiendo el método propuesto en este trabajo, el algoritmo SIFT entrega mejores resultados ya que detecta mayor cantidad de defectos sobre la muestra tanto en el espectro visible como en la imagen multiespectral, en esta última entrega una media armónica entre exhaustividad y precisión del 87,7% para el caso de imágenes procesadas con Sobel y 99% para ecualizador de histograma. En el caso de LBP-Liu las mismas imágenes multiespectrales y con sobel un valor de *F1 score* del 82,6% y con ecualizador un 88,6.

Cabe destacar la importancia de este valor de exhaustividad (recall), ya que indica la información relevante que ha sido extraída, dando mayor importancia a los casos positivos donde realmente existen defectos.

CONCLUSIONES

En relación al control de calidad en la industria, la detección de anomalías o defectos, es un gran desafío en los problemas de visión por computador, especialmente si la correcta detección para su posterior clasificación de defectos incide en la determinación de la calidad del panel y a su vez la calidad incide en el precio.

El presente trabajo entrega unas primeras aproximaciones en la detección de defectos en melamina. Se ha revisado la literatura del área y se han implementado dos algoritmos del estado del arte para detección de cambios de textura y la extracción de las características correspondientes, SIFT y LBP-Liu, para poder entrenar clasificadores a partir de estas características. Una primera evaluación realizada muestra un adecuado rendimiento en clasificación de imágenes con y sin defectos, a partir de las características obtenidas por estos algoritmos, los cuales podrían ser mejorados con el uso de algoritmos de clasificación más avanzados, ámbito que escapa del alcance de este Proyecto de Título. Cabe destacar que este trabajo forma parte de un proyecto mayor enfocado en la detección de defectos a través de imágenes multiespectrales, las cuales actualmente están en proceso de mejoramiento de iluminación del espacio muestral, lo cual podría incidir a futuro en una mejora de los resultados obtenidos por los algoritmos. Esto mismo ha motivado el desarrollo de una aplicación sencilla y con interfaz intuitiva para ajustar su funcionamiento de acuerdo a los requerimientos que vayan surgiendo.

Con todo lo expuesto se cumple el primer objetivo de estudio de algoritmos de detección (capítulo III) los cuales fueron implementados en C# desarrollándose una aplicación que está descrita en los anexos de este informe con su respectivo manual de usuario, cumpliéndose de esta forma el segundo objetivo del proyecto, y por último la evaluación de los algoritmos, objetivo 3, está descrita en la sección 5.3 con un apartado de discusión en la sección 5.4.

Finalmente se sugiere realizar un estudio con una mayor cantidad de imágenes de entrenamiento, más preciso y exhaustivo, para lo cual es necesario la implementación de un clasificador que permita determinar si encuentra el defecto en otras superficies y si corresponde, por ejemplo a papel pegado y no a una mancha de suciedad o de otro elemento. También es necesario contar con un mayor número de imágenes de muestras, puesto que para confiar en las salidas del clasificador es necesario una base de datos con una cantidad mínima de 200 imágenes de prueba.

BIBLIOGRAFÍA Y REFERENCIAS

[1] A. De la Escalera, "Preprocesamiento de imágenes," in *Visión por computador, fundamentos y métodos*, Prentice Hall, 2001, pp. 112–154.

[2] C. Aguilera, F. Barrera, F. Lumbreras, A. D. Sappa, and R. Toledo, "Multispectral Image Feature Points," *Sensors*, vol. 12, no. 12, pp. 12661–12672, 2012.

[3] H. Liu, K. I. Edge, and D. Clifford, "The Multispectral Image Edge Detection Based on Clifford Gradient," 2011.

[4] D. García, *Visión Artificial y Procesamiento digital de imágenes usando Matlab*. Ibarra: Pontificia Universidad católica del Ecuador, 2008.

[5] J. . Báez Rojas and M. . Alonso Pérez, "Uso del sistema HSI para asignar falso color a objetos en imágenes digitales," *Rev. Mex. física E*, vol. 54, no. 2, pp. 186–192, 2008.

[6] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst. Man. Cybern.*, vol. 9, no. 1, pp. 62–66, 1979.

[7] M. Iyer and S. Janakiraman, "Defect Detection in Pattern Texture Analysis," pp. 172–175, 2014.

[8] J. Esqueda, "Fundamentos de Procesamiento de Imágenes Instructor :," 2002.

[9] C. Pinilla, a Alcalá, and F. Ariza, "Filtrado de imágenes en el dominio de la frecuencia," *Rev. Teledetección*, vol. 8, pp. 1–5, 1997.

[10] J. Malik, S. Belongie, T. K. Leung, and J. Shi, "Contour and Texture Analysis for Image Segmentation," *Int. J. Comput. Vis.*, vol. 43, no. 1, pp. 7–27, 2001.

[11] G. Papari and N. Petkov, "Edge and line oriented contour detection: State of the art," *Image Vis. Comput.*, vol. 29, no. 2–3, pp. 79–103, 2011.

[12] L. Liu, L. Zhao, Y. Long, G. Kuang, and P. Fieguth, "Extended local binary patterns for texture classification," *Image Vis. Comput.*, vol. 30, no. 2, pp. 86–99, 2012.

[13] P. M. Pietikäinen and G. Zhao, "Local Texture Descriptors in Computer Vision," *ICCV - 12th Int. Conf. Comput. Vis.*, p. 28, 2009.

[14] A. L. Amet, A. Ertiiziin, and A. Ergil, "Texture defect detection using subband domain co-occurrence matrices," pp. 205–210.

[15] R. Bergman, H. Nachlieli, and G. Ruckenstein, "Detection of Textured Areas in Images Using a Disorganization Indicator Based on Component Counts image analysis An algorithm is presented for the detection of textured areas in digital images . Texture detection has potential application to image enhance," vol. 175, 2007.

[16] I. H. Marilina, B. Nahuel, P. Iván, U. T. N. Facultad, and R. Concepción, "Reconocimiento de Imágenes mediante Scale Invariant Feature Transformation (SIFT)," 2010.

[17] D. G. Lowe, "Distinctive image features from scale invariant keypoints," *Int'l J. Comput. Vis.*, vol. 60, pp. 91–11020042, 2004.

[18] A. Hadid, J. Ylioinas, and M. B. L, "Face and Texture Analysis Using Local Descriptors : A Comparative Analysis," pp. 1–4, 2014.

[19] Documentación Open Source Computer Vision [en línea]

< http://opencv.org/documentation.html>

[Consulta: 01-08-2015 - 30-01-2016]

[20] Documentación EmguCV [en línea]

<http://www.emgu.com/wiki/index.php/Documentation>

[Consulta: 01-08-2015 - 30-01-2016]

[21] MICROSOFT Manual de Lenguaje C# [en línea]

< https://msdn.microsoft.com/es-es/library/zkxk2fwf(v=vs.90).aspx >

[Consulta: 01-08-2015 - 30-01-2016]

[22] PONCE Cruz, Pedro. Inteligencia Artificial con aplicaciones a la ingeniería. México, Alfaomega, 2010. 348p.

[23] Filtros [en línea]

<http://alojamientos.us.es/gtocoma/pid/tema3-1.pdf>

[Consulta: 01-08-2015 - 02-02-2016]

[24] Rayos Ultravioleta [en línea]

< http://definicion.de/rayos-ultravioleta/#ixzz42Y3JJK8L >

[Consulta: 01-03-2016]

[25] Gonzales, Rafael y Woods, Richard. Tratamiento Digital de Imágenes. Madrid, Addison-Wesley, 1996. 0-201-62576-8

[26] M. Tuceryan, A. K. Jain, Texture Analysis, The Handbook of Pattern Recognition and Computer Vision (2nd Edition), by C. H. Chen, L. F. Pau, P. S. P. Wang (eds.), pp. 207-248, World Scientific Publishing Co., 1998.

[27] ELECTRONICS AND COMPUTER SCIENCE UNIVERSITY OF SOUTHAMPTON.
Computer Vision Demostration Website [en línea]
<http://users.ecs.soton.ac.uk/msn/book/new_demo/sobel/>

[Consulta: 01-03-2016 - 10-03-2016]

[28] Aracena-Pizarro, Diego, Daneri-Alvarado, Nicolás. (2013). Detección de puntos claves mediante SIFT paralelizado en GPU.Ingeniare. Revista chilena de ingeniería, 21(3), 438-447

[29] MARMANIS, Haralambos. Algorithms of the Intelligence Web. Stamford, Manning, 2009. 345p.

[30] Betancourt, Gustavo A. (2005) Las Máquinas De Soporte Vectorial (Svms). Scientia Et Technica, Abril-Sin Mes, 67-72.

[31] UNIVERSIDAD AUTONOMA DE BARCELONA
Detector Basado en HOG/SVM [en línea]
< https://es.coursera.org/learn/deteccion-objetos/lecture/JXyER/l4-6-support-vector-machines-svm-desarrollo-matematico >

[Consulta: 01-03-2016 - 10-03-2016]

[32] CARNEGIE MELLON UNIVERSITY
Bag-of-Visual-Words [en línea]
< http://www.cs.cmu.edu/~16385/lectures/Lecture12.pdf>
[Consulta: 11-04-2016]

ANEXO A: APLICACIÓN DESARROLLADA

Dado que el presente proyecto corresponde a una sub etapa de un proyecto FONDEF, y cuenta con protección de propiedad intelectual, los Anexos A y B describirán la aplicación desarrollada en términos de ingeniería de software y manual de usuario respectivamente, omitiendo el detalle de implementación de código y la documentación asociada.

