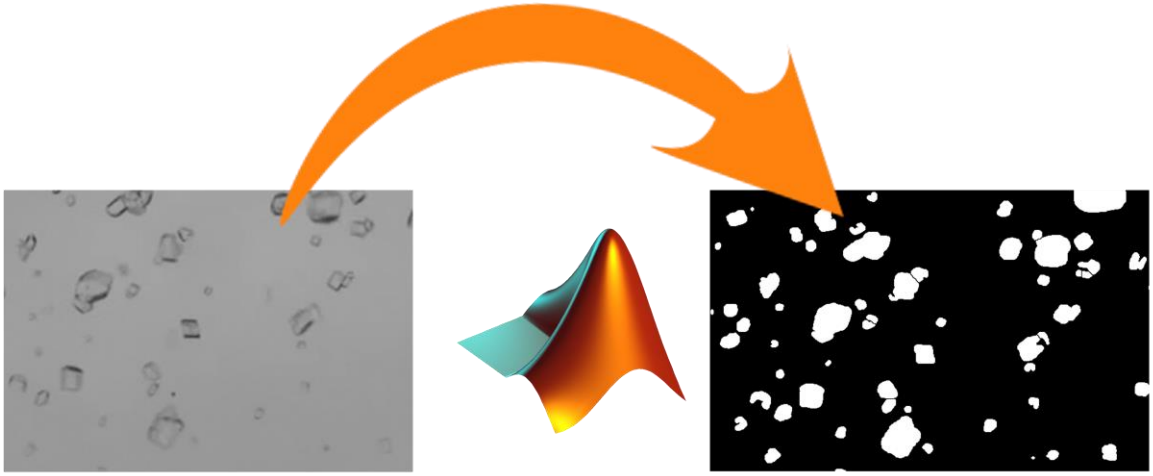
Trabajo Fin de Grado
Ingeniería de Telecomunicación



Análisis automático del proceso de cristalización del grano de sacarosa mediante procesamiento de imágenes

Autor: Lucía Rodríguez García

Tutor: Manuel Vargas Villanueva

Dpto. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Trabajo Fin de Grado
Ingeniería de Telecomunicación

Análisis automático del proceso de cristalización del grano de sacarosa mediante procesamiento de imágenes

Autor:
Lucía Rodríguez García

Tutor:
Manuel Vargas Villanueva
Profesor titular

Dpto. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2021

Trabajo Fin de Grado: Análisis automático del proceso de cristalización del grano de sacarosa mediante procesamiento de imágenes

Autor: Lucía Rodríguez García

Tutor: Manuel Vargas Villanueva

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2021

El Secretario del Tribunal

A mi familia.

A mis amigos

A mis compañeros.

A mis niños.

Agradecimientos

Es difícil reflejar sobre papel todas las gracias a todos aquellos que han hecho posible que hoy esté aquí, cerrando la que probablemente sea la etapa más importante de mi vida. Muchas han sido las dificultades que he ido atravesando con los años, pero siempre con las ideas claras y una meta fija, tal y como me han inculcado desde siempre. Por ello estoy tremendamente agradecida a todos los que habéis estado siempre ahí para ayudarme, guiarme y apoyarme en todo lo que he ido necesitando, nunca habría logrado ser quién soy hoy sin vosotros.

A todos mis amigos, y más que amigos, esa familia elegida; gracias por confiar en mí en muchos momentos que ni siquiera yo lo hacía, en los buenos y en los malos, aguantando mis quebraderos de cabeza. A todos los que han pasado para desgraciadamente no quedarse, pero de los cuales siempre se obtiene una experiencia, ya que la vida es un continuo aprendizaje. A mis compañeros de clase, que no sois pocos los que habéis conseguido que entre todos sumemos uno en tantas ocasiones, por todo lo aprendido no solo en materia académica, habéis conseguido que se haga verdad aquello de que las amistades más fuertes son las que se hacen en la universidad, siento que es y será siempre así.

Por último y no por ello menos importante, gracias a "mis niños", por todo lo que me aguantan, por ser una parte fundamental y necesaria en mi vida. Y es que todos los que me conocéis sabéis que no podía haber acabado estos agradecimientos sin hacerles mención.

Lucía Rodríguez García

Sevilla, 2021

Resumen

Se trabajo se enmarca en el ámbito de la producción de azúcar de mesa para consumo humano. En este largo proceso industrial, uno de los pasos más importantes es el de la cristalización del grano de sacarosa en el interior de los tanques denominados tachos cristalizadores al vacío [1]. Tras los pasos posteriores de centrifugado, refinado, secado y acondicionamiento, se llega a la obtención del azúcar común de mesa, tal y como habitualmente es consumida [2].

Si bien es cierto que hoy en día ya existen soluciones comerciales que permiten una monitorización automática de este proceso, son soluciones relativamente recientes y, en muchas factorías, sigue llevándose a cabo este proceso de supervisión de forma manual, bien bajo la supervisión visual humana constante a través de monitores, o bien mediante la extracción periódica de muestras del tanque.

Este trabajo viene motivado por la inquietud de estudiar la complejidad que supone la puesta en marcha de un sistema automatizado, basado en técnicas de procesamiento de imágenes, para la supervisión en línea del proceso de crecimiento de los cristales de sacarosa. Esto, a su vez, posibilita trabajos futuros que permitan la verificación de que dicho proceso se desarrolle, en todo momento, dentro de los tiempos y parámetros previstos en la fábrica.

El trabajo incluye tres partes. En primer lugar, la descripción del problema a abordar y del estado del arte, con el fin de tener una idea previa y un punto de partida para el desarrollo del software. En segundo lugar, una primera aproximación al problema, basada en la evaluación de técnicas de procesamiento de imágenes para la segmentación de imágenes de muestras secas de cristales de azúcar de mesa. En este estudio se ha hecho especial énfasis en el ensayo de diferentes técnicas de preprocesamiento, mejora y segmentación, así como de extracción de características, buscando la obtención de los mejores resultados para aplicarlos al caso real que se presenta en la industria. En tercer lugar, se aborda, efectivamente, el problema con imágenes de los granos de sacarosa en suspensión tomadas del interior del tanque, procedentes de un vídeo del proceso real de cristalización. Aquí se ha trabajado en la adaptación de los algoritmos previamente considerados o la adopción de nuevas estrategias.

En cuanto al software desarrollado, se ha empleado eminentemente *Matlab*, concretamente la versión *R2019A*. haciendo uso específicamente del *Toolbox de Procesamiento de Imágenes* [3]. Para la segmentación de los granos de sacarosa, se ha visto conveniente complementar las capacidades de dicho Toolbox con la herramienta de libre distribución *ImageJ* [4], orientada inicialmente al tratamiento de imágenes médicas, y cuyos algoritmos de segmentación arrojaban de forma consistente resultados muy superiores a los de Matlab.

Abstract

This work is part of the production of white sugar for human consumption. In this long industrial process, one of the most important steps is that of the crystallization of the sucrose grain inside the tanks called vacuum crystallization tanks [1]. After the subsequent steps of spinning, refining, drying, and conditioning, it comes to obtaining common table sugar, as it is usually consumed [2].

Although it is true that today there are already commercial solutions that allow automatic monitoring of this process, they are relatively recent solutions and, in many factories, this monitoring process continues to be carried out manually, under constant human visual supervision. through monitors, or by periodically removing samples from the tank.

This work is motivated by the concern to study the complexity of setting up an automated system, based on image processing techniques, for online monitoring of the growth process of sucrose crystals. This, in turn, enables future work that allows verification that said process is carried out, always, within the times and parameters set in the factory.

The work includes three parts. First, the description of the problem to be addressed and the state of the art, to have a preliminary idea and a starting point for the development of the software. Secondly, a first approach to the problem, based on the evaluation of image processing techniques for the segmentation of images of dry samples of table sugar crystals. In this study, special emphasis has been placed on the testing of different pre-processing, improvement, and segmentation techniques, as well as the extraction of characteristics, seeking to obtain the best results to apply them to the real case that occurs in the industry. Third, the problem is effectively addressed with images of suspended sucrose grains taken from inside the tank, from a video of the actual crystallization process. Here we have worked on adapting previously considered algorithms or adopting new strategies.

Regarding the developed software, MATLAB has been eminently used, specifically version R2019A. specifically using the Image Processing Toolbox [3]. For the segmentation of sucrose grains, it has been found convenient to complement the capabilities of said Toolbox with the free distribution tool ImageJ [4], initially aimed at the treatment of medical images, and whose segmentation algorithms consistently yielded much superior results those of MATLAB.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Notación

ºBx	Grados Brix
g	Gramos
CSD	Crystal Size Distribution
SBPP	Sugar Beet Pulp Pellet
px	Píxeles
IJ	Image J
D_{eq}	Diámetro Equivalente
D_{max}	Diámetro Máximo De Feret
<	Menor Que
>	Mayor Que
sqrt	Raíz Cuadrada
π	Número Pi
CDF	Cumulative Distribution Function
fps	Frames por segundo
Nº	Número
NaN	Not a Number

1 INTRODUCCIÓN

Este primer capítulo tendrá como objetivo dejar fijadas las ideas por las que se ha llevado a cabo el presente trabajo. Debido a que desarrolla una parte muy concreta del proceso de obtención del azúcar de mesa, se pretende aclarar los objetivos por los cuales el trabajo se concentra en el proceso de cristalización. Para ello, se va a dividir el capítulo en los siguientes tres puntos:

1) Motivación
2) Objetivos
3) Estructura

1.1 Motivación

El trabajo está enmarcado en el proceso industrial de la cristalización del azúcar que se lleva a cabo en factorías azucareras. Concretamente el trabajo viene motivado tras la visita a la azucarera vallisoletana Acor, una de las cooperativas más importantes de Castilla y León fundada en 1962 y con una gran presencia en el mercado español [5].

El proceso de extracción del grano de azúcar es un proceso industrial que consiste en diversas etapas, desde la siembra de la remolacha blanca, materia prima de la que se obtiene el azúcar, hasta que se envasa para ser consumida. En todo este proceso, se puede apreciar que los avances tecnológicos están muy presentes, como puede ser la automatización en las factorías donde se procesa la materia prima para obtener el producto final. Sin embargo, llama la atención que, durante la parte concreta de la cristalización del grano, hoy en día, se sigue controlando a través de la supervisión humana constante, lo cual es una labor altamente tediosa y monótona, lo que puede dar lugar a errores en su objetivo final. Por ello se propone realizar esta labor apoyándose en algoritmos automáticos de procesamiento de imágenes y extracción de datos.

