industriales
etsii

Escuela Técnica Superior de Ingeniería Industrial

UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Escuela Técnica Superior de Ingeniería Industrial

Sistema de visión para la detección de suciedad en cristales AR en una planta termosolar.

TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

Autor: **Ana Peñaranda Verdú**
Director: Jorge Juan Feliu Batlle
Codirector: Pablo Alejandro Martínez Ruiz

Cartagena, 19 de junio de 2017

Universidad Politécnica de Cartagena

A mí familia.

Tabla de contenido

TABLA DE ILUSTRACIONES

ÍNDICE DE TABLAS

CÁPITULO 1

Introducción y objetivos.

1.1 INTRODUCCIÓN.

El presente capítulo tiene como finalidad el planteamiento del proyecto de fin de grado y comentar los objetivos.

En los últimos años el avance de la tecnología y de los sistemas de información están motivando importantes cambios que nos han permitido la automatización de procesos, mejorando así la calidad y disminuyendo costes. Este continuo avance de la tecnología tiene como consecuencia la automatización de sistemas y procesos que anteriormente eran desarrollados manualmente por operarios. Lo que conlleva a una mejora de la calidad de los procesos productivos y, por consiguiente, un aumento de la productividad y disminución de costes asociados. Además, la incorporación de las TIC (Tecnologías de la Información y la Comunicación) permite un mayor control de procesos, una mejora del conocimiento del entorno productivo y la posibilidad de innovación tanto en servicios como en respuestas a las necesidades de la empresa.

La motivación del proyecto es el diseño de un sistema basado en la visión artificial y robótica para la mejora del proceso de limpieza de cristales antireflectantes (AR) de una empresa. Esta es una empresa de producción de energía termosolar con espejos planos (central termosolar de espejos Fresnel).

El procedimiento de trabajo es similar al de cualquier otra planta de generación térmica pero donde se sustituye la caldera de combustible (que puede ser carbón, gas natural, biomasa o combustible nuclear) por un campo solar que utiliza el calor del sol. Dicho calor se emplea para evaporar agua produciendo vapor, cuya presión genera movimiento en una turbina de vapor que a su vez hace rotar un alternador generando electricidad.

La particularidad de esta planta es la tecnología de espejos planos (también llamada Linear Fresnel). La tecnología de espejos planos, a diferencia de la tecnología de espejos cilindro-parabólicos y la de torre central es que son más económicos de fabricar y de mantener, aunque no se alcanzan temperaturas tan altas que incrementan el rendimiento de la turbina [1]. Igualmente, la tecnología consta de evaporación directa, es decir, no utiliza ningún otro fluido como transmisor del calor (como pueden ser aceites térmicos) por lo tanto es totalmente libre de contaminantes [2].

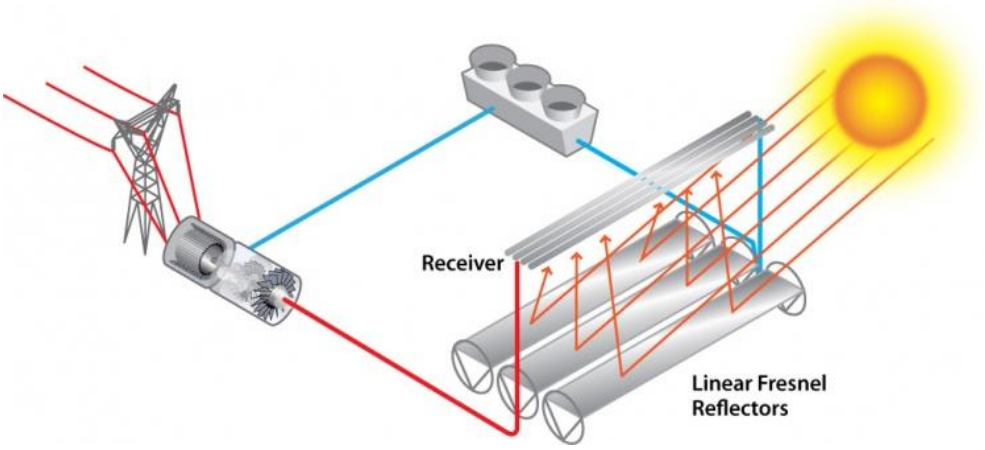


Figura 1: Funcionamiento de planta termosolar con tecnología linear fresnel. Fuente: energy.gov

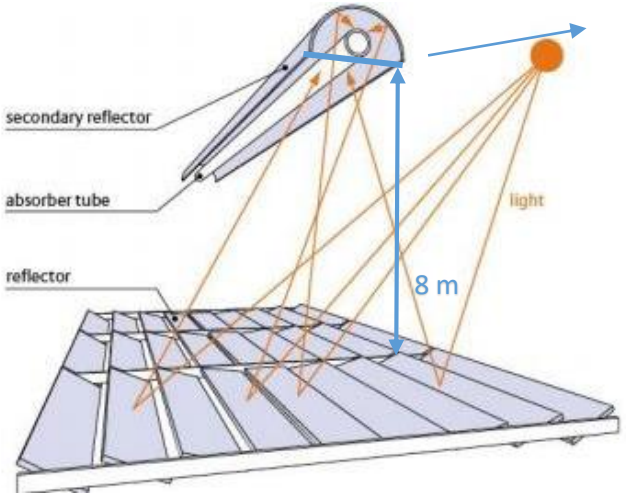
Lo que se pretende conseguir es poder automatizar al máximo el proceso de limpieza de los cristales antireflectantes situados en la parte superior de la estructura (receiver), dichos espejos están al aire libre no ajenos de las condiciones atmosféricas y polvo.

Para que el proceso sea óptimo estos cristales deben de encontrarse en las mejores condiciones, es decir, estar limpios, sin roturas y sin grietas.

- Descripción de la estructura:

  Reflectores lineales primarios: llevan espejos planos normales y simulan la curvatura de los espejos cilindro parabólicos variando el ángulo de cada fila con un solo eje de seguimiento.

  Formado por dos grupos termosolares de 15 MW [3]. Cada uno consta de un campo solar y un sistema de potencia que puede acoplarse adicionalmente al campo solar del otro, con el fin de mejorar el rendimiento de la turbina en periodos de mejor potencia solar.



Justo debajo de la tubería se encuentra el **cristal AR** que queremos analizar.

Figura 2: Tecnología linear fresnel. Fuente: energy.gov

Además, se desarrollará un software (programado en C++) que haga de enlace entre el sistema robotizado y la cámara de visión y que almacene la información (imágenes).

1.2 MOTIVACIÓN

Empecé a hacer este proyecto a propuesta de la empresa. La idea inicial era implementar un sistema de visión para automatizar el proceso de limpieza de los cristales antireflectantes situados en el receiver (reflectores secundarios). Este proyecto se comenzó a desarrollar debido a la curiosidad que sentía sobre el mundo de la visión artificial tras cursar una optativa de ella. La visión artificial ha ido creciendo en la industria debido a las ventajas que conlleva su uso.

Se pretende solucionar un problema que ha rodeado día a día a la planta termosolar, este problema es la limpieza de cristales. La limpieza repercute directamente en la empresa y también a sus trabajadores.

Una de las grandes motivaciones de este trabajo es hacer un desarrollo de software libre, utilizando para ello el lenguaje de programación C++ y el uso de la librería OpenCV. Usar la visión artificial para la supervisión de suciedad en cristales es debido a que una cámara de visión se puede programar mediante un algoritmo similar a lo que vería el ojo humano.

1.3 ANTECEDENTES

Uno de los sentidos más importantes de los seres humanos es la visión. Ésta es empleada para obtener la información visual del entorno físico. De hecho, se deduce que más del 70% de las tareas del cerebro son empleadas en el análisis de la información visual. Casi todas las disciplinas científicas emplean utillajes gráficos para transmitir conocimiento. Por ejemplo, en Ingeniería Electrónica se emplean esquemas de circuitos, a modo gráfico, para describirlos. Se podría hacerlo mediante texto, pero para la especie humana resulta mucho más eficiente procesar imágenes que procesar texto. La visión humana es el sentido más desarrollado y el que menos se conoce debido a su gran complejidad. En el año 1826 el químico francés Niepce (1765-1833) llevó a cabo la primera fotografía, colocando una superficie fotosensible dentro de una cámara oscura para fijar la imagen. Posteriormente, en 1838 el químico francés Daguerre (1787-1851) hizo el primer proceso

fotográfico práctico. Daguerre utilizó una placa fotográfica que era revelada con vapor de mercurio y fijada con trisulfato de sodio [4].

Desde que se inventó la fotografía se ha intentado extraer características físicas de las imágenes. Ejemplos que han transformado la percepción de la ciencia con el procesamiento de imágenes: la Astronomía avanzó enormemente con el análisis de imágenes recibidas por los telescopios, el análisis de radiografías transformó la Medicina.

Sin embargo, el momento histórico que hace que estas técnicas confluyan y den un cuerpo de conocimiento propio, surge en la década de los 80 del siglo XX. La revolución de la Electrónica, con las cámaras de vídeo CCD y los microprocesadores, junto con la evolución de las Ciencias de la Computación hace que sea factible la Visión Artificial. Por tanto, pretende capturar la información visual del entorno físico para extraer características relevantes visuales, utilizando procedimientos automáticos.

La visión artificial o también llamada visión por computador es una disciplina científica que incluye métodos para adquirir, procesar, analizar y comprender las imágenes del mundo real con el fin de producir información numérica o simbólica para que puedan ser tratados por un computador. Tal y como los humanos usamos nuestros ojos y cerebros para comprender el mundo que nos rodea, la visión por computador trata de producir el mismo efecto para que las computadoras puedan percibir y comprender una imagen o secuencia de imágenes y actuar según convenga en una determinada situación [5].

En la actualidad no hay integrados sistemas de control de suciedad en cristales con el uso de visión artificial. Este será el primer proyecto en estudiar y desarrollar de una forma eficaz la inspección de los cristales en plantas termosolares con espejos planos.

En conclusión, la integración de sistemas de visión artificial en los procesos productivos es una apuesta clara en la industria desde hace años. El control de calidad del 100% de la producción, la inspección en tiempo real, la verificación objetiva y constante, la posibilidad de comunicación de resultados a un ordenador o control son grandes beneficios que nos aporta esta tecnología. La optimización de los procesos productivos con sistemas de visión artificial produce una reducción de costes y una mayor calidad y seguridad en el control de los productos.

También, nos permite corregir errores antes de producir residuos, inspección de áreas de difícil acceso o detectar productos defectuosos se convierten en tareas fáciles y automatizadas gracias a la visión artificial.

Además, los costes cada vez más asequibles y los constantes y potentes avances en las tecnologías de visión influyen en la alta demanda de soluciones de visión artificial en la industria.

1.4 REQUERIMIENTOS.

Se pretende adoptar una solución para la supervisión de los cristales de la planta termosolar.

El problema actual es que la empresa realiza la inspección de la planta con un operario. Esta tarea resulta pesada por dos causas; la primera es pasar por debajo de los cristales que alcanzan altas temperaturas y los reflejos que estos producen son incapacitantes; la segunda es el tiempo que conlleva la inspección porque la superficie donde se encuentran los espejos es bastante grande, aproximadamente 300000 metros cuadrados. En definitiva, estos problemas le suponen a la empresa bastante tiempo y daño físico al personal.

Cada cierto tiempo se desmontan y limpian para que la luz del sol penetre y caliente el agua de la tubería, cuanto más limpio este el cristal mayor será el rendimiento.
El inconveniente de la limpieza es que mientras se limpian se para la producción de vapor de agua en ese sector.
Las fases a seguir para desarrollar el trabajo son:

- Estudiar las condiciones ambientales.
- Seleccionar la cámara, lente, objetivo y filtros.
- Ajustar los parámetros de la cámara.
- Capturar imágenes con los parámetros deseados.
- Desarrollar el software para analizar las imágenes.
- Procesar las imágenes: distinguir entre los distintos grados de suciedad.
- Montar el hardware en plataforma móvil para capturar las fotografías.
- Implementar las fases.

El sistema a diseñar constará de una cámara industrial, en ella habrá acoplado un objetivo ya que los cristales AR están a una altura de unos ocho metros. Se estudiará el uso de un filtro para mejorar las imágenes. La conexión al pc será con cable ethernet. Todo esto estará montado sobre una plataforma móvil, la plataforma se colocará en los espejos inferiores (reflectores lineales fresnel), estos espejos son los que se mueven para orientarlos de tal forma que el sol incide sobre ellos y refleja en los reflectores secundarios. Para poder lograr el ángulo deseado dicha plataforma llevará incorporado un nivel y un encoder que nos dará la información necesaria para saber lo que ha avanzado la plataforma y capturar las imágenes.

- Herramientas usadas al detalle:
    - Software:
        - Matlab
        - Visual Studio 2017
        - JaiTools (Software de la cámara)
        - Sistema operativos Linux y Windows 10
        - Librería opencv
    - Hardware:
        - Pc
        - Cámara Go-5000-PGE
        - Objetivo estándar modelo VS-5018H1 de 1900mm
        - Filtro MIDOPT BP470
        - Nivel
        - Encoder
        - Cable ethernet y de alimentación para cámara.

1.5 OBJETIVOS.

El objetivo del proyecto es crear un software para la inspección de los cristales y así disminuir el tiempo en su revisión.
Los cristales se encuentran a la intemperie expuestos a los factores ambientales y tienden a ensuciarse con el polvo.

La idea para mejorar el sistema actual de limpieza es crear un software que determine el grado de suciedad de los cristales, para que nos indique cuando hay que limpiarlos. Se ha pensado definir cinco niveles de suciedad que indiquen:

- Nivel 1: cristal muy limpio, recién limpiado o hace pocas semanas que se limpió.
- Nivel 2-3: cristales limpios que aún permiten el paso de la luz del sol. Por el momento no hace falta que se limpien.
- Nivel 4: cristal sucio, se debería plantear una próxima limpieza.
- Nivel 5: cristal muy sucio, se debería de limpiar inmediatamente. Estos cristales son casi opacos de tanta suciedad que contienen.

El hardware consta de una plataforma móvil con cámara más un pc con sistema operativo Windows.

El software se unificará al hardware con la tecnología ethernet.

Para cumplir con el objetivo principal debemos alcanzar los siguientes objetivos específicos:

1. Elegir la cámara adecuada para el proceso, así como los accesorios que la complementan.
2. Elegir los parámetros adecuados para la captura de imágenes.
3. Desarrollar el software que determine el nivel de suciedad.
4. Capturar las imágenes para probar y mejorar el algoritmo.
5. Probar el software desarrollado junto con el hardware (plataforma móvil) para comprobar posibles errores con la cámara.

Una vez conseguidos estos objetivos ya se podrá lanzar el robot para la inspección de los cristales.

1.6 PLANIFICACIÓN.

Se crea una planificación en diagrama de GANTT de manera resumida con la planificación esperada por tareas principales, esta planificación puede sufrir cambios debido a algún retraso, posibles complicaciones o desvío del objetivo, pero siempre centrado en el tema principal que es la detección de suciedad en cristales AR en una planta termosolar.

1. Lectura de documentación técnica y recopilación de información. Presentación de la propuesta y ajustes técnicos con el tutor.

2. Lectura de documentación y búsqueda de información de procesos industriales basados en visión artificial. Redacción de los primeros puntos para la elaboración del índice del TFE.
3. Revisar documentación técnica y teórica. Puesta en marcha de la cámara e instalación de software.
4. Estudio de las condiciones ambientales y cómo afecta a la toma de fotografías.
5. Elección de los parámetros adecuados para la captura de imágenes.
6. Poner en marcha la creación del algoritmo. Realizar la documentación de los logros y avances en el TFE.
7. Unificación del software y el hardware.
8. Estudio de la robustez del programa creado.
9. Pruebas finales.
10. Redacción y revisión de la memoria.

Diagrama GANTT:

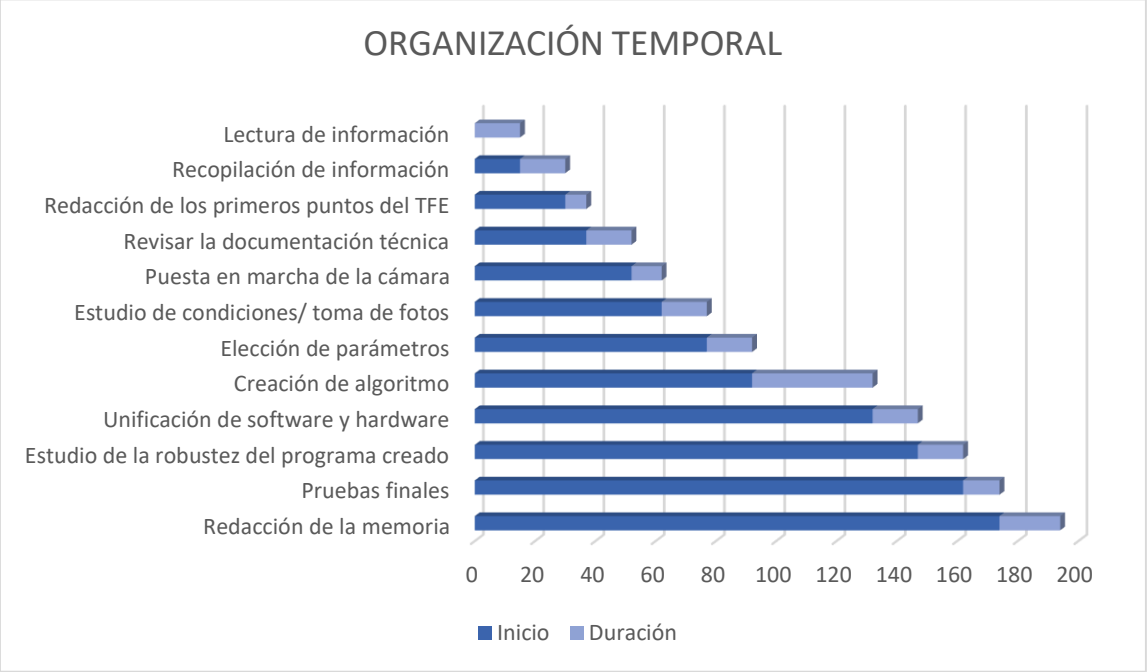


Figura 3: Gantt. Fuente: Propia.

CÁPITULO 2

Conceptos de visión artificial.

2.1 INTRODUCCIÓN

En este capítulo se abordará la búsqueda de información para desarrollar dicho trabajo. Se comentará lo que es un sistema de visión artificial, de que está compuesto y los beneficios que aporta. Así como, también se hablará de los diferentes tipos de cámaras.

2.2 SISTEMAS DE VISIÓN ARTIFICIAL

2.2.1 DEFINICIÓN Y CARACTERÍSTICAS

Se puede definir la "Visión Artificial" como un campo de la "Inteligencia Artificial" que, mediante la utilización de las técnicas adecuadas, permite la obtención, procesamiento y análisis de cualquier tipo de información especial obtenida a través de imágenes digitales. La visión artificial la componen un conjunto de procesos destinados a realizar el análisis de imágenes. Estos procesos son: captación de imágenes, memorización de la información, procesado e interpretación de los resultados [6].

Con la visión artificial se pueden:

- Automatizar tareas repetitivas de inspección realizadas por operadores.
- Realizar controles de calidad de productos que no era posible verificar por métodos tradicionales.
- Realizar inspecciones de objetos sin contacto físico.
- Realizar la inspección del 100% de la producción (calidad total) a gran velocidad.
- Reducir el tiempo de ciclo en procesos automatizados.
- Realizar inspecciones en procesos donde existe diversidad de piezas con cambios frecuentes de producción.

Características:

- Métodos de captación de las imágenes.
    - Digital. La función obtenida tras el resultado de la medida o muestreos realizados a intervalos de tiempo espaciados regularmente, siendo el valor de dicha función un número positivo y entero. Los valores que esta función toma en cada punto dependen del brillo que presenta en esos puntos la imagen original.

- Píxel. Una imagen digital se considera como una cuadrícula. Cada elemento de esa cuadrícula se llama Píxel (Picture element). La resolución estándar de una imagen digital se puede considerar de 512x484 Pixel.
- Nivel de grises. Cuando una imagen es digitalizada, la intensidad del brillo en la escena original correspondiente a cada punto es cuantificada, dando lugar a un número denominado "nivel de gris".
- Imagen binaria. Es aquella que sólo tiene dos niveles de gris: negro y blanco. Cada píxel se convierte en negro o blanco en función del llamado nivel binario o UMBRAL.

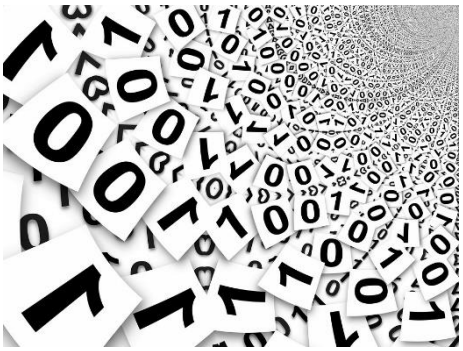


Figura 4: Sistema binario. Fuente: areadetecnologia.com

- Escena. Es un área de memoria donde se guardan todos los parámetros referentes a la inspección de un objeto en particular: Cámara utilizada, imágenes patrón memorizadas, tolerancias, datos a visualizar, entradas y salidas de control, etc.
- Window (ventana de medida). Es el área específica de la imagen recogida que se quiere inspeccionar.

2.2.2 COMPONENTES DE UN SISTEMA DE VISIÓN ARTIFICIAL.

El buen desempeño de un sistema de visión depende en gran parte de los componentes por los que está formado, existiendo seis primordiales para que el sistema funcione adecuadamente.

- Captación: es el proceso a través del cual se obtiene una imagen visual.

- Preprocesamiento: incluye técnicas tales como la reducción de ruido y realce de detalles.
- Segmentación: es el proceso que divide a una imagen en objetos que sean de nuestro interés.
- Descripción: es el proceso mediante el cual se obtienen características convenientes para diferenciar tipos.
- Reconocimiento: es el proceso que asocia un significado a un conjunto de objetos reconocidos.

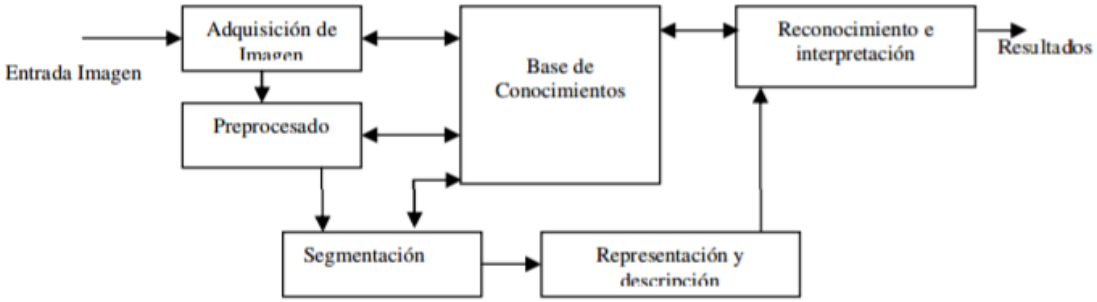


Figura 5: Etapas de un sistema de visión artificial. Fuente: http://blog.i-mas.com

Los componentes básicos de un sistema de visión se resumen a continuación:

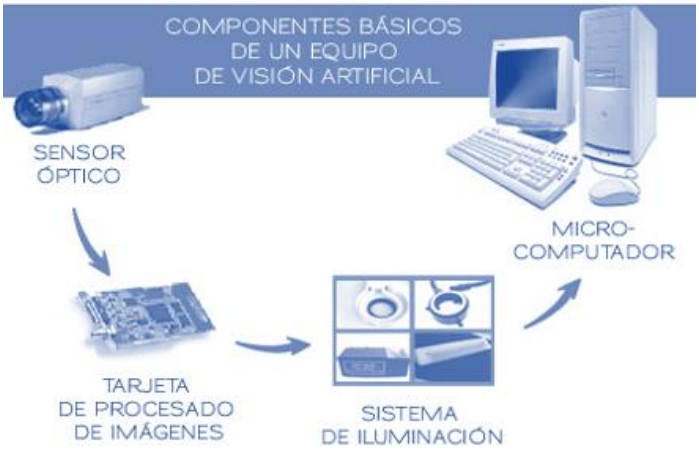


Figura 6: Componentes de un sistema de visión. Fuente: http://blog.i-mas.com

1) **Iluminación**. La iluminación es la parte más crítica dentro de un sistema de visión. Las cámaras capturan la luz reflejada de los objetos. El propósito de la iluminación utilizada en las aplicaciones de visión es:

a. Obtener una imagen en las mejores condiciones para un posterior análisis.
b. Mantener constante la intensidad y la dirección de la luz.
c. Optimizar contraste. Separar características de la imagen del fondo.

Hay un cierto número de consideraciones a tener en cuenta para determinar la mejor iluminación para una aplicación:

- Intensidad de luz necesaria.
- Longitud de onda adecuada.
- Superficie a iluminar.
- Reflectividad del objeto.
- Color del objeto.
- Espacio disponible.
- Tipo de cámara utilizada.

La iluminación podrá ser mediante fibra óptica, fluorescente, led, difusa o láser:

- La iluminación mediante **fibra óptica** proporciona una gran intensidad de luz uniforme, con ausencia de sombras. Es ideal para iluminar objetos de reducidas dimensiones y se puede sujetar al objetivo de la cámara o a la óptica de un microscopio. A los anillos de luz se les puede acoplar filtros de colores, polarizadores/analizadores, y difusores para eliminar reflejos y aumentar el efecto difusor.

- La iluminación mediante **fluorescentes** proporciona una luz brillante, sin sombras. Existen lámparas blancas en distintas temperaturas de color, Y también ultravioletas (UV). Esta iluminación se aplica en entornos que requieren mucha luz, y ningún tipo de sombra, (análisis biológicos, inspección y la microscopía, ensamblaje, inspección de circuitos, industria, laboratorios, visión industrial, fotografía, control de calidad, robótica, etc…)

- La iluminación mediante **diodos led** proporciona una luz difusa muy útil para la aplicación en ciertos objetos. Puede ser de iluminación directa y en anillo.

- La iluminación mediante **láser** se utiliza mayoritariamente en aplicaciones de medida de profundidad y de superficies irregulares. Mediante ópticas especialmente diseñadas, se puede convertir un puntero láser en diferentes formas y tamaños.

2) Cámara y ópticas (Explicadas en el apartado siguiente)
3) CPU

   Se encarga de recoger y mostrar las imágenes capturadas, y de procesarlas para llevar a cabo su cometido. Las tareas a realizar son:

   - Recibir todas aquellas señales de sincronización para que se pueda realizar correctamente la captura de imágenes.
   - Realizar la lectura de las imágenes.
   - Procesar los datos proporcionados por las cámaras para realizar el análisis de imagen.
   - Realizar la interfaz con los usuarios.
   - Comunicar con los sistemas productivos, para detener el proceso en caso de la aparición de algún defecto.
   - Controlar el buen funcionamiento de todos los elementos hardware.

4) Frame grabbers o tarjetas de adquisición.

   Se denomina frame grabber al dispositivo que interconexiona una cámara con un PC, digitalizando y guardando en memoria la imagen adquirida. Las tarjetas se dividen en tres categorías distintas en función de sus características: Frame Grabbers estándar de bajo coste, Frame Grabbers avanzados de altas prestaciones y con características multicanal y Frame Grabbers "inteligentes" con procesadores abordo.

Figura 7: Frame grabber. Fuente: Infaimon.

5) Software

   El software de visión se compone de una serie de herramientas tanto de preproceso como de proceso que analizan la imagen y extraen información de la misma según los algoritmos en los que estén basadas dichas herramientas.

   La mayoría de las aplicaciones se pueden solucionar tanto con librerías de programación como con software basados en interfaz gráfico. La elección del tipo de software depende más de la aplicación que del usuario. Aparecen herramientas de software basadas en redes neuronales, específicas para las aplicaciones ITS o aplicaciones de seguridad de alto nivel.

2.2.3 EJEMPLOS O APLICACIONES.

El objetivo de la visión artificial es extraer características de una imagen para su descripción e interpretación por el PC.

La base del software de un sistema de visión es la interpretación y análisis de los pixeles. El resultado final puede ser, desde la medida de una partícula, a la determinación o lectura de una serie de caracteres (OCR), pasando por cualquier otro proceso que podamos imaginar sobre las imágenes.

Dependiendo de si la aplicación se realiza en entorno industrial o científico los pasos a seguir en un sistema de visión serán algo distintos [7].

Aplicación industrial

- Captura de la imagen.
- Definición de la región de interés donde se realizarán las medidas.
- Inicialización de las tolerancias para determinar si la pieza a determinar es o no correcta.
- Ejecutar las medidas.
- Generar una salida apropiada.

Aplicación científica

- Capturar la imagen.
- Hacer un proceso de mejora.
- Determinar los elementos a medir.
- Ejecutar la medida.

- Almacenar las medidas y realizar procesos gráficos o estadísticos.

Las principales aplicaciones de la visión artificial en la industria actual son: identificación e inspección de objetos; determinación de la posición de los objetos en el espacio; establecimiento de relaciones espaciales entre varios objetos (guiado de robots); determinación de las coordenadas importantes de un objeto; realización de mediciones angulares; mediciones tridimensionales.

Ejemplo:

En automoción: verificación del filtro en un tanque de combustible

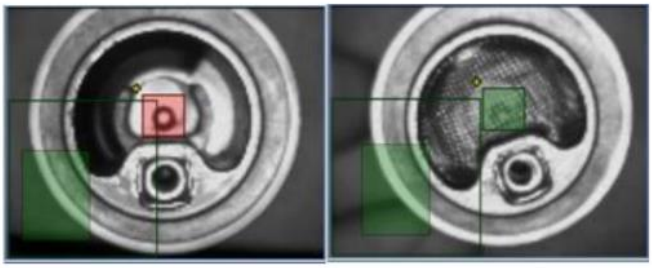


Figura 8: Verificación del filtro de un tanque. Fuente: BcnVision.

2.2.4 BENEFICIOS DE UN SISTEMA DE VISIÓN ARTIFICIAL.

Los sistemas de visión artificial efectúan tareas repetitivas con precisión y rapidez y permiten trabajar fuera del espectro visible distinguiendo detalles no visibles por el ojo humano y aportando numerosos beneficios, siendo los más inmediatos el incremento de la calidad y del rendimiento de la producción y la reducción de costes de mano de obra [7].

Integrados en etapas intermedias de la producción, la reducción de costes es doble: la extracción de la pieza antes de que esté acabada supone un ahorro en materiales y consumo energético [8] y además permite detectar problemas en los dispositivos que originaron el defecto evitando fabricar más piezas defectuosas.

Las prestaciones de la visión mejoran las de las demás tecnologías en todos los campos:

- Calidad en la producción. Producción 100% fiable: verificación objetiva y constante (unitaria).

- Aumentar la producción (optimizar tiempos de producción), reducción de costes, por ejemplo: costes de devolución de pedidos, costes de la imagen de la empresa frente a los clientes, costes de personal, costes de tiempo o costes de procesos productivos.
- Evita errores humanos, falta de atención, errores visuales, absentismo laboral, verificación objetiva y constante o verificación de lugares inaccesibles.
- Detección de defectos sobre la misma línea de producción: Se han desarrollado nuevas tecnologías como la visión multiespectral, que permiten ver más allá de lo que es capaz de captar el sistema visual humano. De este modo, ampliando la respuesta de las cámaras a los espectros ultravioleta e infrarrojo, es posible detectar defectos y problemas.
- Detección de cuerpos extraños: Una de las últimas aplicaciones de la visión artificial es el análisis de imágenes procedentes de señales que son capaces de atravesar la muestra, como los rayos-X, la resonancia magnética nuclear o la termografía. A través de estas técnicas, es posible detectar cuerpos extraños de pocos milímetros.
- Medición sin contacto.
- Sistemas multirecetas.
- Resolver problemas de seguridad y ergonomía.
- Alta velocidad de procesado: Sistema de alta velocidad y resolución capaz de conseguir ratios de inspección de más de 1000 productos por segundo.

2.3 CÁMARAS

La función de una cámara es capturar la imagen proyectada en el sensor, vía las ópticas, para poder transferirla a un sistema electrónico. Las cámaras utilizadas en visión artificial requieren de una serie de características que permitan el control del disparo de la cámara para capturar piezas que pasan por delante de ella en la posición requerida. Son más sofisticadas que las cámaras convencionales, ya que tienen que poder realizar un control completo de: tiempos, señales, velocidad de obturación, sensibilidad, etc. [9]

Se clasifican en función de:

- La tecnología del elemento sensor.