1.1. Objetivos del software

- Objetivo General:

  La aplicación permitirá aplicar algoritmos de procesamiento digital en imágenes y extraer características de defectos presentes en ella.

1.2. Ambiente de Ingeniería de Software

- Metodología de desarrollo

  La metodología es del tipo Evolutivo, siguiendo un modelo de prototipos, partiendo de un diseño rápido se van incorporando distintas funcionalidades (algoritmos) a petición del usuario (jefe laboratorio CIM). De esta manera se obtiene retroalimentación para continuar con las siguientes implementaciones.

- Técnicas y notaciones

  Para modelar la aplicación se utilizará diagrama de casos de uso y Diagramas de flujos de datos

- Herramientas utilizadas

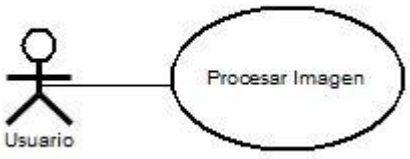
  - Lenguaje de desarrollo: C#
  - Microsoft Visual Studio 2013: Entorno de desarrollo Integrado.
  - EmguCV versión 2.4: wrapper para la librería de procesamiento de imágenes OpenCV
  - Matlab versión R2014a: para diferentes pruebas de procesamiento de imágenes.
  - Power Designer versión 16.1: modelar diagrama de casos de uso.

- Equipo para desarrollo: Computador Personal con

    - Sistema Operativo Windows10
    - Procesador Intel Core i-3, 2.4 Ghz
    - Memoria Ram 4GB
    - Tarjeta gráfica integrada Intel HD Graphics4000

1.3. Análisis

1.3.1. Diagrama de Casos de Uso



Procesar Imagen: este caso de uso permite seleccionar una imagen desde el equipo para la posterior aplicación de algoritmos de procesamiento digital, y permite ver los cambios en tiempo real.

Usuario: Es el encargado de operar en todas las funciones sobre la imagen.

1.3.2. Especificación de los Casos de Uso

Caso de Uso CU001:	Procesar imagen
Actor Principal:	Usuario
Objetivo en Contexto:	Aplicar algoritmos de procesamiento digital de imágenes.
Pre-Condiciones:	Imagen digital de muestra con etiquetado del panel en alguno de los siguientes formatos: tiff, tif, dcm, jpg; jpeg, bmp, png
Post-Condiciones:	Imagen transformada con distintos niveles de grises
Escenario Principal:	

Acción del Actor	Respuesta del Sistema
1. El caso de uso comienza cuando un usuario desea seleccionar una imagen almacenada en algún medio externo a la aplicación.	2. El sistema muestra la interfaz de usuario con un botón "agregar imagen"
3. El usuario selecciona el botón y busca dentro de las carpetas en el equipo o dispositivo externo.	4. El sistema muestra todos los archivos con formato tiff, tif, dcm, jpg; jpeg, bmp, png
5. El usuario selecciona una imagen de muestra.	6. El sistema muestra la imagen en el picturebox Original y la resolución de esta.
7. El usuario selecciona el botón ROI	8. Se muestra una nueva imagen sin el etiquetado y sin el fondo de la muestra en escala de grises. En el *picturebox "Nueva"*
9. El usuario selecciona un método de procesamiento	10. La imagen "Nueva" muestra cambios en sus niveles de grises de acuerdo a la operación seleccionada.
11. El usuario selecciona el botón SIFT	12. La cantidad de *keypoints* y el tiempo en milisegundos son mostrados en una etiqueta bajo la imagen "Nueva"
13. El usuario selecciona el botón LBP-Liu	14. El sistema muestra la misma imagen ROI y abre 4 ventanas con histogramas individuales de la imagen"Nueva".
15. El usuario selecciona cerrar pestaña	16. El sistema cierra la aplicación.
Cursos Alternos:	3. El usuario no selecciona el botón, el sistema permanece a la espera de la ejecución de una actividad. 5.a El usuario selecciona un formato distinto, el sistema envía un mensaje de advertencia "Formato no valido, intente nuevamente". 5.b El usuario selecciona una imagen sin logo. Puede saltar al paso 9. 11. el usuario no selecciona la opción SIFT, la aplicación permitirá operar con cualquiera de las otras opciones. 13. el usuario no selecciona la opción LBP-Liu, la aplicación permitirá operar con cualquiera de las otras opciones. 15. el usuario no cierra la aplicación, la aplicación permitirá operar con cualquiera de las otras opciones.
Referencias cruzadas:	Requisitos RF001, RF002, RF003

6.3.3 Requisitos de la aplicación

A continuación se especifican los requisitos básicos solicitados por el cliente.

RF001: La aplicación debe permitir seleccionar una imagen desde el computador u otro dispositivo.

RF001.1 las imágenes permitidas son en formato tiff, tif, dcm, jpg; jpeg, bmp, png

RF002: La aplicación debe permitir deshacer los cambios hechos en una imagen.

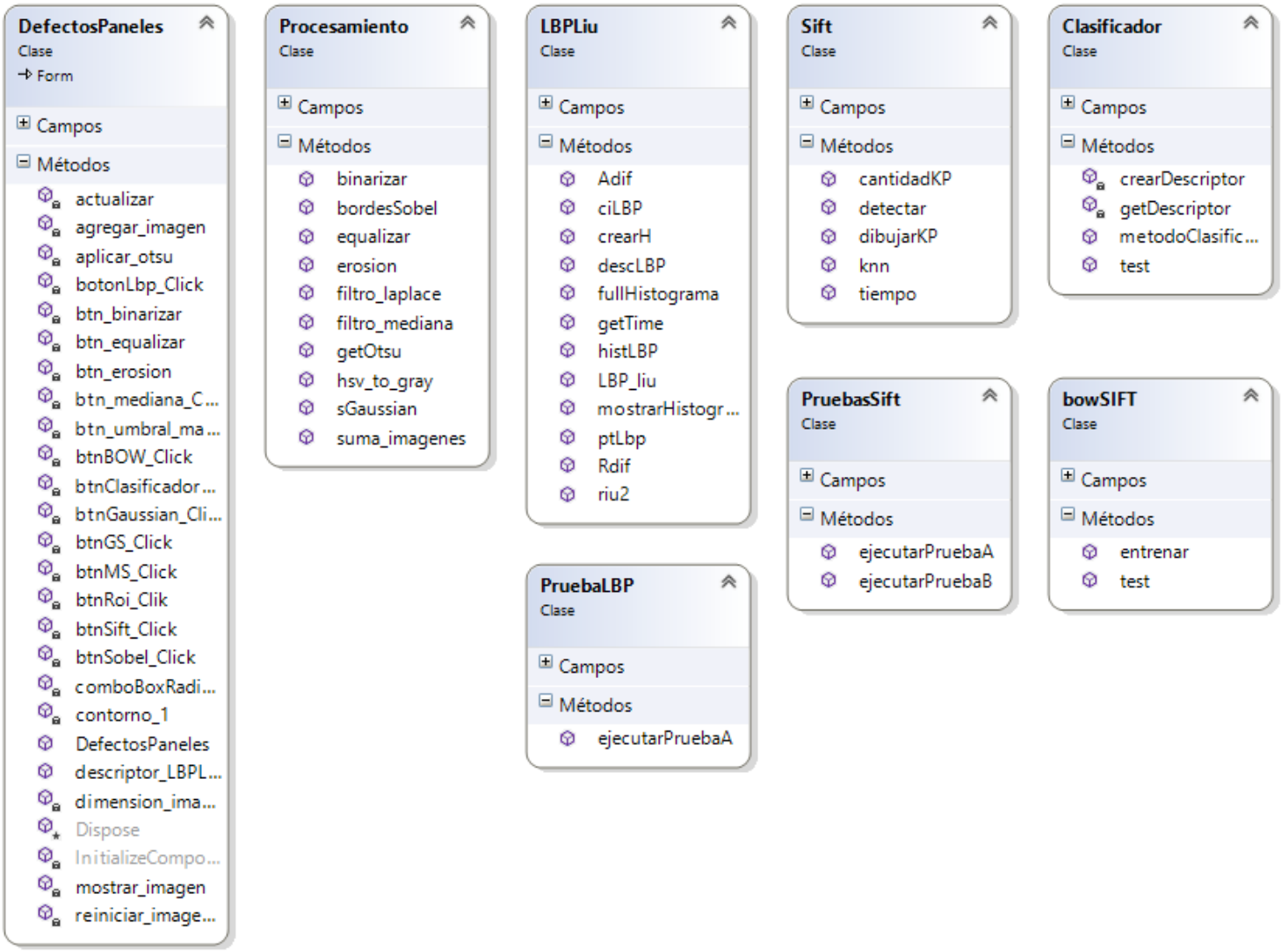
RF002.1: La aplicación contará con un botón de reinicio, donde los cambios en los niveles de gris efectuados sobre la imagen son revertidos.

RF003: La aplicación debe permitir trabajar con imagen con logo y sin logo de etiquetado de panel.