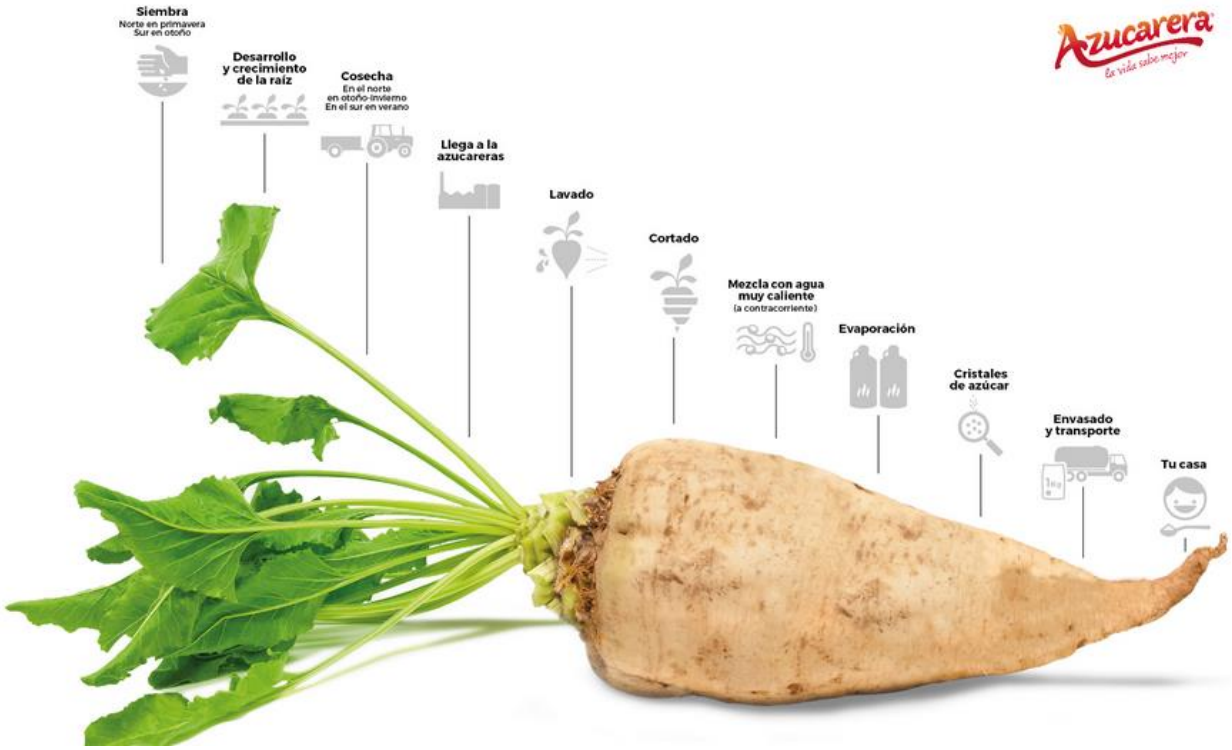


Ilustración 1-1 Etapas del proceso de extracción del azúcar. [24]

Por este motivo, surge la necesidad del desarrollo del presente trabajo, centrado en el proceso de cristalización del azúcar, de forma que se procede a desarrollar un software que consiga monitorizar el proceso de cristalización del grano de sacarosa a través del procesamiento de las imágenes captadas en vídeo directamente del tacho cristalizador. De estas imágenes se espera poder ir viendo el crecimiento progresivo del grano gracias a la acción de la temperatura y presión dentro del tacho, produciéndose así un proceso químico por el cual el grano se cristaliza y entonces irá adquiriendo el calibre necesario para poder pasar a la siguiente fase.

1.2 Objetivos

A pesar de que ya hay algunas opciones comerciales que pretenden solucionar este problema, una gran parte de las factorías azucareras continúan haciendo la monitorización del crecimiento del grano de forma manual con el procedimiento descrito de observación.

Debido a este hecho, el objetivo fundamental de este trabajo consiste en facilitar el proceso de monitorización del crecimiento del grano de sacarosa en la etapa de la cristalización. Gracias a la implantación en esta etapa de un algoritmo que indique el calibre del grano, se podrá tener una idea más exacta de su tamaño en promedio a lo largo del proceso, haciendo más exacto el instante en el que se para el proceso por ya estar formado el cristal.

La mejora que propone este algoritmo es presentar los datos sobre el tamaño teniendo en cuenta su promedio, ya que el vídeo va mostrando una gran cantidad de granos, los cuales, a través de algoritmos de procesamiento de imágenes y reconocimiento de regiones, será capaz de calcular sus parámetros de interés de manera conjunta. Hasta ahora, estas mediciones se realizaban sobre uno o dos granos, superponiendo elementos de medida rudimentarios, como son reglas con poca precisión, sobre el monitor donde se muestran de manera escalada los granos durante el proceso de cristalización.

En definitiva, este trabajo podría implicar una mejora en el proceso de la elección del instante en el que se debe parar el proceso, para así mejorar el calibre del producto final y por consiguiente la calidad de este.

1.3 Estructura

El trabajo se organiza de la siguiente forma:

Una primera parte en la que se introducirá el contexto en el que se desarrolla el proceso industrial por el cual se obtiene el azúcar apto para el consumo humano. Esto es, introducir algunos conceptos necesarios para la mejor comprensión del proceso que incumbe al trabajo y para localizar dentro del mismo la etapa concreta de estudio. Otro aspecto interesante sobre el que indagar, es el de los parámetros usados en el ámbito industrial, para así tener un conocimiento más amplio sobre los datos que se deseará extraer del proceso. Por último, se comentará cómo algunos autores han trabajado sobre aspectos similares a través de la predicción y monitorización en lugar del empleo de la visión artificial, camino seguido por el presente trabajo.

A continuación, en la siguiente sección se tratará de desarrollar las técnicas de visión artificial aplicadas al problema que incumbe, así como comentar las estrategias seguidas por algunos autores para establecer un punto de partida. En adición, se exponen los retos con los que se ha tenido que lidiar y las soluciones adoptadas para el desarrollo del trabajo. A su vez se expondrán los parámetros extraídos de los propios granos para sacar las conclusiones posteriores necesarias y algunas de las estrategias seguidas por diversos autores.

Una vez establecido el contexto y los objetivos sobre los que se desea investigar, en la siguiente sección se lleva a cabo un estudio para la prueba de estas estrategias seguidas sobre imágenes obtenidas por cuenta propia, con los cristales de azúcar ya formados para hacer pruebas de algoritmos con la intención de encontrar la técnica adecuada para obtener los resultados deseados.

Tras hacer pruebas con estas imágenes, se extrapola el algoritmo elegido como óptimo a las imágenes obtenidas de un vídeo real del proceso de cristalización. El vídeo se encuentra disponible en el siguiente enlace, perteneciente a la empresa francesa ITECA, que entre otras funciones tiene la de la producción del azúcar de mesa [6].

El siguiente capítulo consiste en analizar los datos obtenidos de la aplicación de los algoritmos en el tiempo y de la extracción de información de estos sobre las imágenes reales.

Para finalizar, en la última sección del trabajo se trata de comentar las conclusiones a las que se llega tras el análisis de los datos los trabajos futuros pendientes de mejorar, ya que los resultados obtenidos no siempre son los deseados.

2 CONTEXTO DEL TRABAJO EN EL MARCO DE LA INDUSTRIA AZUCARERA

En este apartado del trabajo se va a tratar de poner en contexto tanto el objetivo principal del problema que se abarca como algunos aspectos básicos de las herramientas usadas para abordarlo. Para ello, se comienza por el principio básico, en qué consiste y cómo se consigue cristalizar el grano de sacarosa para llegar al resultado del grano de azúcar de mesa.

2.1 Descripción general del proceso industrial

Con el fin de comprender mejor esta etapa concreta en la que se centra el trabajo, se procede a hacer un esbozo del proceso completo de la obtención del azúcar hasta dicho punto. Para ello, la información a continuación se extrae de la propia página web de la Cooperativa Acor, donde viene con un mayor grado de detalle cada una de las partes por las que pasa [7].

Como bien es sabido, la materia prima del azúcar blanco habitualmente consumida en España y en la mayoría de los países, es conocida por el nombre de "Remolacha Blanca Azucarera" (*Beta Vulgaris Var. Saccharífera*). Se trata de una planta dentro de la familia de las quenopodiáceas, cuyo cultivo es bianual. Sin embargo, para su uso principal de la producción de azúcar, se recolecta en su primer año, cuando en su raíz acumula la mayor cantidad de sacarosa del ciclo. Su segundo año se reserva para la generación de semillas para sus reproducciones futuras [8].

Ilustración 2-1 Remolacha azucarera en primer año de cultivo [9].

Ilustración 2-2 Remolacha azucarera en segundo año de cultivo [9].

Una vez introducida la materia prima, se comienza a enumerar las etapas previas a la cristalización del grano.

2.1.1 Recepción y tratamiento de la remolacha

En esta etapa se recibe en la fábrica la remolacha recolectada y se mide su grado de impurezas para determinar si cumplen los requisitos de calidad establecidos. Posteriormente, se almacenan a la espera de ser procesadas. Para asegurarse de que no se deterioran, se les aplica aire ambiente dependiente de la temperatura exterior.

Con el objetivo de ser transportadas desde los silos de almacenaje, se traslada en canales con agua hasta la fábrica para desterrarlas, lavarlas y extraerles piedras si las hubiera gracias a sistemas implementados para ello en los propios canales. De nuevo, para asegurar unos buenos resultados, se vuelven a lavar, esta vez en centrifugadoras con agua.

El último paso de esta etapa consiste en cortar las raíces y rabillos, de esta manera, el restante (el cuerpo de la remolacha como tal) es lo que se trocea en tiras o cosetas, teniendo como característica principal una gran superficie, necesaria para etapas posteriores. Así pues, la remolacha azucarera se encuentra en las condiciones deseadas para comenzar a ser tratada.

2.1.2 Difusión

En este proceso se extrae de la coseta el jugo de difusión (conteniendo toda la sacarosa que llega a formar el jugo) y se separa la pulpa (materia restante de la coseta). Se consigue dicho resultado gracias a la utilización de agua caliente para poder extraer de la coseta el jugo de interés y haciéndola pasar por un sistema con el agua a contracorriente.