- Cámaras de tubo. Se basan en la utilización de un material fotosensible que capta la imagen, siendo leída por un haz de electrones.
- Cámaras de estado sólido CCD (Charge – Coupled – Device). Se basan en materiales semiconductores fotosensibles para cuya lectura no es necesario un barrido electrónico (más pequeñas que las de tubo).

- La disposición física.
  - Cámaras lineales. Se basan en un sensor CCD lineal
  - Cámaras matriciales. Se basan en un sensor CCD matricial, lo que permite el análisis de imágenes bidimensionales. Hay una cámara específica para cada aplicación, color, monocromo, alta definición, alta sensibilidad, alta velocidad, infrarrojas, etc.

Pasamos a comentar en forma breve el funcionamiento de las más utilizadas.

2.3.1 CÁMARA LINEAL.

El concepto de barrido lineal se asocia a la construcción de una imagen línea a línea, utilizando un sensor lineal, de forma que la cámara se desplaza con respecto al objeto a capturar, o bien el objeto se desplaza con respecto a la cámara.

La tecnología de cámaras lineales fue desarrollada para aplicaciones de inspección de materiales fabricados en continuo, como papel, tela, planchas metálicas, etc. Sin embargo, en la actualidad se está imponiendo en muchos otros procesos productivos y de inspección, que requieren alta resolución.

Las cámaras lineales utilizan sensores que tienen entre los 512 y 8192 pixels, con una longitud lo más corta posible y gran calidad de imagen. El hecho de construir imágenes de alta calidad a partir de líneas individuales, requiere de una alta precisión. La alineación y el sincronismo del sistema son críticos si se quiere obtener una imagen precisa del objeto a analizar [10].

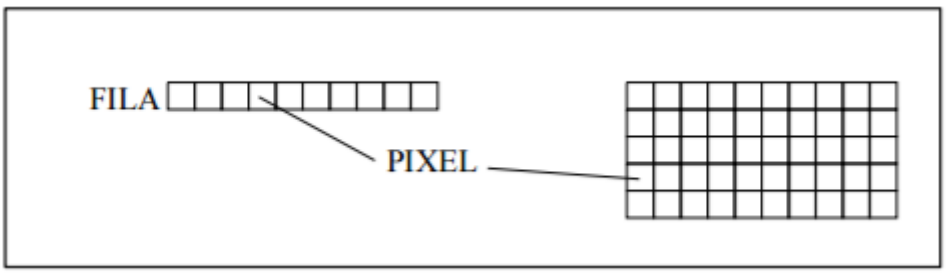


Figura 9: Barrido lineal. Fuente: Infaimon

Su utilización está muy extendida para la inspección de objetos de longitud indeterminada, tipo telas, papel, vidrio, planchas de metal, etc.

- Características técnicas:
    - Número de elementos del sensor. A mayor número de elementos (pixels) mayor tamaño de la óptica.
    - Velocidad. Número de pixels capaces de ser leídos por unidad de tiempo. En las cámaras lineales es un valor mucho más alto que en las matriciales. En las cámaras de última generación se alcanzan velocidades superiores a los 200 Mhz.
    - Cámaras lineales a color. Tienen tres sensores lineales, uno para cada color (rojo verde y azul). Pueden ser de dos tipos:
        - Trisensor. Los sensores CCD están posicionados unos junto a otros separados por un pequeño espacio. Tienen una buena sensibilidad, pero solo pueden utilizarse en aplicaciones con superficies planas.
        - Prisma. Los sensores están posicionados en las tres caras de un prisma. Pueden utilizarse para cualquier tipo de aplicación, pero necesitan de una mayor iluminación.

Figura 10: Cámara lineal. Fuente: Infaimon

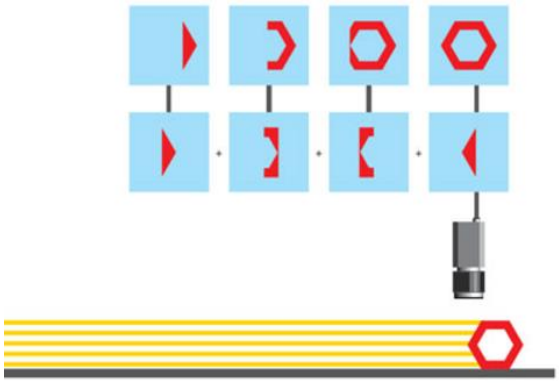


Figura 11: Obtención de imagen mediante barridos sucesivos

2.3.2 CÁMARAS MATRICIALES.

Las cámaras matriciales o de área, pertenecientes al grupo de cámaras de visión artificial, son aquellas en que el sensor de la cámara cubre un área o que está formado por una matriz de píxeles. Una cámara matricial produce una imagen de un área, normalmente con una relación de aspecto de 4 a 3. Esta relación viene de los tiempos de las cámaras Vidicón y de los formatos de cine y televisión. Actualmente existen muchas cámaras que ya no mantienen esta relación y que no siguen los formatos de la televisión, o se adaptan a los nuevos formatos de alta definición 16:9.

El sensor cubre un área que está formada por una matriz de pixels. Los sensores de cámaras modernos son mayoritariamente CCD (Charge Coupled Devices) que utilizan material sensible a la luz para convertir los fotones en carga eléctrica. Miles de diodos sensibles se posicionan de forma muy precisa en una matriz y los registros de desplazamiento transfieren la carga de cada píxel para formar la señal de video. El tamaño de los CCD está definido en pulgadas, sin embargo, su tamaño real no tiene nada que ver con su valor en pulgadas, sino que están basados en la relación de los primeros con el tamaño de los tubos Vidicon [11].

Figura 12: Cámara matricial. Fuente: Infaimon

Características de los sensores:

- Factor de relleno. El factor de relleno es el porcentaje del área de píxel que es sensible a la luz. El caso ideal es 100%, cuando los píxels activos ocupan el 100% del área del sensor. Sin embargo, circuitos como los registros de lectura y los circuitos anti-blooming reducen este factor, en algunas ocasiones hasta al 30%. El efecto de esta reducción se traduce en una menor sensibilidad y en efectos de aliasing. Para mejorar esto, muchos sensores con bajo factor de relleno (normalmente CCD) utilizan microlentes que cubren cada uno de los píxels incrementando la efectividad del factor de relleno [12].

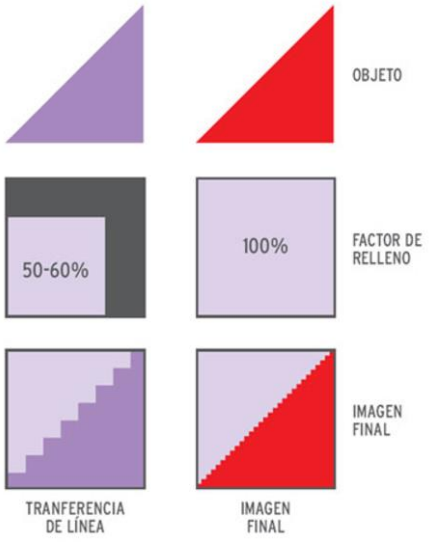


Figura 13: Factor de relleno. Fuente: Infaimon

- Tipo de transferencia. Según la forma de transferencia de la información.
  - Transferencia Inter-línea (ITL). Son los más comunes, utilizan registros de desplazamiento situados entre las líneas de píxel para almacenar y

transferir los datos de la imagen lo que permite una alta velocidad de obturación.

- Transferencia de cuadro. Disponen de un área dedicada al almacenamiento de la luz, la cual está separada del área activa, esto permite un mayor factor de relleno, aunque se pierde velocidad de obturación.
- Cuadro entero (Full Frame). Son los CCD que tienen una arquitectura más simple. Emplean un registro paralelo simple para exposición de los fotones, integración de la carga y transporte de la carga. Se utiliza un obturador mecánico para controlar la exposición. El área total del CCD está disponible para recibir los fotones durante el tiempo de exposición. El factor de relleno de estos tipos de CCD es del 100%.

2.3.3 CAMARAS COLOR

Tanto en aplicaciones industriales como científicas cada vez se están utilizando más las cámaras color. Aunque el proceso de las imágenes a color es más complejo. Este tipo de cámaras pueden proporcionar más información que la cámara monocromo.

- Cámaras Color 1CCD

  Las cámaras color de 1 CCD incorporan un sensor con un filtro en forma de mosaico que incorpora los colores primarios RGB. Este filtro de color es conocido como filtro Bayer. De hecho, el sensor es un sensor monocromo al que se le ha superpuesto el filtro de color. La forma en que se disponen los colores R, G y B es como se muestra en la figura 14. Como se puede ver hay el doble de pixels con filtro verde que con filtro azul o rojo. Esto es así para hacer más semejante el sensor a la visión humana que es más sensible al verde.

  Debido al carácter del propio filtro, es evidente que en los pixels donde se sitúa el filtro rojo, no tienen señal ni de verde ni de azul. Para subsanar la falta de estos colores en estos pixels, se construye una señal RGB a partir de los pixels adyacentes de cada color. Este cálculo se realiza en el interior de la cámara mediante un circuito DSP específico, que permite realizar la operación en tiempo real y dependiendo de la cámara permite obtener una señal analógica o digital en RGB en su caso [12].

Algunas cámaras transmiten la señal del sensor a de forma digital hacia el exterior de la cámara y el ordenador mediante un programa apropiado realiza la transformación y permite visualizar la imagen en color.

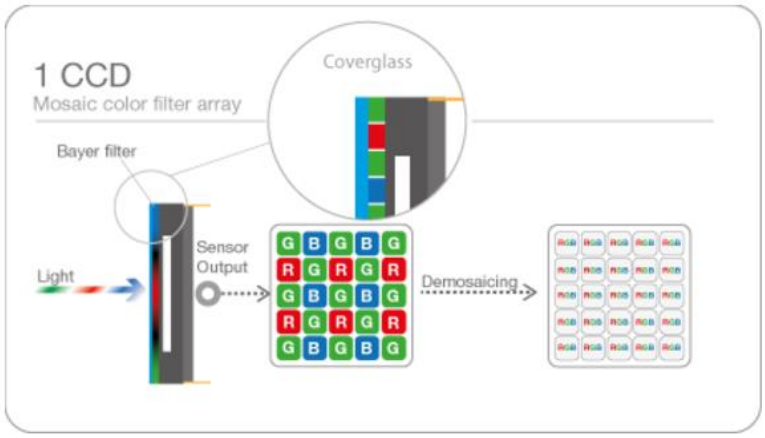


Figura 14: Filtro de cámara a color. Fuente: Infaimon

- Cámara Color 3CCD

Las cámaras color 3CCD incorporan un prisma y tres sensores. La luz procedente del objeto pasa a través de la óptica y se divide en tres direcciones al llegar al prisma. En cada una de los tres extremos del prisma se encuentra un filtro de color (rojo, verde y azul) y un sensor que captura la luz de cada color que viene del exterior. Internamente la cámara combina los colores y genera una señal RGB similar a la que ve el ojo humano. La fidelidad de las imágenes de las cámaras de 3CCD es muy superior a las de las cámaras de 1 CCD, pero hay un par de inconvenientes inherentes al sistema. Por una parte, este tipo de cámaras requieren más luz debido a que el prisma hace que sea menor la cantidad de iluminación que incide sobre los sensores, y por otra se genera un efecto de aberración cromática debida a la propia estructura del prisma. Este efecto puede ser subsanado colocando las ópticas diseñadas específicamente para este tipo de cámaras.

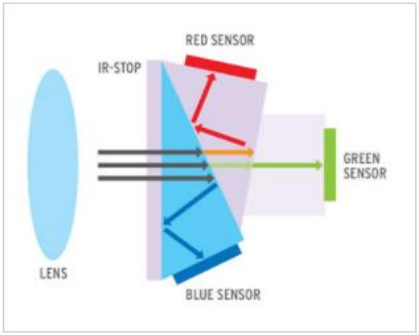


Figura 15: Cámara 3 CCD. Fuente: Infaimon

2.4 ÓPTICA

Las ópticas se utilizan para transmitir la luz al sensor de la cámara de una forma controlada y así obtener una imagen enfocada del objeto. El objetivo es conseguir la mejor imagen posible que nos permita realizar mediciones precisas, con un alto contraste y con la menor distorsión geométrica posible [13].

- Para determinar cuál es el objetivo apropiado para cada tipo de aplicación se debe tener en cuenta algunos parámetros como: Tamaño y especificaciones del sensor de la cámara.
- Distancia entre la cámara y el objeto.
- Campo de visión y tamaño del objeto.

Con estos datos y utilizando fórmulas matemáticas podemos calcular la óptica más apropiada para cada aplicación. Muchos fabricantes de objetivos también facilitan programas de cálculo para determinar cuál es el objetivo idóneo para la aplicación que necesitamos. A la práctica, le elección de la óptica correcta no es algo tan obvio y deben tenerse en cuenta también otros aspectos como la iluminación, la concordancia entre la calidad de la óptica y el sistema de visión en general y sobre todo las necesidades particulares de cada una de las aplicaciones.

2.4.1 ÓPTICA ESTANDAR.

Las ópticas estándares son las más utilizadas en aplicaciones de visión artificial. Las gamas diseñadas para aplicaciones industriales suelen estar disponibles en distintas distancias focales desde 3.5 a 200 mm y se caracterizan por ser robustas y resistentes a vibraciones y golpes. Estas ópticas están optimizadas para enfocar desde infinito hasta pocos centímetros del objeto. Aun así, por buenas que sean, las ópticas siempre realizan alguna distorsión en la imagen captada. Estas distorsiones o aberraciones son el cambio de la representación geométrica de un objeto en el plano de la imagen. Por ejemplo, un rectángulo puede aparecer como una figura geométrica curvada hacia dentro (almohadilla) o hacia fuera (barril).

Figura 16: Óptica estándar. Fuente: Infaimon

La distorsión de la imagen puede causar serios problemas en las aplicaciones de visión artificial. En las características técnicas de las ópticas acostumbran a expresar en porcentaje la distorsión producida por la misma.

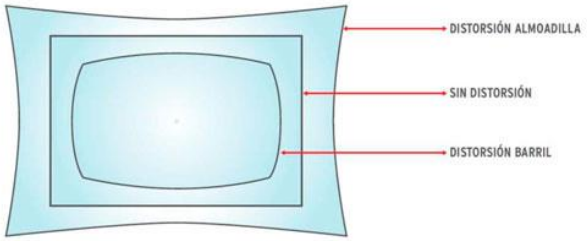


Figura 17: Distorsión de la óptica. Fuente: Infaimon

Dependiendo de la forma geométrica del objeto a analizar podemos tener problemas con las mediciones del mismo ya que las ópticas estándares se ven afectadas por el ángulo de adquisición de la imagen. Para solucionar este problema existen ópticas telecéntricas con las que se asegura una visión totalmente paralela de los objetos a analizar.

Son ideales para la mayoría de cámaras estándar, este tipo de ópticas diseñadas para aplicaciones industriales utilizan cristales de calidad y son de construcción robusta, para poder resistir las vibraciones y golpes sin que se desenfoque o cambie la apertura están provistas de tornillos de fijación para el iris y el enfoque [14].

2.4.2 ÓPTICA TELECÉNTRICA.

Los objetivos telecéntricos ofrecen las mejores prestaciones en términos de resolución, telecentricidad, profundidad de campo y distorsión y se utilizan para el desarrollo de

aplicaciones de medición de alta precisión. Para la construcción de una óptica telecéntrica la lente tiene que ser igual o mayor que el objeto y de gran calidad. Esto hace que este tipo de ópticas sean aproximadamente 10 veces más caras que las ópticas estándar. Están diseñadas especialmente para medir objetos con cierta profundidad de campo.

Las ópticas telecéntricas eliminan la distorsión haciendo que la luz incida completamente perpendicular (colimada) al sensor. El resultado es que se obtiene una magnificación igual independientemente de la distancia del objeto y eliminando cualquier efecto de perspectiva.

Figura 18: Óptica telecéntrica. Fuente: Infaimon

Las ópticas telecéntricas están basadas en el principio de proyección paralela. En este caso el centro de perspectiva se encuentra en el infinito. Como si se tratara de una representación ideal en un dibujo técnico, estas lentes telecéntricas aseguran una visión virtualmente paralela de objetos de tres dimensiones y de los detalles que deben medirse en estos objetos. Esto significa que los objetos del mismo tamaño y a diferentes distancias de la óptica aparecerán siempre con el mismo tamaño en la imagen.

Se disponen de ópticas telecéntricas adaptables a cámaras estándar con adaptador a rosca C y también se dispone de ópticas telecéntricas para cámaras de altas resoluciones tanto matriciales como lineales.

Las ópticas estándares provocan que algunos factores limiten la exactitud y la repetitividad de la medición provocando:

- Cambios de aumento debido a los movimientos del objeto.
- Distorsión de la imagen.
- Errores de perspectiva.
- Baja resolución de imágenes.

- Incertidumbre sobre las posiciones de los bordes, debido a la geometría de la iluminación.

Estas limitaciones pueden resultar factores críticos en algunas aplicaciones de visión artificial. Algunos de estos aspectos pueden ser eliminados o reducidos mediante la utilización de ópticas telecéntricas.

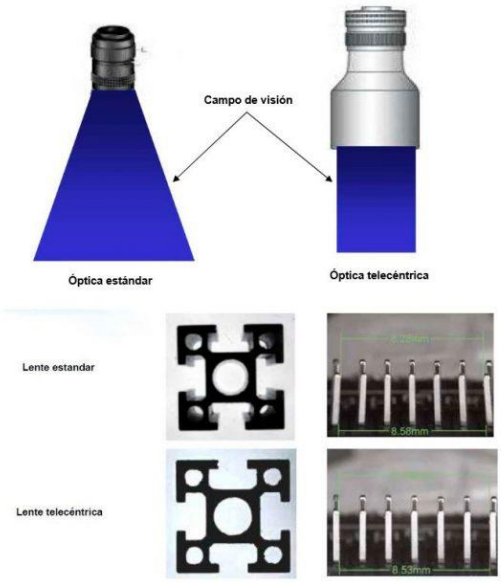


Figura 19: Óptica estándar vs telecéntrica. Fuente: bcnVisión

2.5 FILTRO

Para entender el funcionamiento de un filtro, tenemos que entender qué es la luz y cómo se comporta desde un punto de vista meramente físico. La luz visible es una porción del espectro electromagnético de una determinada longitud de onda [14].

Anteponiendo un filtro en la óptica, podremos seleccionar qué longitud de onda le llega a los fotoelectroreceptores del sensor, captando únicamente aquella que resulten más favorables para el objetivo del proyecto.

En muchas aplicaciones de visión artificial es necesario que la cámara solo reciba una determinada longitud de onda procedente del objeto. Para conseguir esto, pueden utilizarse fuentes de luz monocromáticas, o bien pueden utilizarse filtros que solo dejen pasar o que corten ciertas longitudes de onda. Los filtros más utilizados en los sistemas de visión artificial son los filtros de paso o de corte de infrarrojo [9].

Sin embargo, en aplicaciones de color, utilizando cámaras monocromo, se usan filtros para resaltar los colores y de esa forma mediante una cámara monocromo resaltar un determinado color. También son frecuentemente utilizados en combinación con sistemas de luz estructurada láser, con el fin de ajustar la captura de la cámara a la longitud de onda del láser [15].

Figura 20: Filtro. Fuente: Infaimon

Conceptos básicos:

- Los filtros paso banda: permiten el paso selectivo de longitudes de onda comprendidas en un rango determinado.
- Los filtros paso bajo y paso alto: son filtros de corte, ya que permiten el paso de las longitudes de onda por debajo o por encima de un determinado valor.
- Filtros de corte: Bloquean el paso de ciertas longitudes de onda.

Otra de las utilidades de los filtros es atenuar o eliminar el ruido transmitiendo únicamente la componente de nuestro interés.

El uso de los filtros se recomienda habitualmente, aunque sea sin función de filtración, con el fin de proteger la óptica principal ante suciedad o deterioros.

CÁPITULO 3

Selección de hardware y software.

3.1 HARDWARE.

En la selección del hardware se debe elegir primero el método de captura más adecuado al proceso descrito anteriormente.

En un principio se barajaban varias ideas:

- Dron más cámara de video → descartado por inestabilidad para capturar las fotografías.
- Cámara digital CANON modelo EOS 7D → también descartada porque este tipo de cámaras no tienen una interfaz adecuada para conectarla al sistema de adquisición de imágenes.

Figura 21: Canon EOS 7 D. Fuente: Propia

Las fotos con la cámara CANON con el diafragma cerrado quedan así:

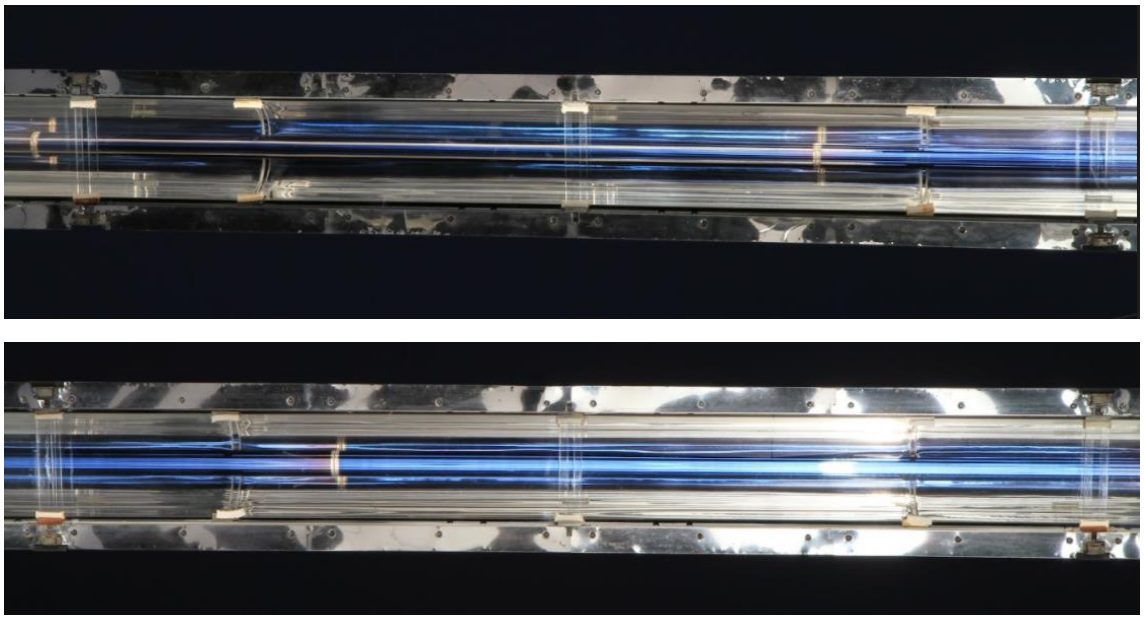


Figura 22: Fotografía cámara CANON. Fuente: Propia

- Cámaras de visión artificial: se decide usar la monocromo porque para detectar el nivel de suciedad es suficiente usar la escala de grises. Dentro de este tipo están las matriciales (produce la imagen de un área) o lineales (construcción de la imagen línea a línea).

Se concluye usar la cámara GO-5000-PG

3.1.1 CÁMARA.

La cámara GO-5000-PG tiene un tamaño reducido, versatilidad con excelentes prestaciones y el precio más económico del mercado en cámaras de 5MP global shutter, lo que la convierte en la cámara ideal para el desarrollo de nuevos proyectos con cámaras de visión artificial.

Se trata de una cámara de 5 megapíxeles CMOS con una carcasa muy compacta. La combinación de las capacidades de ROI y binning convierten esta pequeña cámara en la ideal para un amplio rango de posibilidades, desde una cámara VGA súper rápida (hasta 450 imágenes por segundo) hasta una cámara de alta sensibilidad utilizando binning para crear tamaño de pixel de 10µm o incluso 20µm.Sus características son [16]:

- Dimensiones: 29x29x41.5mm.
- Peso: 46g.
- Sensor CMOS de altas prestaciones es capaz de realizar hasta 22 imágenes por segundo a la máxima resolución (5 megapíxeles).
- Cuenta con Global-shutter, tamaño de pixel de 5µm.
- Control de ganancia analógica y lookup table integrado.
- Salida Gigabit Ethernet de 8/10/12-bit.
- Alimentación vía Power-over-Gigabit Ethernet.

A la cámara irá acoplada una óptica estándar modelo VS-5018H1, esta óptica está optimizada para enfocar desde infinito a pocos centímetros del objeto. Se usa uno de 1900mm para enfocar todo el cristal y un poco de los laterales [14].

Figura 23: Cámara JAI más óptica. Fuente: Propia

3.1.2 FILTRO

Utilizando cámaras monocromas, se usan filtros para resaltar los colores y de esa forma mediante resaltar un determinado color.

Debido a los reflejos que aparecen en las fotos se opta a la elección de un filtro, para eliminar los excesos de brillo debidos a los reflejos en el cristal.

- Un **filtro poralizado** sirve para reducir los reflejos que aparecen, realza el azul del cielo en contraste con el blanco de las nubes. Su funcionamiento es filtrar el haz de luz para quedarse con un solo plano.

Mucha luz que se ve en las imágenes es luz rebotada, reflejada en la superficie. Donde más se perciben estos reflejos es en superficies metálicas, agua o cristales.

El inconveniente es que las fotos se echan en perpendicular y el filtro poralizador no elimina el haz de luz con este ángulo, sino que la deja pasar.

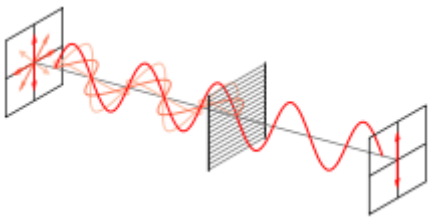


Figura 24: Funcionamiento filtro poralizador. Fuente: Wordpress

Así que, se utilizará un filtro pasa banda MIDOPT-BP470-40:

- Los **filtros BP470** mejoran la visualización de temas iluminados por la iluminación LED azul y cualquier emisión de fluorescencia azul excitada por UV

(430-500nm). Su anchura de banda ancha y alta transmisión de pico (T ≥ 90%) produce imágenes significativamente de mayor contraste. Para la mayoría de las aplicaciones de fluorescencia UV de visión de máquina, el uso de un filtro BP470 es absolutamente necesario [17]. Nosotros lo seleccionamos para aclarar todo lo que es de color azul. Así eliminamos el color del cielo y resalta más el cristal.

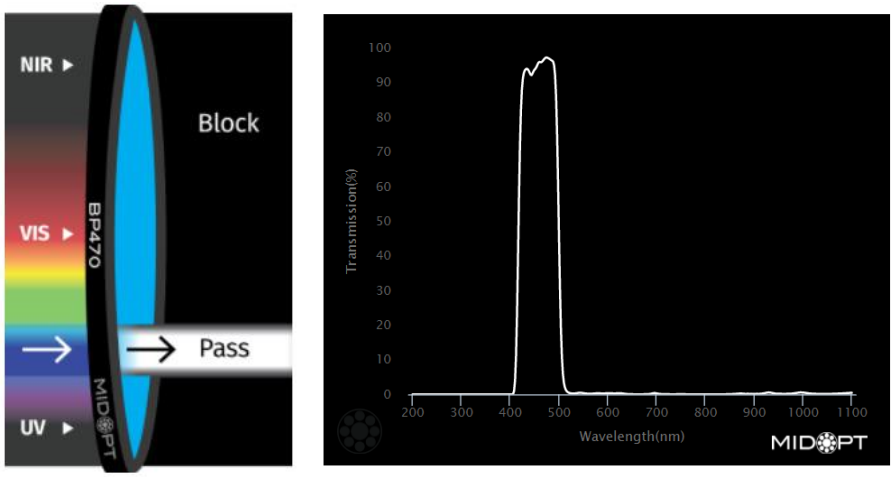


Figura 25: Filtro pasa banda azul. Fuente: Infaimon.

3.2 SOFTWARE.

3.2.1 JAI CAMERA CONTROL TOOLS.

Para poder empezar a usar la cámara de visión artificial se debe descargar el software específico.

El software incluye:

- Componentes SDK necesarios para integrar fácilmente cámaras basadas en GenICam JAI en aplicaciones de visión.
- Controlador de filtro para la transmisión rápida y eficaz de los datos basados en paquetes.
- La herramienta de control JAI. Una aplicación genérica que puede utilizarse para evaluar y controlar todas JAI Spark Series y cámaras de la serie Go, así como otras cámaras GigE Vision y GenICam conformes con independencia de la serie. La interfaz gráfica de usuario permite al usuario ver y activar las características y funciones disponibles de la cámara conectada. Funciones de transmisión y visualización varían en función de la interfaz [18].

Entorno de trabajo:

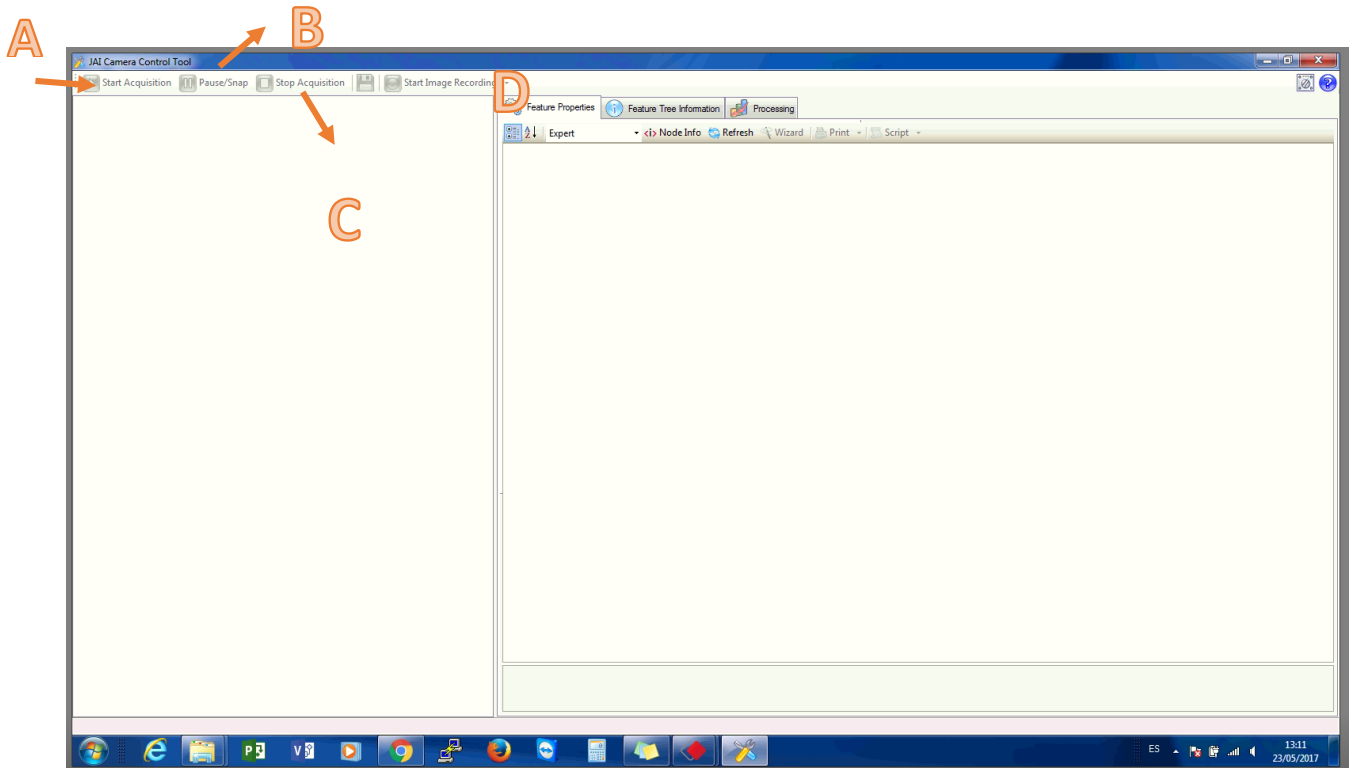


Figura 26: Programa JAI CAMERA CONTROL TOOLS. Fuente: Propia.