RF003.1: La aplicación contará con un botón llamado ROI que permitirá extraer el área de interés.
RF003.2: La extracción será por medio del usuario. La aplicación no discernirá en primera instancia si la imagen seleccionada contiene el logo o no.

1.4. Modelado

1.4.1. Clases implementadas



Descripción de clases:

Clase	**Descripción**
DefectosPaneles	Clase principal que contiene las funcionalidades de los eventos de los botones e inicializa la aplicación.
Procesamiento	Permite aplicar diferentes métodos de procesamiento de imágenes digitales sobre una imagen de entrada dada.
LBPLiu	Recibe una imagen es escala de grises y un radio (2,3 ó 4) y luego crea descriptores de ella.
PruebaLBP	Permite aplicar suavizado gaussiano + filtro Sobel a una imagen, luego aplicar el algoritmo LBP-liu y registrar el tiempo en milisegundos.
SIFT	Recibe una imagen es escala de grises, luego detecta los *keypoints*, los dibuja, guarda la cantidad y registra el tiempo en milisegundos.
PruebasSift	Permite aplicar suavizado gaussiano + filtro Sobel o filtro mediana + filtro Sobel a una imagen, luego aplicar el algoritmo SIFT y registrar el tiempo en milisegundos y la cantidad de *keypoints*.
Clasificador	Recibe imágenes de entrenamiento, les aplica el algoritmo LBP y crea un modelo clasificador. Luego recibe imágenes de prueba, las compara con el modelo establecido y exporta los resultados en un archivo con extensión .txt.
bowSIFT	Recibe imágenes de entrenamiento, les aplica el algoritmo SIFT, crea un BOWTrainer y va guardando en él cada modeloDescriptor de las imágenes. Agrupa todos los vectores del BOWTrainer en un diccionario y crea un SVM. Luego recibe imágenes de prueba, las compara con el modelo establecido y exporta los resultados en un archivo con extensión .txt.

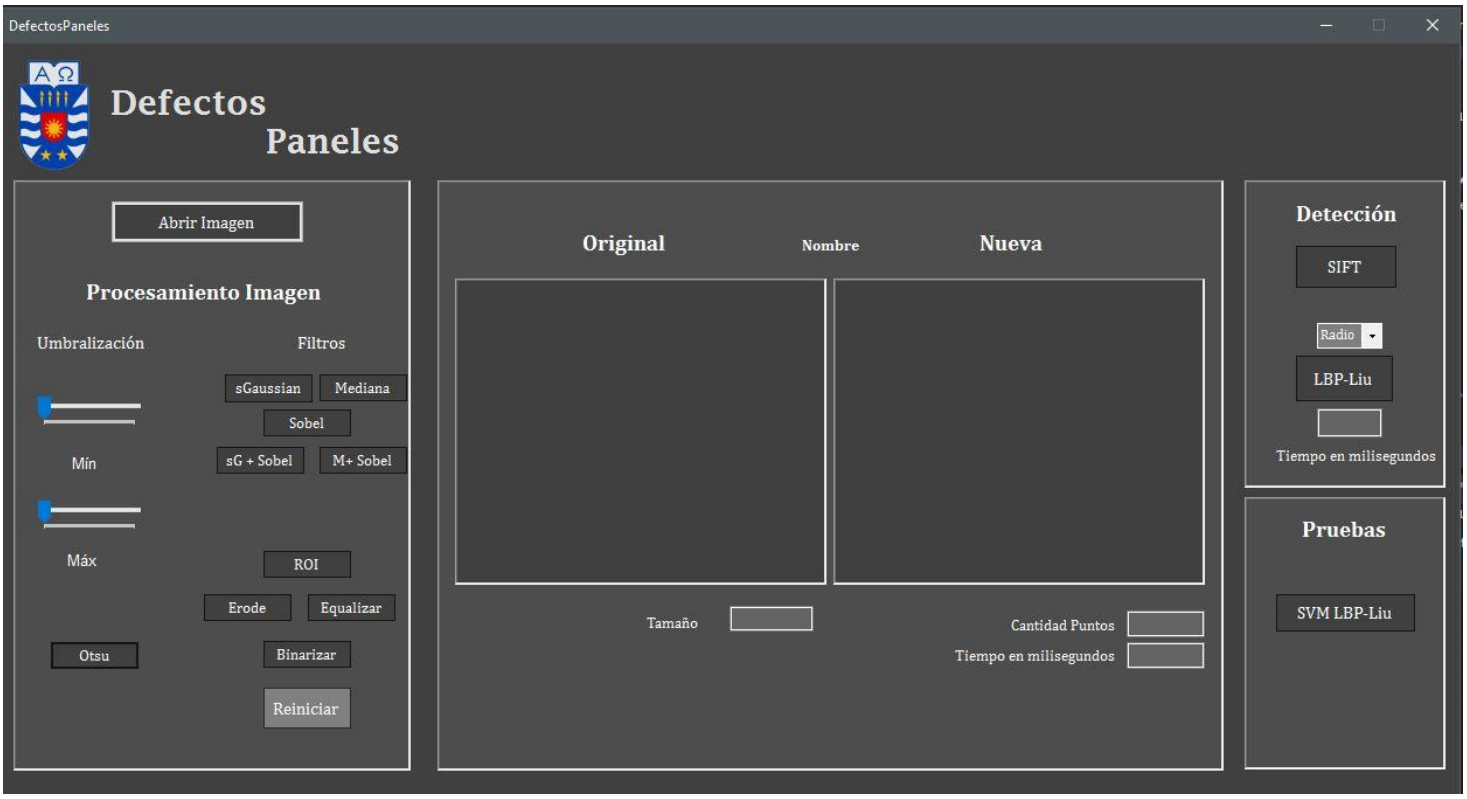
ANEXO B: MANUAL DE USUARIO

1.1 Defectos Paneles

Defectos Paneles es una aplicación de escritorio que permite cargar una imagen desde un dispositivo externo, aplicar procesamiento digital y ejecutar los algoritmos SIFT y LBP-Liu sobre ella.

1.2 Pantalla principal

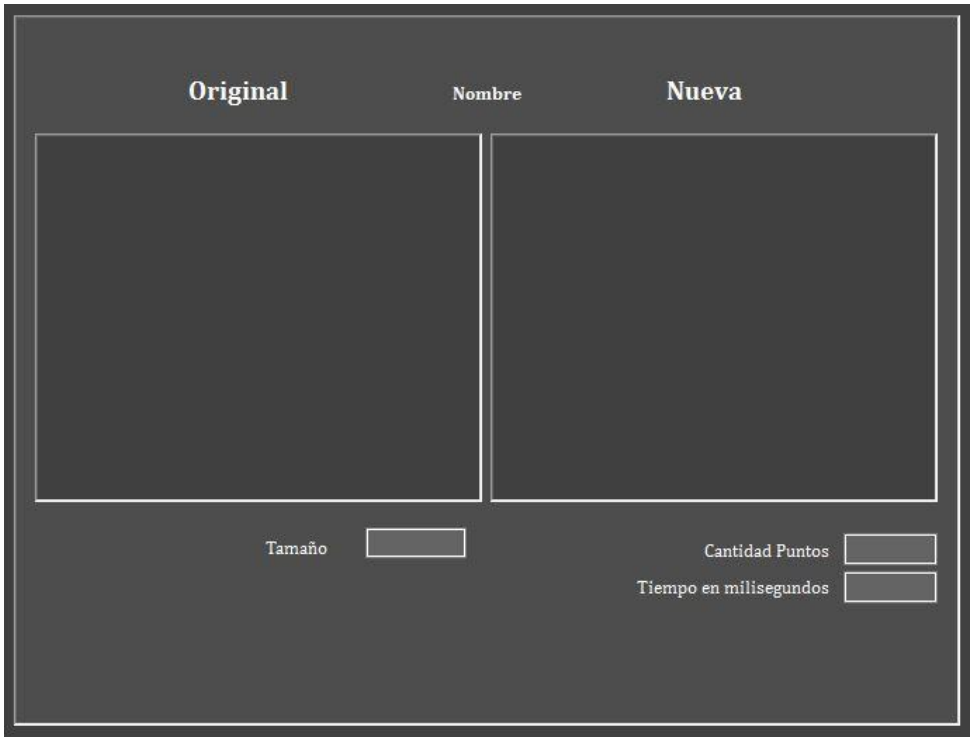
Al abrir la aplicación se iniciará la siguiente pantalla con 3 paneles principales:



1.3 Paneles

1.3.1 Panel central

En este panel es posible visualizar una imagen seleccionada y los cambios efectuados sobre ella.