2.1.3 Prensado, secado y granulado de la pulpa

Lo primero que se lleva a cabo es la recuperación de la pulpa previamente obtenida de la difusión. Esta pulpa se introduce en un sistema de prensado para que expulse el agua que le pueda quedar. Dicha agua, que aún contendrá sacarosa, será introducida de nuevo en los difusores con la misma finalidad.

La pulpa, ya prensada, procede a ser secada a través de un horno, donde va rotando y perdiendo su humedad debido a la temperatura que alcanza. Para concluir, la pulpa seca se trata para obtener pequeños gránulos denominados *pellets*, los cuales se comercializarán como combustible biomasa bajo las siglas de SBPP. De esta manera, se reduce la generación de residuos en el proceso y podrá sacarse beneficio extra con la misma cantidad de materia prima [10].

2.1.4 Depuración del jugo

Del jugo obtenido en la etapa de difusión, se deben separar las partículas en suspensión que se encuentran en él para su descarte, equilibrar su pH (ya que se trata de un ácido que podría llegar a destruir la sacarosa que contiene), extraer los azúcares (el objetivo principal del proceso) y eliminar los coloides (sustancias con facilidad para adherirse a otras). Para lograr esto, se procede a seguir los siguientes pasos:

2.1.4.1 Preencalado

Se añade una lechada de cal para equilibrar el pH mediante la alcalinización.

2.1.4.2 Encalado

Se trata de una segunda lechada de cal para que, mediante un proceso químico, actúe sobre los componentes no azucarados y estabilice el jugo. De este modo, pasará a no ser letal para los azúcares.

2.1.4.3 Primera Carbonatación

Se le añade por borboteo al jugo encalado un gas procedente de un horno de cal con anhídrido carbónico, de esta manera, se separa la sacarosa contenida en el sacarato cálcico, extraído del proceso. Como junto a este sacarato cálcico se tienen otras partículas sin interés, para deshacerse de ellas, se pasa dicho compuesto por una serie de filtros o decantadores.

2.1.4.4 Segunda Carbonatación

Esta se hace de forma análoga a la primera, teniendo como resultado la reducción de la alcalinidad del compuesto. A continuación, se filtra y se sulfata con motivo de reducir su viscosidad y coloración.

2.1.5 Evaporación del jugo

Este proceso se lleva a cabo debido a la gran proporción de agua en el jugo ya depurado y la poca de materia seca (relacionado con el concepto grados Brix, que se definirá más adelante), la cual es el interés final. Se lleva a cabo mediante el aumento de temperatura del jugo, siempre por debajo de 100ºC. De esta forma, la presión en el interior irá cambiando progresivamente, evaporando el agua del jugo y por tanto concentrándose.

El resultado obtenido se le denomina jarabe, una sustancia en la que se encuentran disueltos azúcares. Este jarabe pasa por último por un proceso de filtración para impurezas que pueda haber y se encuentra en condiciones de entrar en la fase de cristalización.

2.1.6 Cristalización

Este es el punto de partida del presente trabajo, la etapa en la que a partir de la sustancia denominada jarabe, la cual contiene los azúcares de la remolacha azucarera, se conseguirán formar los cristales, conocidos como azúcar de mesa.

Este proceso es complejo debido a que la intención principal, tanto de la Cooperativa Acor (empresa en la que se ha fundamentado el trabajo) como otras empresas, es la de ofrecer un producto de alta calidad. Para ello, se aplican tres fases de cristalización al jarabe, tanto para aumentar su calidad como para aprovechar todas las cualidades presentes en las mieles (soluciones sobresaturadas de azúcares).

La presente etapa se lleva a cabo fundamentalmente dentro de los tanques destinados a la cristalización, como se ha comentado antes, los tachos o tachas al vacío. Se trabaja bajo la condición del vacío debido a que así se evita la descomposición térmica del azúcar[1] y el incremento de color [11].

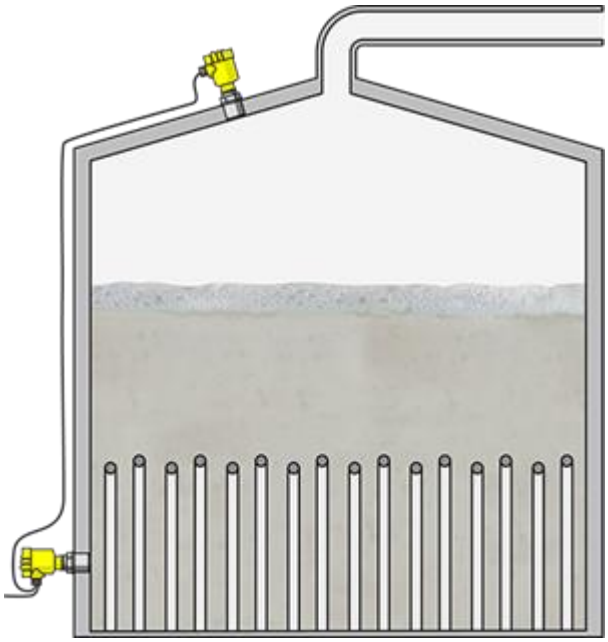


Ilustración 2-3 Tacho al vacío para el proceso de la cristalización del azúcar [25].

A continuación, se procede a describir la etapa de cristalización de forma detallada enumerando a cada uno de los pasos que se siguen. Con el objeto de una mejor comprensión del proceso que puede resultar algo complejo debido a la retroalimentación de algunas etapas con la miel restante, se adjuntan la siguientes imágenes esquemáticas publicadas en el vídeo anteriormente citado, de donde se ha obtenido gran parte de la información de esta sección [11].

[1] Gracias a que trabajar dentro de los tachos permite una cristalización a temperaturas más bajas, el azúcar no corre el riesgo de descomponerse una vez cristalizado por estar sometido a altas temperaturas.

Ilustración 2-5 Esquema del proceso de cristalización del azúcar por Acor. Parte 1ª.

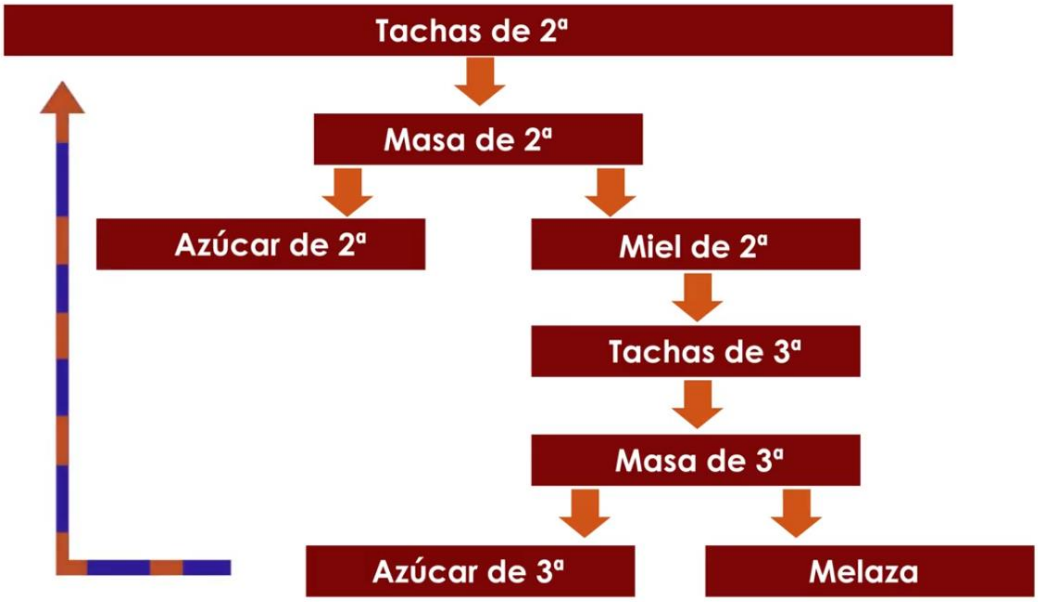


Ilustración 2-4 Esquema del proceso de cristalización del azúcar por Acor. Parte 2ª.

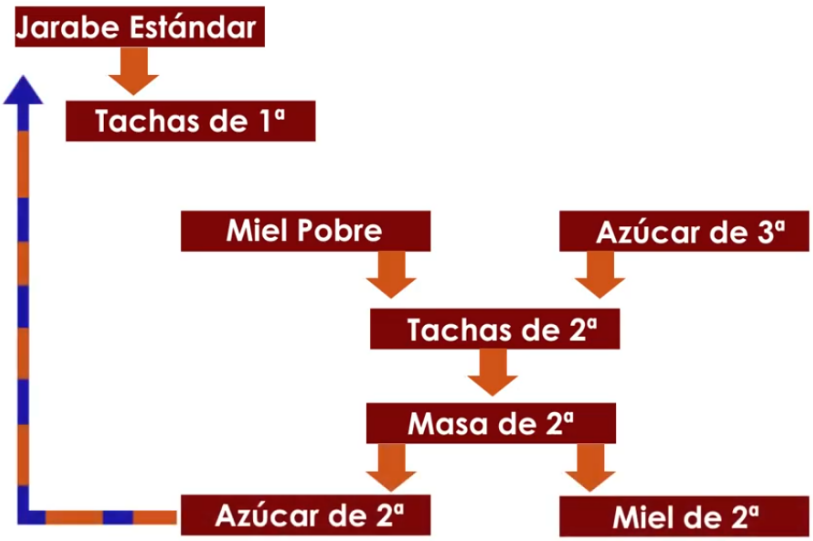


Ilustración 2-6 Esquema del proceso de cristalización del azúcar por Acor. Parte 3ª.

2.1.6.1 Primera Cristalización

Los tachos al vacío funcionan de forma discontinua. Esto significa que, en un primer momento, se introducen en ellos el jarabe en el fondo del tanque, evaporándolo hasta conseguir una sobresaturación deseada. A continuación, se introducen cristales de azúcar de pequeño tamaño (150 micras de diámetro)[2], esto hará que el jarabe introducido al principio se cristalice sobre la superficie de los pequeños cristales usados como semillas, haciendo que, a través de la temperatura y la presión, su tamaño aumente progresivamente al ir disminuyendo la cantidad de líquido en la disolución. A la vez, se le sigue añadiendo jarabe, para que este proceso no cese, pero controlando la sobresaturación en todo momento.

De este proceso se obtienen unos primeros cristales y una solución que no ha cristalizado en el fondo del tacho, respectivamente la masa cocida de primera y la miel madre.