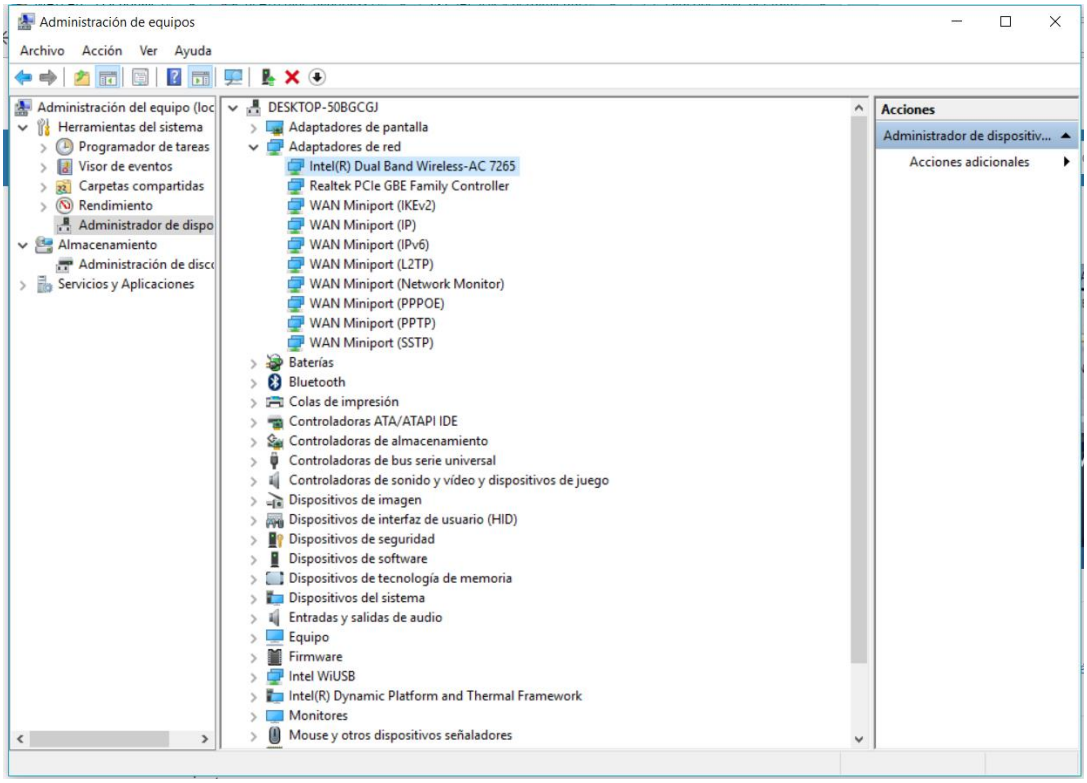
- Punto A→ este botón sirve para comenzar a grabar.
- Punto B → pausar la grabación o hacer una captura.
- Punto C → parar la grabación.
- Punto D → guardar la captura.

Antes de comenzar a usar el programa hay que asegurarse de unos ajustes:

1. Configuración de la tarjeta GigE

   El paso previo a la conexión de una cámara GigE que requiere altas tasas de transferencia es la configuración de las opciones de la tarjeta GigE.

   Para ello iremos a Menú Incio > Equipo y con el botón derecho le daremos a Administrar. Con esto se nos abrirá la siguiente ventana de Administración de Equipos:



Seleccionaremos el Administrador de dispositivos y buscaremos dentro de Adaptadores de red la tarjeta Gigabit que tenga instalada el ordenador (o tarjeta de INTEL).

Haremos doble clic sobre la tarjeta y se nos abrirá la ventana siguiente de propiedades donde elegiremos la pestaña de Opciones avanzadas:

Tendremos que hacer las siguientes comprobaciones:

- ✓ Verificar que los buffers de recepción y de transmisión están al valor máximo. Si no lo estuvieran le daremos a la flecha incrementando su valor hasta el límite superior 2048.
- ✓ Habilitar la moderación de interrupciones.
- ✓ Los paquetes jumbo deben estar al límite máximo 9014 bytes.
- ✓ La velocidad de obstrucción de la interrupción debe estar en el valor extremo.

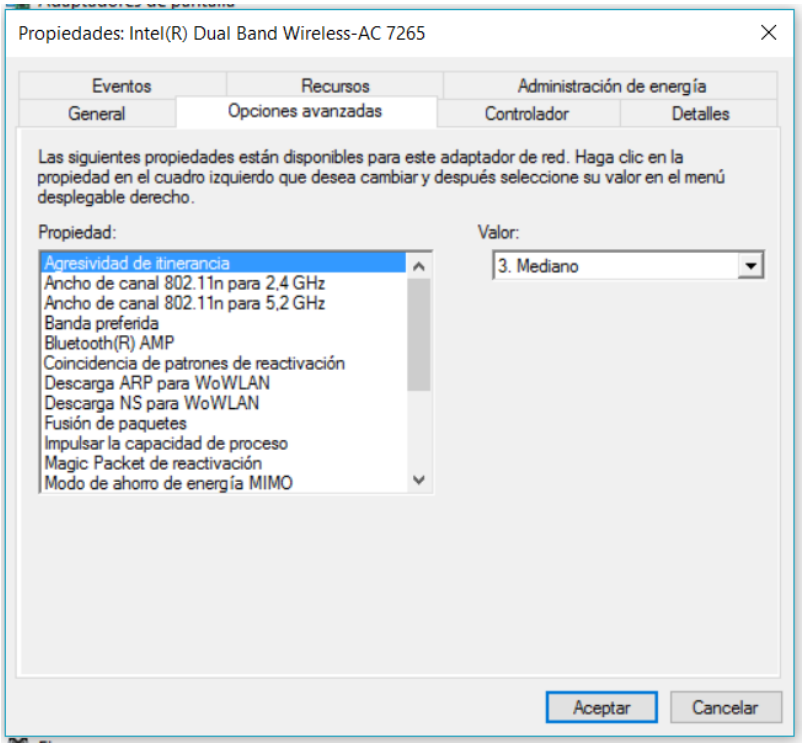


Figura 27: Propiedades avanzadas de tarjeta gráfica. Fuente: Propia.

En algunas ocasiones los valores de buffers, moderación de interrupciones y velocidad de obstrucción se encuentra dentro de la línea de 'Opciones para la mejora del funcionamiento'. Cuando seleccionamos esa línea le damos al botón de Propiedades que aparece y modificamos los parámetros.

2. NET FRAMEWORK

Es una tecnología de la empresa Microsoft, que permite mejor compatibilidad de varias aplicaciones con el sistema Windows, cuenta con librerías y es un lenguaje de programación, que permite el diseño de otras aplicaciones.

Net Framework sirve para la correcta ejecución de programas. Es un administrador que permite la ejecución correcta de los programas escritos específicamente con esta plataforma, por lo que, si no se cuenta con este programa instalado, o existe una versión antigua del mismo, debe de instalarse la versión adecuada, para que los programas creados con esta tecnología, puedan ejecutarse correctamente, o de lo contrario surgirán errores en la ejecución de los mismos, impidiendo su uso por parte del usuario.

Para usar JAI CAMERA CONTROL TOOLS necesitamos la versión NET FRAMEWORK 3.5. Si surgen problemas de este tipo se hará lo siguiente:

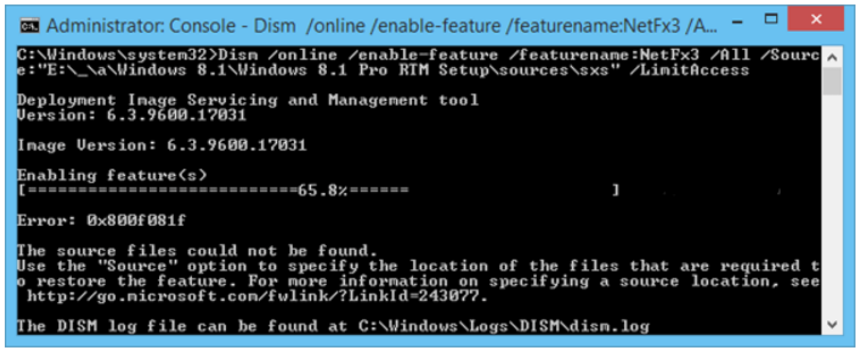


Figura 28: Error NET FRAMEWORK.

a. Escribe gpedit.msc en Ejecutar o en el campo de búsqueda del menú inicio y pulsa Intro. Se abrirá el Editor de Directivas de Grupo Local.
b. Iremos a Configuración del equipo > Plantillas Administrativas > Sistema
c. Luego se especificará la configuración para la instalación de componentes opcionales y la reparación de componentes. Esta opción tiene que estar marcada como No Configurada. Haz clic derecho en ella para activar

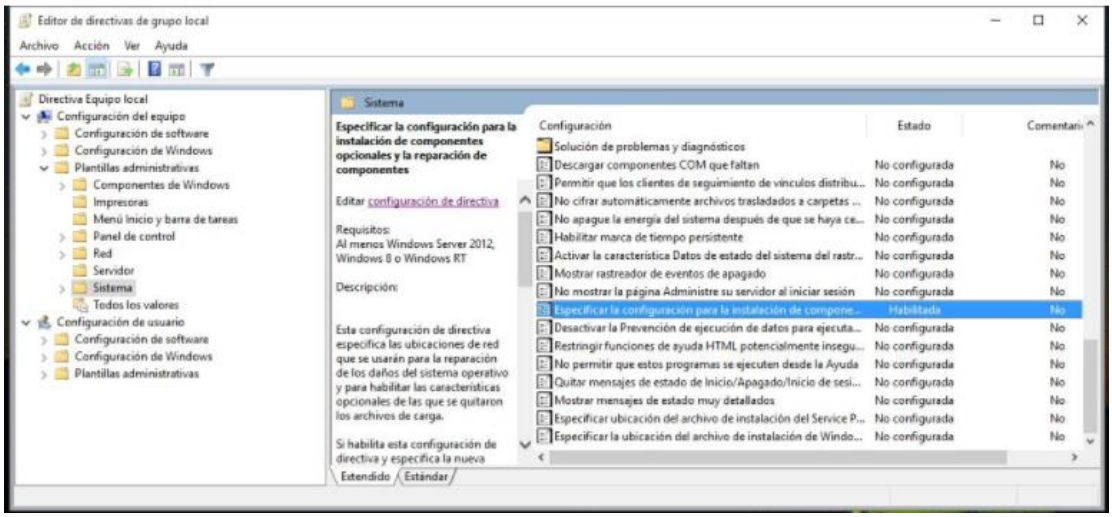


Figura 29: Editor de directivas de grupo local.

3.2.2 MATLAB 2015a.

La plataforma de MATLAB está optimizada para resolver problemas de ingeniería y científicos. Además, dispone de una toolbox denominada Image Processing Toolbox. Esta consiste en una librería de funciones asociada al tratamiento de imágenes, basada en matrices. Está formada por un conjunto de funciones adicionales que amplían la capacidad del entorno numérico de Matlab y proporcionan un conjunto completo de algoritmos y herramientas gráficas para el procesado, análisis, visualización de imágenes etc. [19].

Entre las funciones principales destacan: mejora y filtrado de imágenes; análisis, como segmentación y extracción de funciones; transformaciones geométricas; herramientas interactivas, incluyendo histogramas y selecciones de ROI (región de interés).

Este programa será usado para el desarrollo del software para análisis de las imágenes. Para poder usar la cámara GigE es necesario instalar el paquete de apoyo de adaptador según la cámara utilizada. Para este objetivo se empleará "Image Acquisition Toolbox" de Matlab. Las etapas serán:

1. Instalación de Image Acquisition Toolbox:

   Seleccionamos en la pestaña Home > Resources > Add-Ons > Get Hardware Support Packages. Ahí nos aparecerá una ventana como esta y seleccionaremos instalar GigE visión hardware.

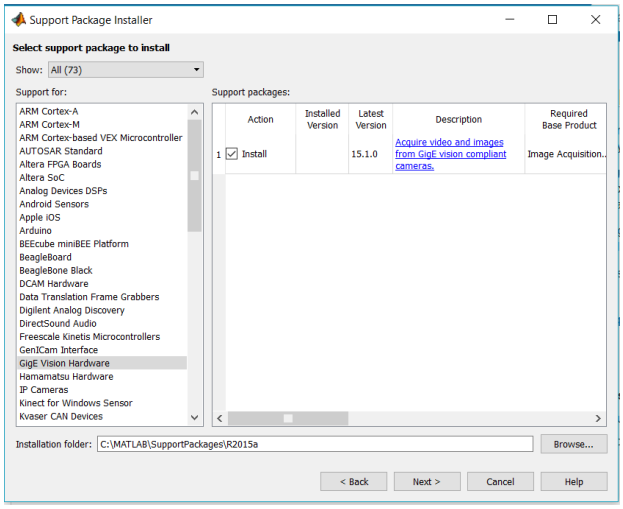


Figura 30: Instalación de paquetes Matlab.

2. Obtener la información del dispositivo de video.

   Para crear el objeto de vídeo. Se utiliza el comando:

```
info = imaqhwinfo

info =

    InstalledAdaptors: {'gige'  'winvideo'}
       MATLABVersion: '8.5 (R2015a)'
         ToolboxName: 'Image Acquisition Toolbox'
      ToolboxVersion: '4.9 (R2015a)'
```

   Nos indica los tipos de drivers con los que se puede trabajar.

```
>> imaqhwinfo('gige')

ans =

      AdaptorDllName: 'C:\MATLAB\SupportPackages\R2015a\gigevisionhardware\toolbox\imaq\supportpackag...
   AdaptorDllVersion: '4.9 (R2015a)'
         AdaptorName: 'gige'
           DeviceIDs: {1x0 cell}
          DeviceInfo: [1x0 struct]
```

3. Crear un objeto de video de entrada.

   Para trabajar con la cámara de vídeo hay que crear un objeto que controle la cámara, conectando ésta con Matlab:

   >>vidobj = videoinput(‘gigecam’)

4. Adquirir los datos de la imagen. Para capturar imágenes se escribirá lo siguiente en la ventana de comandos:

   >> g= gigecam

```
Display Summary for gigecam:

       DeviceModelName: 'GO-5000M-PGE'
          SerialNumber: 'U502445'
             IPAddress: '169.254.1.157'
           PixelFormat: 'Mono8'
  AvailablePixelFormats: {'Mono8'  'Mono10Packed'  'Mono12Packed'  'Mono10'  'Mono12'}
                Height: 2048
                 Width: 2560
               Timeout: 10
```

   >> preview (g)

   >> imwrite(snapshot(g), 'blup.bmp')

   >> imshow ('blup.bmp')

3.2.3 VISUAL STUDIO 2017.

Para poder usar Visual Studio hay que instalar una biblioteca en el ordenador. Esta biblioteca es llamada **OpenCV**.

OpenCV es una biblioteca libre de visión artificial originalmente desarrollada por Intel. Se utiliza en infinidad de aplicaciones. Desde sistemas de seguridad con detección de movimiento, hasta aplicaciones de control de procesos donde se requiere reconocimiento de objetos. Esto se debe a que su publicación se da bajo licencia BSD, que permite que sea usada libremente para propósitos comerciales y de investigación con las condiciones en ella expresadas.

Figura 31: OpenCV

Open CV es multiplataforma, existiendo versiones para GNU/Linux, Mac OS X y Windows. Contiene más de 500 funciones que abarcan una gran gama de áreas en el proceso de visión. Esta biblioteca pretende proporcionar un entorno de desarrollo fácil de utilizar y altamente eficiente. Esto se ha logrado realizando su programación en código C y C++ optimizados, aprovechando además las capacidades que proveen los procesadores multinúcleo. OpenCV puede además utilizar el sistema de primitivas de rendimiento integradas de Intel, un conjunto de rutinas de bajo nivel específicas para procesadores Intel.

Pasos a seguir para la instalación de OpenCV en el entorno de Visual Studio.

Lo primero es ir al Panel de Control > Sistema y Seguridad > Sistema

Se abre una ventana como esta: → Propiedades del Sistema

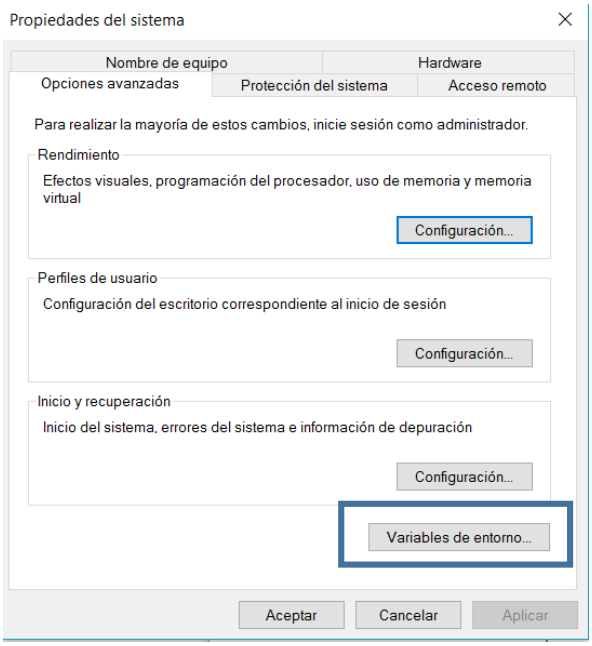


Figura 32: Propiedades del sistema Windows

Hacemos clic en Variables de entorno y creamos una variable de entorno nueva en el Path para OpenCV, con la ruta donde se encuentra la carpeta de opencv.

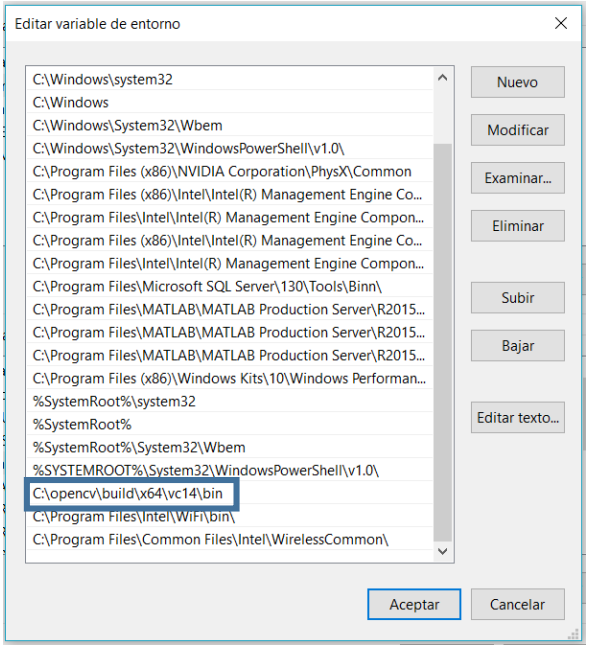


Figura 33: Editor de variables entorno.

Lo siguiente será crear una plantilla en visual donde quede configurada la biblioteca OpenCV.

En Visual Studio hacemos clic en Archivo → Nuevo Proyecto → Aplicación de consola win 32. Creamos un proyecto vacío.

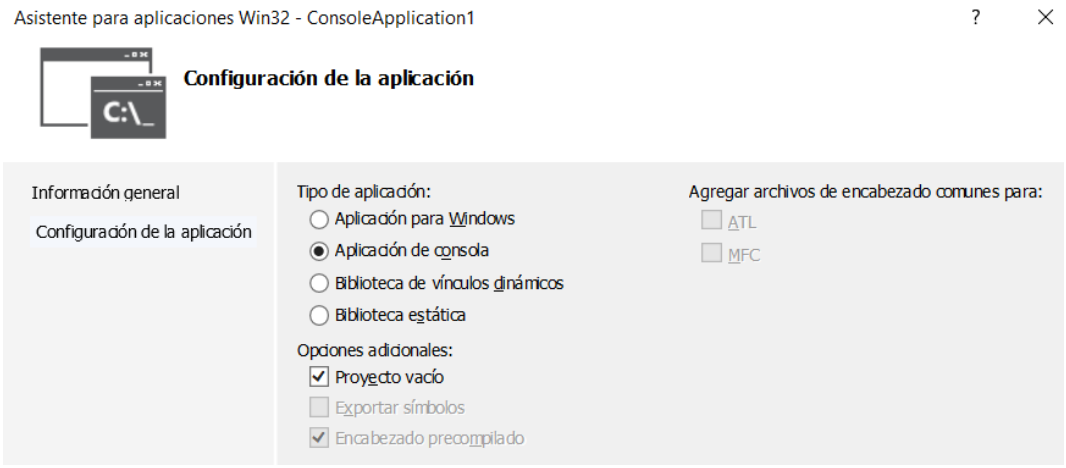


Figura 34: Configuración de la aplicación.

Una vez creado el proyecto iremos al Explorador de soluciones y haremos clic con el botón derecho para abrir la ventana de propiedades del proyecto e incluir los directorios necesarios para poder trabajar con OpenCV

- En Propiedades de configuración → C/C++ → General → Directorios adicionales se escribe : C:\opencv\build\include
- En Vinculador →General, seleccionar Directorios de bibliotecas adicionales y escribir para el Visual 2017: C:\opencv\build\x64\vc14\lib

El siguiente paso sería crear un código ejemplo para ello hay que agregar un archivo .cpp a la solución.

CÁPITULO 4

Desarrollo del software.

4.1 INTRODUCCIÓN

En este capítulo se tratará toda la programación del proceso. Por ello, durante el desarrollo de la parte práctica, se ha ido programando con el objeto de depurar posibles fallos e ir conformando un proceso simple para dotarlo poco a poco de toda su funcionalidad.

El software del pc, hará la captura de la imagen para después procesarla y darnos como resultado final el nivel de suciedad del cristal AR.

4.2 SELECCIÓN DE IMÁGENES

Al empezar a programar nuestro algoritmo lo primero que hacemos es analizar las imágenes para ver sus características.

Sus características principales serán las que aporta la cámara, por ejemplo, el tamaño, el formato para guardar la imagen y que están a escala de gris (cámara monocroma).

Una escala de grises es una escala empleada en la imagen digital en la que el valor de cada píxel posee un valor equivalente a una graduación de gris. Las imágenes representadas de este tipo están compuestas de sombras de grises.

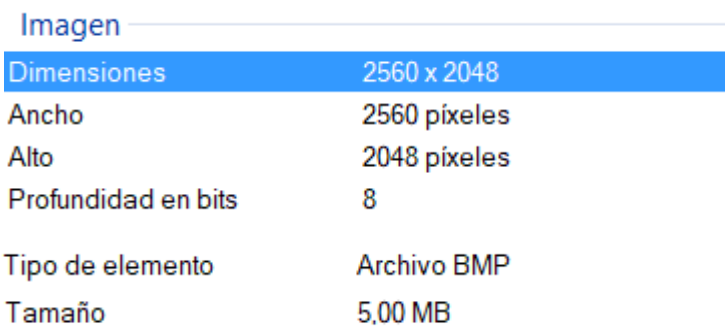


Figura 35: Propiedades de las imágenes. Fuente: Propia.

Formato BMP: es un formato de imagen de mapa de bits, propio del sistema operativo Microsoft Windows. Puede guardar imágenes de 24 bits (16,7 millones de colores), 8 bits (256 colores) y menos. Puede darse a estos archivos una compresión sin pérdida de calidad.

Un archivo BMP es un archivo de mapa de bits, es decir, un archivo de imagen de gráficos, con píxeles almacenados en forma de tabla de puntos que administra los colores como colores reales o usando una paleta indexada. El formato BMP ha sido

estudiado de manera tal que permite obtener un mapa de bits independiente del dispositivo de visualización periférico.

En visión artificial es importante que las fotos tomadas se parezcan bastante, es decir, que tengan características parecidas, así, se podrán analizar mejor.

Las fotos se capturan a la misma hora del día y en el mismo cristal AR, para conseguir características similares se harán pruebas con distintas fotos que se muestran a continuación:
El AR tiene un tamaño de 1365mm x320 mm, la pieza de junta de 295mm x 60mm y un espesor de 6 mm.

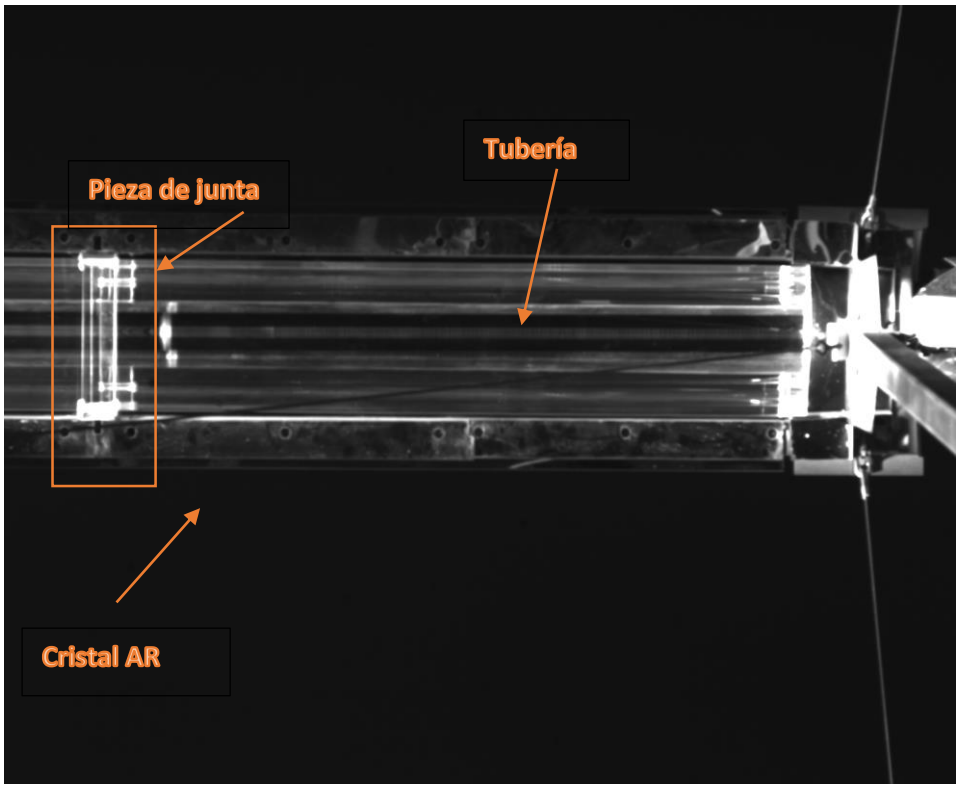


Figura 36: Cristal AR, Nivel 1_Te:150. Fuente: Propia.

Características de la imagen→Tiempo de exposición de 150 y una radiación de 897 w/m^2. Esta imagen es considerada un nivel 1 (está limpio el cristal)

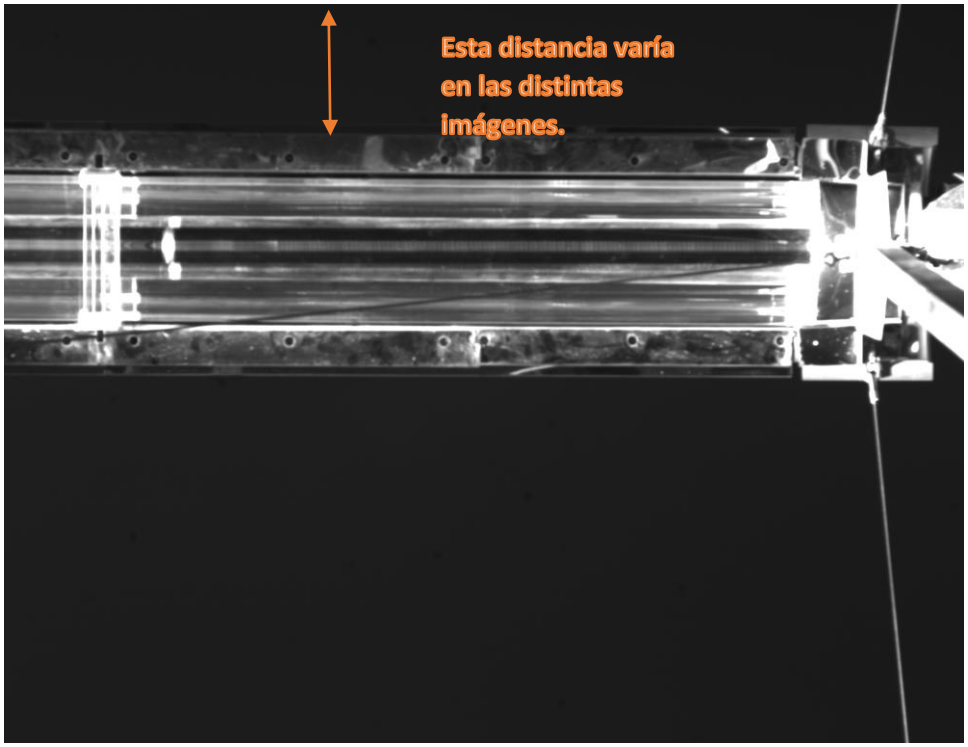


Figura 37: Cristal AR. Nivel 1_Te:350

Características de la imagen→Tiempo de exposición de 350 y una radiación de 897 w/m^2. El nivel de la imagen anterior es considerado 1.

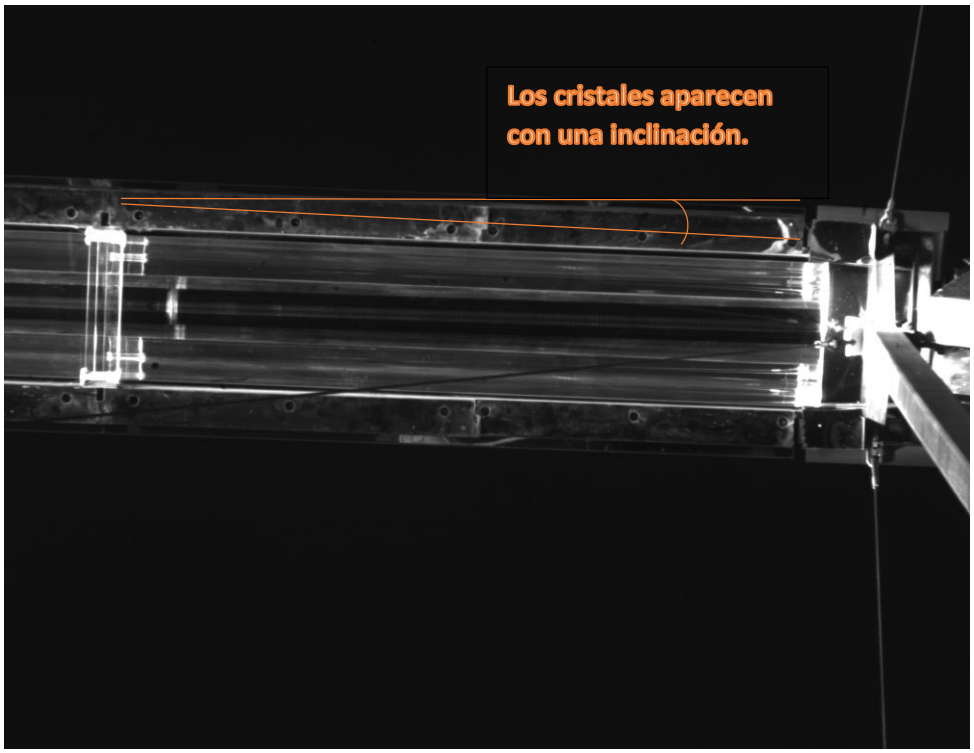


Figura 38: Cristal AR, Nivel 1_Te:415

Características imagen→Tiempo de exposición de 415 y una radiación de 797 w/m^2

Diferencias aparentes encontradas:

- La distancia de la esquina superior de la fotografía y el cristal AR varía.
- El cristal AR puede aparecer con una inclinación en la fotografía.
- Sabiendo que son el mismo cristal, tomadas a la misma hora del día y variando el tiempo de exposición unas imágenes tienen más brillo, es decir, reflejos del sol.

Factores que influyen en la toma de fotos:

- Radiación solar: como el cristal a analizar es el de la parte de arriba donde se refleja y concentra el sol, nos influye directamente en las fotos.
  Cuando hay mayor radiación el cristal se ve con menor definición y empiezan a surgir unos brillos blancos en las fotos (debidos a los reflejos solares). Por este motivo la radiación deberá ser un parámetro fijo. Se concreta realizar las fotografías de 12:00 a 14:00 porque durante estas horas permanece constante. A primera hora del día la radiación es pequeña para apreciar bien las imágenes, una vez que amanece la radiación aumenta minuto a minuto llegando su a punto máximo a medio día y manteniéndose ahí estable durante más tiempo.

- Tiempo de exposición: este parámetro también altera las imágenes, a mayor tiempo de exposición quedan más blancas. Así que el tiempo de exposición deberá ser un parámetro predeterminado.
  Tiempo de exposición es la cantidad de tiempo que el sensor está expuesto a la luz. Este es el control que se utiliza para ajustar la cámara a las condiciones de luz de la imagen a capturar.

Las anteriores fotos fueron tomadas en diciembre. Pero cuando se realizó la foto en un nivel 5 se puedo apreciar que la imagen se hace ilegible, es decir no se puede evaluar nada en la foto. Se hace difícil su análisis.