Contiene:

- Recuadro Original: Lado izquierdo se visualiza la imagen de entrada.
- Recuadro Nueva: Lado derecho, visualiza la imagen original con los cambios efectuados sobre ella.
- Nombre: Nombre de la imagen.
- Tamaño: Tamaño de la imagen (ancho x alto).
- Cantidad Puntos: Cantidad de *keypoints* detectados por el algoritmo SIFT.
- Tiempo en milisegundos: Tiempo en detectar los *keypoints.*

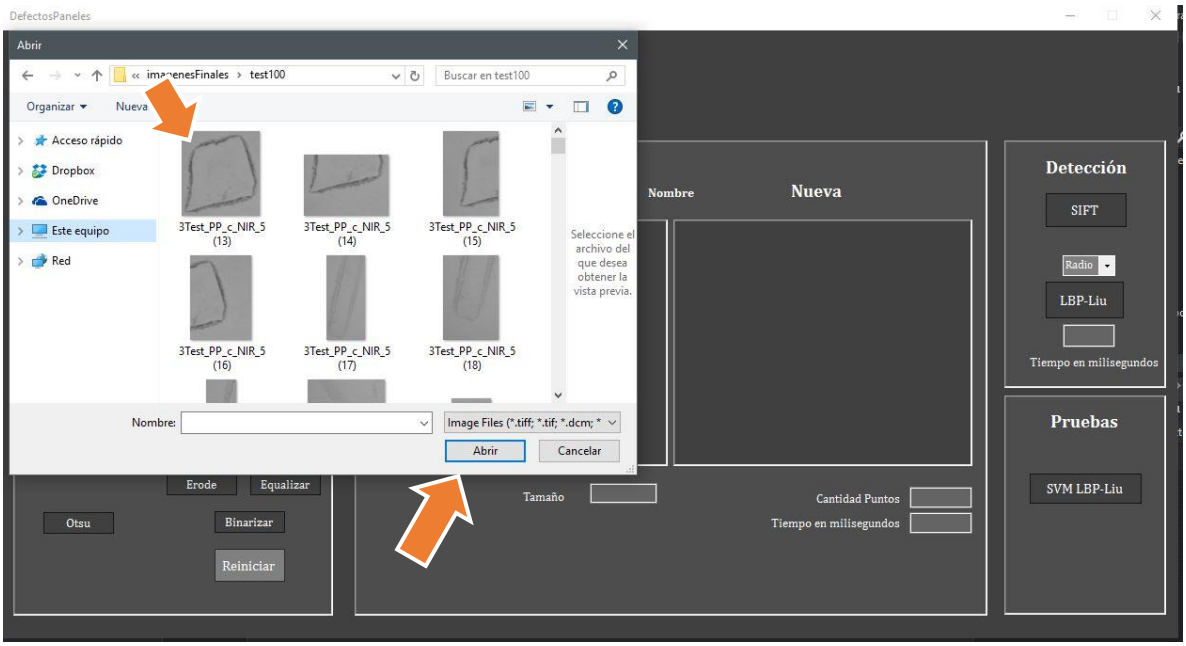
1.3.2 Panel izquierdo

Permite seleccionar una imagen almacenada en el computador o en un dispositivo externo. Luego aplicar distintos métodos de procesamiento digital.

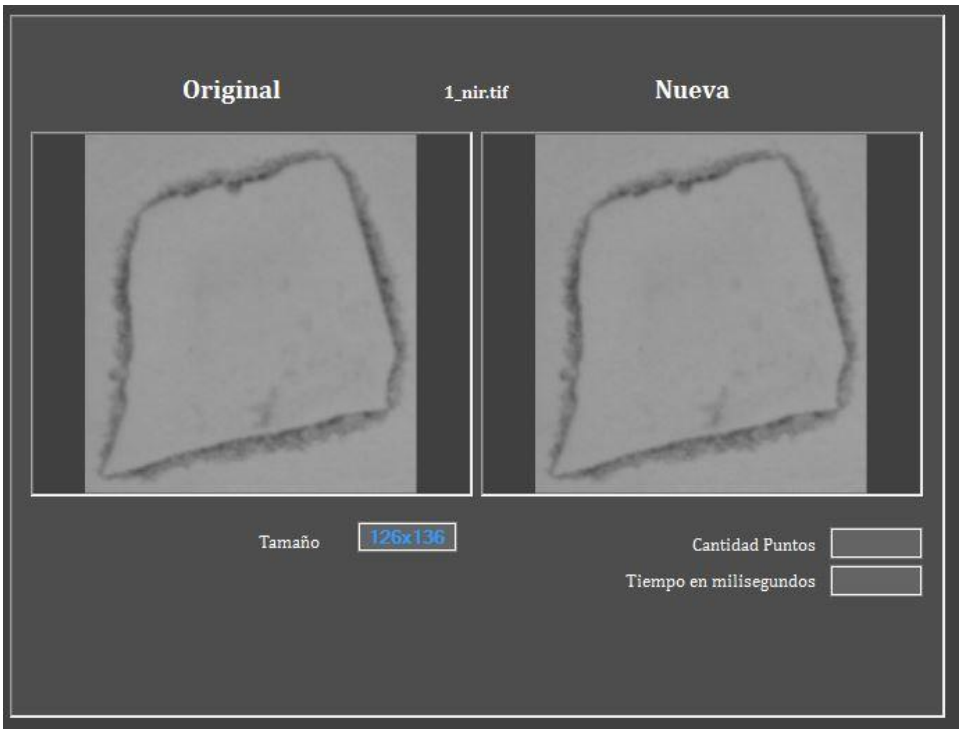
Contiene las siguientes funcionalidades:

- Botón Abrir Imagen: Al clickear sobre él abre una ventana emergente que permite navegar por las diferentes carpetas para buscar una imagen.

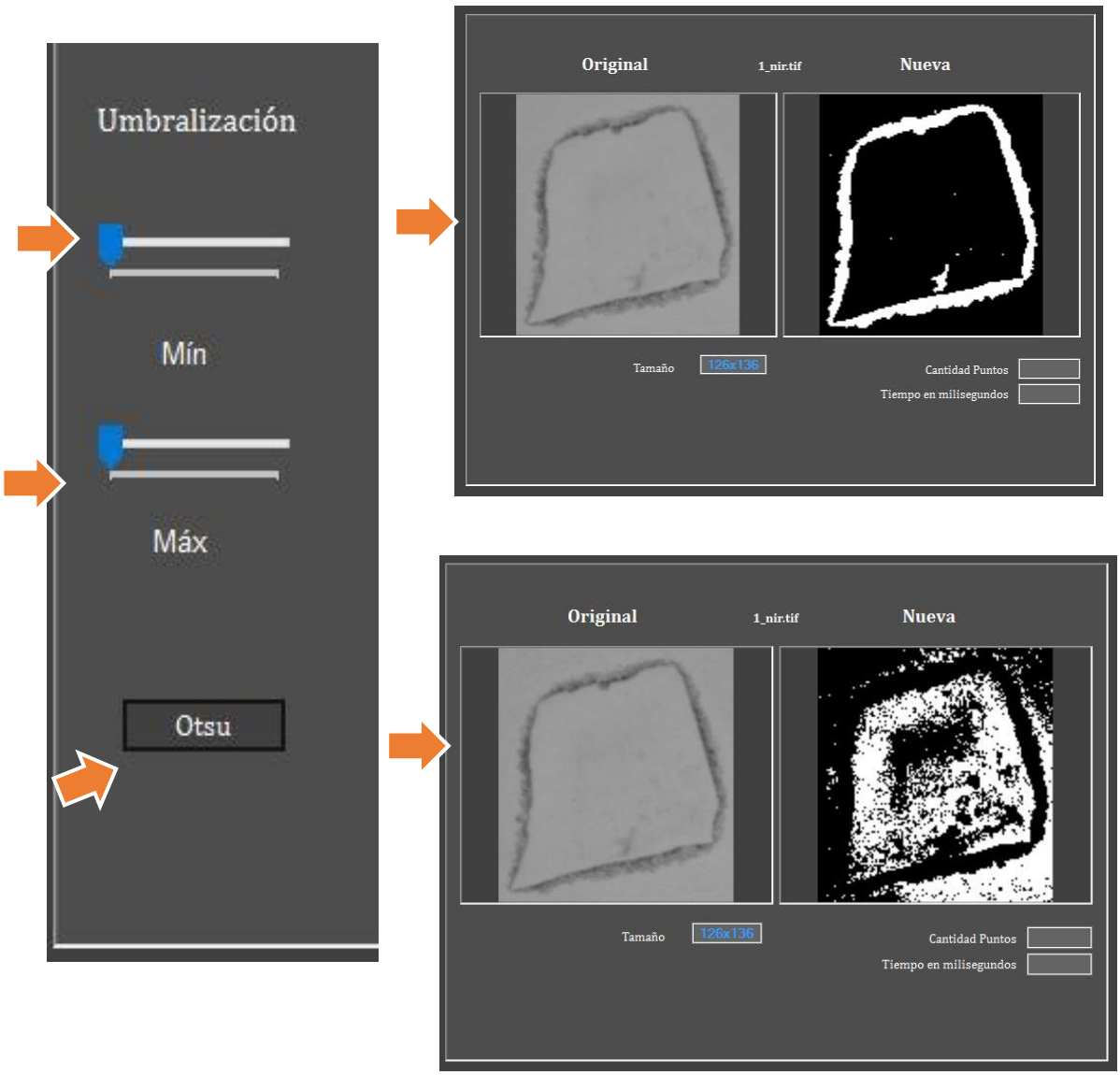
Mostrará solamente los archivos permitidos en la aplicación: formato tiff, tif, dcm, jpg; jpeg, bmp, png



Al seleccionar la imagen, luego se presiona el botón abrir y se muestra la imagen en ambos recuadros del panel central.