2.1.6.2 Primera Centrifugación

Con el objeto de separar ambas partes, se someten a la centrifugación, que se hará en dos fases. Esta parte del proceso se lleva a cabo en los conocidos como centrífugas. Son elementos presentes en la industria que consisten en que haciendo girar la miel conteniendo los cristales de forma que, debido a la fuerza centrífuga, el contenido se desplaza hacia las paredes, donde habrá una malla metálica que deje pasar la solución de la miel madre y no filtre los cristales. A la vez que se filtra, se les aplica agua a los cristales de azúcar para eliminar cualquier rastro de miel en su superficie. A continuación, se le aplica vapor para eliminarla.

[2] Denominada "siembra". Se obtiene a través de una cristalización fría en laboratorio a partir de granos de azúcar de diámetro 10 micras.

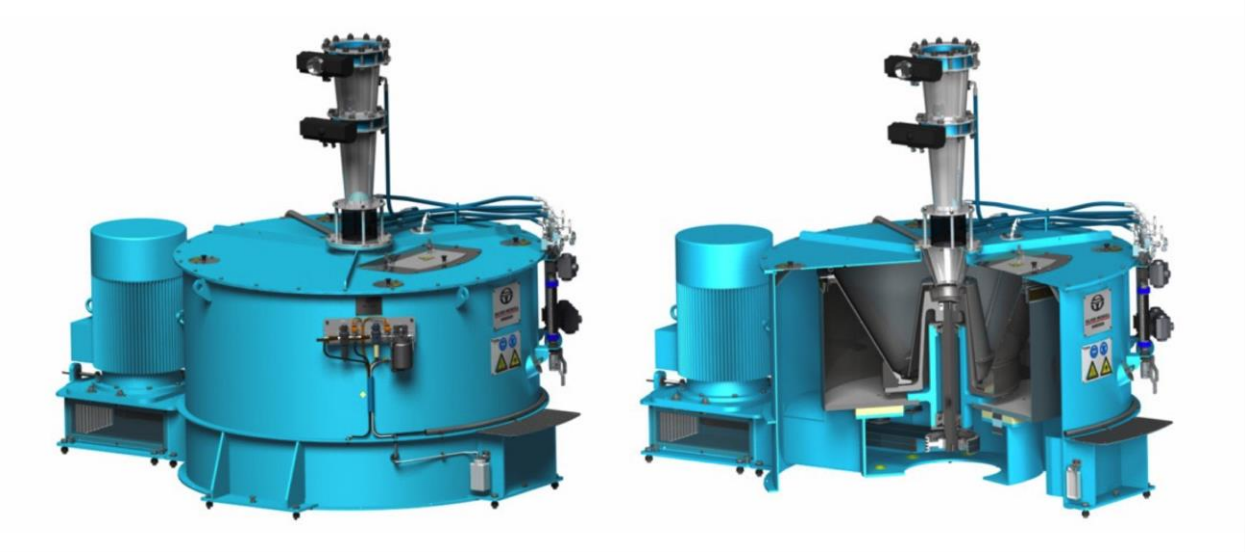


Ilustración 2-7 Centrífuga de azúcar [26].

Como en procesos anteriores, se obtienen por separado dos sustancias, los cristales y la miel. De esta primera centrifugación se obtiene la denominada "azúcar blanquilla", que será secada a través de aire frío y caliente, posteriormente se le aplicará una criba para asegurar su calidad. Por otro lado, se tiene la miel que ha sido filtrada por la malla metálica de la centrifugación, la cual será sometida a una segunda cristalización.

2.1.6.3 **Segunda Cristalización**

Se lleva a cabo en tachos de segunda, en los cuales se introduce la miel y cristales ya formados de un tamaño aproximado de 150 micras de diámetro como en la primera cristalización. Análogamente, a través del proceso de cristalización, estos granos van engordando hasta conseguir el tamaño requerido.

De igual manera, se obtiene una masa de segunda cocción, formada por azúcar de segunda cocción y miel de segunda cocción.

2.1.6.4 **Segunda Centrifugación**

Se aplica, tal y como se ha citado, una segunda centrifugación con el mismo objeto que la primera, ser capaces de separar el azúcar de la miel. En esta centrifugación, ambos elementos siguen caminos diferentes a los de la primera, pues el azúcar de segunda cocción es mezclado en jarabe, el que se va introduciendo en las tachas de primera, mientras que la miel que se filtra en la centrifugadora pasa a una tercera etapa de cristalización para seguir sacándole rendimiento.

2.1.6.5 **Tercera Cristalización**

Se realiza de la misma forma, alimentándose en un principio de miel pobre para llenar el fondo del tacho y posteriormente se le añade la miel de segunda cocción. Se realiza su cocción hasta obtener una masa cristalizada de tercera cocción.

2.1.6.6 **Tercera Centrifugación**

De este proceso de centrifugado se obtienen dos productos, una solución denominada "melaza", la cual no tiene la capacidad de ser cristalizable, y el azúcar de tercera cocción, la cual se mezcla con miel de segunda cocción para realizarse un lavado de sus cristales.

2.1.7 Finalización del proceso

Tras este largo y complejo proceso de cristalizaciones y centrifugaciones consecutivas, se obtienen los cristales de sacarosa, popularmente conocidos como granos de azúcar de mesa. Se trata de azúcar blanquilla y su tamaño deseado para su comercialización oscila entre los 0,4mm y los 0,6mm de diámetro. Se consiguen gracias a los pasos seguidos, en concreto por los de evaporación del jugo y su posterior enfriamiento.

Cabe destacar, que de este proceso al igual que se obtiene el producto final comercializable y apto para el consumo humano, también se obtiene la melaza, que no es más que las soluciones que no han cristalizado. Esta melaza es un subproducto y por tanto se comercializa con el objeto de servir para la preparación de alimentos para el ganado en algunos casos [12]. En otros, se fermenta para producir alcohol y levaduras [13].

2.2 Parámetros más relevantes

En este apartado se va a realizar una abstracción de los términos más puramente químicos o que no incumban en el ámbito en el que se enmarca el trabajo. Esto quiere decir que los parámetros que se tendrán en cuenta serán los que afecten de una manera directa en la monitorización, descartándose aquellos relacionados con cuestiones químicas como pueden ser la aplicación de compuestos en disoluciones.

Estos parámetros principales son:

2.2.1 Temperatura

La cristalización es un proceso que se lleva a cabo en soluciones con gran concentración de azúcar en ellas. A través del aumento de temperatura dentro de los tanques denominados tachos, este proceso podrá darse a temperaturas no tan altas como serían necesarias bajo otras condiciones, en diferentes recipientes.

En los tachos al vacío, en concreto en la factoría Acor que ha compartido sus datos, la temperatura a la que funcionan los tachos en su interior durante las diferentes etapas está comprendida en el rango de 70-80ºC. Esta es, sin lugar a duda, una de las cualidades más destacadas del uso de tachos al vacío, la capacidad de poder evaporar la miel de su interior para así cristalizarse a bajas temperaturas, ya que no es la temperatura la encargada directa de la cristalización, sino la saturación de la disolución.

El objetivo principal de la cristalización en frío, como así se llama a esta técnica, es la baja proporción de aglomeraciones del cristal durante su proceso como así lo demuestra un artículo publicado por diversos autores como Faria [14]. En él se puede observar a través de diferentes gráficas y en diversas circunstancias, como al aumentar la temperatura a la que se trabaja, a medida que se cristaliza más masa, crece de forma exponencial el grado de aglomeración, siendo el uso más adecuado en torno a 25ºC.

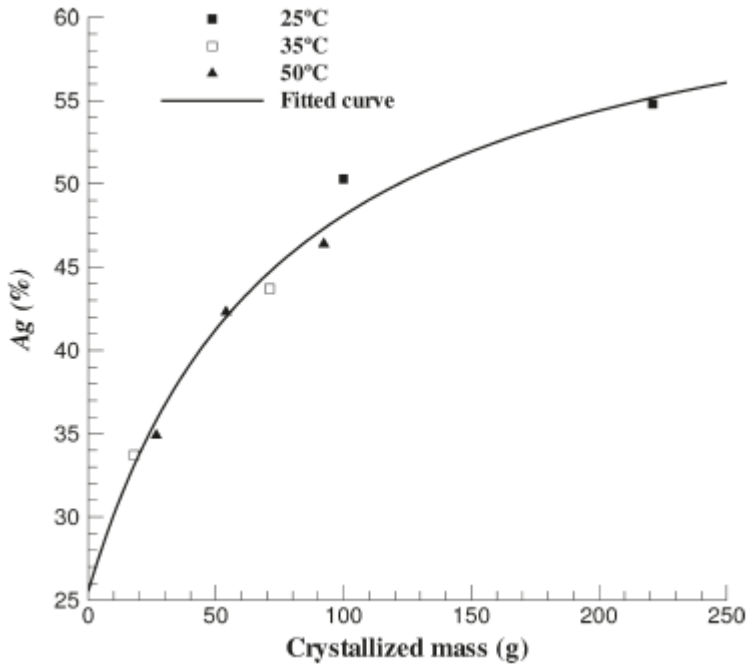


Ilustración 2-8 Grado de aglomeración según masa y temperatura [14].

2.2.2 Diámetro

También denominado calibre del grano es un factor muy importante en la cristalización del grano, ya que será el parámetro que indique la finalización del proceso al alcanzar el valor deseado. Como se ha visto en la sección anterior, en el proceso de cristalización se trabaja con diferentes magnitudes de este parámetro.

En las etapas de cristalización se aprecia que se comienza el proceso mediante a adición de granos ya cristalizados previamente en un laboratorio[3]. Estos granos son los que se han denominado anteriormente como siembra, sirven de base para la cristalización. El proceso que se desarrolla con estos cristales es el de la cristalización sobre ellos, ya que a partir de un diámetro de 150 micras (siembra inicial), se llega a los diámetros deseados en la industria azucarera, entre 0,4mm-0,6mm.

Una vez que se alcanza el diámetro deseado, se actúa sobre la masa variando su presión y, por ende, variando así su sobresaturación, así se detendrá el proceso de cristalización y dejará de aumentar el tamaño de los granos.

Por este hecho, será el diámetro el factor que será continuamente estudiado en el trabajo, es lo que se monitoriza constantemente. No se hará siempre directamente con el diámetro del grano, ya que puede dar lugar a errores, sino que se estudiarán características derivadas del mismo que arrojen mayor grado de información acerca de su tamaño y la evolución de este.