El nivel 5 con una radiación como las anteriores imágenes de 840-890 W/m^2 y un tiempo de exposición intermedio de 415 se vería así:

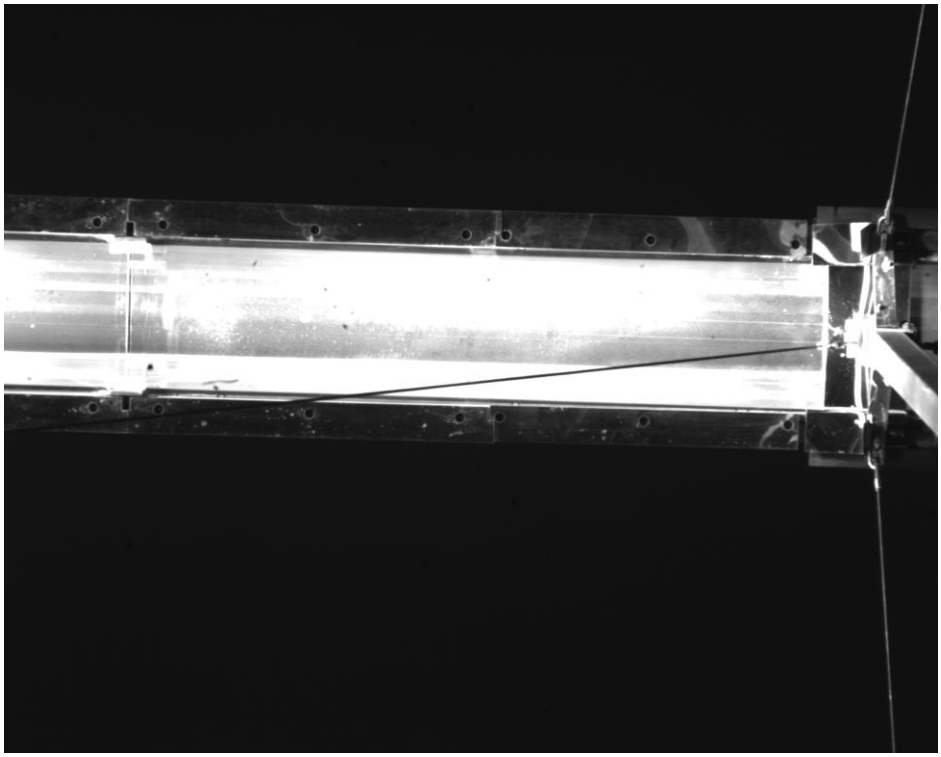


Figura 39: Cristal AR, Nivel5_Te:415

A partir de este momento se decide poner el filtro pasa banda azul de la marca MIDOPT-BP470-40 que mejora la visualización en los cristales iluminados, realza el azul y elimina el brillo.

Una vez colocado el filtro en la cámara las imágenes se oscurecen demasiado y se decide aumentar el tiempo de exposición.

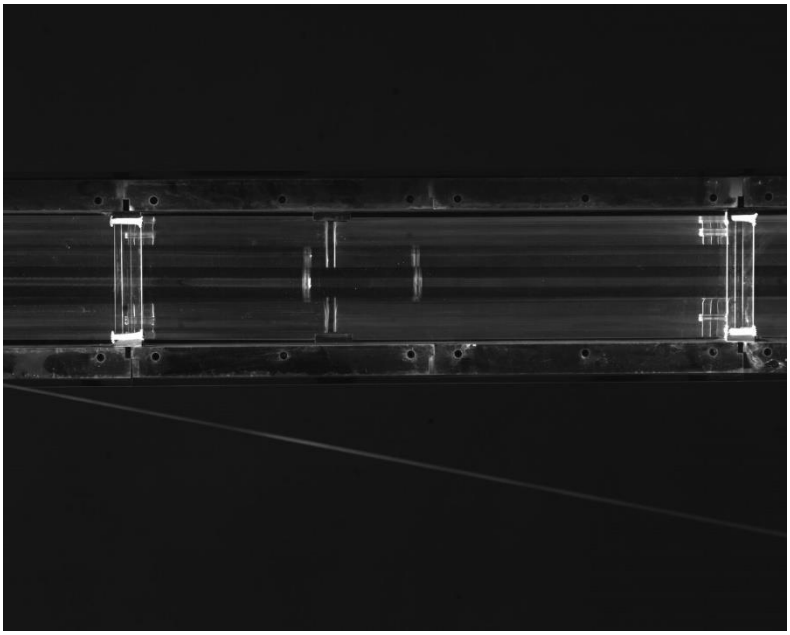


Figura 40: Cristal AR con filtro. Te:800

Desde este momento se establece realizar las fotos a la misma hora de 12:00 a 14:00 con la radiación de 800W/m^2 a 900W/m^2 y un tiempo de exposición de 1900. A continuación se muestra una tabla con los distintos niveles:

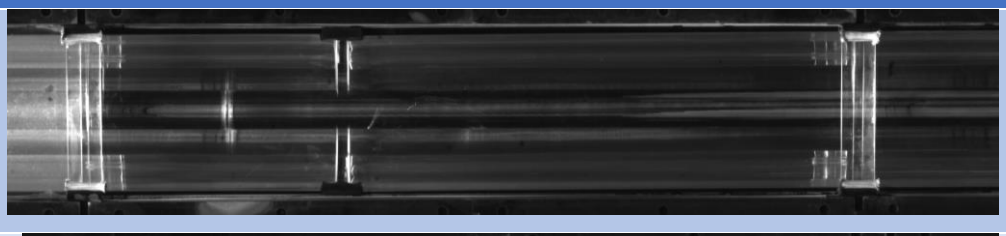
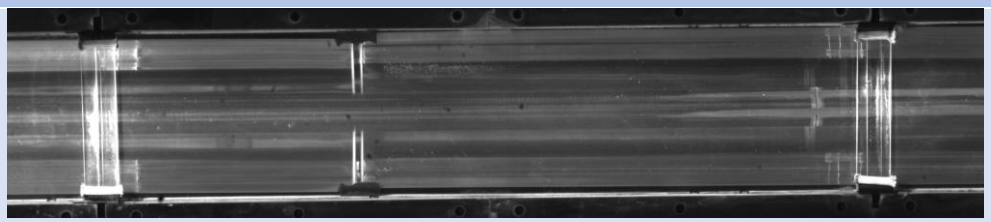
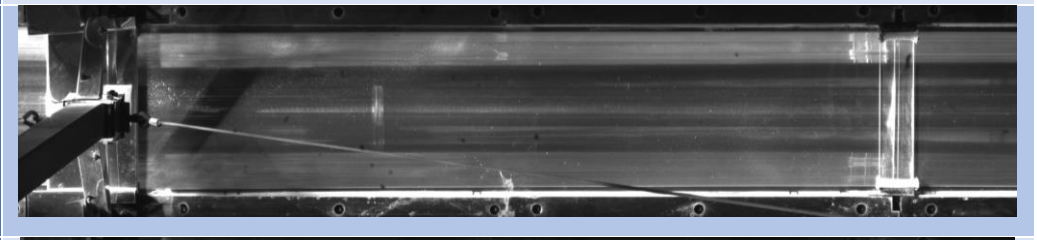
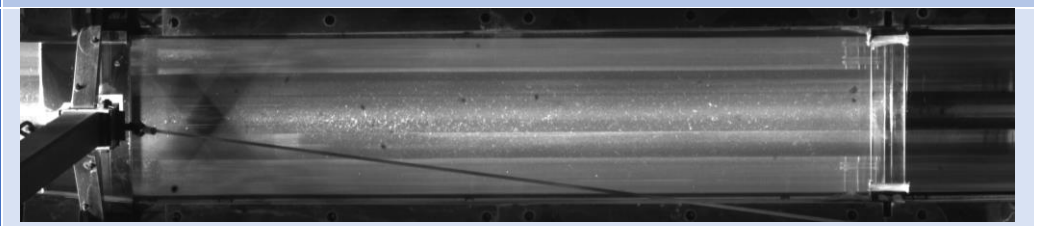
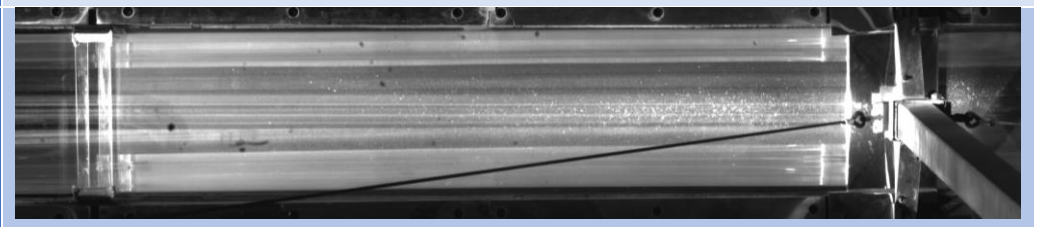
Nivel	Cristal AR
1	
2	
3	
4	
5	

Tabla 1: Comparación de niveles.

En la anterior tabla podemos ver que el nivel de gris cambia para los distintos niveles, esto nos ayudará a desarrollar el algoritmo de análisis.

Selección de la zona a analizar

En este punto se decidirá la zona de análisis de las imágenes y se comentan los distintos problemas que pueden surgir:

- En la esquina superior e inferior suele acumularse una capa de polvo.

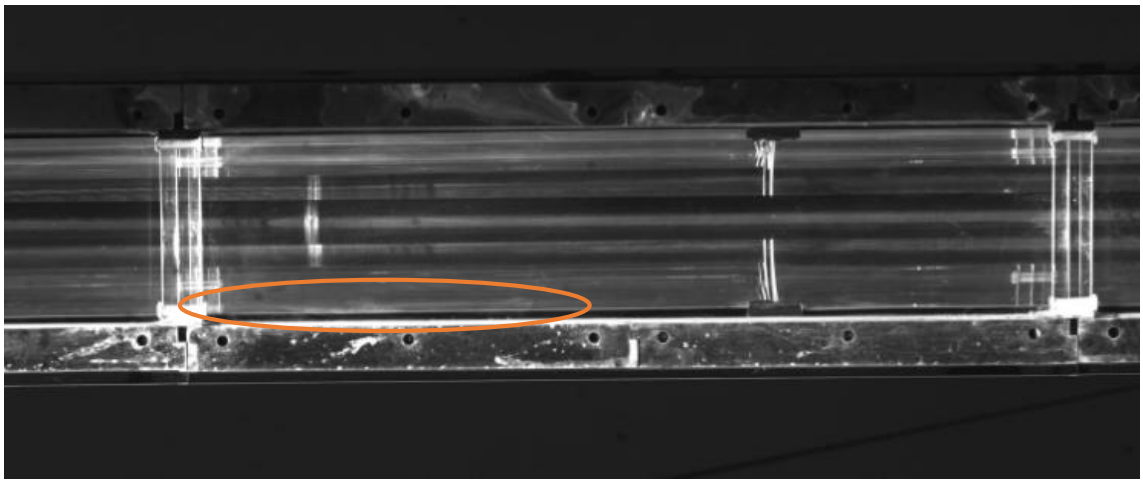


Figura 41:Selección de la zona a analizar.

- Debajo de la tubería suele aparecer su reflejo, esto puede dar falsos resultados, parece que hay dos tuberías.

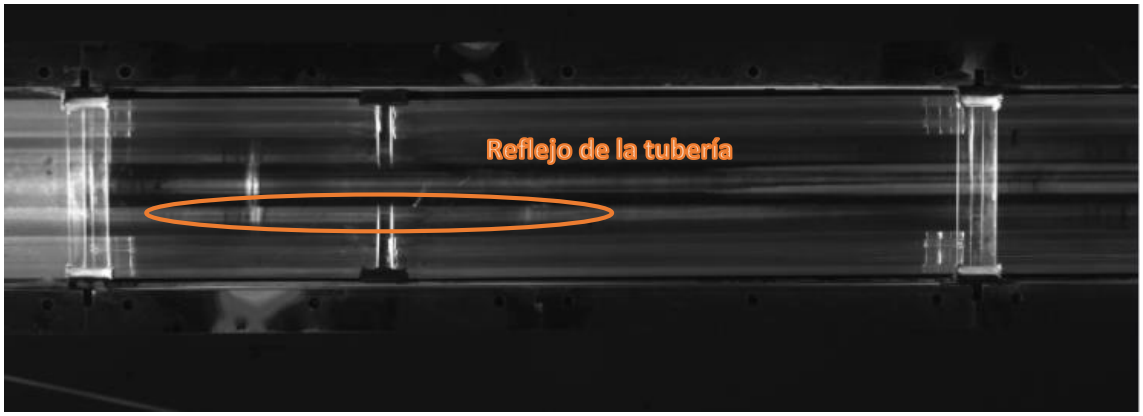


Figura 42: Selección de la zona a analizar.

- Las piezas de junta como no se puede limpiar bien se ven demasiado blancas y darían valores falsos si se analizan ya que tras varias imágenes se puede comprobar que cuando aumenta el nivel de suciedad también aumenta el blanco y el brillo en la imagen. Si estas piezas fueran analizadas podrían aumentar el nivel de suciedad.

Debidos a estos problemas se decide no analizar la tubería, tampoco la esquina inferior (para descartar los restos de polvo acumulados en las esquinas) y recortar ambos lados de las imágenes para quitar las piezas de junta.

PROGRAMACIÓN EN MATLAB

Con las técnicas de procesado se pretende mejorar o realzar las propiedades de las imágenes para facilitar las siguientes operaciones de la Visión Artificial, tales como las etapas de segmentación, extracción de las características y finalmente la interpretación automática de las imágenes.

Las técnicas de preprocesado se basan bien en técnicas derivadas del procesamiento lineal de señales o bien en un conjunto de procedimientos heurísticos que han dado resultados satisfactorios.

Se decide hacer un barrido de la imagen en vertical para analizar el nivel de gris y extraer características de las imágenes. El barrido consiste en un corte transversal al cristal de ancho 1 pixel y de alto todo el cristal para representar su nivel de gris.

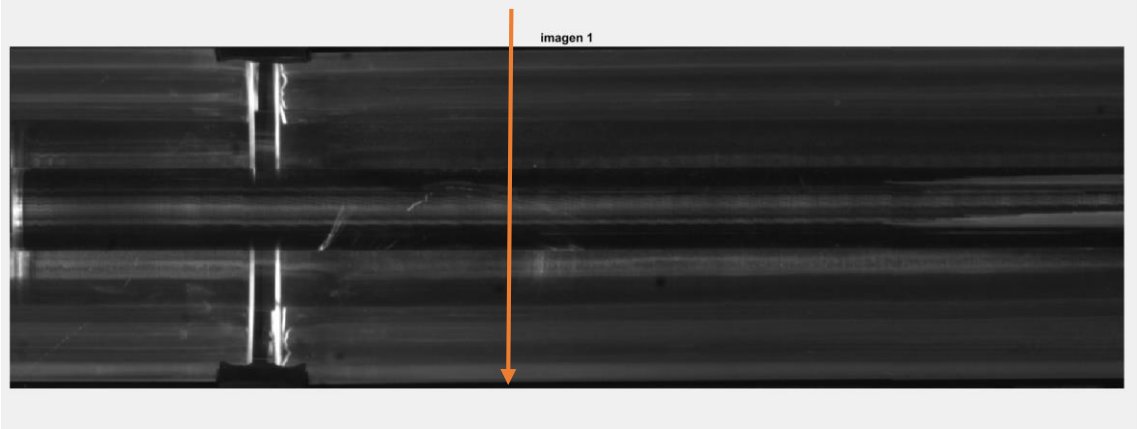


Figura 43: Barrido del cristal para análisis.

Cristal con nivel 1 el resultado fue el siguiente:

- Se puede apreciar que el pico maximo es donde se encuentra la tubería.
- El otro pico que nos encontramos es el del reflejo de la tubería.
- En las esquinas el valor de gris también pasa de 70 esto es debido a la acumulación de suciedad como se dijó anteriomente.
- Justo encima y debajo de la tubería los niveles son más constantes y bajos.

En la gráfica siguiente se muestra el nivel de gris frente a la posición del pixel de un cristal AR con nivel 1:

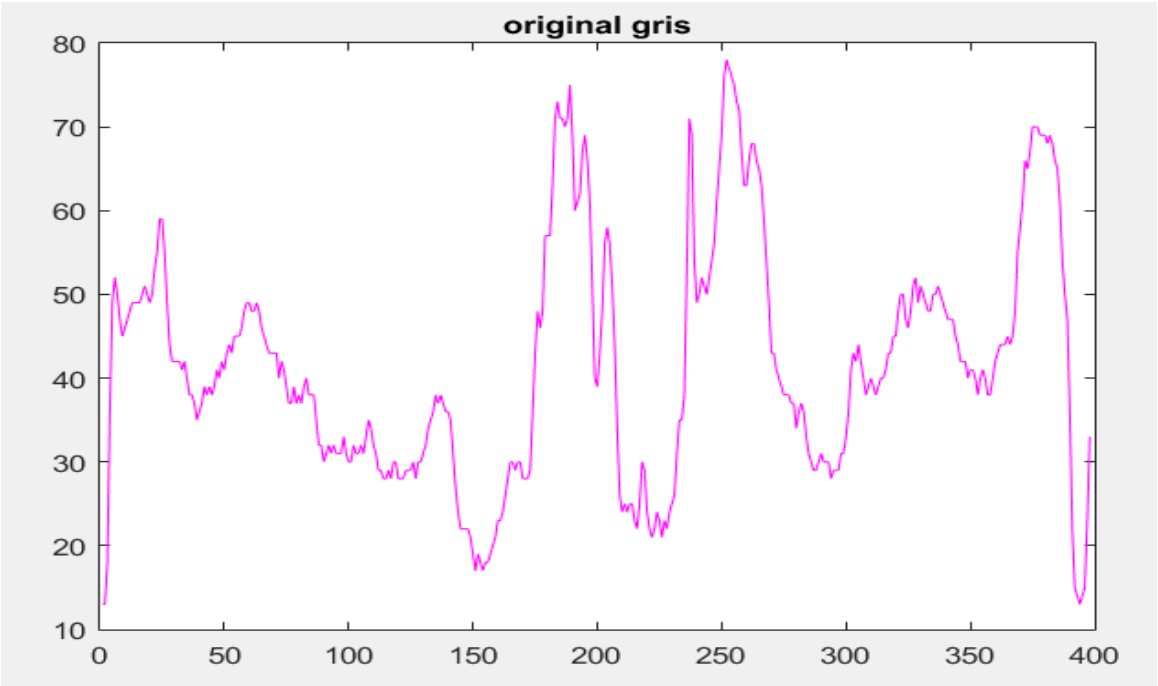


Figura 44: Representación del nivel de gris

A partir de ahora la zona a analizar será exactamente debajo de la tubería, evitando los problemas comentados anteriormente: la tubería, su reflejo y las esquinas inferior y superior.

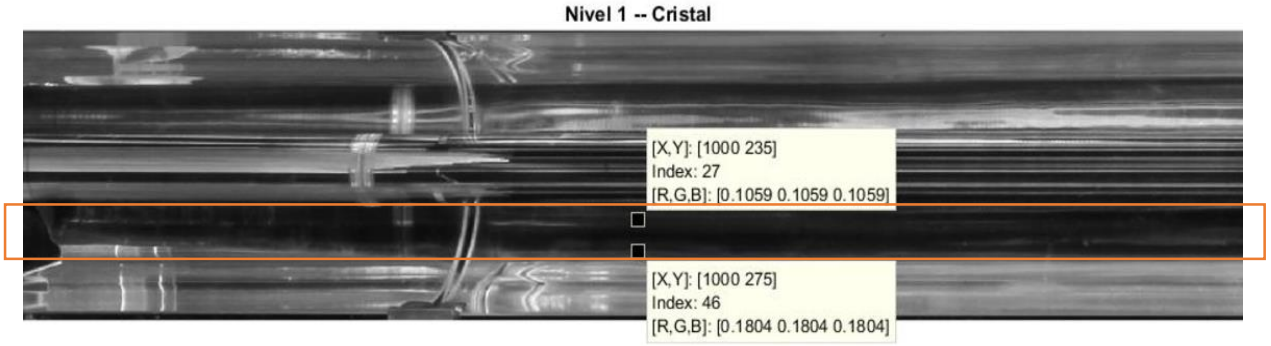


Figura 45: Zona de análisis del Cristal

En esta nueva fase se trata de agrupar los píxeles, por algún criterio de homogeneidad, para dividir la escena en regiones de interés. Estas áreas deben de tener algún significado físico. Por tanto, la segmentación de una imagen es un proceso de extracción de los datos de interés insertados en la escena capturada.

A continuación, se muestran las gráficas comparativas de los niveles que queremos distinguir, la primera sería la representación del nivel de gris frente a la posición de los pixeles entre el nivel 1 y 2.

Las características que podemos destacar cuando aumenta el nivel de suciedad (cantidad de polvo acumulado) también aumenta el valor del nivel de gris y que la diferencia de estos valores es de aproximadamente 20. Con estas características ya podemos replantearnos cómo diferenciar los niveles.

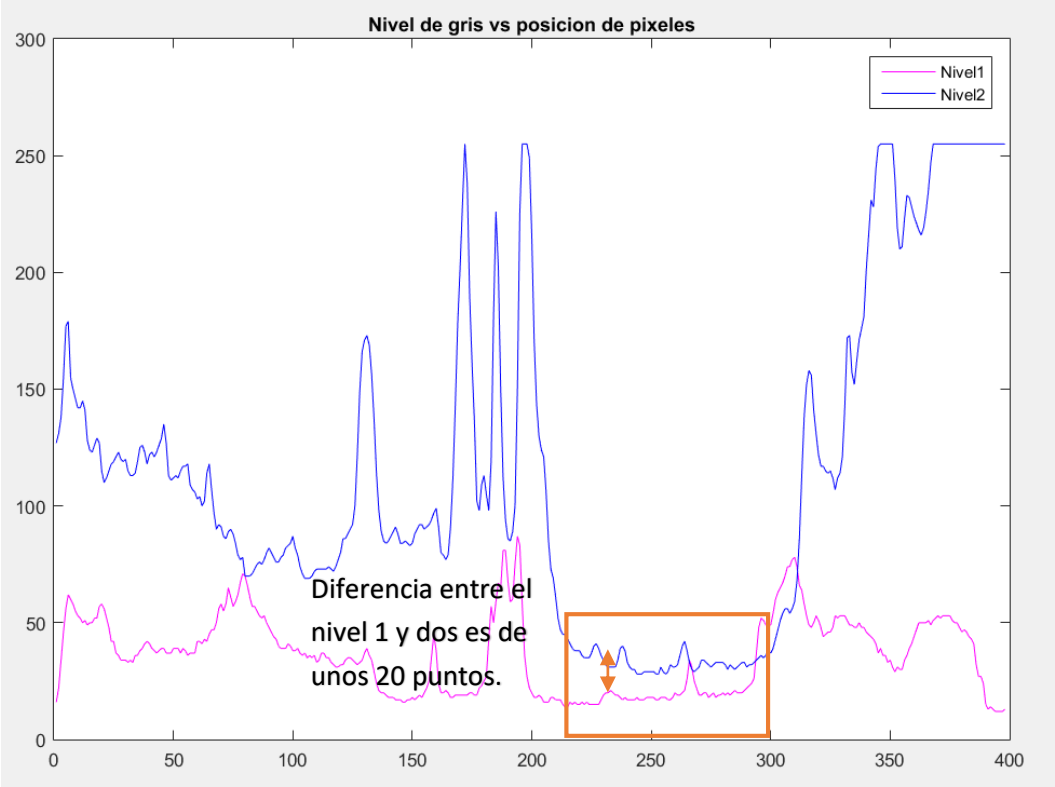


Figura 46: Representacion del valor de gris del nivel 1-2. Fuente: Propia.

La siguiente gráfica es la representación del nivel de gris frente a la posición de los pixeles entre el nivel 1 y 3. Aquí la diferencia entre los dos niveles es mayor, se está hablando de un valor alrededor de 60.

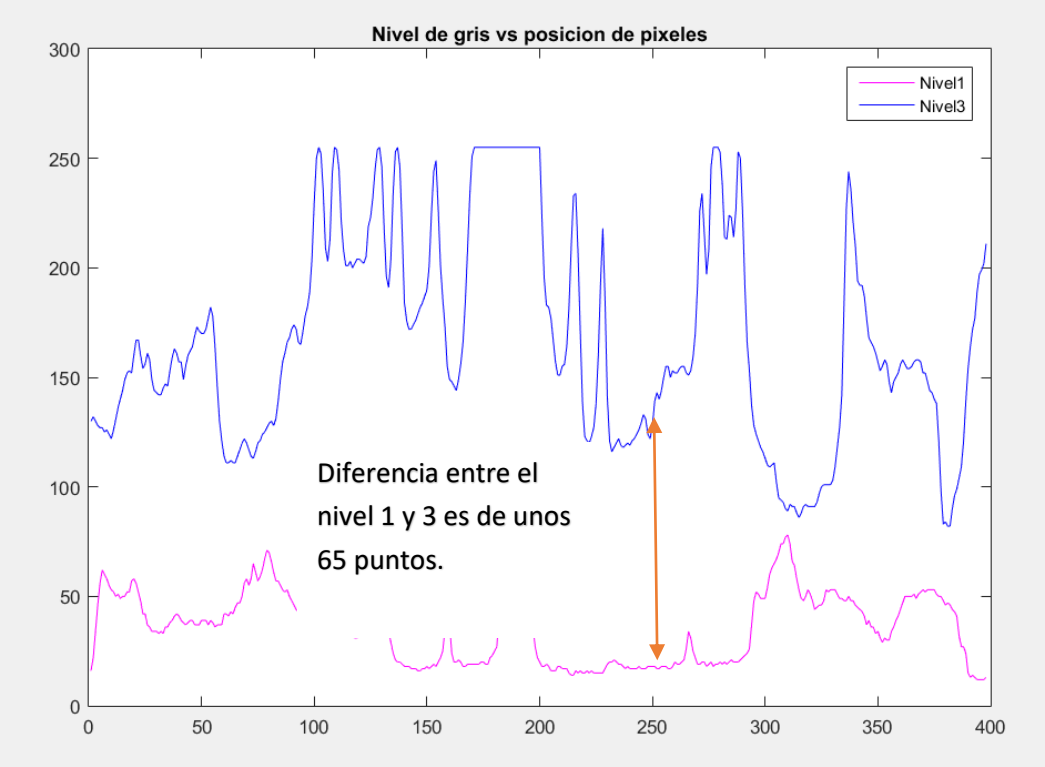


Figura 47: Representación del valor de gris de nivel 1-3

Gráfica de la representación del nivel de gris frente a la posición de los pixeles entre el nivel 1 y 4. La diferencia entre ambos niveles está cerca de 85.

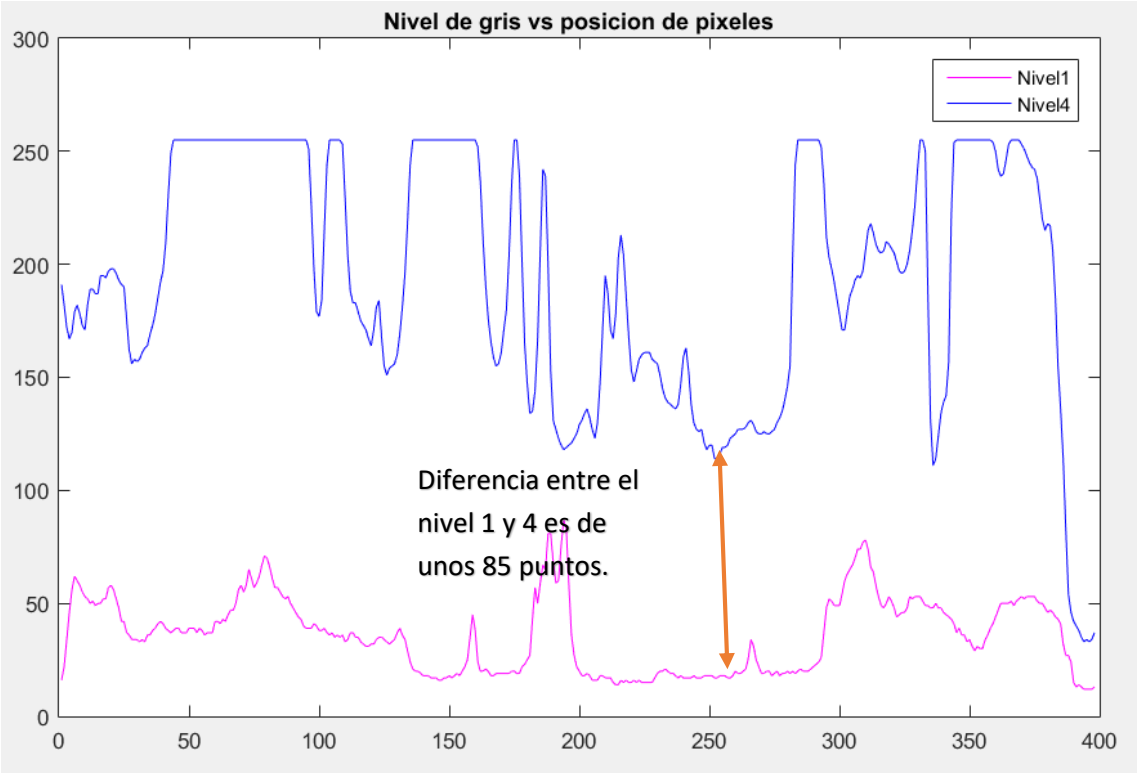


Figura 48: Representación del valor de gris de nivel 1-4

Por último, se muestra la gráfica de la representación del nivel de gris frente a la posición de los pixeles entre el nivel 1 y 5.

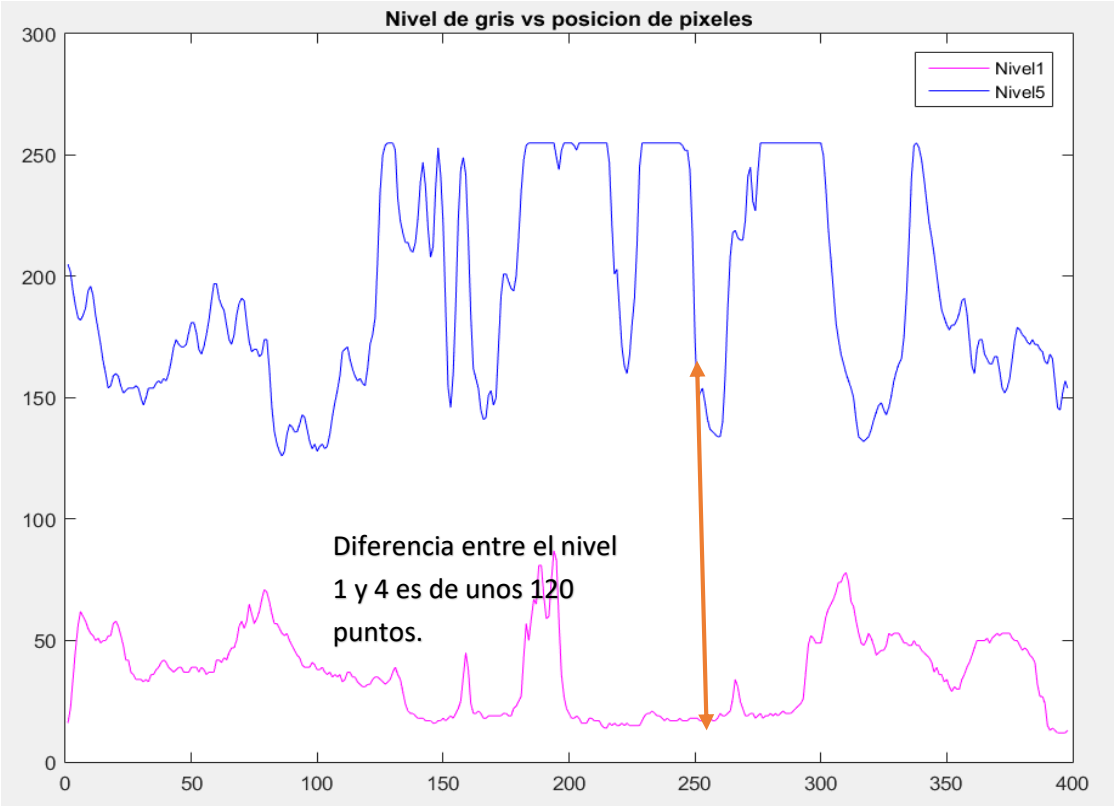


Figura 49: Representación del valor de gris de nivel 1-5

Con estos resultados podemos hacernos una idea de cómo desarrollar el algoritmo para hacerlo lo más robusto ante posibles cambios. Antes de empezar a programar lo mejor es hacer una representación gráfica del proceso para detallar el algoritmo.