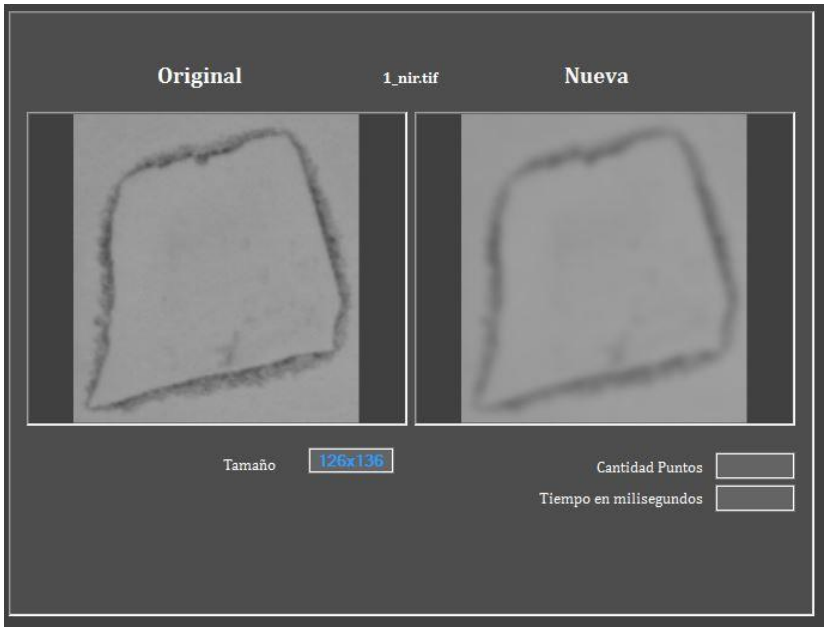


La sección umbralización permite seleccionar el valor mínimo y máximo de umbralización sobre la imagen, el cual es visualizado en tiempo real sobre la imagen de entrada.
El botón Otsu, permite aplicar este algoritmo sobre la imagen de entrada y visualizarlo.

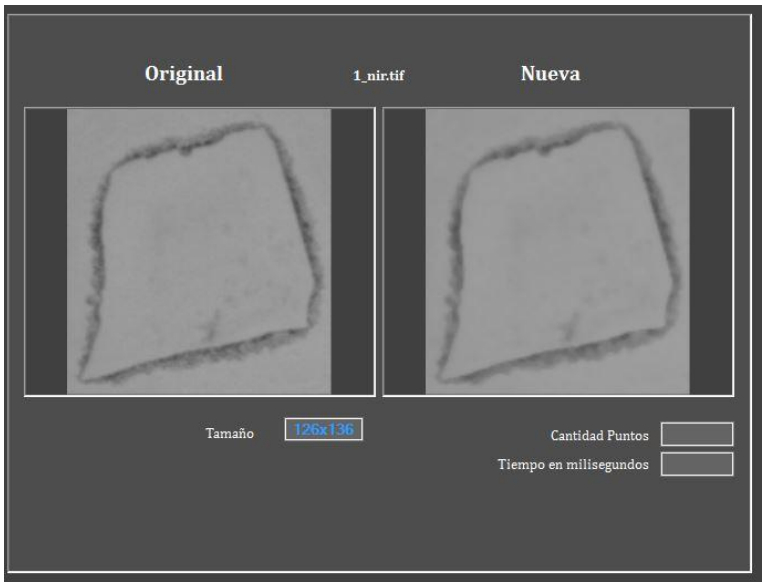


La sección **Filtros** tiene las siguientes funcionalidades:

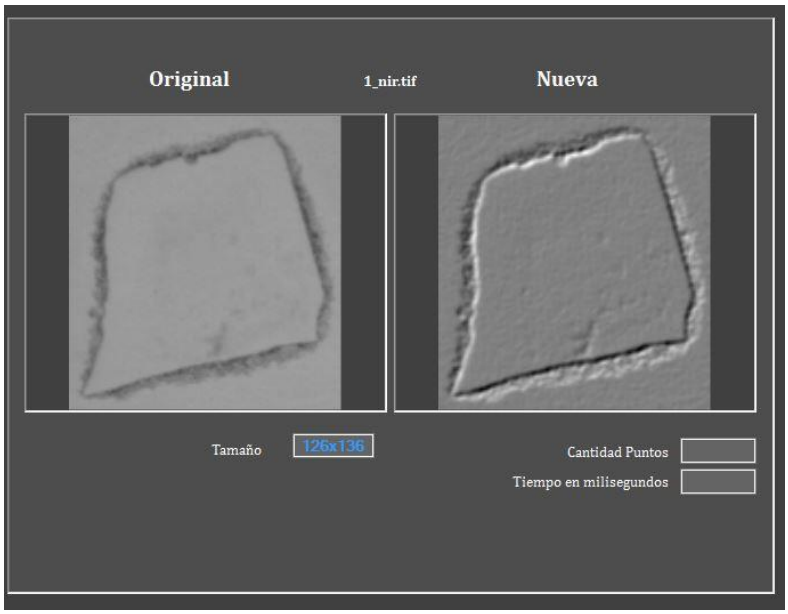
- Botón sGaussian: permite aplicar suavizado gaussiano sobre la imagen.



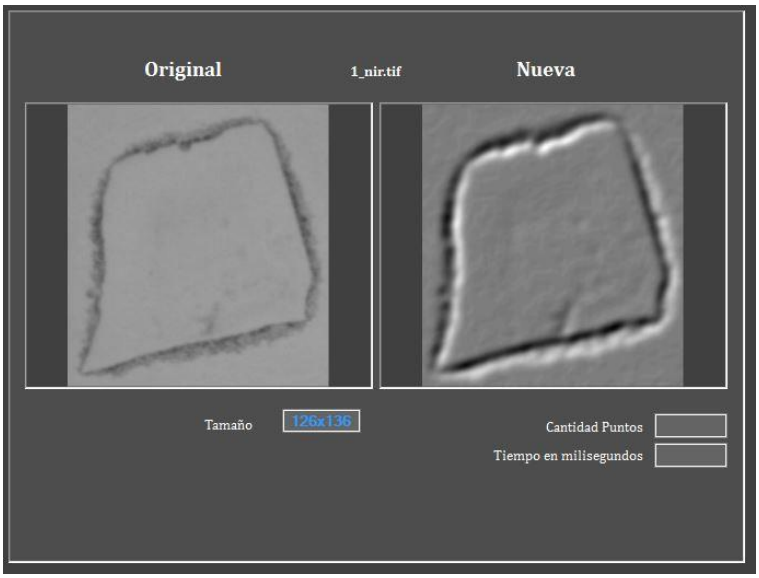
- Botón Mediana: permite aplicar filtro de la mediana sobre la imagen.



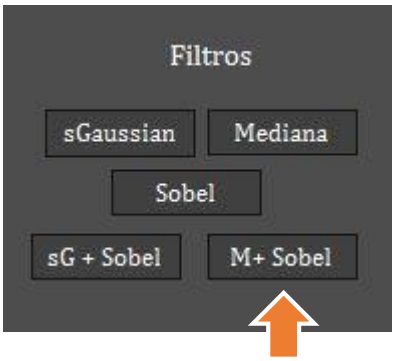
- Botón Sobel: permite aplicar filtro sobel sobre la imagen.

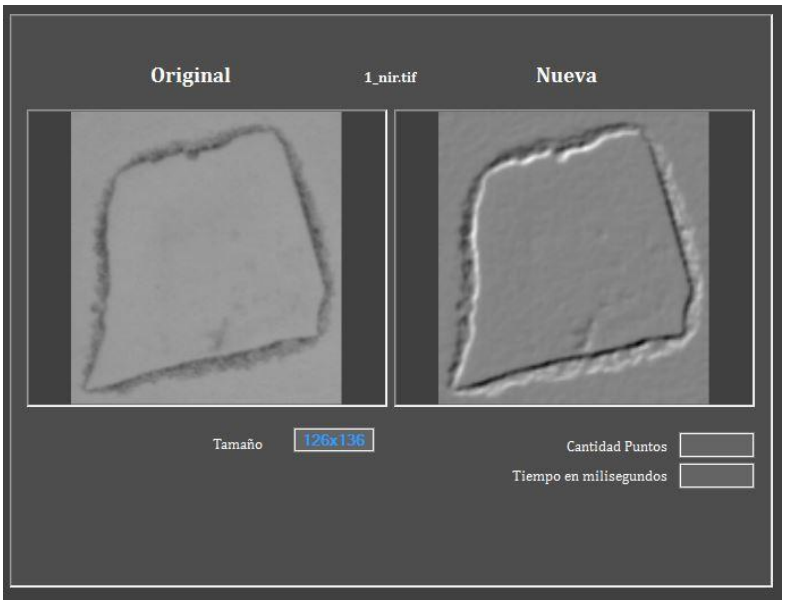


- Botón sG + Sobel: permite aplicar suavizado gaussiano y luego filtro sobel sobre la imagen.