2.2.3 Brix

Se trata de una medida generalmente usada en la industria azucarera, pero también conocida, por ejemplo, en la viticultura o la industria frutícola. Con este parámetro se mide la cantidad de masa seca contenida en una solución líquida. En el caso que incumbe al trabajo, la medición será la de sacarosa presente en el líquido.

Se representa en grados Brix (ºBx), midiendo la cantidad de gramos de masa seca en 100 gramos de solución líquida [15]. De esta manera se dirá que 30ºBx se corresponde a que en una solución habrá 30g de sacarosa por cada 100g de solución (jarabe).

[3] Cristales obtenidos a través de una cristalización en frío con una suspensión de azúcar de unas 10 micras.

Esta magnitud será de especial importancia para tener en cuenta durante el desarrollo de la monitorización, ya que a medida que la cristalización avanza, se espera que aumente la cantidad de materia seca en la solución. Esto afectará de forma negativa en la aplicación de técnicas de visión artificial en las imágenes obtenidas dado que se hará cada vez más complejo poder identificar un grano de otro (separación de regiones), bien por su proximidad o bien por el fenómeno conocido por aglomeración[4]. Por ello, cabe esperar que los algoritmos aplicados tengan mayor robustez en sus inicios que cuando el proceso alcanza su fin.

2.2.4 Presión

Como ya se ha citado en apartados anteriores, la presión es la que regula de forma indirecta la cristalización del grano de sacarosa. De forma indirecta debido a que su variación actúa en el interior del tacho regulando la sobresaturación.

2.3 Predicción y monitorización del grano de sacarosa

En esta sección se describirán las técnicas usadas ajenas a la visión artificial para saber en qué momento se ha de parar el proceso de cristalización al haber crecido lo suficiente los granos de sacarosa. Se destacan dos técnicas dentro de esta modalidad.

La primera consiste en la llevada a cabo en la propia factoría Acor, la cual depende de la inspección, continuada y monótona de un trabajador, a partir de las imágenes mostradas en tiempo en un monitor. Para ello, este monitor consta de una rejilla superpuesta en su pantalla, la estrategia trata de ir comparando el tamaño del grano que se muestra a través de un vídeo, tomado del interior del tanque, con el espacio de separación de las rejillas, hecho a medida para que cuando el grano tenga las dimensiones de dicha rejilla en su mayoría, indique que se ha obtenido el tamaño deseado y por ello se dé por finalizada la etapa de cristalización en la que se encuentre.

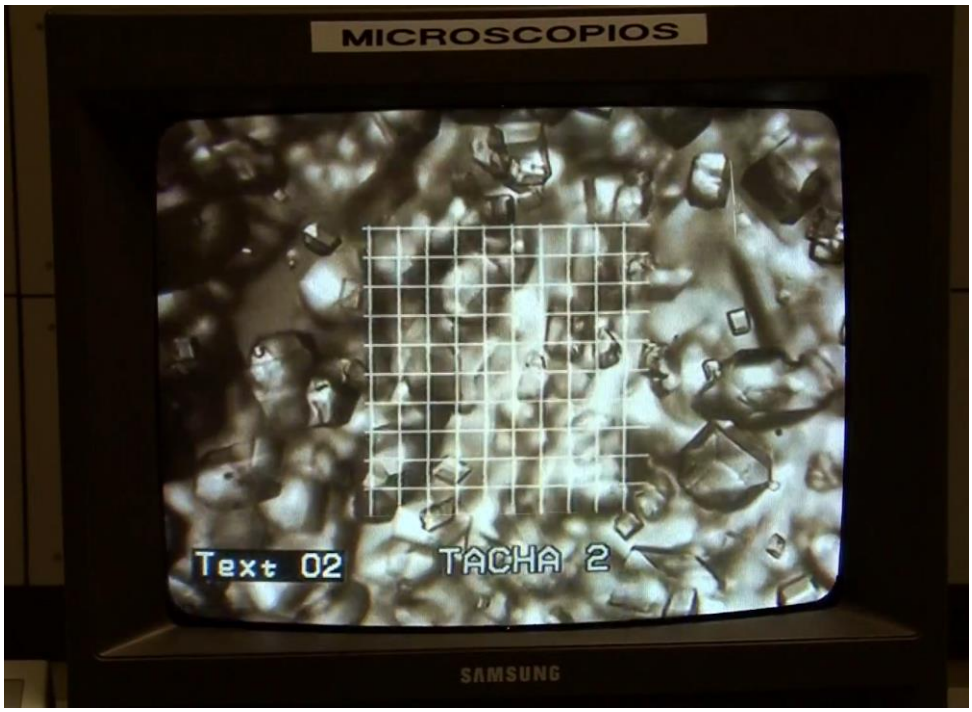


Ilustración 2-9 Monitor con rejilla para medir el grano de sacarosa (1º) [11]

[4] Este tema será tratado de forma mucho más extensa en una sección próxima.

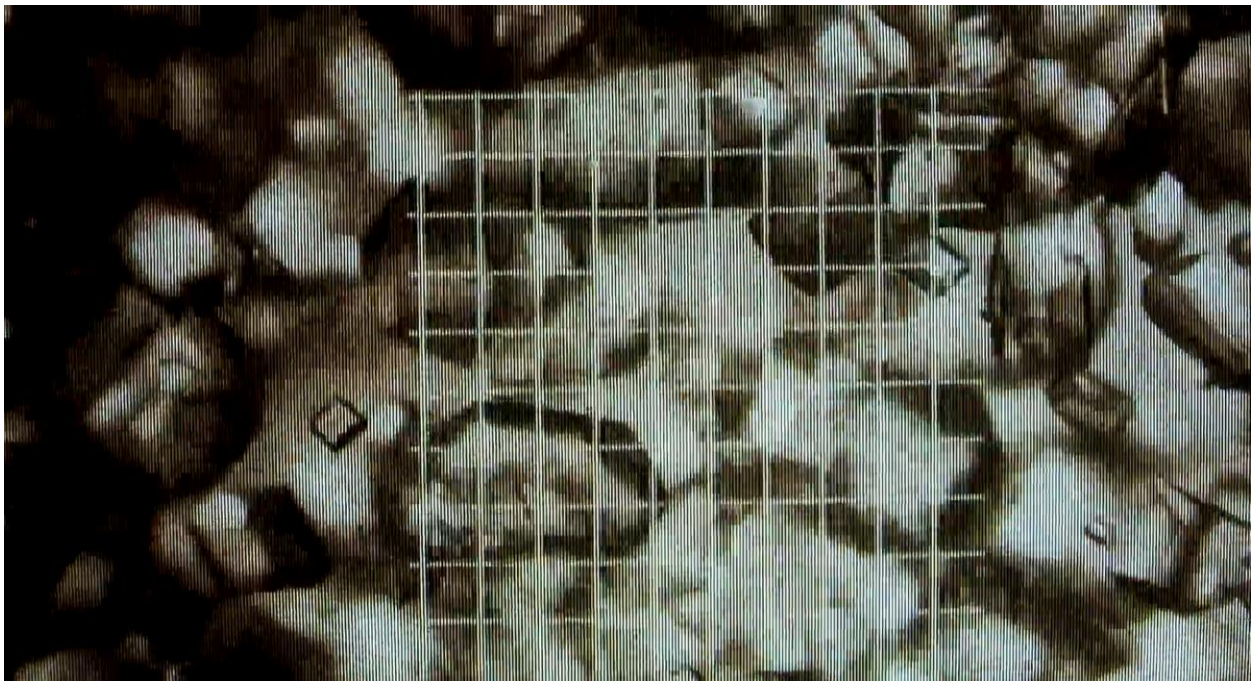


Ilustración 2-10 Monitor con rejilla para medir el grano de sacarosa (2º) [11]

La segunda técnica usada son las basadas en modelos del comportamiento dinámico del proceso de cristalización, tal y como se describen en los artículos de Martins [16] y Nagy [17]. Estos artículos tratan de cómo aplicar un modelo matemático basado en balance de masa para que, mientras la cristalización está teniendo lugar, y a través de parámetros relacionados con el contenido del cristal y la pureza del líquido, determinar la evolución que tendrán estos cristales.

Las características fundamentales en este tipo de sistemas en los que se basa esta monitorización son la distribución del tamaño del cristal (Crystal Size Distribution o CSD), la forma que este obtiene y su pureza.

Sin embargo, modelos como los que propone Martins, basa su control de la cristalización en la pureza de la mezcla, ya que a medida que avanza la cristalización, la concentración de sacarosa decrece y por ello la mezcla en conjunto pasa a ser menos pura. Para esta medida, se usan lecturas a través de microondas, capaces de determinar el porcentaje de impurezas en la mezcla.

2.4 Monitorización a través de la visión artificial

Este es el punto de partida del presente trabajo, y estas son las técnicas en las que se va a basar su desarrollo. Cabe destacar la importancia de los avances en el campo de la visión artificial y la automatización gracias a los cuales este tipo de estudios son posibles, pues se requiere para ello tanto potentes equipos para la captación y procesado de las imágenes con las que se trabaja como para trabajar con los conceptos estadísticos involucrados en el proceso.

Es por este motivo por lo que los grandes avances en el campo de la monitorización, ya no solo en el ámbito de la industria azucarera, si no de la industria en general, se han dado en las dos últimas décadas. Los artículos en los que se ha basado y que sirven como pilar para las ideas de este trabajo, quedan comprendidos en los últimos veinte años, periodo donde se ha visto incrementado el uso de la visión artificial, en parte por los avances técnicos y por consiguiente la mejora de la velocidad de procesado de las computadoras, y en parte por los nuevos sistemas fotográficos, de gran precisión y rapidez, disponibles en el mercado.

Para empezar, se va a detallar el proceso de obtención de las imágenes con las que se trabajará en el proceso. Así pues, dichas imágenes provienen de una cámara microscópica y una lente de gran aumento que se encuentran ubicadas en lo que se denomina «celda de flujo».

Este concepto se define como una parte del tacho por la cual el azúcar pasará en suspensión.

Es importante destacar, que va a tener poca profundidad ya que la finalidad principal es que no se produzcan demasiadas aglomeraciones de granos (problema de la superposición de estructuras que se comentará más adelante).

Por otro lado, hay que comentar que la cámara se sitúa en un orificio de apenas 5mm de lado. Sin embargo, este factor no es un inconveniente para su resolución (captada a través de un sensor tipo CCD) que arroja una calidad aproximada de 1024x1024 píxeles.