4.2.1 Flujograma.

El diagrama de flujo nos ofrece una descripción visual de las actividades implicadas del proceso, mostrando la relación secuencial entre ellas, facilitando la rápida comprensión de cada actividad y su relación con las demás [20], el flujo de la información y los materiales, las ramas en el proceso, la existencia de bucles …

El flujograma quedaría así:

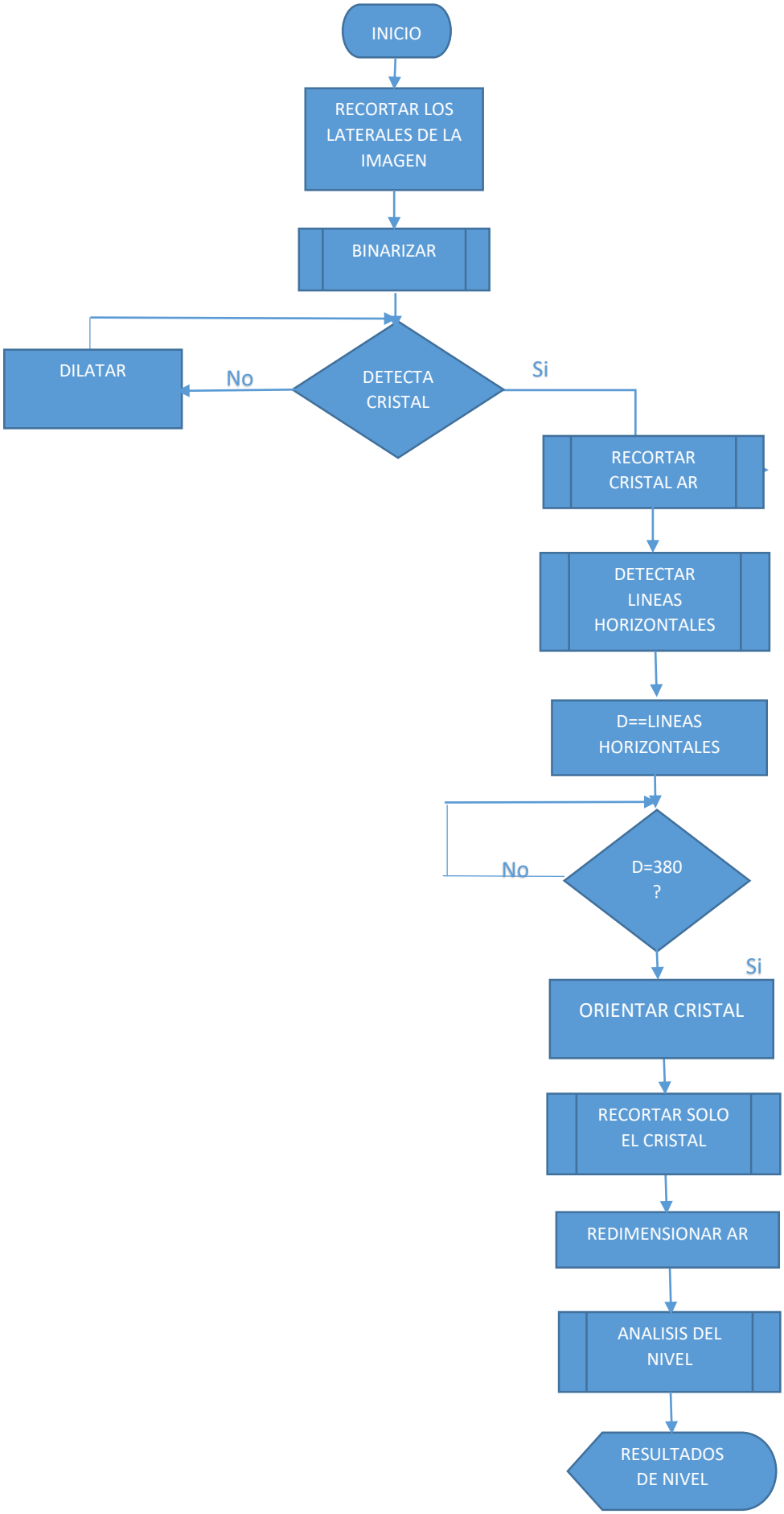


Figura 50: Flujograma del algoritmo.

4.2.2 Explicación punto a punto de los apartados del flujograma:

- Recortar los laterales

  Lo primero es usar la función imread para leer la imagen desde el archivo especificado y almacenarla con el nombre deseado.

  La imagen a leer debe encontrarse en la carpeta de trabajo de Matlab. Los formatos de imagen soportados por Matlab son: TIFF, JPEG, GIF, BMP, PNG, XWD.

```
>> gray = imread('fotos\img.bmp');
```

  Ver el tamaño de la matriz que forma la imagen y saber los pixeles a recortar.

  gray1 *2048x2560 uint8*

  El tipo de dato de la matriz es uint8: Enteros de 8 bits en el rango de [0,255] (1 byte por elemento)

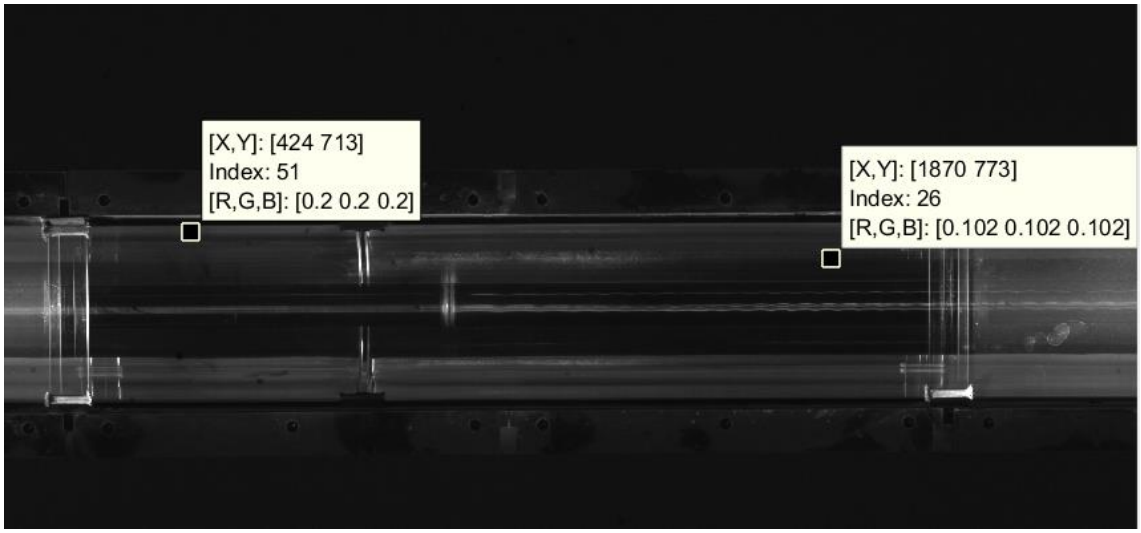


Figura 51: Imagen para recortar

  Para recortar se usa la función imcrop, rect es un vector de posición de cuatro elementos [xmin ymin width height] que especifica el tamaño y la posición del rectángulo de recorte.

  Donde xmin y ymin forman el punto de la esquina superior izquierda de la región a seleccionar.

```
>> REC= imcrop (gray, [420,0,1870-420,2560]);
```

  rect

- Binarización:

  Un método para diferenciar el objeto (en este caso el cristal) del fondo de la imagen es mediante una simple binarización.

  A través del histograma obtenemos una gráfica donde se muestra el número de pixeles por cada nivel de gris que aparece en la imagen.

  Para binarizar la imagen, se deberá elegir un valor adecuado (umbral) dentro de los niveles de grises, de tal forma que el histograma forme un valle en ese nivel. Todos los niveles menores al umbral calculado se convertirán en negro y todos los mayores en blanco [5].

  Suponiendo que queremos aislar el cristal de su entorno, será necesario detectar un umbral adecuado. Podemos considerar la fijación del umbral como una operación que implica pruebas con respecto a una función T de la forma:

  $$T=T[x, y, p(x,y), f(x,y)]$$

  Siendo f(x, y) la intensidad en el punto (x,y) y p(x,y) la intensidad media en la región cerrada.

  Así, se creará una imagen binaria g(x,y) definiendo:

  $$g(x,y) = \begin{cases} 0 \; si \; f(x,y) > T \\ 1 \; si \; f(x,y) \leq T \end{cases}$$

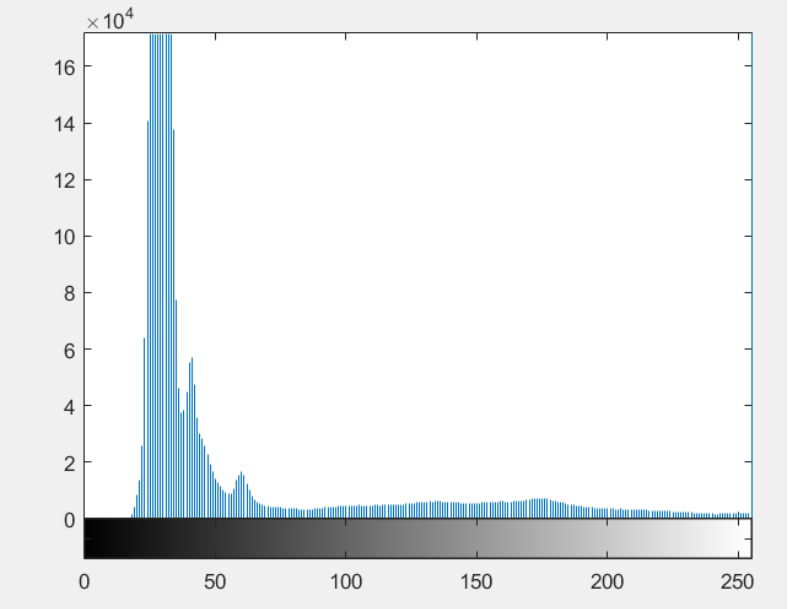


Figura 52: Umbralización del histograma.

Para obtener dicho umbral, uno de los métodos más utilizado es el conocido **método de Otsu** (Nobuyuki Otsu, 1979), que utiliza métodos estadísticos para la resolución de dicho problema. Consideremos dos a G1 y G2 como dos grupos de pixeles divididos por un umbral T. La varianza dentro de los grupos será,

$$G_b^2 = n1(T)G_1^2(T) + n2(T)G_2^2(T)$$

Y la varianza entre grupos,

$$G_b^2 = n1(\mu_1 - \mu)^2 + n2(\mu_2 - \mu)^2$$

Para: $\mu = n1\mu_1 + n2\mu_2$

siendo:

- μi: media del conjunto Gi.
- ni: fracción total de pixeles del conjunto Gi.

Obtenidas ambas varianzas, obtendríamos el cociente siguiente:

$$J = \frac{G_b^2}{G_w^2}$$

Por tanto, el objetivo sería encontrar un umbral T que maximice el cociente de varianzas J, es decir, maximizar la varianza entre clases y minimizar la varianza.

- En Matlab

  La función graythresh de Matlab calcula el umbral T mediante la aplicación del método de Otsu, introduciendo una imagen de intensidades imagen como parámetro de entrada. Además, la función im2bw binariza la imagen con un umbral T previamente hallado, obteniendo una imagen binaria bw.

  El código sería:

```
>> umb = graythresh(Rec);
   bwn = im2bw(Rec,umb);
```

  El programa nos mostraría:

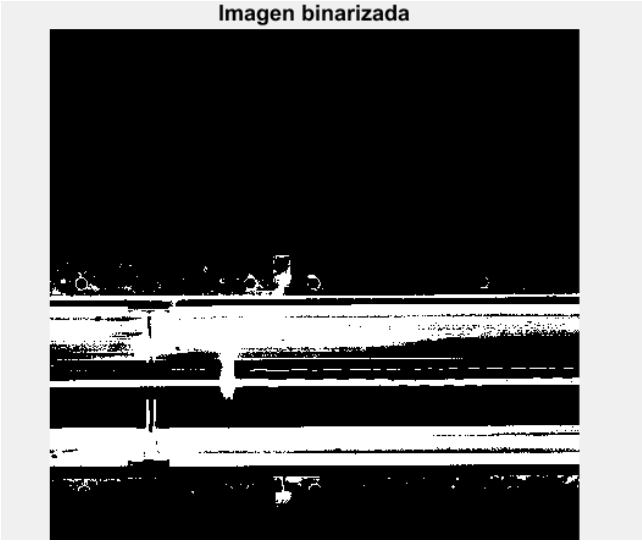


Figura 53: Imagen binarizada_Met. Otsu

- Dilatación

  Para que el reconocimiento sea posible debemos intentar formar una única región conectada. Para esto, utilizaremos la operación morfológica dilatación. Función en Matlab imdilate.

  La dilatación es la expansión de un conjunto de pixeles conectados, provocada por la transformación producida por un elemento estructurante sobre los pixeles vecinos del objeto.

  Pretendemos dilatar las regiones conformadas hasta que el conjunto forme una sola región conectada. Para la formación de una sola región conectada, desarrollamos el siguiente bucle:

```
1
2   while true
3       m=m+10;
4       se=strel('square', m);
5       BW = imdilate(bwn, se);
6       [L,Ne]= bwlabel (BW);
7
8       if Ne==1
9           BW=bwn;
10      else
11          break;
12      end
13
14  end
```

  Mostramos el resultado final:

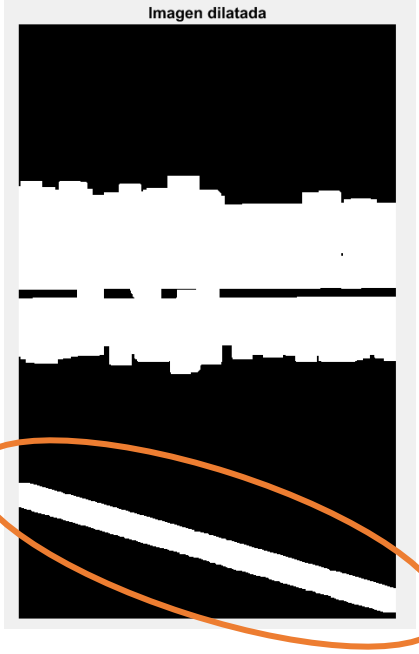


Figura 54: Imagen dilatada

En algunas imágenes aparecerá esta área, es un alambre que saldrá en los cristales de las esquinas. Para eliminarlo y que no nos aparezca mensaje de error se usará la función bwareaopen de Matlab. Bwareaopen elimina todos los componentes conectados (objetos) que tienen menos de 200000 píxeles de la imagen binaria BW, la producción de otra imagen binaria, Area_open. La conectividad por defecto es ocho para dos dimensiones.

```
>> Area_open= bwareaopen(BW, 200000);
```

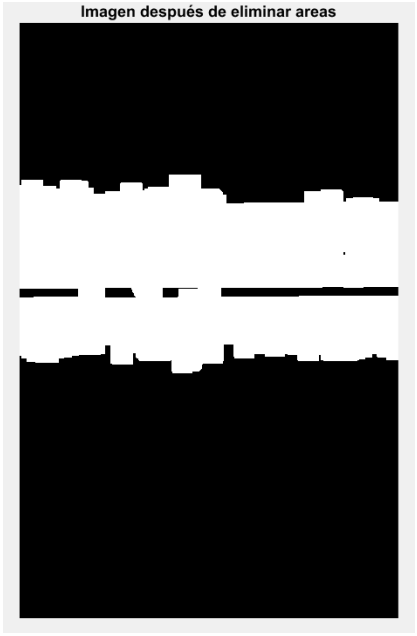


Figura 55: Imagen después de eliminar áreas.

- Extracción de la región →Recortar el cristal AR

  Una vez la imagen ha sido pre-procesada, el primer paso será extraer la región deseada (el cristal) [5].

  El objetivo consiste en etiquetar todos los componentes conexos de forma que el resultado final sea una región. Se entiende por componente conexo como todos los pixeles de valor 1.

  La función regionprops permite extraer distintas propiedades de cada una de las regiones que se detecten por el algoritmo interno de la función, tales como su centroide, área etc. Una de estas propiedades es "boundingbox" que es la caja rectangular de perímetro mínimo que contiene la región etiquetada. La función devuelve las coordenadas de dicha caja, por tanto, podríamos "recortar" cada una de las regiones para extraer las características de cada una de forma individual. El código implementado es el que sigue:

```
[L_1, Ne_1] = bwlabel(Area_open);
prop_1 = regionprops(L_1,'Area','Orientation','BoundingBox','Centroid');
%Funcion imcrop para recortar el rectangulo delimitado por BoundigBox
Recor_1 = imcrop(Rec_1,[prop_1.BoundingBox(1),prop_1.BoundingBox(2),
          prop_1.BoundingBox(3),prop_1.BoundingBox(4)]);
```

La figura recortada quedaría así:

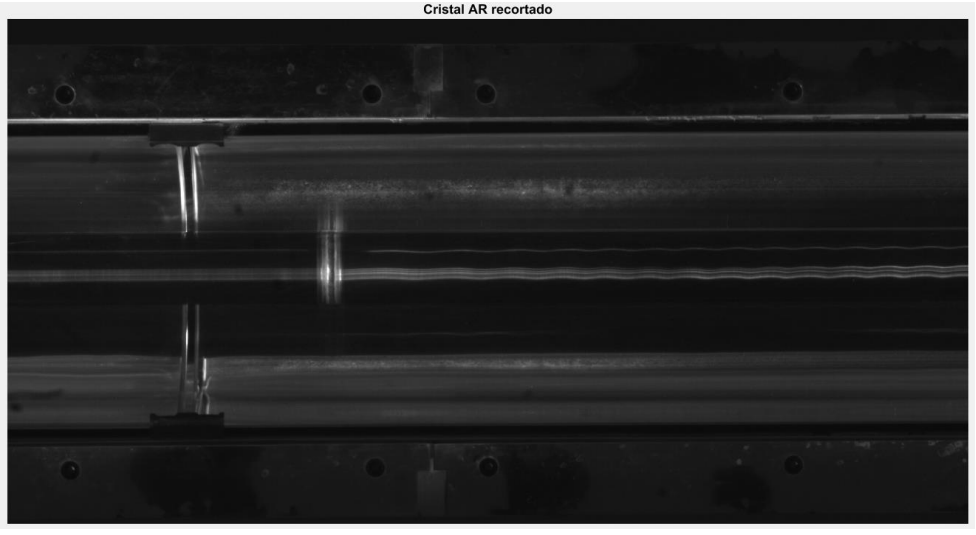


Figura 56: Cristal AR recortado.

- Detección de bordes.

Los bordes de una imagen digital se pueden definir como transiciones entre dos regiones de niveles de gris significativamente distintos. Suministran una valiosa información sobre las fronteras de los objetos y puede ser utilizada para segmentar la imagen, reconocer objetos, etc. La mayoría de las técnicas para detectar bordes emplean operadores locales basados en distintas aproximaciones discretas de la primera y segunda derivada de los niveles de grises de la imagen [21].

La derivada de una señal continua proporciona las variaciones locales con respecto a la variable, de forma que el valor de la derivada es mayor cuanto más rápidas son estas variaciones. En el caso de funciones bidimensionales f(x,y), la derivada es un vector que apunta en la dirección de la máxima variación de f(x,y) y cuyo módulo es proporcional a dicha variación. Este vector se denomina gradiente.

Queremos detectar las líneas horizontales para poder eliminar la parte refractaria del cristal. En Matlab la función que detecta bordes es edge:

```
Sobel=edge(Rec, 'Sobel',0.03,'horizontal');
```

Thresh

Esta función encuentra los bordes de una imagen de distintos niveles de intensidad. El resultado es una imagen binaria del mismo tamaño que la imagen original en la cual, "1" significa que ha detectado un borde y "0" es que no lo ha detectado. El parámetro thresh indica el umbral de binarización. Si se elige el umbral de binarización, hay que ser consciente de que la función edge normaliza la imagen antes de procesarla, llevándola al intervalo [0,1]. La imagen resultante sería esta:

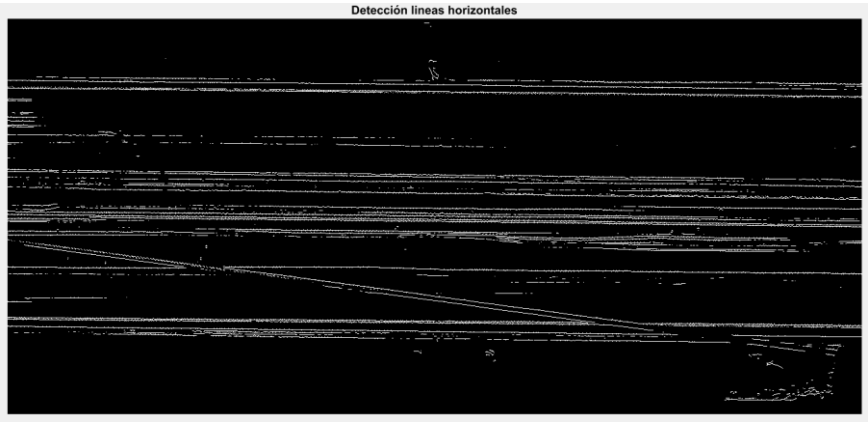


Figura 57: Sobel horizontal.

En la figura 57 aparecen muchas líneas que ensucian la imagen, si volvemos a usar la función bwareaopen podemos eliminar las áreas no deseadas.

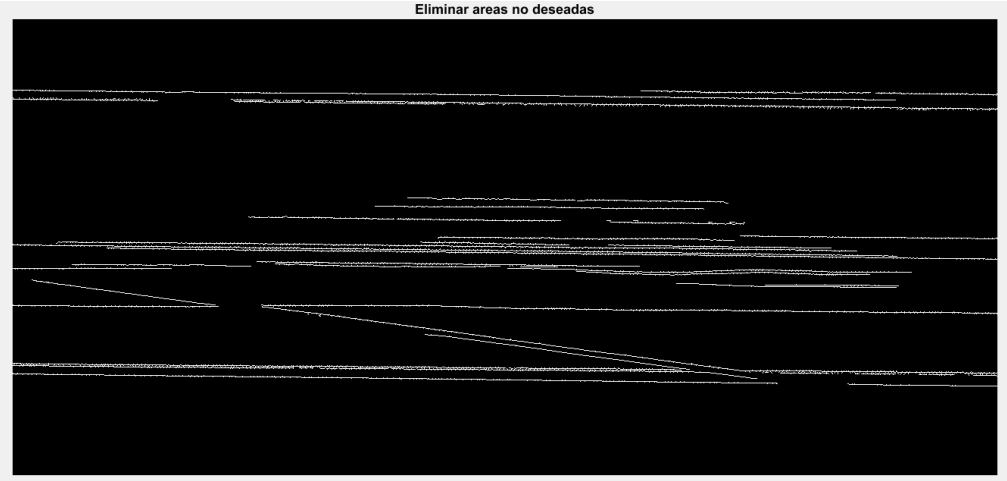


Figura 58: bwAreaOpen eliminación de áreas.

La figura anterior es el resultado de eliminar las áreas menores a 160.

- Erosión.

  Por culpa de una mala umbralización, o por efecto del proceso de filtrado, en ocasiones pueden quedar pixeles no conectados a las regiones en las imágenes binarizadas. Para que éstos pixeles no provoquen una posterior extracción de características defectuosas, es conveniente aplicar un proceso de erosionado sobre la imagen binarizada.

  La función strel permite crear elementos estructurantes para su uso en operaciones morfólogicas sobre imágenes. Nosotros utilizaremos el elemento línea. Un elemento de mayor tamaño podría llegar a "cortar" regiones, lo que provocaría posteriores errores en el reconocimiento. La función imerode realiza la erosión con el uso de un elemento estructurante previamente creado. El código implementado es el siguiente [19].

```
se_3=strel('line',45,0);
ero_b=imerode(Area_open_b, se_3);
```

  El resultado es el siguiente:

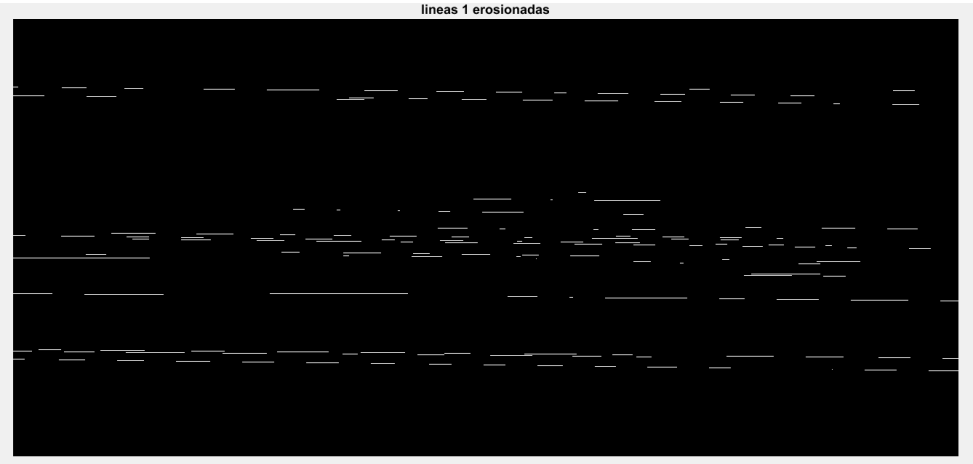


Figura 59: Erosión de líneas.

- Creación de una máscara.

  La mayoría de los filtros usan una matriz de convolución. Con el filtro matriz de convolución podemos crear una máscara personalizada. Esta mascara es bidimensional. La siguiente ecuación define la convolución de A *y* B (imagen y la máscara) [22]*:*

$$C(j,k) = \sum_p \sum_q A(p,q)B(j-p+1, k-q+1)$$

  p y *q* se extienden sobre todos los valores que conducen a subíndices legales de A *(p, q)* y B *(j-p + 1, k-q + 1)*.

  En Matlab una máscara de convolución que dilata sería de la forma:

```
msk=[0 0 0 0 0;
0 1 1 1 0;
0 1 1 1 0;
0 1 1 1 0;
0 0 0 0 0;];

V=conv2(double(dil_b),double(msk));
```

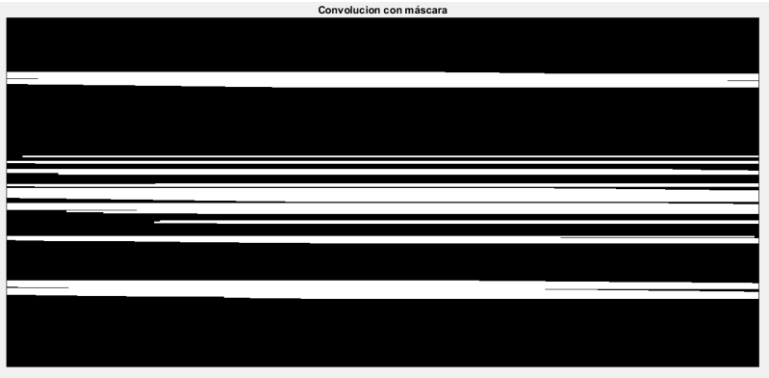


Figura 60: Máscara de convolución.

Después de tratar las líneas en horizontal es importante distinguir cual es la línea que separa el cristal de la parte refractaria.

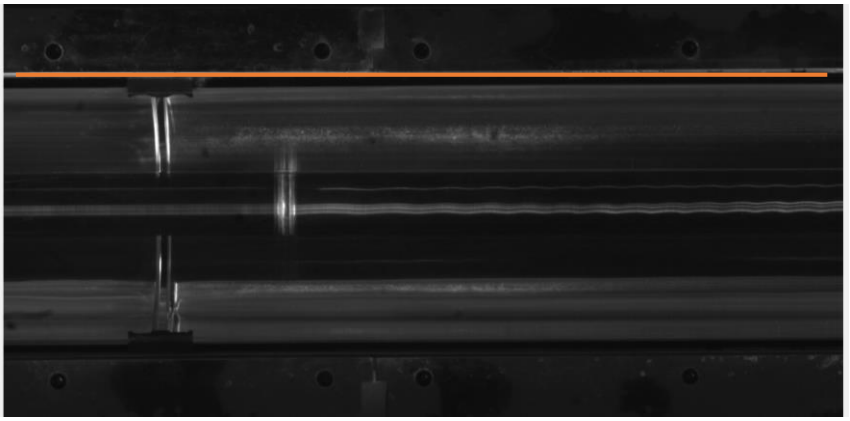


Figura 61: Detección de línea para recortar.

Etiquetamos las líneas detectadas, posteriormente tratadas, y guardamos sus áreas en una variable de menor a mayor:

```
[L_c, Ne_c] = bwlabel(B);
prop_c = regionprops(L_c,'Area','BoundingBox','Orientation','Centroid');
AREAMax_Cristal=sort([prop_c.Area],'ascend');
```

Después guardamos en una variable la posición de las líneas en el eje y, BBox (BoundingBox (2)) creamos un bucle for para encontrar cual es la línea que nos representa la parte refractaria. Normalmente esa línea se encuentra entre los pixeles 30 y 115. Desarrollamos el siguiente bucle:

```
for i=1:length(BBOX)
    for k=2:length(BBOX)
        X=abs(BBOX(i)-BBOX(k));
        if X>380 && X<472
            IndiceCristal=i;
            IndiceCristall=k;
            if (BBOX(IndiceCristal)<115) && (BBOX(IndiceCristal)>30)
                g=i;
            elseif (BBOX(IndiceCristall)<115) && (BBOX(IndiceCristall)>30)
                g=k;
            end
        end
        if X==0
            g=1;
        end
    end
end
```

Donde g sería el índice de la línea buscada, una vez encontrado el índice ya podemos recortar con imcrop como se ha explicado anteriormente.

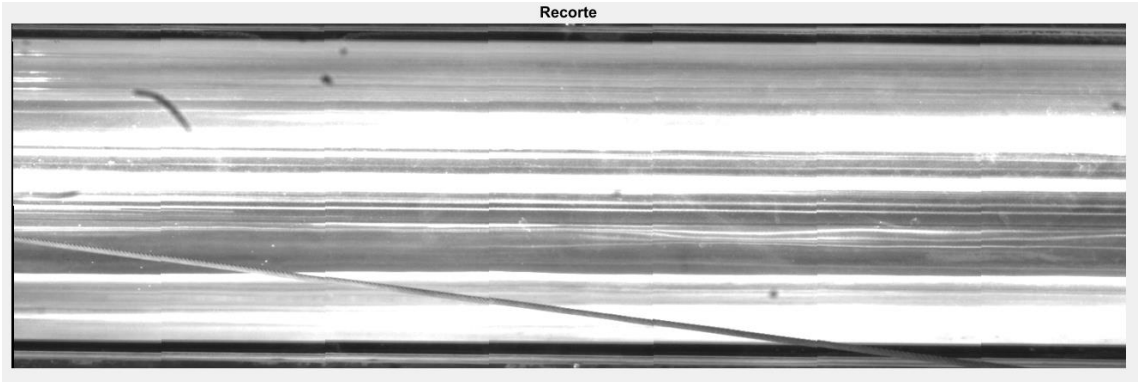


Figura 62: Recorte de la parte refractaría.

- Orientación

  Matlab usa la función imrotate para girar imagen unos grados en una dirección en sentido antihorario alrededor de su punto central. Para girar la imagen hacia la derecha, hay que especificar un valor negativo al ángulo. Imrotate hace que la imagen de salida sea lo suficientemente grande como para contener toda la imagen girada. Utiliza la interpolación del vecino más cercano, el establecimiento de los valores de los píxeles en los que la imagen de salida que se encuentran fuera de la imagen girada a cero [21].

  Para saber que ángulo hay que girar basta con usar regionprops con la propiedad de orientación. El código quedaría así:

```
[L_e, Ne_e] = bwlabel(lineas);
h = regionprops(L_e,'Orientation');

if h(1).Orientation>0
    Rot_h = imrotate(Reco_2,(-h(1).Orientation)/2);
    figure, imshow(Rot_h), title('Cristal  - Angulo Corregido')
    C=Rot_h;
elseif h(1).Orientation == 0
    figure, imshow(Reco_2), title('Cristal - Sin rotar')
    C= Reco_2;
elseif h(1).Orientation < 0
     Rot_h_n = imrotate(Reco_2,(-h(1).Orientation)/2);
     figure, imshow(Rot_b_n), title('Cristal - Angulo Corregido N')
     C=Rot_h_n;
end
```

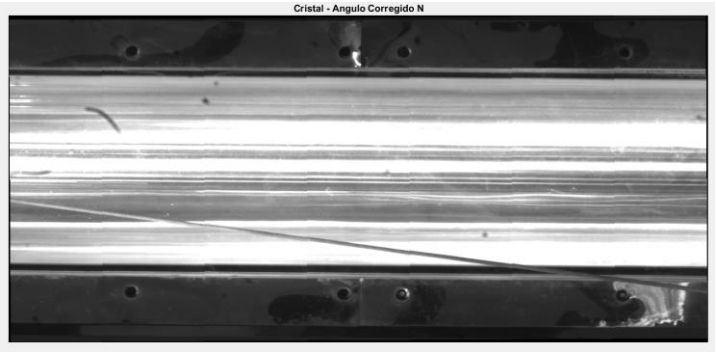


Figura 63: Rotación de figuras.