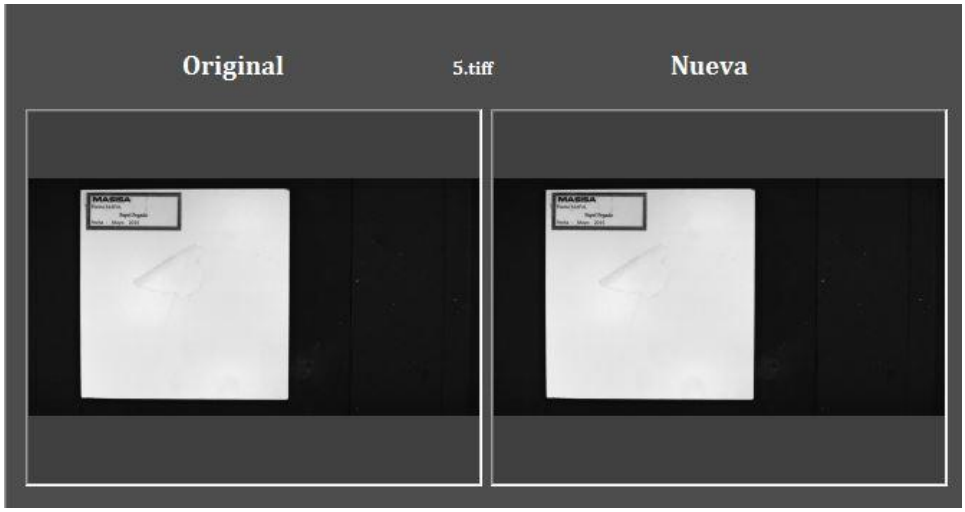
- Botón M + Sobel: permite aplicar filtro de la mediana y luego filtro sobel sobre la imagen.

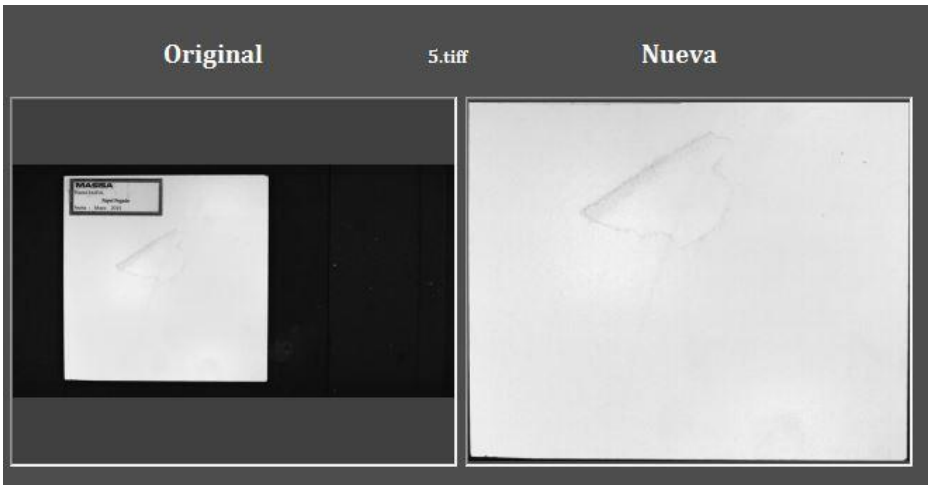




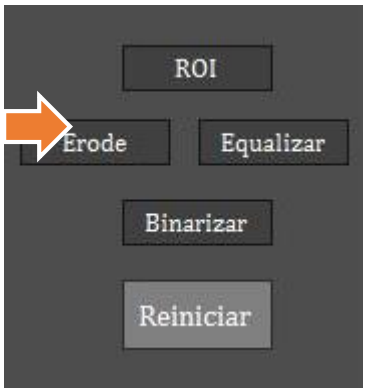
Sección inferior:

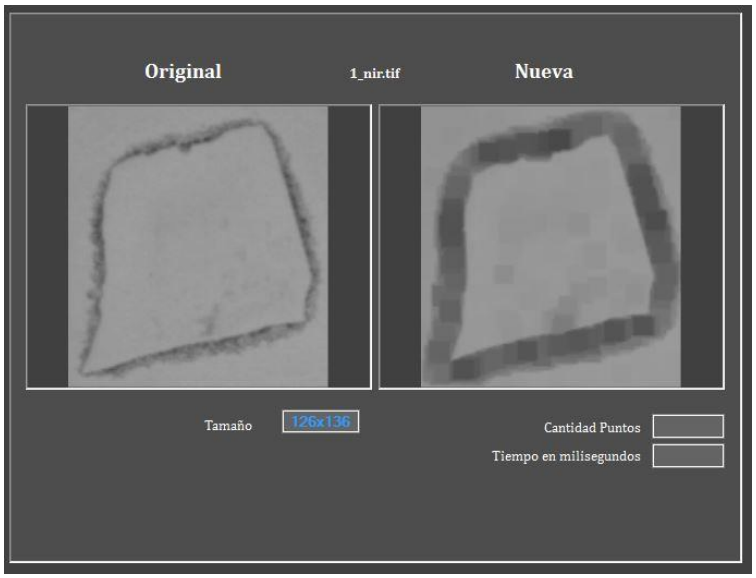
- Botón ROI: permite eliminar el fondo y la etiqueta de una muestra.



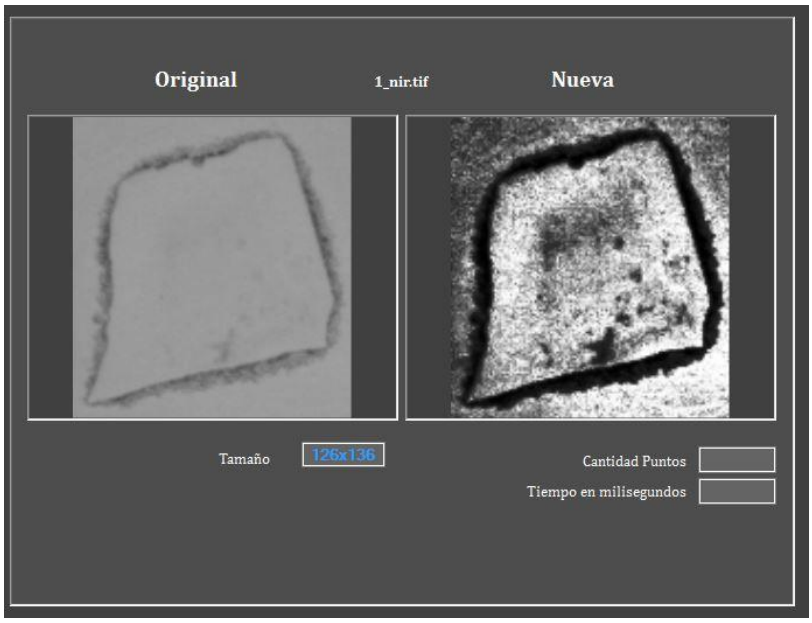


- Botón Erode: permite aplicar erosión sobre la imagen.

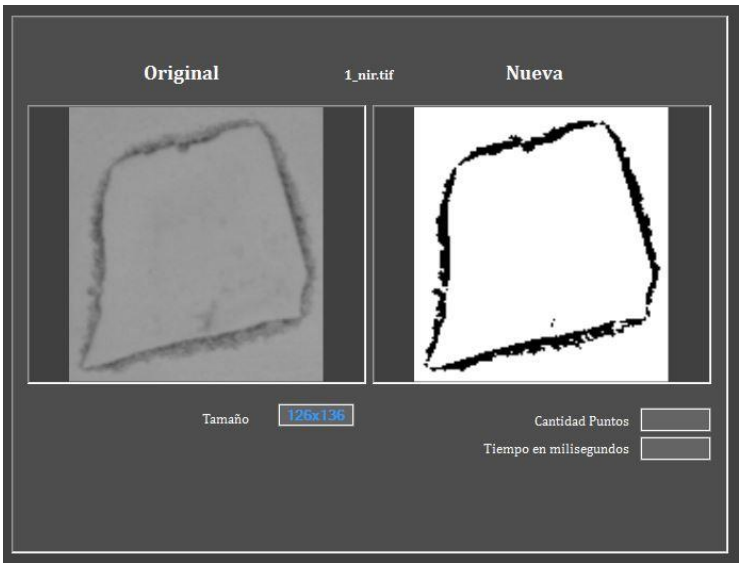




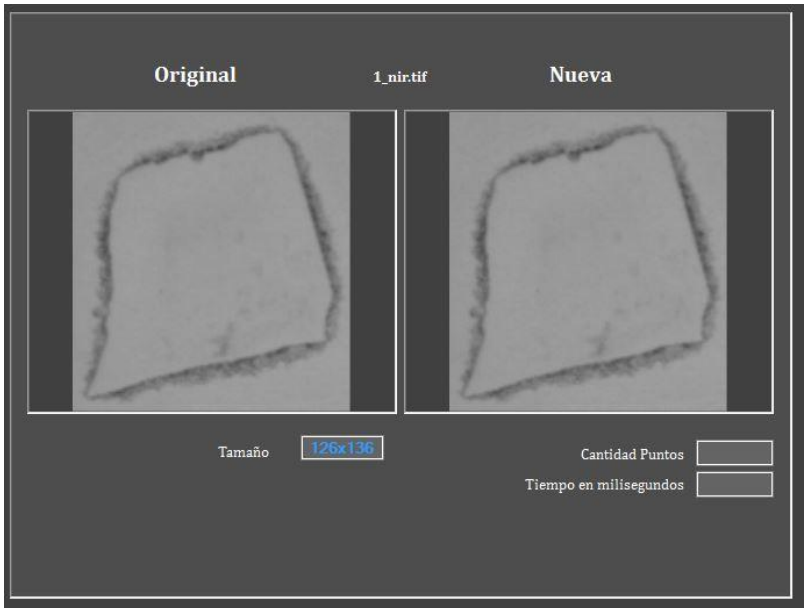
- Botón Equalizar: permite ecualizar el histograma de la imagen.