Siguiendo esta línea, la adquisición de la imagen se hará a través de un montaje del sistema adecuado, el cual está formado por tres elementos que se indican a continuación:

1) El sistema de captación (la cámara microscópica ya descrita)
2) El tubo transparente (la celda de flujo)
3) Un sistema de retroiluminación (consistente en una fuente de luz situada detrás del tubo y un disipador de luz o expansor de haz).

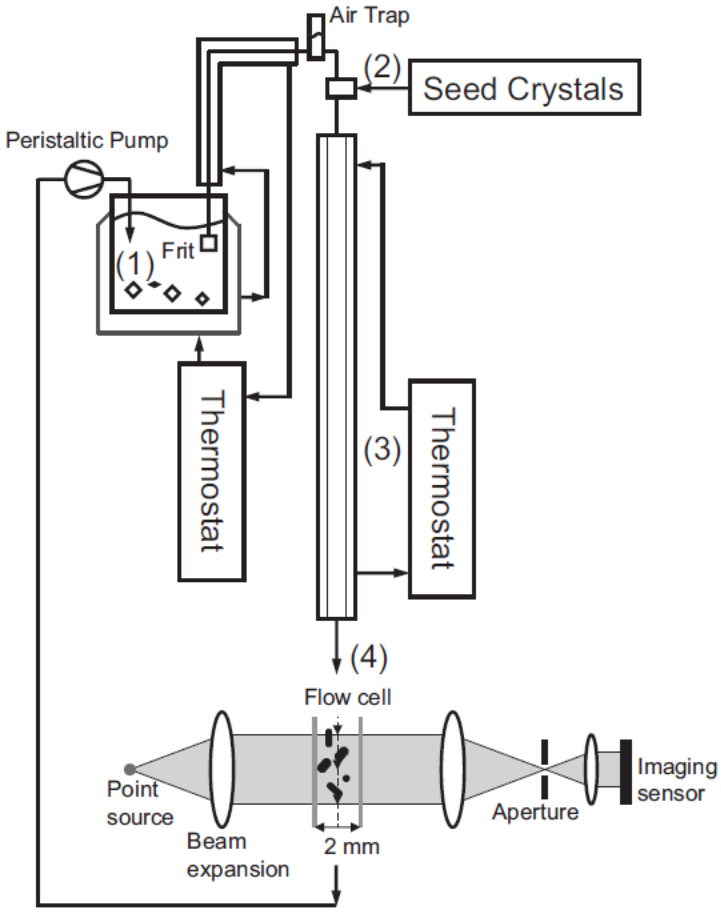


Ilustración 2-11 Esquema de adquisición de las imágenes de partida [18].

La utilidad de esta última parte se centra en hacer que la luz que ilumina por la parte trasera sea lo más difusa posible, de forma que se produzca el mínimo número de sombras en los granos y el fondo de la imagen se vea de color blanco [18].

A pesar de estos intentos, en el desarrollo de las soluciones del trabajo se va a producir de forma inevitable una complicación que añade este sistema de iluminación: el efecto de oscurecimiento de los bordes de la imagen debido a que se ilumina desde su punto central.

Por otra parte, dentro de las técnicas de visión artificial empleadas en el sector de la industria azucarera, se pueden distinguir tres grandes familias fundamentalmente. Cada una de ellas se encuentra basada en un principio de obtención de los datos de interés, teniendo en común que todas parten del montaje descrito anteriormente. Se describen brevemente como:

2.4.1 Técnicas basadas en la sustracción del fondo

El fundamento de esta familia de técnicas se basa en la suposición acertada en la mayoría de los fotogramas, de que el fondo de la imagen se mantiene constante y que lo que varía de un fotograma a otro son los granos de azúcar que se desplazan suspendidos en la mezcla. De esta forma, se obtiene el histograma aproximado a través de la captación de una imagen del fondo sin granos, para restarlo en cada fotograma a la imagen completa, quedando así nada más en la imagen que los elementos de interés, los granos de azúcar.

De esta forma, los cambios de un fotograma al siguiente del vídeo serán debidos a los cambios que se quieren estudiar. Para ello cuenta con algoritmo de reconocimiento de regiones que identifiquen la aparición de nuevos cristales en la imagen, es decir, capaz de identificar cambio en la morfología [17]. Una vez analizados los parámetros necesarios se cotejan con los resultados obtenidos mediante la predicción con un modelo matemático, estimando de este modo la forma en la que crecerá la población de granos.

Se hace de esta forma y no por simple umbralización puesto que puede que algunos granos de azúcar tengan pixeles dentro del rango que abarca el fondo. Esto se debe a que como se comentó en apartados anteriores, los granos de azúcar no son opacos, son traslucidos, forman sombras, y estas sombras pueden tener el mismo nivel de gris que el fondo [18].

2.4.2 Técnicas basadas en la reconstrucción 3D de los cristales

Esta familia de técnicas es en la que se basa el artículo de Borchert [19] del cual se ha obtenido la mayor parte de la información. Estas consisten en obtener un descriptor en forma de vector de la signatura del contorno de la proyección 2D que se tiene del grano. Se obtiene esta signatura del contorno a partir de la distancia del centroide de la silueta del objeto al contorno como una función del ángulo. Muestreando a un cierto paso angular, se obtiene el vector descriptor de la signatura.

Por cuestiones vinculadas a la presencia de ruido o irregularidades en el contorno de la silueta de los granos proyectados (debida, en algunos casos a las propias irregularidades del proceso de crecimiento del grano o a la adhesión de pequeñas partículas al contorno del grano), la signatura puede presentar variaciones indeseables, desde el punto de vista de su descripción. Es por ello por lo que se puede recurrir a trasladar esta descripción a un dominio donde las pequeñas variaciones pueden desacoplarse fácilmente de las descripciones de mayor escala, como pueda ser la transformada de Fourier, de la cual se tomarán únicamente los coeficientes representativos de las cualidades más dominantes de la signatura. El hecho de descartar el resto de los coeficientes hace que se puedan despreciar las irregularidades del contorno que distorsionarían los resultados del estudio.

La siguiente imagen muestra los dos tipos de parámetros principales que se emplean para la reconstrucción del grano en 3D, por una parte, las mediciones del centroide al borde y por otro los valores absolutos de los coeficientes de Fourier.

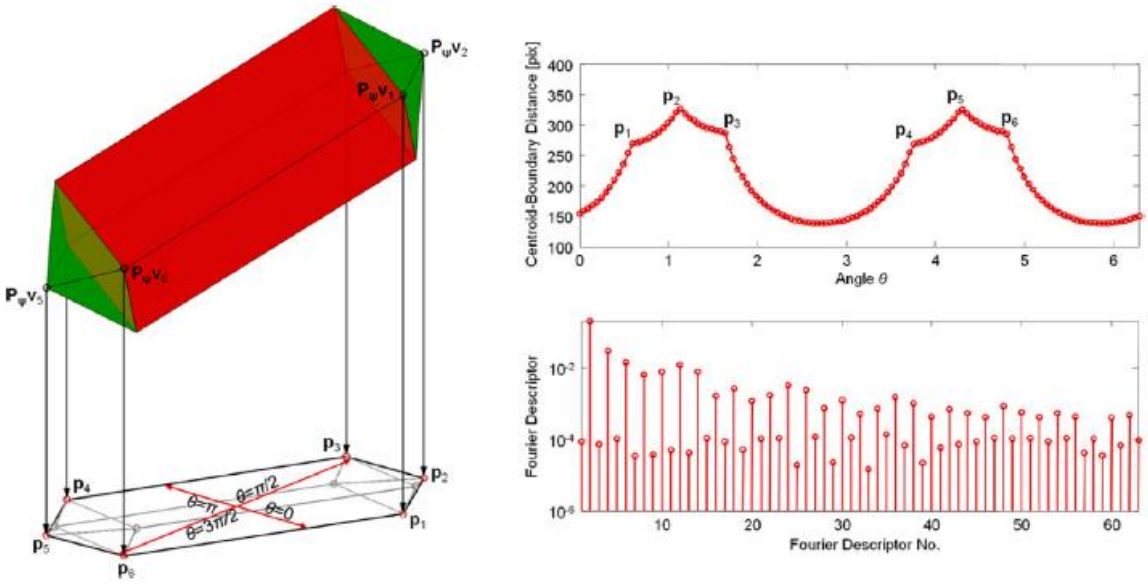


Ilustración 2-12 Parámetros usados para la reconstrucción 3D del grano [19].

En la imagen se pueden identificar los dos parámetros, en primer lugar, se encuentra la distancia del centroide al borde, en función del ángulo expresado en radianes. Esta medida, como ya se ha comentado, sería suficiente, pero no se hace del todo fiable debido a que el grano puede presentar impurezas y máximos no esperados tal y como se ha comentado en párrafos previos. Por ello, no se puede contar con los parámetros *Pi* como máximos locales tan solo. Con el objetivo de aclarar la forma del grano, el segundo parámetro que se utiliza son los descriptores de Fourier tomando los primeros 64 coeficientes (tal y como propone Borchert). De esta manera, se decidirán los parámetros *Pi* junto con el estudio de la variabilidad de los coeficientes de Fourier.

2.4.3 Técnicas basadas en la segmentación a través de formas planas

En esta familia es la que se basa el presente trabajo, siempre con las consideraciones oportunas de otras técnicas para una visión más completa del problema y de su posible solución. Estas técnicas se apoyan en el uso del procesamiento de la imagen tal y como es captada por el sistema de adquisición, sin considerar la tercera dimensión del grano, la profundidad, y obteniendo los resultados deseados usando parámetros como son radios, áreas, diámetros y diámetros de Feret (se verán y describirán en el próximo capítulo).

Para la obtención de estos parámetros se hará una segmentación en bruto, mediante el preprocesamiento de la imagen para su mejora de cara a la aplicación de algoritmos de segmentación como pueden ser la umbralización. Tras separar el grano del fondo, habrá que tener en cuenta una serie de consideraciones como son la aglomeración y las transparencias del grano para solucionarlo con técnicas de tratamiento de imagen avanzados. Una vez se tengan los granos binarizados y listos para su estudio es cuando se podrán obtener las medidas pertinentes para la extracción de conclusiones.

Estas son solo una pinceladas sobre las técnicas más relevantes en el panorama citado, serán ampliadas en el siguiente capítulo, donde se detallarán las técnicas concretas usadas para la solución que propone el trabajo y el porqué de su elección.