- Recorte final

  Si nos fijamos en la figura 62 podemos apreciar que el cristal aún no está recortado del todo bien, en la parte superior e inferior queda un hueco oscuro que nos separa el cristal de la parte refractaria. Así que volvemos a ejecutar la función para detectar las líneas horizontales que nos quedan. Pero ahora el parámetro tresh lo dejaremos automático que por defecto en Matlab es 0.04, el resultado sería el siguiente:

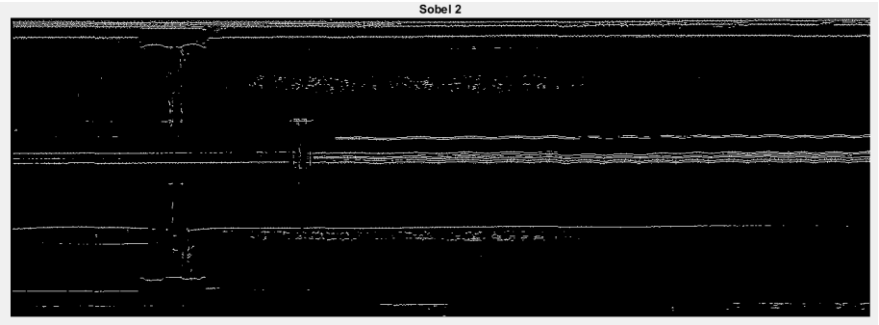


Figura 64: Segundo sobel.

Eliminamos las áreas pequeñas que molestan para el análisis.

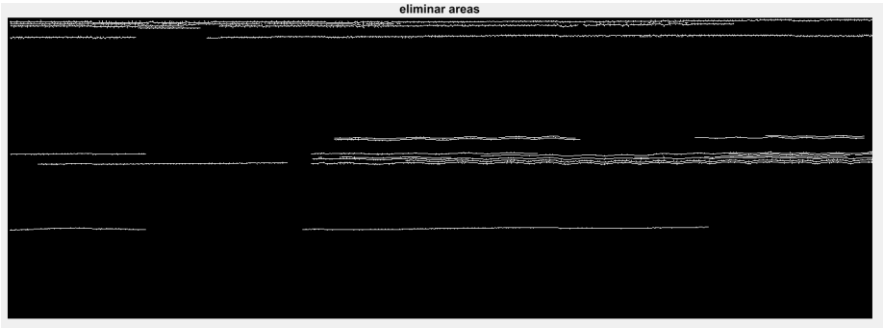


Figura 65: Eliminar áreas.

Posteriormente se realizaría un tratamiento a las líneas resultantes tras aplicar la función sobel horizontal y se realizaría un bucle for parecido al explicado anteriormente para quitar el sobrante de abajo y de arriba.

No en todas las imágenes hace falta volver a recortar, pero para que el algoritmo sea robusto se ha decido hacer un segundo recorte.

El resultado del cristal recortado de algunas imágenes quedaría de esta forma:

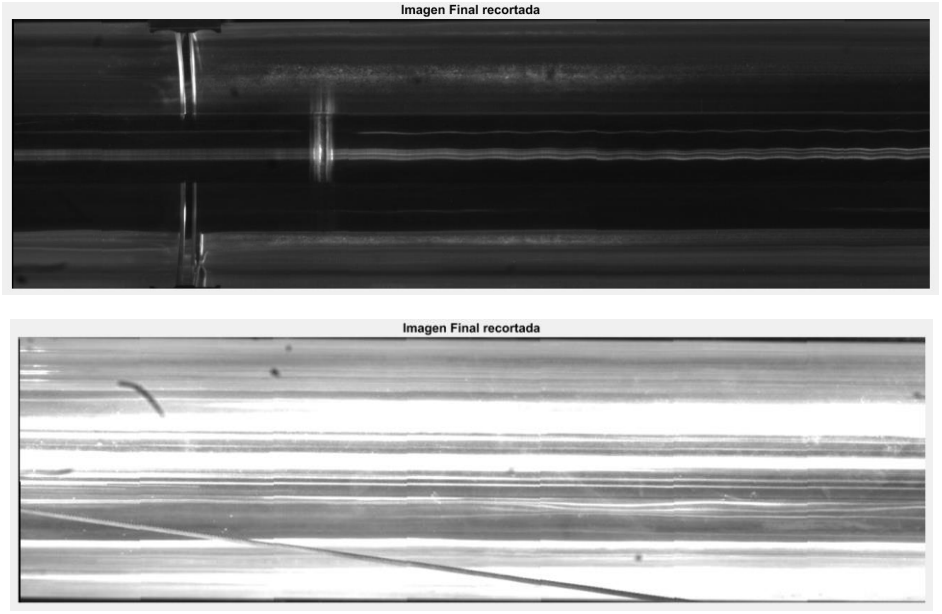


Figura 66: Imágenes recortadas.

La imagen recortada está redimensionada para que todas tengan el mismo tamaño y no haya diferencias entre ellas. En Matlab se hace con la función imresize que devuelve la imagen Final_2 que tiene el número de filas y columnas especificadas por el vector de dos elementos [398, 1300].

```
Final_2 = imresize(Fina_1,[398, 1300]);
```

- Análisis de las imágenes

  Se crea una función para recortar los cristales que será fija para todos los cristales. El algoritmo recorta el cristal de referencia y el que analizamos.

```
function [Rec]=Recortar(Imagen)
```

  En este apartado se realizan dos técnicas para detectar el nivel de suciedad de las imágenes.

  1. Hallando la media del nivel de gris de cada imagen. Para lograr esto se elige una imagen de referencia (nivel 1) y la que queremos analizar. Se crea una matriz con la sección a analizar de cada imagen. En este caso sería la parte de debajo de la tubería como se comentó en el anterior apartado. Inmediatamente se restan ambas matrices y se calcula la media del nivel de gris en columnas, quedándonos un vector. A ese vector se le calculará la media. Este valor nos será útil para ver qué diferencias existen entre los diferentes niveles.

En Matlab quedaría de la forma:

```
[~, columnas]=size(Final);
img_1=Final(260:340,:);
img_2=Final_2(260:340,:);
Diff=img_2-img_1;
[fil, col]=size(Diff);
Sum=sum(Diff);
Add=Sum/fil;
for k=1:columnas
    Sumal=Add';
    Total= sum(Sumal);
    [Fil, Tam]=size(Suma);

end
Media=Total/Tam
```

2. Análisis de texturas con filtro de Gabor. La idea principal es aplicarle a la imagen uno o más filtros que permitan extraer de ella características representativas de las texturas que la componen, para luego clasificar las salidas filtradas mediante un clasificador vectorial [23].

   Los filtros de Gabor recibieron mucha atención, debido a que se demostró que son una muy buena aproximación del comportamiento de ciertas células de la corteza visual de algunos mamíferos, las cuales tienen la función de procesar texturas [24]. Además, estos filtros mostraron poseer propiedades de localización óptima en el dominio espacial, siendo esto algo deseable en los problemas de segmentación de las imágenes texturadas [25].

- Filtro Gabor.

  Gabor es un filtro pasabanda en 2D, si le asignamos una determinada frecuencia y dirección obtenemos una reducción del ruido a la vez de preservar la dirección de la imagen original.

  Este filtro es aplicable en huellas digitales, ya que debemos tomar en cuenta que las imágenes a estudiar presentaran localmente una orientación y frecuencia definidas.

  La forma general del filtro de Gabor está dada por funciones [26]:

  Una función armónica bidimensional cuya oscilación se extiende perpendicularmente al vector $f_0 = (f_{0x}, f_{0y})$ con frecuencia f_0, con origen en $x_0(x_0, y_0)$y fase ϕ

$$G(x, x_0, f_o, \theta, \sigma_X, \sigma_Y, \phi) = e^{-\pi\left(\frac{X'^2}{\sigma_x^2}+\frac{Y'^2}{\sigma_Y^2}\right)} e^{(i2\pi f_0(X-X_0)+\phi)}$$

Donde $x' = (x', y')$, resulta una traslación y un giro de ejes , es decir:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} cos(-\theta) & sen(-\theta) \\ -sen(-\theta) & cos(-\theta) \end{pmatrix} \cdot \begin{pmatrix} x - x_0 \\ y - y_0 \end{pmatrix}$$

Podemos construir varios filtros de Gabor variando la longitud de onda λ, orientación θ, la fase ϕ y las desviaciones típicas de la gaussiana σ, usando:

$$G(x, y, \lambda, \theta, \ \phi, x_0, y_0) = e^{\frac{(x-x_0)^2+(y-y_0)^2}{-2\sigma^2}} sen(\frac{2\pi}{\lambda}(xcos\theta - y\, sen\theta) + \phi)$$

En Matlab el filtro se le aplicaría al cristal recortado y la función seria la siguiente:

```
phi = 2*pi/3;
theta = 0.5;
sigma = 0.65*theta;
filterSize = 6;

G = zeros(filterSize);

for i=(0:filterSize-1)/filterSize
    for j=(0:filterSize-1)/filterSize
        xprime= j*cos(phi);
        yprime= i*sin(phi);
        K = exp(2*pi*theta*sqrt(-1)*(xprime+ yprime));
        G(round((i+1)*filterSize),round((j+1)*filterSize)) = exp(-(i^2+j^2)/(sigma^2))*K;
    end
end

J = conv2(I2,G);
imagesc(imag(J))
```

El resultado obtenido es el que se muestra a continuación:

- Aplicación del filtro de Gabor a un cristal considerado nivel 1:

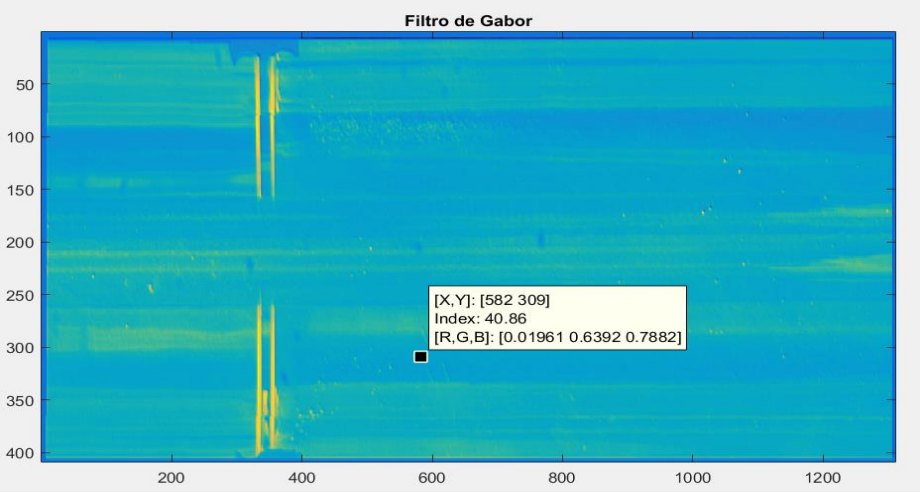


Figura 67: Filtro de Gabor a nivel 1.

- Aplicación del filtro de Gabor a un nivel 2:

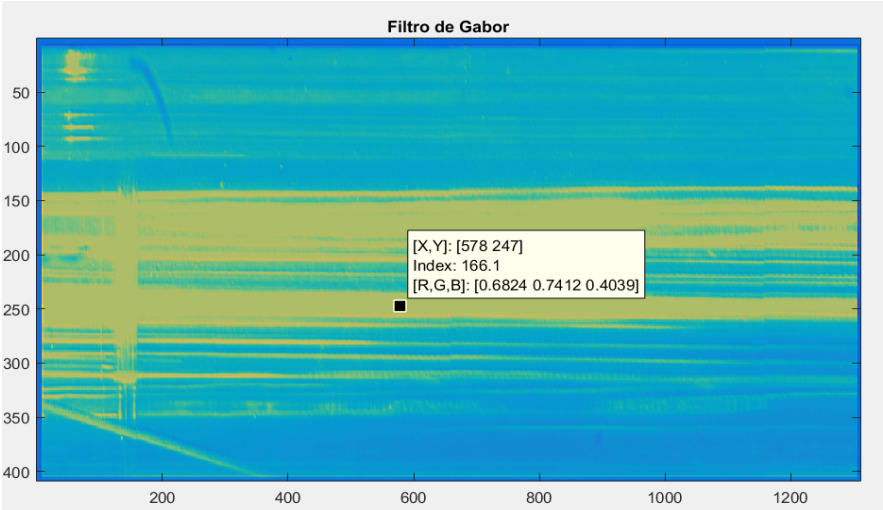


Figura 68: Filtro de Gabor a nivel2.

- Aplicación del filtro de Gabor al nivel 5:

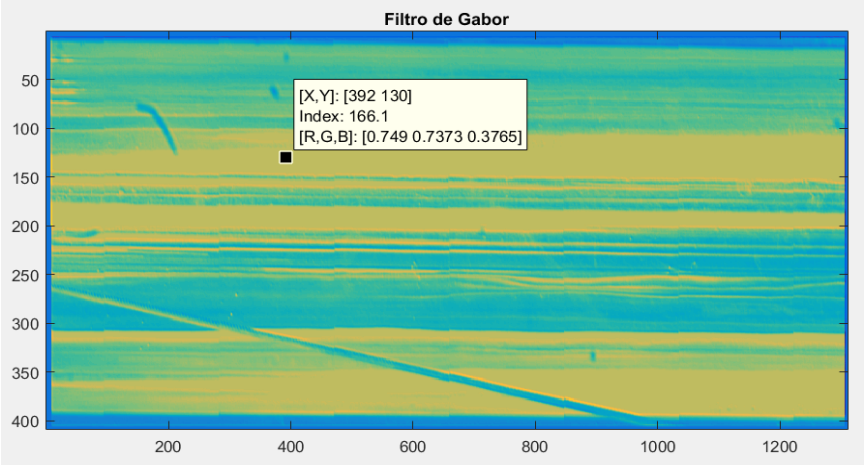


Figura 69: Filtro de Gabor a nivel5.

Los resultados están claros, el filtro nos resalta las zonas con mayor suciedad.

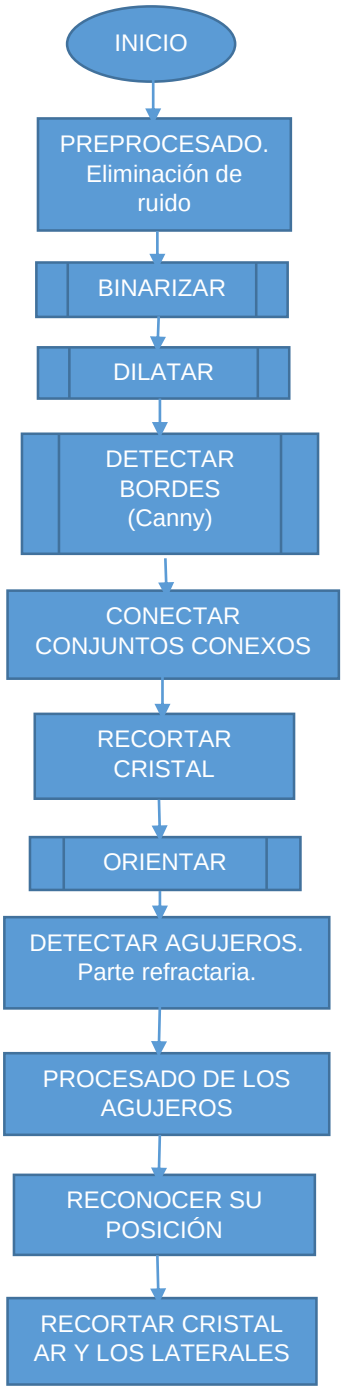
En el nivel 1 la imagen es casi toda azul con un valor de 40, conforme aumenta la suciedad en los cristales AR van apareciendo unas zonas de un color naranja de un valor 166 que nos indican donde esta acumulado el polvo.

Las zonas coloreadas de naranja serían las zonas que aparecen blanquecinas en las fotografías originales. Estas zonas tienen un tono opaco que no deja pasar la luz reflejada del sol para calentar el agua de la tubería y convertirla en vapor de agua.

Por esto es importante que los cristales AR estén limpios para que el rendimiento de la planta sea óptimo.

4.3 NUEVAS IDEAS.

Se está desarrollando otro algoritmo para recortar los cristales AR, pero debido a los reflejos que hay en las imágenes aún no es totalmente eficiente.



A continuación, se da una explicación del algoritmo:

- Pre-procesado de la imagen
  - Eliminación del ruido.

    Los píxeles de la nueva imagen se generan calculando la mediana del conjunto de píxeles del entorno de vecindad del píxel correspondiente a la imagen origen. De esta forma se homogeneizan los píxeles de intensidad muy diferente con respecto a la de los vecinos. Este tipo de filtro es bastante indicado cuando se tiene ruido aleatorio. En Matlab la instrucción empleada para realizar el filtro de la mediana es medfilt2 [27].

    B = medfilt2(A)

    Donde A es la matriz de entrada a la que se le aplica el filtro de la mediana utilizando por defecto una vecindad de 3X3.

  - Binarización con el método Otsu. Usando el comando graythresh.

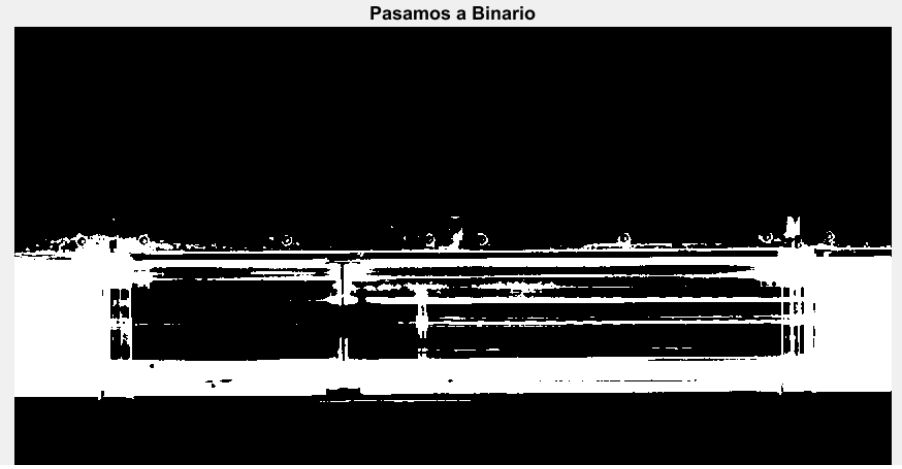


Figura 70: Binarización del segundo algoritmo.

  - Dilatación. Para hacer la región a detectar más grande.

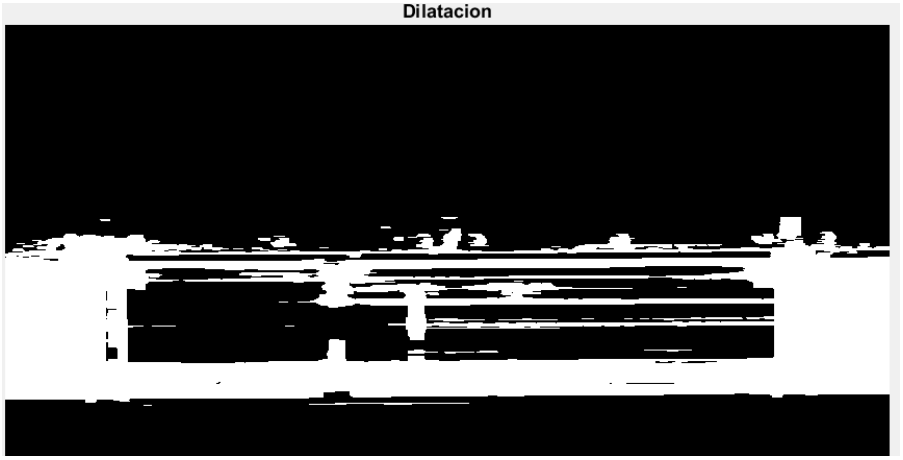


Figura 71: Dilatación en el segundo algoritmo.

Este será el código a escribir en Matlab:

```
B = medfilt2(Resize, [5 5]);

umb=graythresh(B);
Cd=im2bw(B,umb);

Se=strel('line',25,0);
P=imdilate(Cd,Se);
```

- Detección de bordes con el método canny. Operador de primera derivada al igual que sobel funciona con el gradiente.

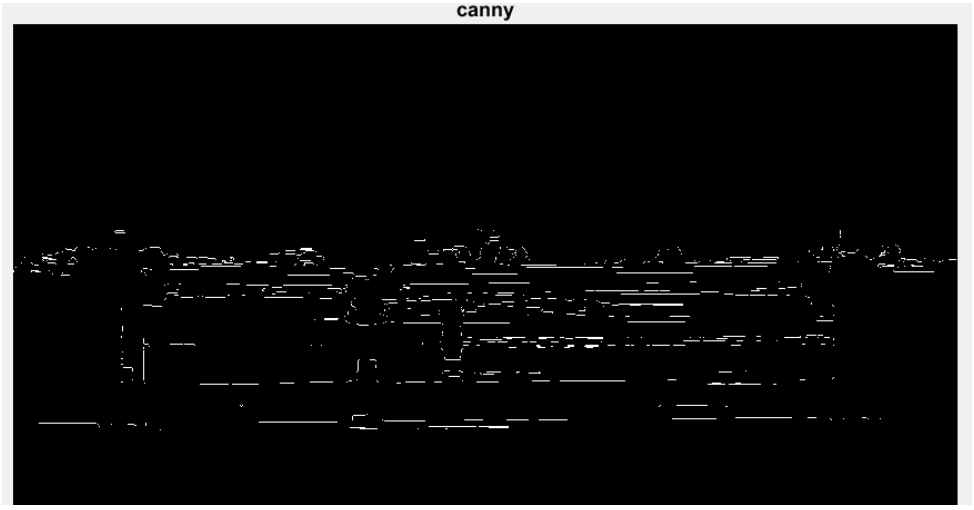


Figura 72: Detección de bordes con Canny.

- Para rellenar los huecos de los conjuntos conexos usamos el comando el Matlab imfill que rellenará los huecos en la imagen binaria.

```
L =bwlabel(E);
d2 = imfill(L, 'holes');
```

Figura 73: Rellenar conjuntos conexos.

- Una vez conectada toda el área del cristal se pasa a su detección usando el comando regionprops para detectar el área mayor y recortar con imcrop. Esta vez usaremos la propiedad centroide para el recorte que nos detecta el centro del objeto a recortar.

```
Rec=imcrop(Resize,[prop(1).Centroid(1)-(2560/2),prop(1).Centroid(2)-(680/2),2560,670]);
```

El resultado es:

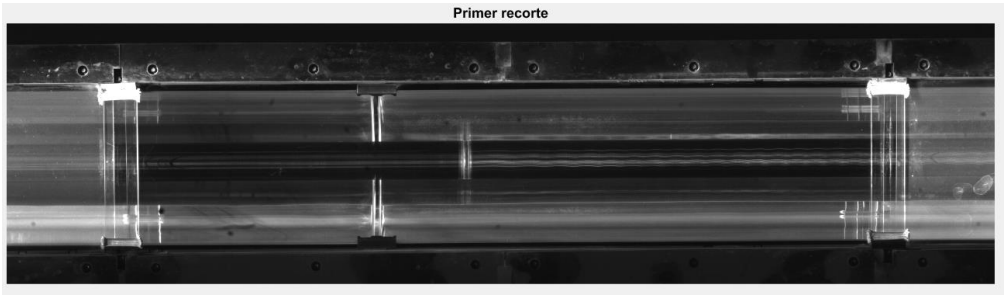


Figura 74: Primer recorte del cristal.

- Orientación del cristal, como anteriormente se hizo.
- Después se pasará al segundo recorte. Se realizará detectando los agujeros (tornillos para la sujeción del cristal).
  Realizamos operaciones morfológicas de erosión y dilatación para extraer los contornos del cristal.

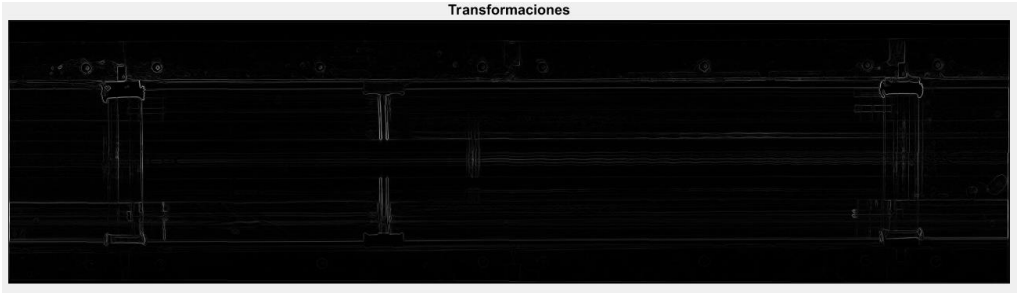


Figura 75: Transformaciones morfológicas.

Posteriormente se usará la máscara de convolución explicada en puntos anteriores para resaltar el contorno extraído.

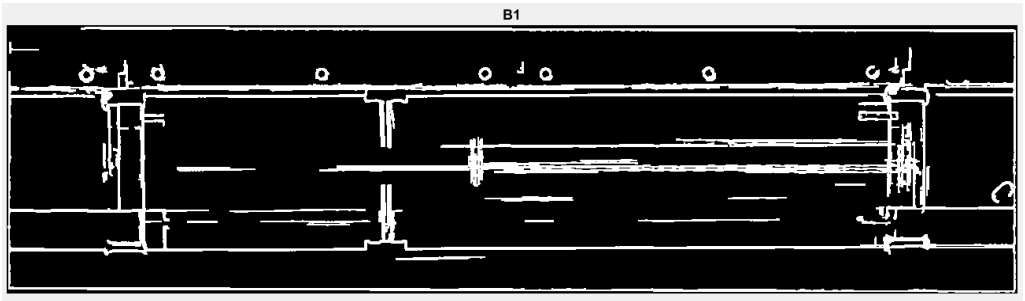


Figura 76: Aplicación de máscara de convolución.

Como lo que pretendemos es usar los agujeros para recortar el cristal volvemos a usar el comando imfill para rellenar los agujeros.

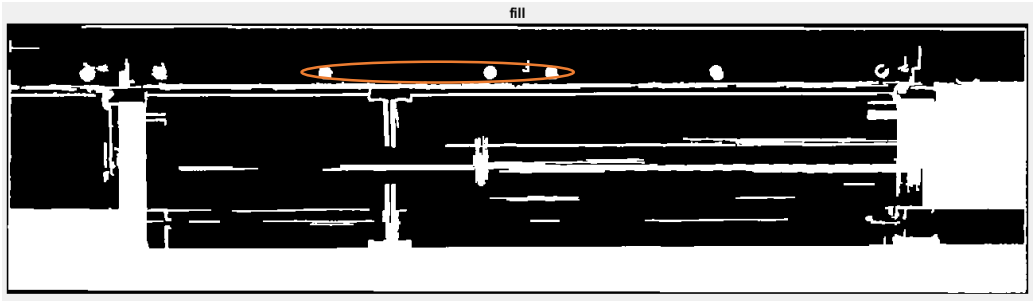


Figura 77: Reconocer áreas redondas del cristal.

Lo siguiente será reconocerlos con regionprops.

```
[LFill,N_Area]=bwlabel(Fill);
propFill=regionprops(LFill,'Area','BoundingBox','Centroid', 'Eccentricity');

for indiceCirculo = 1:N_Area
    if (propFill(indiceCirculo).Centroid(2) < 165) || (propFill(indiceCirculo).Centroid(2) > 545)
        indice = indiceCirculo;
        if (propFill(indice).Eccentricity < 0.55)
        found = indice;
        area = propFill(found).Area;
        radio = sqrt(area/pi);
        radio = 1.2 * radio;
        centroide = propFill(found).Centroid;
        t = 0:pi/20:2*pi;
        x = centroide(1) + radio*cos(t);
        y = centroide(2) + radio*sin(t);
        hold on
       plot(x,y,'linewidth',3);
        end
    end
end
```

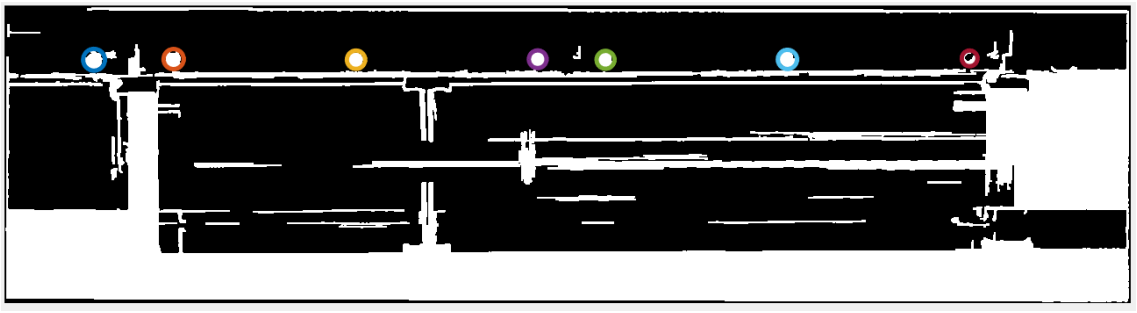


Figura 78: Reconocimiento de los círculos.

Por último, necesitamos el índice de los agujeros para usar imcrop con las sus propiedades.

Si se detecta el agujero de arriba en la posición dos (el alto) de imcrop se suman 70 pixeles que es la distancia calculada de un agujero a el cristal AR.

```
Rec = imcrop(Final_S,[0,propFill(IndiceUp).BoundingBox(2)+70,2560,400]);
```

Si es en la parte de abajo se resta 440 que es la parte sobrante de abajo:

```
Rec = imcrop(Final_S,[0,propFill(IndiceDown).BoundingBox(2)-440,2560,400]);
```

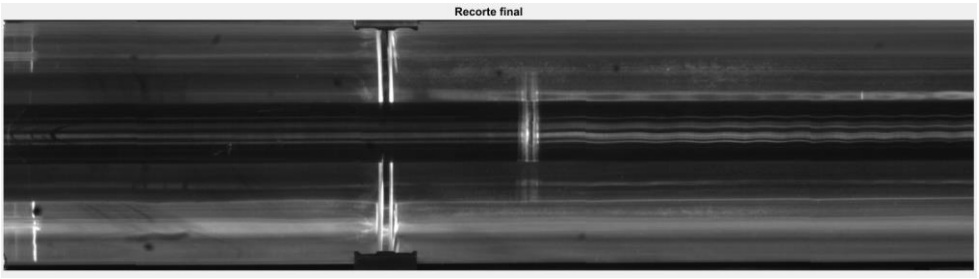


Figura 79: Recorte final con el segundo algoritmo. Fuente: Propia.

4.4 INTERFAZ GRÁFICA

Para poder realizar la adquisición de imágenes a partir de un sistema de visión instalado en el equipo desde Matlab, es necesario crear un objeto de video asociado al mismo. Para ello, podemos introducir la siguiente línea de código:

```
>> obj=videoinput('gige',1);
```

De esta forma conectamos la cámara instalada en nuestro ordenador a Matlab, pudiéndose alterar distintos parámetros de configuración del dispositivo, parámetros como la resolución, el tratamiento de la señal de video, el número de fotografías por disparo (framesPerTrigger), etc. Para la adquisición de imágenes, Matlab ofrece dos alternativas: mediante capturas, o mediante grabación o extracción de frames. Se ha comprobado que la mayoría de cámaras tardan un tiempo en adaptarse a la luminosidad del entorno, una vez iniciado el proceso de captura. Este hecho, unido a que el tiempo de fotografiado será siempre mayor que al de la extracción de frames de video, nos hace descartar la primera opción. Para la extracción de frames, es necesario iniciar la grabación con el objeto de video. Esto puede realizarse mediante la función start de Matlab [28].

```
>> start(obj);
```

Posteriormente, previsualizamos el objeto de video mediante el comando preview.

```
>> preview(obj);
```

Para la extracción de un frame a tiempo real, utilizamos el comando getsnapshot sobre el objeto:

```
>> getsnapshot(obj);
```

Es muy importante utilizar este comando después de haber inicializado la previsualización del objeto, para que la cámara pueda adaptarse a las condiciones de luminosidad del entorno. De lo contrario, la calidad de la imagen será muy pobre.

Creación de una interfaz gráfica en Matlab.