- Botón Binarizar: permite aplicar binarización sobre la imagen.



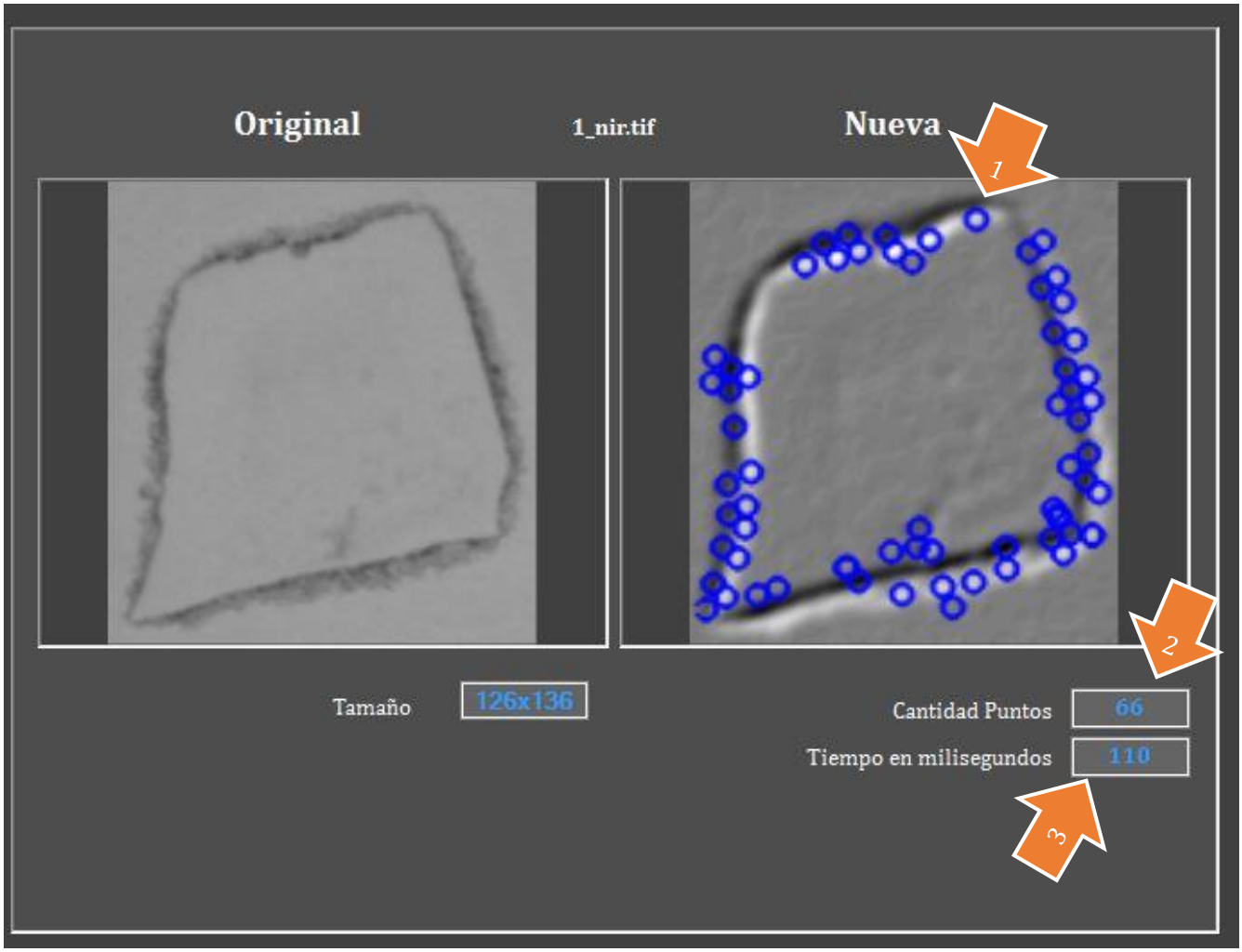
- Botón Reiniciar: permite deshacer todos los cambios aplicados sobre la imagen.



1.3.3 Panel derecho

Permite aplicar un algoritmo de detección sobre una imagen previamente procesada o sin procesamiento digital mostrada en el recuadro derecho.

- Botón SIFT: permite aplicar el algoritmo SIFT sobre la imagen. Al presionarlo muestra:
  1. Los puntos encontrados sobre la imagen Nueva.
  2. La cantidad de *keypoints* encontrados.
  3. El tiempo de detección en milisegundos.

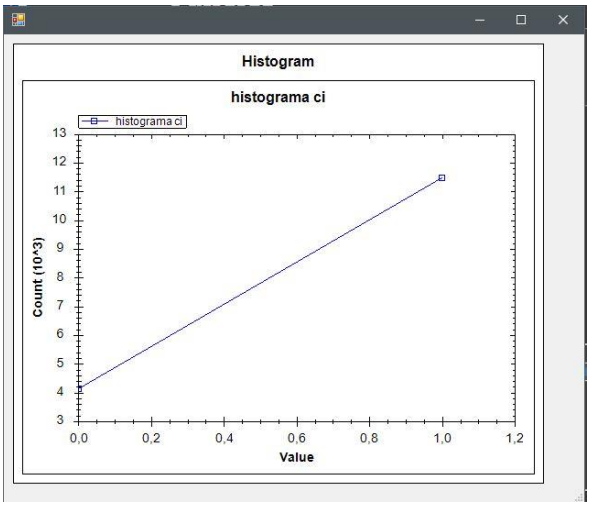


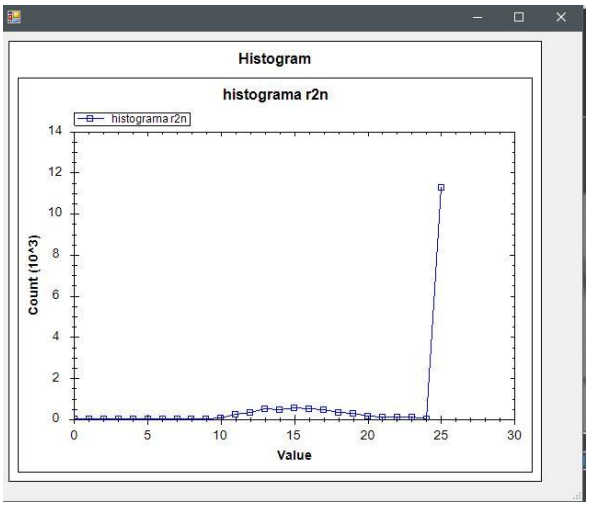
- Botón LBP-Liu : permite seleccionar un radio y luego aplicar el algoritmo LBP-Liu sobre la imagen.

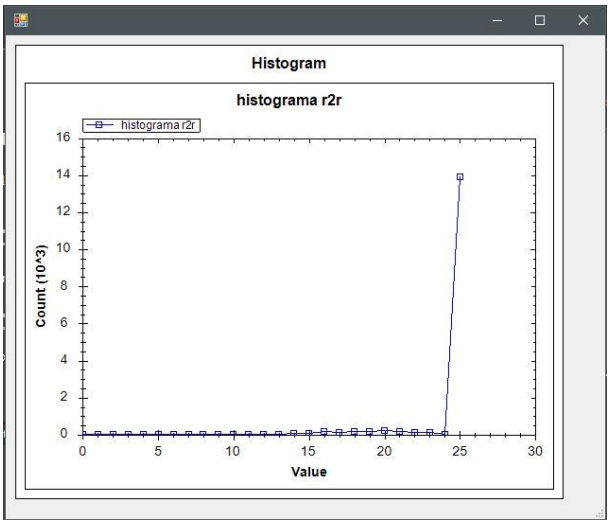
  Al presionar sobre Radio, aparece un listado con las opciones de radio para aplicar el algoritmo.