3 TÉCNICAS DE VISIÓN ARTIFICIAL APLICADAS A LA CRISTALIZACIÓN DEL GRANO DE SACAROSA

Tal y como se adelantó en el capítulo anterior, este capítulo tendrá como objetivo fundamental la descripción con un alto grado de detalle de las técnicas en las que se cimienta el trabajo.

El itinerario seguido en este capítulo será del tipo cronológico, ya que se empezará por describir algunas soluciones planteadas por autores que previamente han trabajado en este ámbito, pilar fundamental para el siguiente apartado, que consiste en la solución que finalmente se ha adoptado en este trabajo. Por último, como tiende a ocurrir en trabajo de investigación de esta índole, se comentarán los retos más importantes que habrá que tener en cuenta a la hora de implementar dicha solución.

3.1 Estrategias seguidas por algunos autores

Con la intención de hacer del trabajo una solución totalmente completa y funcional, los algoritmos usados en el mismo han sido fundamentados en algunos expuestos por investigaciones tanto referentes a la propia cristalización del azúcar como a otras que sin ser ese su motivo principal, hacen referencia a técnicas de visión artificial adaptables al problema que se abarca.

En su gran mayoría, las técnicas, funciones e ideas empleadas para el procesamiento y tratamiento de las imágenes son propias del conocimiento sobre el tema de la autora y tutor del presente trabajo, se ha optado, en algunos casos, por seguir las recomendaciones de algunos autores y páginas web para solucionar los problemas que se han ido presentando durante el desarrollo de los algoritmos.

El pilar fundamental del trabajo se encuentra en el artículo de Faria donde se hace un análisis en profundidad del problema de la caracterización del cristal de sacarosa, fácilmente extrapolable al trabajo [20]. En el artículo, se detalla el proceso completo, pero haciendo énfasis especialmente en la etapa donde el grano de sacarosa se cristaliza. La solución propuesta por estos autores consiste en seguir los pasos descritos en el siguiente esquema:

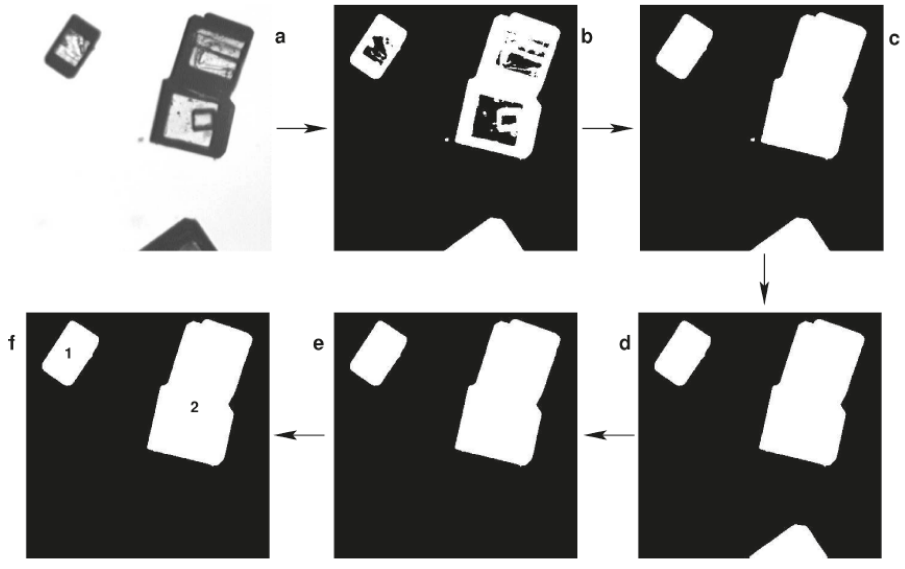


Ilustración 3-1 Esquema del preprocesamiento sobre la imagen [20].

En él, se puede observar que a través de la imagen de partida (a), puede llegarse con la aplicación de algunas técnicas al reconocimiento de regiones de los objetos presentes en la imagen. Para ello, en un principio, con la imagen (a) en escala de grises, se aplica una umbralización adecuada de la imagen, para así obtener la versión binaria de la imagen de partida (b). El problema siguiente que se presenta en esta imagen son las transparencias que presentan los granos de sacarosa, ya que no son totalmente opacos y al estar alimentados de una luz por su parte posterior, estos se pueden apreciar con zonas transparentes en sus caras, siendo las aristas del cubo que forma el grano las que quedan marcando la forma del grano.

La solución por la que se opta para resolver este problema es el relleno de los objetos, que, al aplicarse, da como resultado la imagen (c), donde se puede apreciar que los granos de la imagen no se encuentran huecos. Continuando con la mejora de la imagen, llama la atención que, debido a la umbralización, en la mayoría de los casos, quedan imperfecciones, píxeles de la imagen que han sido reconocidos como granos, pero que se corresponden a impurezas y píxeles con diferente nivel de gris, en este caso sobre el fondo de la imagen. Para combatir este problema, se aplican algoritmos de eliminación de ruido, dando como resultado la imagen (d). La siguiente tarea será la eliminación de granos situados en los bordes, tanto los que se encuentren cortados por ellos, como los que simplemente están en contacto, quedando así la imagen (e). Este paso se lleva a cabo para evitar el falseamiento de los datos aportados por las imágenes, ya que si se reconoce como grano entero uno que no lo es porque se encuentra cortado por el borde de la imagen, esto podría actuar bajando la media del tamaño de los granos en la imagen. Por ello, se opta por la eliminación de todos los granos que colinden con los bordes. De esta manera, las estadísticas serán los más fieles posibles a la información capturada por el sensor.

La última parte de este esquema es simplemente el reconocimiento y numeración de las regiones de la imagen para poder trabajar con ellas por bloques, y poder obtener las estadísticas para cada una de ellas y en conjunto. Este resultado lo vemos en la imagen (f), donde se ha superpuesto en cada grano su número de región.

Esta es sin dudas una estrategia que arroja muy buenos resultados para el tema que abarca el trabajo, pero tanto por la prueba propia de los algoritmos descritos como a través de la investigación realizada sobre la materia, se encuentra un problema que puede dificultar mucho la obtención de resultados.

El problema que se plantea se localiza en la fase del proceso anteriormente descrito a la hora del reconocimiento de regiones. En este paso, se espera como resultado el etiquetado de regiones por parte del programa usado, es decir, poder tener una región por cada uno de los granos que aparecen en la imagen. No es un procesado trivial, ya que los granos de la imagen, en su amplia mayoría, se encuentran aglomerados, o lo que es lo mismo, superpuestos. Este problema puede deberse bien a que estén lo suficientemente juntos para ser reconocidos como una única región, o bien porque se encuentren en diferentes planos de profundidad. Como se describió en el apartado 2.4, para la mejor toma de imágenes, los granos pasarán por un tubo muy estrecho, con poca profundidad y el diámetro suficiente para captar la imagen. Esta decisión viene incentivada por obtener imágenes con pocos granos, lo menos superpuestos posible, para la mejor obtención de estadísticas fieles al punto del proceso, ya que, si esta captura de imágenes se realizara a través de una escotilla en el propio tacho, los resultados no arrojarían datos reales, ya que se encontrarían superpuestos todos los granos en sus diferente planos de profundidad.

Con el objetivo de solucionarlo, se dispone del conocido algoritmo de segmentación avanzada Watershed, el cual se describirá con mayor detalle en el siguiente apartado junto al resto de soluciones propuestas. Esta solución ha sido elegida entre otras muchas técnicas de segmentación por su recomendación de algunos autores y por la propia documentación del programa Matlab (MathWorks) por ser un algoritmo que, con poco consumo de recursos, es capaz de lograr resultados de severa calidad.

3.2 Solución adoptada

La solución con la que se va a trabajar tiene un grado de complejidad tal que se decidió hacer unas pruebas previas de la misma sobre un escenario más favorable antes de probarla con las imágenes originales con las que se trabaja en la industria azucarera. Este escenario previo ha sido montado por la alumna con el material que disponía en casa (cámara y materiales de papelería) y con material cedido por asociaciones pertenecientes a la Universidad de Sevilla (flashes y trípodes fotográficos). En capítulos posteriores se procederá a explicar con mayor grado de detalle cómo está montado dicho escenario, ya que el apartado presente se centra en dar una idea general de cuáles son los puntos básicos por los que pasa la solución adoptada para el desarrollo del algoritmo.

Esta solución será la misma, tanto para el escenario previo de estudio como para las imágenes originales del tacho. Como es de esperar, debido a que la naturaleza de las imágenes y su formación no es exactamente la misma, el algoritmo de la solución se ve alterado a la hora de pasar de las imágenes procedentes del montaje propio a las obtenidas por las cámaras situadas en el tubo del tacho. En cualquier caso, se tratan de modificaciones con carácter de adaptación del algoritmo, adición de técnicas para mejorarlo, ya que el esquema general seguido en la obtención de la solución es el siguiente:

1) Lectura del fotograma a analizar.
2) Preprocesamiento y mejora de la imagen.
3) Algoritmo de binarización y filtrado propuesto por Faria para la mejora de la imagen [20].
4) Algoritmo Watershed para la segmentación de regiones.
5) Continuación del algoritmo propuesto por Faria para el reconocimiento de regiones [20].
6) Extracción de características de los granos de la imagen.
7) Representación de los datos y resultados obtenidos.

Después de la presentación de este esquema general, se procede a la descripción en líneas generales de cada uno de los pasos implicados, tanto el motivo de su inclusión como su utilidad de cara a la presente solución.

3.2.1 Lectura de la imagen

Se parte del hecho de que las imágenes con las que se trabaja son obtenidas de los fotogramas de un vídeo, ya que con el presente trabajo se pretende dar una solución dinámica y capaz de ejecutarse a la vez que se produce el fenómeno de la cristalización dentro del tanque.

Hay que diferenciar la naturaleza de las imágenes en ambas soluciones, ya que, en el escenario de pruebas, las imágenes son obtenidas por una cámara fotográfica con una resolución de lente de 24mpx, de manera estática, dentro de un montaje apropiado en términos de luminosidad y movimiento.

Sin embargo, las imágenes reales con las que se trabaja en la industria azucarera son tomadas de los fotogramas de un vídeo, cuya iluminación se hace a contraluz, con la finalidad de resaltar el contorno de los granos de sacarosa que se van formando y aumentando su tamaño, suspendidos en el jarabe.