- Vamos a new >> Graphical User Interface

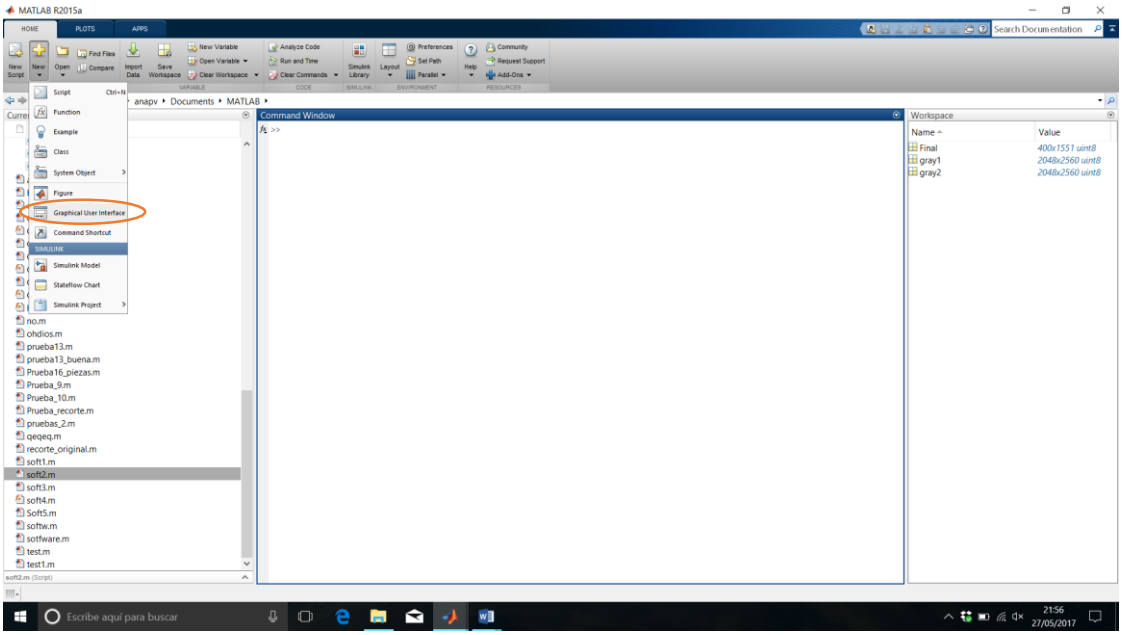


Figura 80: Creación de GUI. Fuente: Propia.

- Creamos una en blanco por defecto:

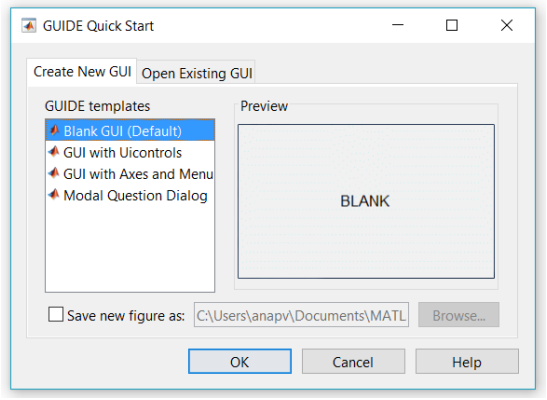


Figura 81: Gui en blanco.

- Se nos abriría la ventana del entorno de la GUI:

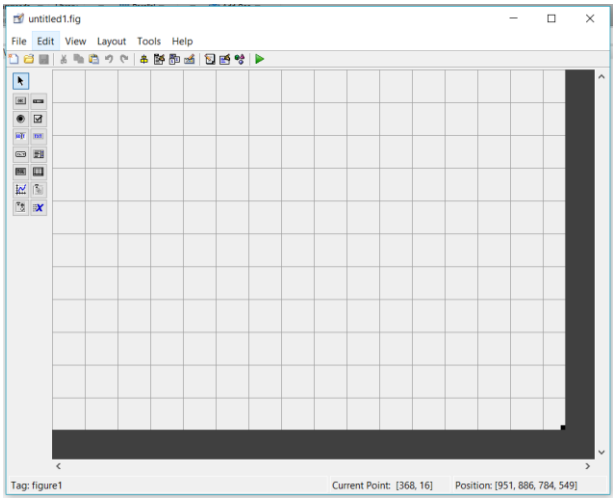


Figura 82: Entorno de trabajo de GUI.

Mediante el editor de diseño de GUIDE, podemos diseñar gráficamente la interfaz de usuario. GUIDE genera entonces de manera automática el código de Matlab para construir la interfaz, el cual modificamos para programar el comportamiento de nuestra app.

A fin de ejercer un mayor control sobre nuestro diseño y el desarrollo, también creamos código en Matlab que defina las propiedades y los comportamientos de todos los componentes usados. Matlab contiene funcionalidad integrada que nos ayudará a crear la GUI para nuestra app de forma automática. Podemos agregar cuadros de diálogo, controles de interfaz de usuario (como botones y controles deslizantes) y contenedores (como paneles y grupos de botones) [29].

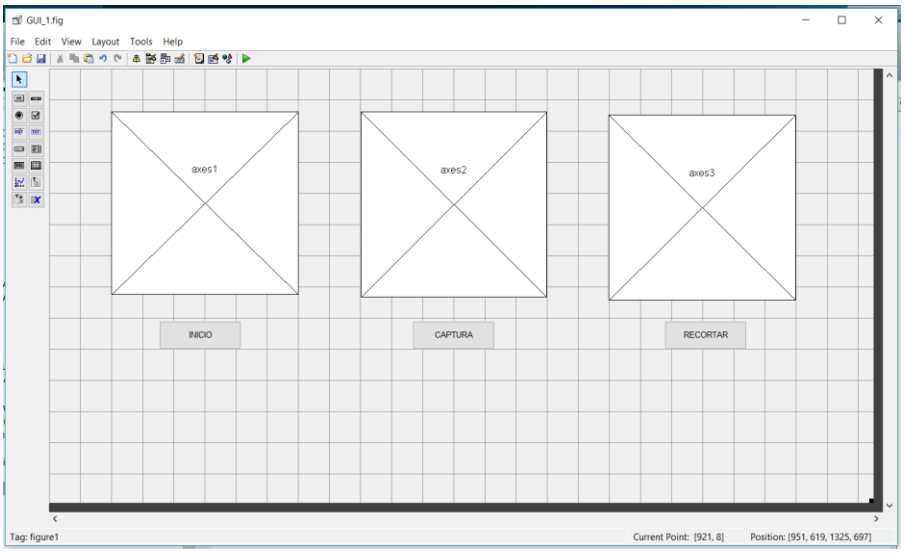


Figura 83: Creación de botones en la GUI. Fuente: Propia.

Una aplicación GUIDE consta de dos archivos .m y.fig. El archivo .m es el que contiene el código con las correspondencias de los botones de control de la interfaz y el archivo. fig contiene los elementos gráficos. Cada vez que se adicione un nuevo elemento en la interfaz gráfica, se genera automáticamente código en el archivo.m.

Los dos primeros botones sirven para iniciar la cámara de video y capturar una frame respectivamente. Mientras que el tercero sirve para mostrar la imagen recortada. La asignación u obtención de valores de los componentes se realiza mediante las sentencias get y set. Todos los valores de las propiedades de los elementos y valores de las variables transitorias se almacenan en una estructura llamada *handles.* Y la función que guarda los datos de la aplicación es *guidata.* El código para grabar y capturar la imagen es el siguiente:

```
% --- Executes on button press in Inicio.
function Inicio_Callback(hObject, eventdata, handles)
global ini
ini=1;
VIDEO (hObject, eventdata, handles)

% --- Executes on button press in Captura.
function Captura_Callback(hObject, eventdata, handles)
global cap
cap=1;
axes(handles.axes2)
imshow(cap)
set(handles.axes1,'UserData',cap);
VIDEO (hObject, eventdata, handles)


function VIDEO (hObject, eventdata, handles)
global ini
global cap
if ini==1
    vid=videoinput('gige', 1);
    triggerconfig(vid,'manual');
    start(vid)
    while 1
        axes(handles.axes1)
        snapshot=getsnapshot(vid);
        imshow(snapshot)
           if cap==1
            break;
        end

    end
    cap=0;
      delete (vid)
end
```

La interfaz se vería así:

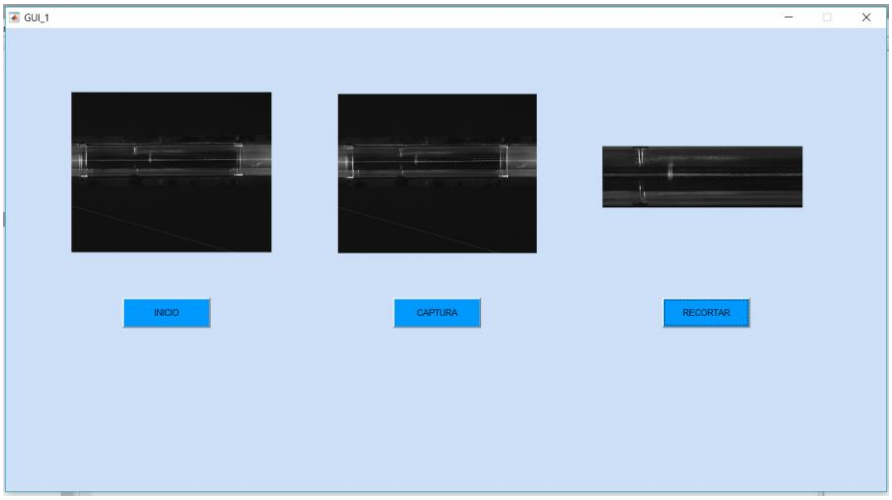


Figura 84: GUI_1. Fuente: Propia.

Se decide que también nos muestre el nivel de suciedad que tiene la imagen recortada. Para esto se le añade un botón más para que nos imprima el nivel por pantalla, a ese botón solo hay que añadir la función:

```
set(handles.text5,'String', Nivel);
```

y añadir en la interfaz un *static text* que mostrara el valor.

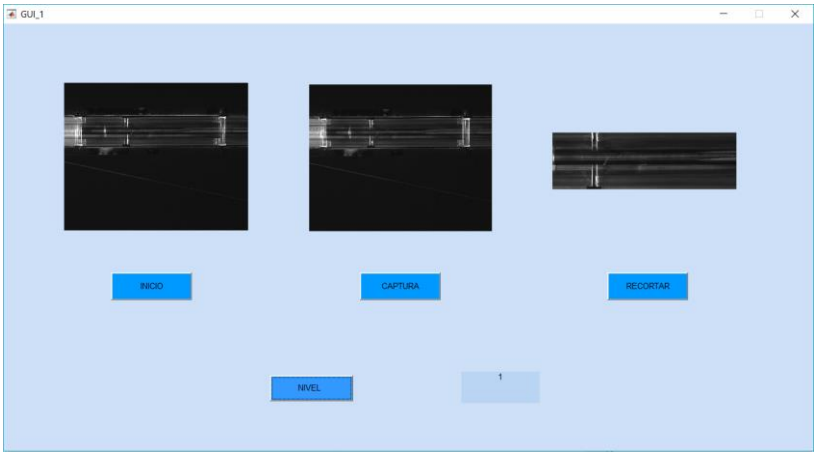


Figura 85: GUI_Nivel de suciedad 1. Fuente: Propia.

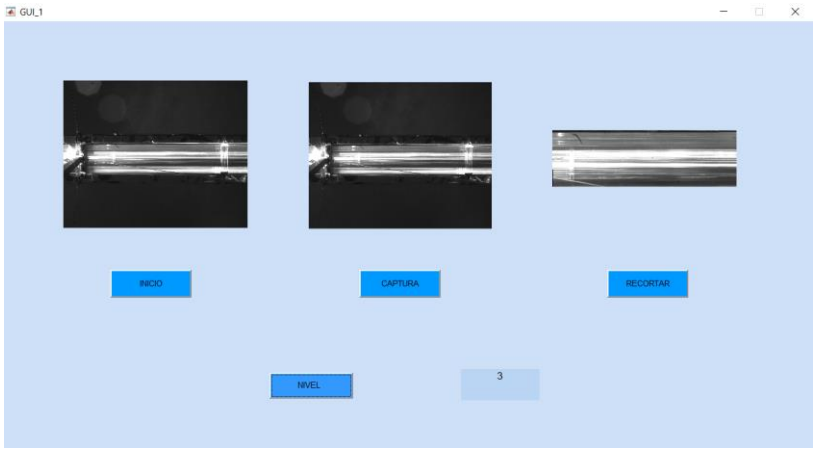


Figura 86: GUI_Nivel de suciedad 3. Fuente: Propia.

Para hacer nuestra interfaz más completa se decide añadir más elementos a la interfaz. Así como dos botones para seleccionar en que época del año nos encontramos y un static text que nos muestre el porcentaje de suciedad.

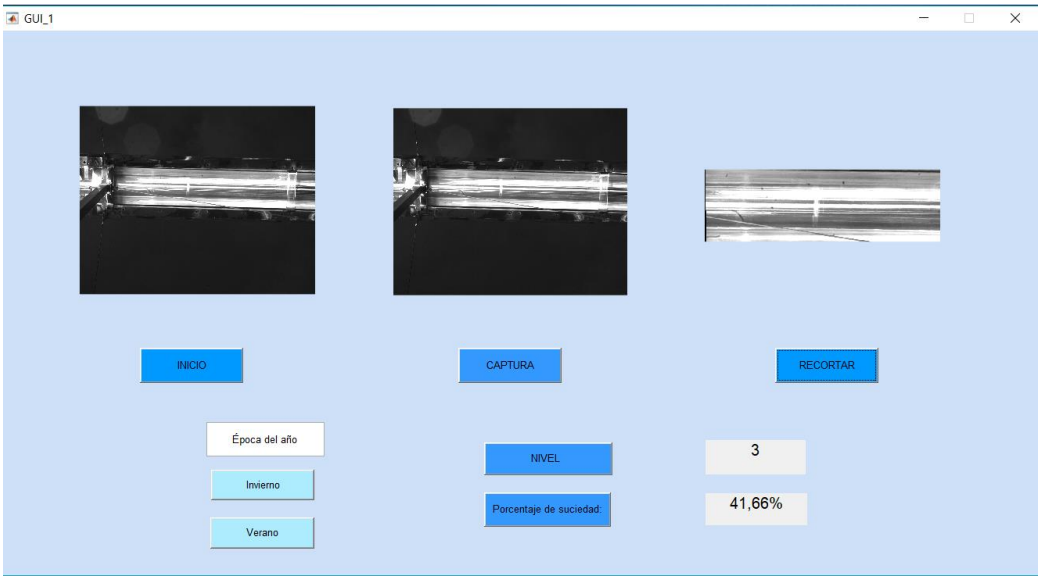


Figura 87: Interfaz Gráfica en Matlab.

4.5 PROGRAMACIÓN EN VISUAL STUDIO

En este apartado del trabajo explicaremos cómo codificar a lenguaje C el algoritmo explicado en el apartado 4.3 de programación en Matlab para que esté escrito en un lenguaje adecuado para un entorno de producción, sobre todo por su velocidad de ejecución al ser un lenguaje compilado, mientras que Matlab es un entorno de desarrollo de laboratorio principalmente.

La librería OpenCV proporciona un marco de trabajo de alto rendimiento para el desarrollo de aplicaciones de visión por computador en tiempo real mientras que Matlab trabaja a bajo-medio rendimiento si contamos con un buen procesador en nuestro ordenador. También proporciona varios paquetes de alto nivel para el desarrollo de aplicaciones de visión. Todos los paquetes se pueden agrupar en librerías de C/C++ [30].

Ahora pasaremos de usar la librería de OpenCV de la Toolbox para Matlab a usar la librería HighGUI para poder programar en el entorno de Visual Studio.

Explicación del proceso de recorte del cristal AR:

Antes de empezar a programar hay que incluir las librerías en el encabezado:

```
1   #include <iostream>
2   #include <opencv2/opencv.hpp>
3   #include "opencv2/core/core.hpp"
4   #include<math.h>
```

```
23  int main(int, char** argv)
24  {
25      Mat imagen; //Mat es básicamente una clase con dos partes de datos
26                  //la cabecera de matriz un puntero a la matriz que contiene el valores de píxel
27      Size tam;
28      //namedWindow("Ventana", CV_WINDOW_NORMAL); sirve para crear una ventana y mostrar la imagen
29      imagen = imread("../data/2.bmp");
30      //imshow("Ventana", imagen);
31      tam = imagen.size();
32      cout << "Ancho :" << tam.width << "\t" << "Alto:" << tam.height << endl;
33
34      // Transforma a gray si esta en RGB, sino lo mantiene en gray
35      Mat gray;
36      if (imagen.channels() == 3)
37      {
38          cvtColor(imagen, gray, CV_BGR2GRAY);
39      }
40      else
41      {
42          gray = imagen;
43      }
```

- Recortar los laterales.

  Es simple, se crea una región de interés. Roi funciona extrayendo una submatriz del rectángulo que le indicamos [30].

```
Mat roiImg = resimg(Rect(415, 0, 1300, 2560));
namedWindow("Rec", CV_WINDOW_NORMAL);
imshow("Rec", roiImg);
```

Figura 88: Recortar laterales con OpenCV.

- Binarización:

  Igualmente se usa el método Otsu para umbralizar.

```
threshold(roiImg, bw, 0, 255, THRESH_BINARY | THRESH_OTSU);
```

Figura 89: Binario con OpenCV.

Para eliminar las áreas pequeñas indeseadas se usa la operación morfológica *open*, primero hay que crear una estructura para decir que elementos queremos eliminar y luego usar MORPH_OPEN.

```
//  Openig...
Mat const structure_elem = cv::getStructuringElement(MORPH_RECT, cv::Size(12, 8));
Mat open_result;
morphologyEx(bw, open_result, MORPH_OPEN, structure_elem);
```

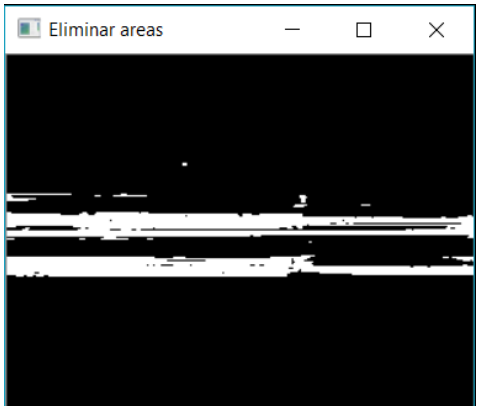


Figura 90: Area_Open con OpenCV.

- Dilatación.
  Creamos una función porque durante el proceso se dilatará varias veces.

```
//funcion dilatacion
void Dilation(int, void*)
{
    int dilation_type;

    Mat element = getStructuringElement(MORPH_RECT, Size(6 * dilation_size + 1, 6 * dilation_size + 1),
        Point(dilation_size, dilation_size));
    //Aplicar dilatacion
    dilate(bw, dilation_dst, element);

    //namedWindow("Dilatacion", CV_WINDOW_NORMAL);
    //imshow("Dilatacion", dilation_dst);

}
```

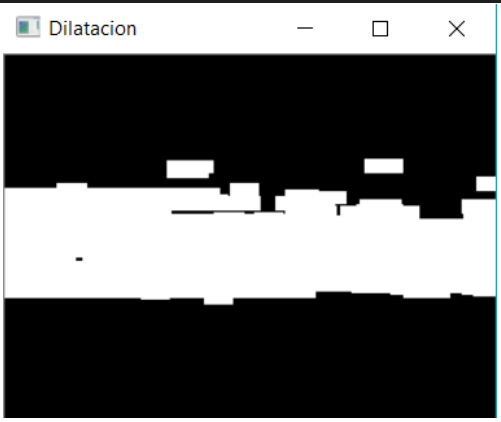


Figura 91: Dilatación con OpenCV.

- Extracción de la región.

  Lo que hacemos es detectar el contorno del rectángulo mayor y lo recortamos con roiImg.

  FindContours detecta los contornos de una imagen en binario y los guarda en un vector de puntos [31].

  El código quedaría de la siguiente forma:

```
//  Recortar el cristal...

Mat canny_output;
vector<vector<Point> > contorno;
vector<Vec4i> niveles;
RNG rng(12345);

// Buscar los contornos de la imagen, se almacenan en contours
findContours(dilation_dst, contorno, niveles, CV_RETR_TREE, CV_CHAIN_APPROX_SIMPLE, Point(0, 0));

//Extraemos contornos
    int contador = 0;
Rect bounding_rect;
vector<Rect> boundRect(contorno.size());
vector<vector<Point> > contorno_poly(contorno.size());
Mat drawing = Mat::zeros(dilation_dst.size(), CV_8UC3);
```

```
for (int i = 0; i < contorno.size(); i++)
{
    contador = contorno.size();
    cout << "contador contornos:" << contador << endl;
    cout << " Area =  " << contourArea(contorno[i]) << endl;
    bounding_rect = boundingRect(contorno[i]);
    cout << "BoundingRect:" << bounding_rect << endl;
    approxPolyDP(Mat(contorno[i]), contorno_poly[i], 3, true);
    boundRect[i] = boundingRect(Mat(contorno_poly[i]));

    Scalar color = Scalar(rng.uniform(0, 255), rng.uniform(0, 255), rng.uniform(0, 255));
    drawContours(drawing, contorno_poly, (int)i, color, 1, 8, vector<Vec4i>(), 0, Point());
    rectangle(drawing, boundRect[i].tl(), boundRect[i].br(), color, 2, 8, 0);

    Img_rec = roiImg(Rect(boundRect[i].tl(), boundRect[i].br()));
    namedWindow("Imagen recortada", WINDOW_NORMAL);
    imshow("Imagen recortada", Img_rec);
```

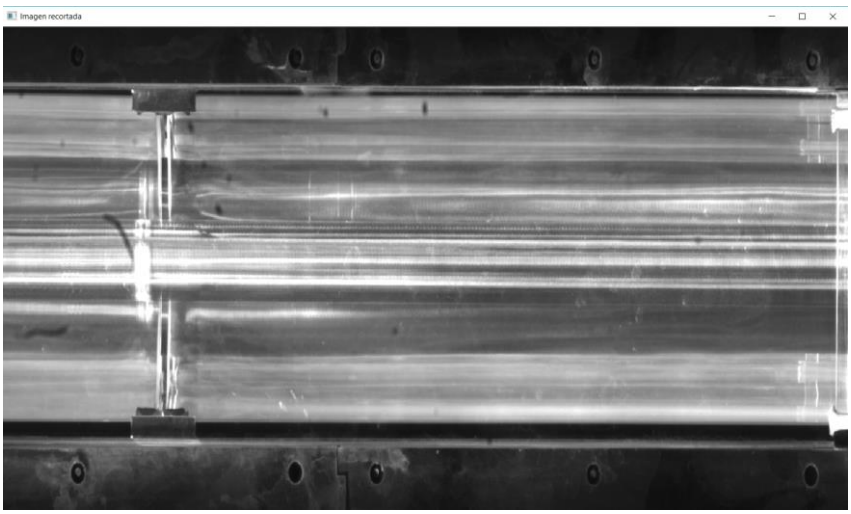


Figura 92: Imagen recortada.

- Detección de bordes.

  Para detectar las líneas en horizontal aplicaremos la función sobel en dirección horizontal y mostraremos las líneas detectadas [31]:

```
/// Generar grad_x and grad_y
Mat  grad_x;
Mat  abs_grad_x;

/// Gradient x
    int scale = 1;
int delta = 0;
int ddepth = CV_16S;
Sobel(Img_REC, grad_x, ddepth, 1, 0, 3, scale, delta, BORDER_DEFAULT);
convertScaleAbs(grad_x, abs_grad_x);
    imshow("sobel", abs_grad_x);
/// Muestra lineas eh horizontal
Mat bw1;
adaptiveThreshold(Img_REC, bw1, 255, CV_ADAPTIVE_THRESH_MEAN_C, THRESH_BINARY, 15, -2);

Mat horizontal = bw1.clone();
int horizontalsize = 500;
Mat horizontalStructure = getStructuringElement(MORPH_RECT, Size(horizontalsize, 1));
dilate(horizontal, horizontal, horizontalStructure, Point(-1, -1));
imshow("horizontal", horizontal);
```

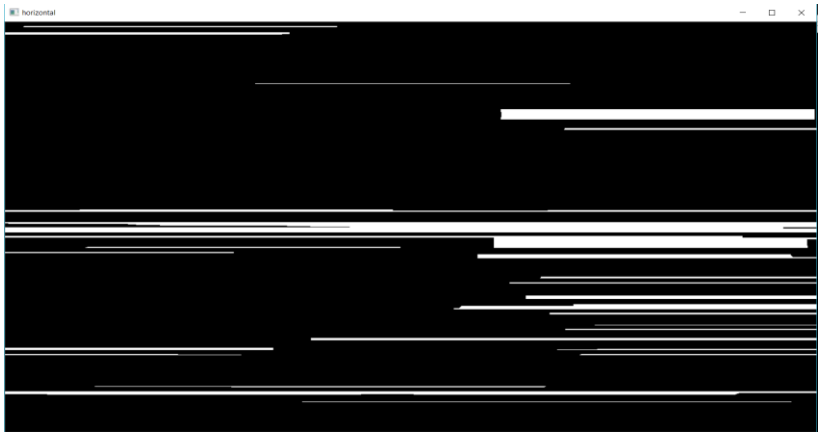


Figura 93: Detección de líneas con OpenCV.

O usar Canny y dibujar los contornos:

```
// Detectar los bordes con canny
Canny(horizontal, canny_sal, thresh, thresh * 2);
    // Buscar los contornos de la imagen, se almacenan en contours
findContours(canny_sal, lineas, hierarchy, CV_RETR_TREE, CV_CHAIN_APPROX_SIMPLE, Point(0, 0));

// Mostrar la Imagen modificada por la funcion findContours
imshow("Modificada", canny_sal);

// Dibujar los contornos encontrados
Mat dibujar = Mat::zeros(canny_sal.size(), CV_8UC3);
for (size_t i = 0; i< lineas.size(); i++)
{
    Scalar color = CV_RGB(0, 255, 0);
    drawContours(dibujar, lineas, (int)i, color, 2, 8, hierarchy, 0, Point());
}
```

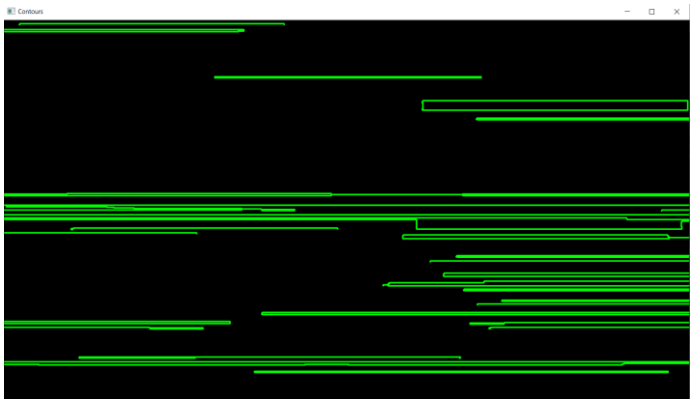


Figura 94: Contornos con OpenCV.

Una vez detectadas las líneas usamos boundingRect que es como BoundingBox de Matlab para guardar la posición de las líneas en horizontal. Esto se hizo en el primer recorte. A continuación, creamos un bucle for para encontrar la línea que representa la parte refractaria. Esta línea suele estar entre los valores 30 y 110.

Se recorta otra vez con roiImg:

```
Img_rec = roiImg(Rect(boundRect[i].tl(), boundRect[i].br()));
```

boundRect[i].tl() → nos da la anchura del rectángulo a recortar

boundRect[i].br() → nos da la altura.

Una vez ejecutado el algoritmo el cristal recortado quedaría así:

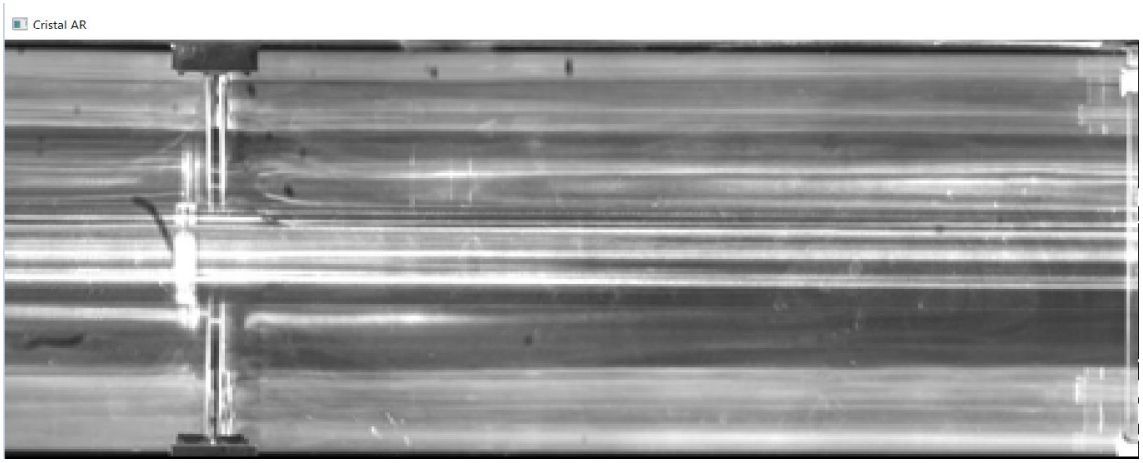


Figura 95: Cristal AR con OpenCV.

CÁPITULO 5

Resultados.

5.1 INTRODUCCIÓN

En este capítulo se recopila toda la información de los resultados obtenidos en las pruebas que se han realizado durante el proceso de clasificación de los niveles de suciedad para llegar a unas conclusiones finales. Además, se expondrá alguno de los principales problemas que han surgido durante la realización de este proyecto y se propondrán mejoras que se pueden tratar en futuros proyectos para mejorar el proceso.

5.2 PRUEBAS Y RESULTADOS

Una vez el proceso ha sido programado en su totalidad y se ha comprobado que la funcionalidad cumple con los objetivos marcados en un principio, se han realizado pruebas en las que se ha tenido en cuenta los siguientes puntos:

- Si el software ha sido capaz de recortar el AR a partir de la fotografía.
- Número de intentos de recorte por parte del software.
- Número total de imágenes recortadas correctamente.
- Tiempo total del proceso.

A partir de estas pruebas se ha elaborado un cuadro resumen de esos puntos críticos para poder obtener finalmente las conclusiones y proponer qué se podría mejorar en proyectos futuros.

Las fotografías se nombran:

XX.S.M.C

Dónde XX es la fila, S es el sector, M es el módulo y C es el cristal.

Prueba 1: 14 diciembre

Número de foto	¿Recorta?	Intentos de lectura
9.1.1.1	Sí	2
9.1.1.2	Sí	1
9.1.1.3	Si	1
9.1.1.4	Sí	1
Tiempo del proceso	2 s	
Recortadas correctas	4	
Tiempo medio de recorte	1,5 s	

Tabla 2: Prueba 1

Prueba 2: 4 enero

Número de foto	¿Recorta?	Intentos de lectura
8.1.6.1	Sí	1
8.1.6.2	Sí	3
8.1.6.3	Si	1
8.1.6.4	Sí	1
Tiempo del proceso	2,1s	
Recortadas correctas	4	
Tiempo medio de recorte	1,65s	

Tabla 3: Prueba 2

Prueba 3: 14 febrero

Número de foto	¿Recorta?	Intentos de lectura
13.1.2.1	Sí	1
13.1.2.2	Sí	2
13.1.2.3	Si	1
13.1.2.4	Sí	4
Tiempo del proceso	2,3s	
Recortadas correctas	4	
Tiempo medio de recorte	1,9s	

Tabla 4: Prueba 3

Prueba 4: 27 marzo

Número de foto	¿Recorta?	Intentos de lectura
4.1.1.1	Sí	1
4.1.1.2	Sí	3
4..1.1.3	Si	1
4.1.1.4	Sí	1
Tiempo del proceso	2,1s	
Recortadas correctas	4	
Tiempo medio de recorte	1,30s	

Tabla 5: Prueba 4

Resultados de pruebas:

Pruebas realizadas	4	
Imágenes recortadas	16	
Recorte correcto	16	100%
Media de intentos	1,56	
Tiempo medio del proceso	1,58s	

Tabla 6: Resultados de las pruebas

Los fallos de lectura que se dan son por los cambios de época que nos cambian las imágenes, pero entre prueba y prueba se ha ido mejorando el algoritmo para que sea más robusto y a la vez sencillo, para que el tiempo de procesado sea mínimo.