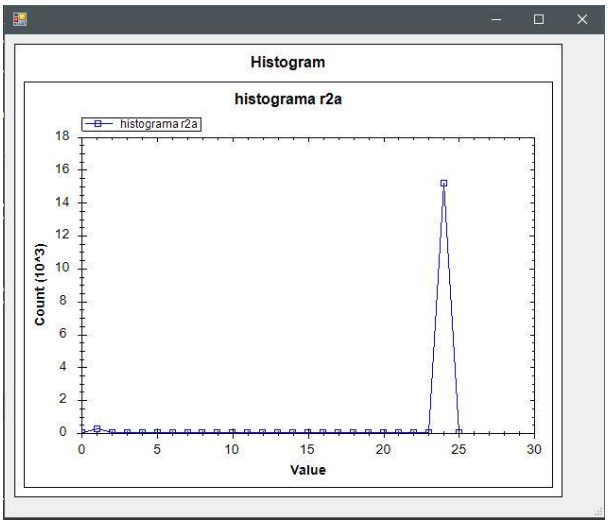
Luego se presiona el botón LBP-Liu y aparece el tiempo en milisegundos en ejecutarse

Enseguida se muestran los histogramas individuales de este algoritmo:

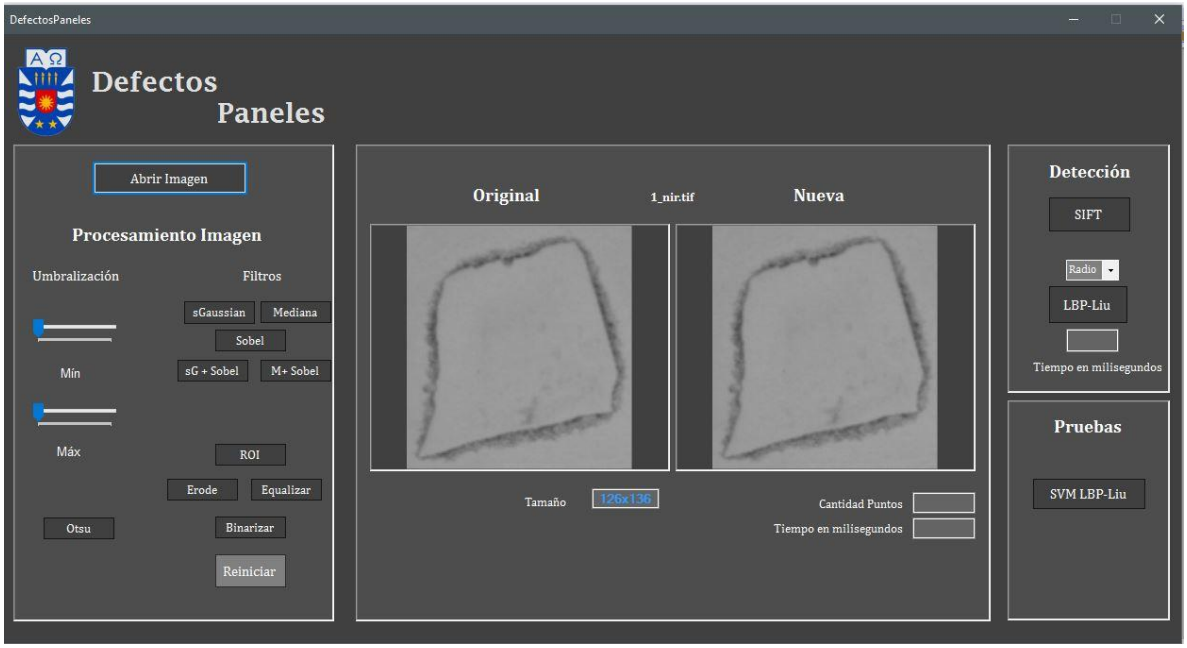




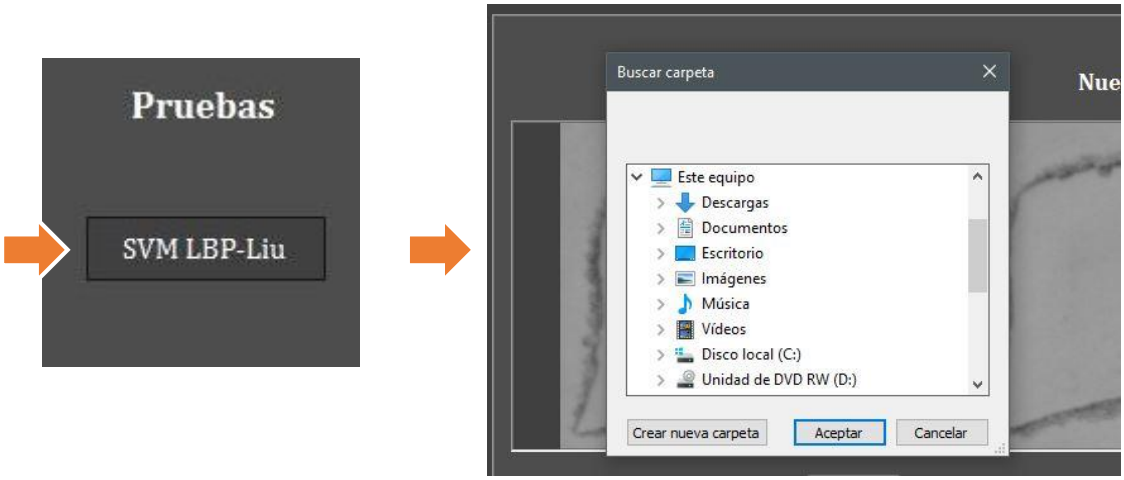




Luego de cerrar estas ventanas, se vuelve a la ventana principal.

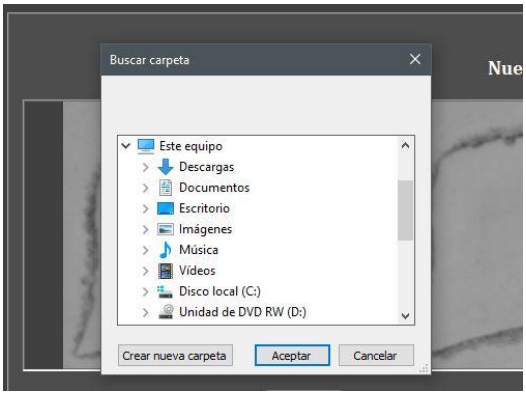


La sección Pruebas, contiene un botón llamado SVM LBP-Liu, el cual permite entrenar un clasificador SVM con imágenes procesadas con algoritmo LBP-Liu y luego ver el resultado de ejecutar un conjunto de pruebas .



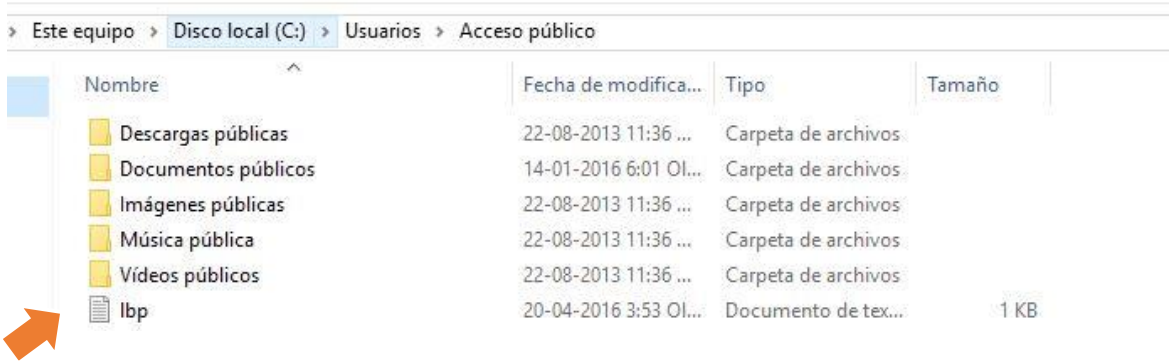
Al presionarlo se abre una ventana que permite buscar y seleccionar la carpeta que contiene muestras de entrenamiento:

- La carpeta **debe** contener la primera mitad con muestras positivas y la segunda mitad con muestras negativas. Luego del entrenamiento se abre nuevamente una ventana:



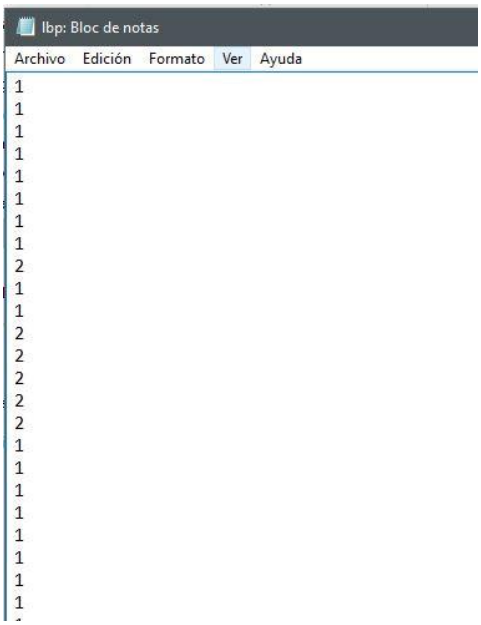
- Esta ventana permite buscar la carpeta con las muestras a probar. La carpeta **debe** contener la primera mitad con muestras positivas y la segunda mitad con muestras negativas. Así el clasificador SVM realiza las comparaciones con las muestras entrenadas previamente.

- Luego de la detección, se exportará en la ruta "C:\Users\Public" un archivo llamado "lbp.txt"



El archivo contiene los resultados de la clasificación, donde:

- 1= muestra detectada como clase defecto
- 2= muestras detectada como clase sin defectos



La primera mitad corresponde a los elementos marcados con defecto y la segunda mitad los sin defecto, por lo tanto se debe revisar si efectivamente en los resultados la primera

mitad corresponde a 1 y la segunda a 2, los valores marcados como distinto dentro de sus rangos, son las falsas detecciones del algoritmo.

Luego de ejecutar esta funcionalidad, se vuelve automáticamente a la ventana principal y es posible minimizar o cerrar la aplicación.