A continuación, se muestran los resultados obtenidos durante las distintas épocas del año:

- Se hizo una prueba con la cámara CANON:
  17 de noviembre

Nivel de la imagen.	Nivel de suciedad
nivel 2	80
nivel 3	60
nivel 4	91
nivel 5	134

Tabla 7: Resultados con cámara Canon

Para probar la cámara se hicieron diferentes fotos a los cristales, pero sabiendo cuál era su nivel de suciedad y así comprobar que valores nos salían. El problema es que la cámara se calibra sola al echar las fotos y el nivel tres supuestamente estaría más limpio que el dos. Por ello se descartó la cámara.

- Imágenes sin filtro: 15 de diciembre

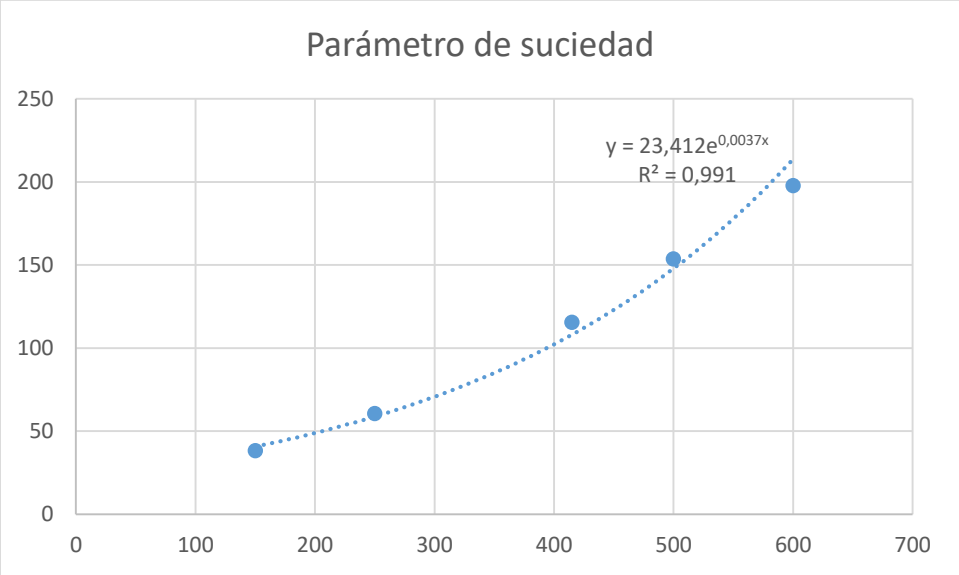
En esta tabla solo se muestra el parámetro de suciedad, pero sin distinguir los niveles, el valor de la columna nivel no es el valor que nos da el programa, sino que es el que apreciamos al echar la foto.

F	S	M	C	Nivel	Radiación	T exp	Parámetro de suciedad
9	1	1	1	1	801	150	0
9	1	1	2	2	825	150	0
8	1	6	1	5	814	150	38,32
9	1	1	4	2	800	250	0
8	1	6	1	5	830	250	60,67
9	1	1	1	2	852	415	2,26
9	1	1	2	3	799	415	11,4
9	1	1	3	4	805	415	126,7
9	1	1	4	5	815	415	115,57
9	1	1	4	2	800	500	2,79
8	1	7	4	3	820	500	21,39
7	1	1	1	4	802	500	140,73
8	1	6	1	5	817	500	153,82
9	1	1	4	2	820	600	3,3
8	1	7	4	3	810	600	38,57
7	1	1	1	4	890	600	154,57
8	1	6	1	5	800	600	197,92

Tabla 8: Resultados sin filtro del parámetro de suciedad con tiempos de exposición distintos

En la siguiente gráfica se representa nivel de gris (nivel de suciedad) frente el tiempo de exposición en el mismo cristal para ver cómo afecta el tiempo de exposición a la toma de imágenes.

El tiempo de exposición influye exponencialmente en el nivel de suciedad de las imágenes. Sigue una distribución exponencial ya que el coeficiente de determinación $R^2 = 0.99$



Por el motivo de la variación de los parámetros con el tiempo de exposición se fija un tiempo de exposición a 600.

Se establecen unos valores para los distintos niveles, como se explicó anteriormente el parámetro de suciedad es la diferencia de los niveles de gris de la imagen de referencia con nivel uno y de la que queremos analizar.

Nivel	Parámetro suciedad
1	0-15
2	15-45
3	45-75
4	75-110
5	>110

Tabla 9: Distintos niveles de suciedad en invierno

F	S	M	C	Rad.	T. exp	Parámetro suciedad	Nivel se suciedad
8	1	6	1	800	600	143,77	5
8	1	6	2	800	600	123,12	5
8	1	6	3	800	600	85,49	4
8	1	6	4	800	600	77,25	3_4
8	1	7	1	800	600	55,21	3
8	1	7	2	800	600	53,84	3
8	1	7	3	800	600	103,57	4

Tabla 10: Prueba de niveles de imágenes sin filtro.

- Imágenes con filtro: 1 enero
    - La tabla siguiente muestra el nivel de gris de la misma imagen (nivel 1) para diferentes radiaciones y tiempos de exposición.

F	S	M	C	Rad	T. exp	Nivel de gris
9	1	1	1	750	2100	29,21
9	1	1	1	800	2100	37,31
9	1	1	1	850	2100	42,34
9	1	1	1	900	2100	49,45
9	1	1	1	950	2100	52,21
9	1	1	1	750	2000	27,99
9	1	1	1	800	2000	36,65
9	1	1	1	850	2000	41,09
9	1	1	1	900	2000	43,31
9	1	1	1	950	2000	48,5
9	1	1	1	750	1900	27,52
9	1	1	1	800	1900	30,86
9	1	1	1	850	1900	38,22
9	1	1	1	900	1900	39,63
9	1	1	1	950	1900	40,1
9	1	1	1	750	1800	24,64
9	1	1	1	800	1800	29,2
9	1	1	1	850	1800	30,8
9	1	1	1	900	1800	37,47
9	1	1	1	950	1800	41,47

Tabla 11: Resultados de imagen con nivel 1 cambiando radiación y T. exp

    - La tabla siguiente muestra el nivel de gris de la misma imagen (nivel 5) para diferentes radiaciones y tiempos de exposición.

F	S	M	C	Rad	T. exp	Nivel de gris
8	1	6	1	750	2100	78,62
8	1	6	1	800	2100	89,80
8	1	6	1	850	2100	115,60
8	1	6	1	900	2100	123,00
8	1	6	1	950	2100	135,05
8	1	6	1	750	2000	75,00
8	1	6	1	800	2000	102,60
8	1	6	1	850	2000	121,77
8	1	6	1	900	2000	125,00
8	1	6	1	950	2000	139,01
8	1	6	1	750	1900	73,90
8	1	6	1	800	1900	95,47
8	1	6	1	850	1900	112,08
8	1	6	1	900	1900	115,78
8	1	6	1	950	1900	124,75
8	1	6	1	750	1800	71,00
8	1	6	1	800	1800	97,07
8	1	6	1	850	1800	116,08
8	1	6	1	900	1800	120,87
8	1	6	1	950	1800	124,77

Tabla 12: Resultados de imagen con nivel 5 cambiando radiación y T. exp

Está prueba se realizó tras poner el filtro para elegir un tiempo de exposición que nos de los valores de la tabla 9.

Se elegirá un tiempo de exposición de 2000 y la radiación se mantiene como antes entre 800-900 w/m^2.

- Imágenes del 28 de marzo

F	S	M	AR pos	Radiación	T. exp	Nivel	No. foto	N. suciedad	N. de gris del cristal
9	1	2	1	996	2100	**2**	1	51,03	104,39
					2000		1.1	45,40	98,28
					1900		1.2	31,70	78,18
					1800		1.3	16,04	61,65
9	1	1	4	996	2100	**2**	2	41.31	92,92
					2000		2.1	29,90	69,07
					1900		2.2	22,20	65,57
					1800		2.3	21,50	51,48
9	1	7	3	1000	2100	**2/3**	3	57,20	109,35
					2000		3.1	46,30	101.17
					1900		3.2	25,00	74,90
					1800		3.3	23,00	109.35
4	1	1	1	1000	2100	**3**	4	91,00	146,21

					2000		4.1	76,90	131,90
					1900		4.2	72,71	127,65
					1800		4.3	65,07	119,90
4	1	1	3	1000	2100	**3**	5	45,62	98,40
					2000		5.1	41,3	93,37
					1900		5.2	38,0	89,10
					1800		5.3	33,7	84,33
2	3	8	2	1000	2100	**4**	6	113.06	168,78
					2000		6.1	92,27	147,80
					1900		6.2	89,06	145,06
					1800		6.3	79,04	134,66
2	3	8	3	1010	2100	**5**	7	178,87	234,96
					2000		7.1	172,8	228,88
					1900		7.2	170.1	226,10
					1800		7.3	162.4	218,44
2	3	8	4	1010	2100	**3**	8	96,90	152,4
					2000		8.1	92,30	147,73
					1900		8.2	83,50	138,8
					1800		8.3	76,3	131,37
9	21	5	3	1015	2100	**2**	9	59,10	115,91
					2000		9.1	57,20	107,42
					1900		9.2	55,20	110,24
					1800		9.3	55,09	110,2
9	21	5	4	1015	2100	**2**	10	59,89	123,01
					2000		10.1	48,40	112,14
					1900		10.2	47,12	101,7
					1800		10.3	45,51	114,81
27	1	1	3	1015	2100	**4**	11	96.70	151.73
					2000		11.1	87.10	141.97
					1900		11.2	82.20	136.92
					1800		11.3	76.15	130.74
					1500		11.4	55.8	109.73
27	1	1	4	1015	2100	**2**	12	23.0	74.96
					2000		12.1	20.2	71.70
					1900		12.2	17.07	68.5
					1800		12.3	14.5	64.5

Tabla 13: Resultados de nivel de suciedad en marzo

Podemos apreciar que en la anterior tabla los valores han cambiado por la época del año. Estos valores han aumentado considerablemente, se decide establecer otra tabla de referencia con los parámetros que nos diferencian los niveles y dependiendo de la estación en la que estemos se seleccionaran:

Nivel	Parámetro suciedad
1	0-25
2	25-60
3	61-90
4	91-120
5	>120

Tabla 14: Distintos niveles de suciedad en verano

- Prueba para comprobar los distintos niveles en mayo.

  Los resultados obtenidos fueron los correctos ya que los niveles que nos daba el programa eran los que se veían al echar las fotografías.

  Se han conseguido recortar todas las fotografías y se ha podido distinguir los cinco niveles de suciedad sin ningún problema.

Nro. Foto	Nivel. Suciedad	Nivel.programa
1	63.18	**3**
2	26.23	**2**
3	37.19	**2**
4	58.8	**2**
5	84.39	**3**
6	110.07	**4**
7	114.08	**4**
8	54.4	**2**
9	93.0	**3**
10	81.4	**3**
11	81.6	**3**
12	101.4	**4**
13	138.9	**5**
14	172	**5**
15	102.9	**4**
16	162.01	**5**
17	129.26	**5**
18	126.20	**5**
19	41.9	**2**
20	75.2	**3**

Tabla 15: Prueba de nivel en verano

5.3 PROBLEMAS DURANTE EL DESARROLLO

Una vez comentados los resultados obtenidos es conveniente resaltar qué inconvenientes se han presentado durante el desarrollo del proceso, cómo se han intentado solucionar y finalmente se propondrán ideas de mejora que se podrían abordar en proyectos futuros.

1. El primer problema que aparece al realizar el proyecto es la fluctuación de la radiación que en pocos minutos sube y con el paso de una nube baja bruscamente y cambia las fotografías.

   Para evitar este problema se han realizado las fotos al medio día porque la radiación a partir de las doce de la mañana sube y se mantiene constante. Y en días nublados no se realizarán fotos.
2. Los valores del nivel de gris han variado en las distintas épocas del año.

   En los meses de otoño invierno los valores son inferiores a los que nos dan en primavera verano. Esto es debido a que la radiación del sol en verano es más fuerte y el reflejo que se ve en el cristal es mayor. Hablamos de dos ideas para solucionarlo:

   - Limpiar un cristal y marcarlo como nivel 1 en cada temporada, es decir, cambiar la imagen de referencia.
   - Cambiar los valores del parámetro de suciedad para cada temporada como en las tablas 9 y 14. Esta solución fue la adoptada en este proyecto.

   Comparación de dos cristales para ver el reflejo del sol incidiendo en ellos, la primera imagen es de diciembre y no se ve brillo por el sol y la segunda es de verano y si tiene brillo. Esto es lo que hace que aumente el parámetro.

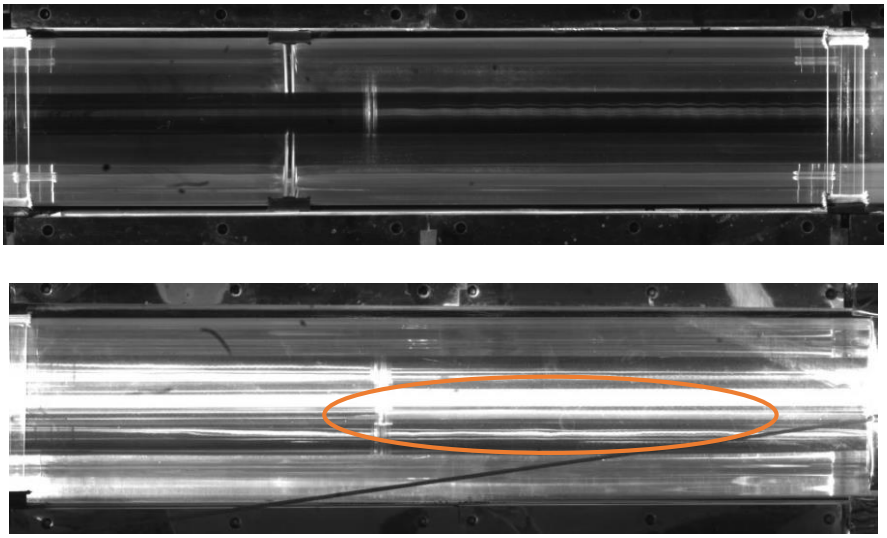


Figura 96: Comparación de cristales. Fuente: Propia.

CÁPITULO 6

Conclusión y trabajos futuros.

6.1 INTRODUCCIÓN

En este capítulo se exponen las conclusiones obtenidas tras la realización del presente proyecto. Además, se citan los posibles trabajos futuros para la mejora y ampliación del proyecto.

6.2 CONCLUSIÓN

En cuanto los objetivos marcados a lo largo de este proyecto se ha desarrollado un sistema que nos detecta el nivel de suciedad de los cristales AR de la planta termosolar haciendo uso de un sistema de visión. Los resultados obtenidos durante todo el proceso han sido satisfactorios. Cabe destacar que el programa ha ido cambiando para su mejora, de acuerdo a las pruebas realizadas, se ha obtenido un porcentaje de recorte y análisis del 100%. También se pudo distinguir los cinco niveles o grados de suciedad mediante las pruebas realizadas en las imágenes en escala de grises para determinar su textura, la textura que tiene nos determina el grado de suciedad.

Se han desarrollado diferentes algoritmos con el fin de conseguir detectar los cristales AR en diferentes filmaciones con la cámara móvil. Mediante el estudio del estado del arte, se comprobó los diferentes tipos de cámaras del mercado y sus accesorios.

Se ha integrado una cámara de visión artificial en el proceso comunicada directamente al PC. La programación se ha realizado en Matlab para su posterior paso a C, por lo que ha sido necesario mucho tiempo de documentación y muchas pruebas para que funcionara adecuadamente.

Además, se cumplieron los objetivos para la selección de la cámara, sus accesorios y la elección de los parámetros adecuados para la toma de fotografías.

Asimismo, se está comprobando que la eficiencia de la planta tiene una relación directa con el brillo que nos sale en las imágenes. En invierno la eficiencia está alrededor del veinte por ciento, sin embargo, en verano es del sesenta por ciento.

Las imágenes que se han mostrado a lo largo del proyecto nos mostraban unos valores más altos de brillo en verano, por este motivo aumentaba su parámetro de suciedad. Aunque esto no nos indica que estén más sucios, sino que aumenta el brillo debido a los reflejos del sol. Que haya mucho brillo en la imagen significa que se está reflejando más el sol y que esto se trasmite en mayor eficiencia en la creación de vapor de agua.

6.3 TRABAJOS FUTUROS

Seguidamente se presentan algunas propuestas para mejorar este proyecto. Aunque se ha diseñado un sistema automatizado que realiza la labor para la que fue pensado aún quedan cosas por mejorar.

- Crear un algoritmo que funcione con nubes. Esto mejorará la utilidad del algoritmo y se podría usar en un rango de días más amplio.
- Estudio de la relación de la eficiencia con el brillo y el nivel de suciedad para ver cómo se ven afectados unos parámetros con otros.
  Parece interesante la relación que tienen y en este proyecto solo se han dado unas breves pinceladas a este tema. Ya se mejoró el problema que surgía al cambiar de época (de estación) con la detección de los niveles de suciedad.
  Se hará un calibrado del algoritmo en las distintas estaciones del año.
- Completar y mejorar el código que recorta el cristal a partir de los agujeros que se encuentran en la parte refractaria del cristal.
- Montar la plataforma para que pueda ser usada por la empresa para hacer pruebas in situ y en tiempo real. Hasta el momento se han colocado todos los elementos en la plataforma, pero no se mueve de forma autónoma.
- Aunque el código se ha migrado a C no se ha creado una interfaz gráfica en Visual Studio para conectar la cámara, se podría completar el proyecto realizando la interfaz como se hizo en Matlab y poder hacer las pruebas en tiempo real.

CÁPITULO 7

Bibliografía.

BIBLIOGRAFÍA

[1] D. R. Mills y G. L. Morrison, «Compact linear Fresnel reflector solar thermal powerplants,

» de *Solar Energy*, 2000, pp. 68 (3), 263-283.

[2] J. Me, Z. Qiu, Q. Li y Y. Zhang, «Opthical Design of Linear Fresnel Reflector Solar concentrators.,» de *Energy Procedia*, 2012.

[3] «ohlindustrial,» [En línea]. Available: http://www.ohlindustrial.com/proyectos/planta-termosolar-30-mw-puerto-errado-2-murcia/.

[4] D. Santillan, «Visión artificial II,» *Ideas roboticas avanzadas.*

[5] R. C. Gonzalez y R. E. Woods, «Digital Image Procesing,» United States of America, Prentice Hall, 2002, pp. 3-7, 88-103, 693-730.

[6] P. Arevalo y S. Herrera, Robótica industrial prototipo y sistemas de visión artificial., Ecuador: EAE, Editorial Academia Española, 2012.

[7] J. A. S. Sánchez, Avances en robótica y visión por computador, Cuenca: Edición de la universidad de Castilla la Mancha, 2002.

[8] W. B. Rauch-Hidin, «Aplicaciones de la inteligencia artificial en la actividad empresarial,

la ciencia y la industria.,» 1989, pp. 555-600.

[9] Etitudela. [En línea]. Available: http://www.etitudela.com/celula/downloads/visionartificial.pdf.

[10] INFAIMON. [En línea]. Available: http://www.infaimon.com/es/camaras-lineales-4.

[11] «INFAIMON,» [En línea]. Available: http://www.infaimon.com/es/camaras-matriciales.

[12] FORCOM, «ASERV,» [En línea]. Available: http://aserv.udg.edu/Portal/Uploads/4103862 /CAMARAS_Infaimon.pdf.

[13] BCNVISION, «Blog de Visión Artificial,» [En línea]. Available: http://www.bcnvision.es /blog-vision-artificial/opticas-para-vision-artificial/.

[14] INFAIMON. [En línea]. Available: http://www.infaimon.com.

[15] [En línea]. Available: http://www.iberoptics.com/filtros-vision-artificial_77.

[16] «Infaimon,» [En línea]. Available: http://www.infaimon.com/es/go-5000-pge.

[17] [En línea]. Available: http://midopt.com/filters/bp470/.

[18] «jAi,» [En línea]. Available: http://www.jai.com/en/support/jai_sdk_and_control_tool.

[19] MATLAB, «MathWork,» [En línea]. Available: https://es.mathworks.com/products/ matlab.html.

[20] «Aiteco,» Artículo , [En línea]. Available: https://www.aiteco.com/que-es-un-diagrama -de-flujo/.

[21] «Matlab,» [En línea]. Available: https://es.mathworks.com/discovery/edge-detection.html.

[22] U. P. d. Madrid, «elai.upm,» [En línea]. Available: http://www.elai.upm.es/webantigua/spain/Asignaturas/MIP_VisionArtificial/ApuntesVA/ cap3ProcesadoImagv1.pdf.

[23] A. K. Jain y F. Farrokhinia, «Unsupervises Texture Segmentation Using Gabor Filters,» Pattern Recognition, 1991, pp. 1167-1189.

[24] R. M. Haralick, «Statistical and structural approaches to texture.,» de *vol 67,* 1979, pp. 786-804.

[25] K. Karu, «Is there any texture in te image?,» de *vol 29*, 1996, pp. 1437-1446.

[26] [En línea]. Available: http://iie.fing.edu.uy/investigacion/grupos/gti/timag/trabajos/2003/huellas/html/node16.html.

[27] [En línea]. Available: http://www4.ujaen.es/~satorres/practicas/practica2.pdf.

[28] Upm, «Upm,» [En línea]. Available: http://www.elai.upm.es/webantigua/spain/Asignaturas/MIP_VisionArtificial/PracticasVA/prac2VA_AdquisicionGUI.pdf.

[29] M. S. G. R. Alemán y J. A. Otero Hernández, «Guia Matlab,» [En línea]. Available: http://metodosnumericoscem.weebly.com/uploads/2/5/9/7/25971049/guia_simple_guide_matlab.pdf.

[30] C. A. Lindley, Practical image procesing in C., Canada: Wiley, 1991.

[31] A. Kaehler y G. Bradski, Learnung openCV 3. Computer vision in C++ with the opencv library, O'RELLEY, 2008.

CÁPITULO 8

Anexos.

GO-5000-PGE

5-megapixel CMOS global shutter

- *Large format 5 MP CMOS imager (global shutter)*
- *Up to 22 fps at full resolution*
- *5.0 µm square pixels*
- *Small size (29 x 29 x 41.5 mm, excluding lens mount)*
- *8/10/12-bit output in choice of monochrome or raw Bayer color models*
- *60 dB linear dynamic range with built-in HDR modes up to 100 dB (monochrome only)*
- *Analog and digital gain control for less quantized noise in low-light situations*
- *Exposure control from 10 µs to 8 seconds in 1 µs steps*
- *2X and 4X binning for increased sensitivity (monochrome only)*
- *Single and multi-ROI modes for flexible windowing and use of 2/3" or smaller optics*
- *Accepts power over GigE Vision interface or separate 6-pin connector*
- *C-mount lens mount*
- *Automatic Level Control (ALC) for dynamic lighting conditions*

Specifications for GO-5000-PGE

Go Series

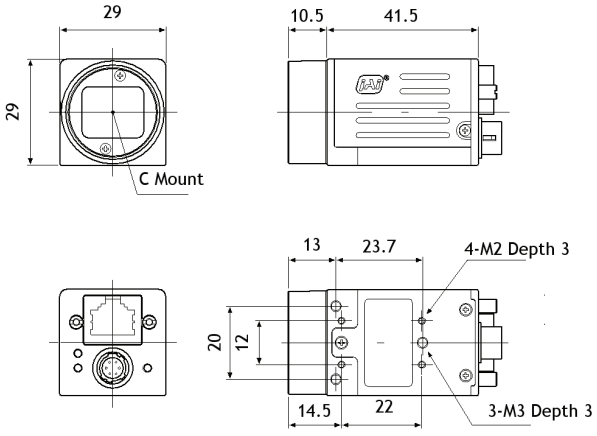
Specifications	GO-5000-PGE
Sensor	1" CMOS global shutter (Lince5M)
Pixel clock	48 MHz
Frame rate, full frame	22 frames/sec. @ 8-bit
Active area	12.8 mm (h) x 10.2 mm (v), 16.39 mm diagonal
Cell size	5.0 µm (h) x 5.0 µm (v)
Active pixels	2560 (h) x 2048 (v)
Read-out modes Full ROI (mono) ROI (color) Binning	2560 (h) x 2048 (v) up to 22 fps Any start line, any height in 1-line steps, with X offset and width in 8-pixel steps Any start line, any height in 2-line steps, with X offset and width in 8-pixel steps 1x2, 2x1, 2x2, 4x4 (monochrome only)
EMVA 1288 Parameters Absolute sensitivity (mono) Absolute sensitivity (color) Maximum SNR (mono) Maximum SNR (color)	10-bit output format 20.17 p (λ = 525 nm) 51.25 p (λ = 525 nm) 41.30 dB 38.12 dB
Traditional SNR* mono color	>55 dB (0 dB gain) >53 dB (0 dB gain, green,)
Video signal output mono color (raw)	8/10/12-bit monochrome 8/10/12-bit raw Bayer
Gain (digital)	Manual/automatic 0 dB to +24 dB
Gain (analog)	1x, 2x, 4x (mono or individual Bayer channels)
White balance (GO-5000C)	Manual, one-push auto, or continuous (3000K to 9000K)
Gamma	0.45, 0.6, 1.0 or 32-point LUT (16-pt. color)
Synchronization	Internal
Trigger input	Opto In, Pulse Generators, Software, User Output, Actions
Trigger modes	EPS, Trigger Width, Timed RCT (with ALC), Sequence
Electronic shutter Timed exposure Auto shutter	 10 µs to 8 sec in 1 µs steps 1/22 to 1/10,000 sec.
Auto Level Control (ALC)	Shutter range from 1/22 to 1/10,000, gain range from 0 dB to +24 dB Tracking speeds and max values adjustable.
High Dynamic Range function	4 built-in HDR slopes. Selectable up to ~100 dB (monochrome only).
Pre-processing functions	Blemish compensation (256 pixels)
Operating temperature	-5°C to +45°C
Storage temperature	-25°C to +60°C
Humidity	20 - 80% non-condensing
Vibration	10 G (20Hz to 200Hz XYZ)
Shock	80 G
Regulations	CE (EN61000-6-2, EN61000-6-3), FCC Part 15 class B, RoHS/WEEE
Power 6-pin connector PoE	 12V to 24V DC ± 10%. 2.5W typical @ 12V 35V to 57V DC. 3.19W typical @ 55V
Lens mount	C-mount
Dimensions (H x W x L)	29 mm x 29 mm x 41.5 mm (excl. lens mount)
Weight	46 g

Ordering Information

GO-5000M-PGE	Monochrome camera with GigE Vision
GO-5000C-PGE	Color camera with GigE Vision

*Traditional SNR is based on random noise in a single frame, where EMVA SNR measurements consider more comprehensive noise sources and variance over time. For a more complete description, see the manual.

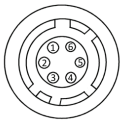
Dimensions



Outside size tolerance ± 0.3mm

Connector pin-out

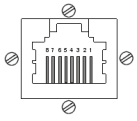
DC In / Trigger



HIROSE HR-10A-7R-6PB(73)

Pin	Signal
1	+12V to +24V DC Input
2	Opto In 1
3	Opto Out 1
4	Opto Out 2
5	Opto Common
6	GND

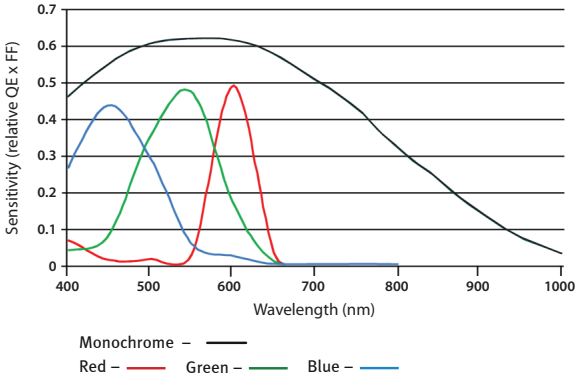
GigE Vision Interface



RJ-45 with locking screws

Pin	Signal
1	TRD+ (0)
2	TRD- (0)
3	TRD+ (1)
4	TRD+ (2)
5	TRD- (2)
6	TRD- (1)
7	TRD+ (3)
8	TRD- (3)

Spectral Response



Note: Color response includes IR-cut filter.

Europe, Middle East & Africa
Phone +45 4457 8888
Fax +45 4491 3252

Asia Pacific
Phone +81 45 440 0154
Fax +81 45 440 0166

Americas
Phone (Toll-Free) 1 800 445 5444
Phone +1 408 383 0300

Visit our web site on www.jai.com

See the possibilities

13-May-13
SPSV0132A1

Mega Pixel CCTV Lens

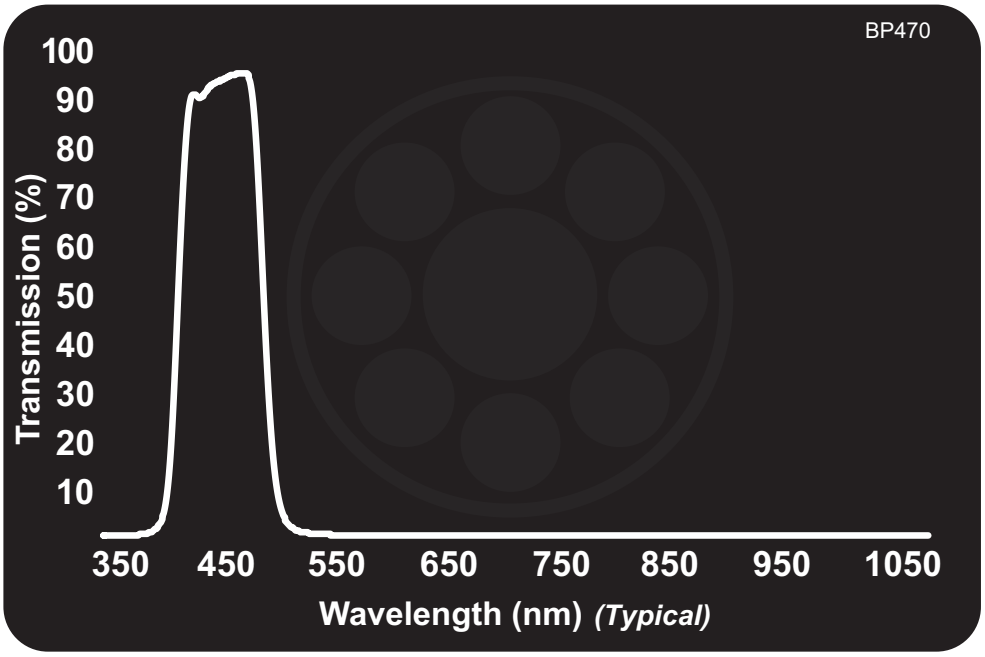
Model	VS-5018H1
Focal Length(f)	50 mm (49.93)
Maximum Aperture Ratio	1:1.8 (1.84)
F/#	1.8 ～ 16
Angle of view (1″)	11.0° X 14.6° (Diagonal:18.1°)
M.O.D	500 mm
Flange back	17.526 mm
Operation of Iris & Focus	manual
Mount	C-mount
Filter Thread	M 40.5 P= 0.5
Wavelength	Visible (400nm - 700nm)
TV distortion (1″)	-0.01 %
Sensor size (max.)	1″
Weight (approx.)	135 g
Dimension	Φ 44 (MAX) × L= 44.5 mm

VS Technology Corporation

MidOpt BP470

WWW.MIDOPT.COM
INFO@MIDOPT.COM

MIDOPT
MIDWEST OPTICAL SYSTEMS, INC.



DATA POINTS (TYPICAL)

Wavelength (nm)	Transmission (%)	Wavelength (nm)	Transmission (%)	Wavelength (nm)	Transmission (%)	Wavelength (nm)	Transmission (%)
1100	0.15	910	0.00	720	0.04	530	0.82
1090	0.12	900	-0.01	710	0.02	520	2.18
1080	0.09	890	-0.04	700	0.01	510	8.21
1070	0.05	880	-0.07	690	0.01	500	34.94
1060	0.04	870	-0.02	680	0.01	490	81.31
1050	0.02	860	0.01	670	0.01	480	94.50
1040	0.02	850	0.01	660	0.03	470	94.13
1030	0.01	840	0.01	650	0.02	460	93.10
1020	-0.01	830	0.01	640	0.01	450	91.99
1010	-0.01	820	0.02	630	0.00	440	89.39
1000	-0.01	810	0.02	620	0.00	430	87.79
990	-0.02	800	0.01	610	0.00	420	50.23
980	-0.02	790	0.02	600	0.00	410	8.94
970	-0.01	780	0.02	590	0.00	400	1.48
960	0.00	770	0.02	580	0.01	390	0.34
950	-0.03	760	0.04	570	0.05	380	0.12
940	-0.04	750	0.10	560	0.15	370	0.07
930	-0.04	740	0.10	550	0.27	360	0.05
920	0.00	730	0.06	540	0.44	350	0.06