Esta diferencia será determinante a la hora de aplicar los algoritmos de reconocimientos de regiones debido a que, en el escenario propio, se sufrirá de manera más acusada el fenómeno de las aglomeraciones a pesar de tomar regiones de las imágenes con poca densidad de granos. Por otro lado, en el escenario real no siempre será posible determinar el contorno completo del grano para poder aplicarle las técnicas de visión artificial adecuadas, debido a que no tengan suficiente grosor o la iluminación proporcionada no sea su óptima.

Una vez hecha esta diferenciación, se procede a la explicación de cómo tratar cada uno de los casos. Para la lectura de las imágenes del escenario previo se simula un crecimiento de los granos de sacarosa de manera artificial. Esto se llevará a cabo mediante un zoom digital o su equivalente recortes concéntricos. Esta idea se lleva a cabo dentro del propio programa, partiendo de una imagen original (que previamente se ha leído con la orden orientada al procesamiento de imágenes *imread*) se obtienen las sucesivas a partir de recortes de ellas. Habrá que tener en cuenta que, al aplicar esta técnica, se irán perdiendo cada vez más granos de las regiones del borde de la imagen, pero que, por el mismo motivo, veremos un crecimiento continuo de los granos restantes. Este crecimiento será posible gracias al hecho de escalar las imágenes con la orden de Matlab *imresize* cuya función consiste en hacer que una matriz como es la imagen pase de tener unos píxeles originalmente a otra cantidad dada como argumento de entrada.

Por otro lado, se tienen las imágenes procedentes de un vídeo, que en un escenario real de pruebas in situ de este algoritmo se tendría en *streaming,* pero que, para la resolución de este trabajo, se tendrá almacenado en disco como vídeo. Para trabajar con este formato se usarán las funciones *VideoReader* para almacenar el video en la memoria del programa y *readFrame* para obtener como una imagen el fotograma con el que se va a trabajar.

Una vez que se obtenga de una forma o de otra, anteriormente descritas, se está en el mismo punto a pesar de que las situaciones de partida sean diferentes, y por ello se prosigue de las misma manera para ambas situaciones. El siguiente paso consiste en transformar la imagen, leída por defecto como imagen en color RGB, a escala de grises, para lo cual se empleará la función de Matlab *rgb2gray,* obteniendo así la imagen como una sola matriz con los píxeles de esta como un número escalar comprendido entre 0 y 255, formato UINT8.

A partir de este momento, se tiene la imagen actual con la que se trabajará almacenada como una variable en memoria dentro del programa. De ella se obtendrán los datos necesarios y tras ellos, se leerá la siguiente imagen en la misma variable.

Este es uno de los fundamentos básicos de la visión artificial, que permite que en lugar de trabajar con variables del tamaño de una matriz imagen, se pueda trabajar con vectores con datos de estas imágenes, con características que la definen y dicen de ellas todo lo que se quiere saber para sacar conclusiones en masa. En pocas palabras, se pasa de tener un gran bloque de matrices imagen a una tabla con los valores de los parámetros que las caracterizan y en algunos casos, pueden llegar a clasificarlas.

3.2.2 Preprocesamiento y mejora de la imagen

En este apartado, se realiza una operación que a simple vista resulta sencilla, pero que puede ayudar a arrojar resultados con bastante mejor calidad. Se trata de aplicar a las imágenes de partida dos operaciones de procesamiento de imagen con la intención de mejorar la aplicación de futuros algoritmos a ellas, sobre todo algoritmos del ámbito de la segmentación de la imagen.

Para ello, se deberá tener en cuenta una peculiaridad presente entre ambos escenarios. Debido a la forma de obtención de ambos grupos de imágenes, en las obtenidas en el escenario propio, se tiene un fondo oscuro y el grano de azúcar ya formado de colores claros. Sin embargo, las imágenes tomadas a través del vídeo de la fábrica, se encuentra el fondo blanco, debido a que se trata de la iluminación, y los granos oscuros, ya que lo que se ve de ellos es su contorno a contraluz. Este hecho no afectará para este apartado, ya que las operaciones que se aplican en él funcionan de la misma forma sobre un fondo claro que sobre un fondo oscuro. Se considerará la inversión de las imágenes en apartados posteriores.

3.2.2.1 Operación de *opening*

En un primer lugar, se realiza sobre las imágenes una operación morfológica, tal y como se adelantó en capítulos anteriores, llamada *opening.* Esta operación tiene como objetivo la eliminación de puntos espurios de la imagen, como pueden ser pequeñas motas en el fondo del escenario previo o bien motas flotando de manera irregular sobre el jarabe en el que se encuentran suspendidos los granos. A su vez, su aplicación sobre imágenes en escalas de grises tiene un efecto beneficioso para el caso que se está estudiando. Se trata de la expansión de los contornos de la imagen, ya que el *opening* actúa sobre ella aplicando en cada píxel el máximo de su máscara. Con ello, se consigue que al estar los granos blancos sobre un fondo negro (puesto que en el caso del escenario es naturalmente así y en el vídeo se ha invertido la imagen original para trabajar con ella) los contornos de estos granos se expandan y sean más fáciles de reconocer y segmentar.

Se trata de una operación morfológica debido a que su aplicación, desde el punto de vista más detallado, se realiza a través de la aplicación pixel a pixel en la imagen de una pequeña máscara, En este caso, se cuenta con una máscara de tamaño 3 píxeles (un elemento estructurante de dimensión 3x3) para las imágenes del vídeo que tienen una resolución inferior, y una máscara de 5 píxeles (un elemento estructurante de dimensión 5x5) para las del escenario puesto que su resolución duplica a las anteriores. La intención de la operación es la de eliminar puntos que no se corresponden con el objetivo principal de problema y expandir los contornos sin afectar a su estructura original, y no el de cerrar otro tipo de oquedades objetivo para el que están pensadas otras máscaras de tamaños superiores. Con esta simple operación se consigue una imagen más limpia y nítida, carente a su vez de algunas pequeñas alteraciones que podrían haber sido consideradas como granos por error.

Al usar este procesado de *opening,* se está evitando también considerar como granos pequeños fragmentos desprendidos de granos, o granos como tal partidos durante el proceso. Este hecho será provechoso, pues considerar como granos los casos expuestos, podría conllevar el falseo de los resultados que se pretenden obtener más adelante, con los cuales habrá que tomar una decisión de cara al proceso industrial.

Tras argumentar su uso, esta operación se puede realizar de una forma muy sencilla con el programa Matlab a través de la función *imopen* la cual dependerá como parámetros tanto de la imagen a la que se aplica como de un elemento estructurante, creado en un paso anterior por el comando *strel*. Para este trabajo ha sido usada una máscara con estructura en forma de disco por sencillez, ya que en una máscara de 3x3 píxeles, no es algo determinante la estructura de esta.

Se muestran los resultados de la aplicación de esta máscara a las imágenes en cada uno de los casos:

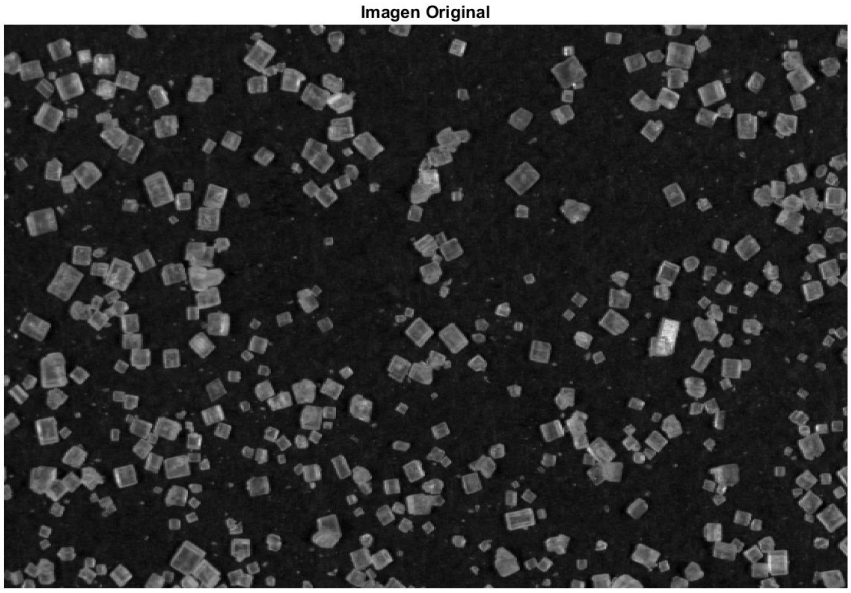


Ilustración 3-2 Imagen del escenario antes de la operación morfológica.

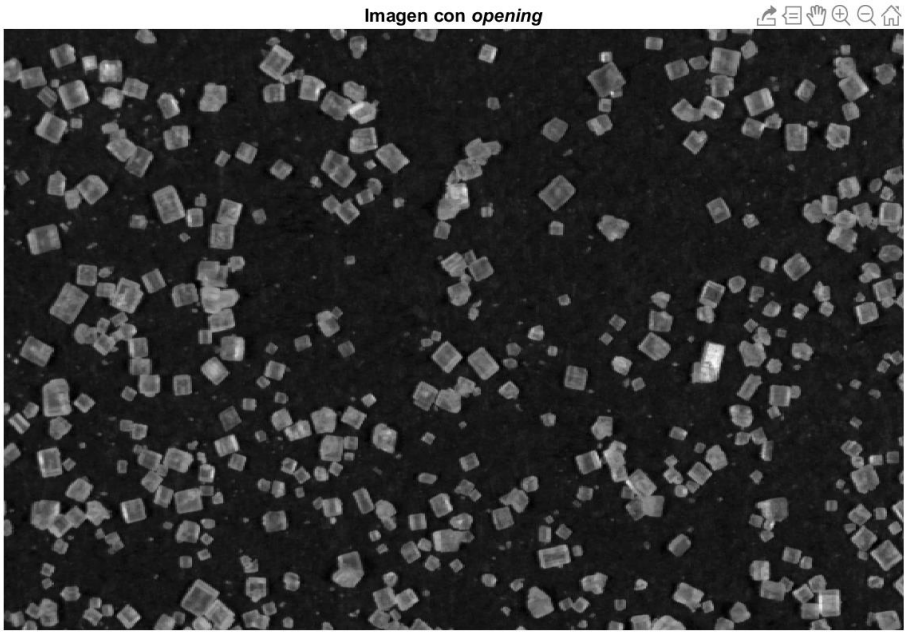


Ilustración 3-3 Imagen del escenario después de la operación morfológica.

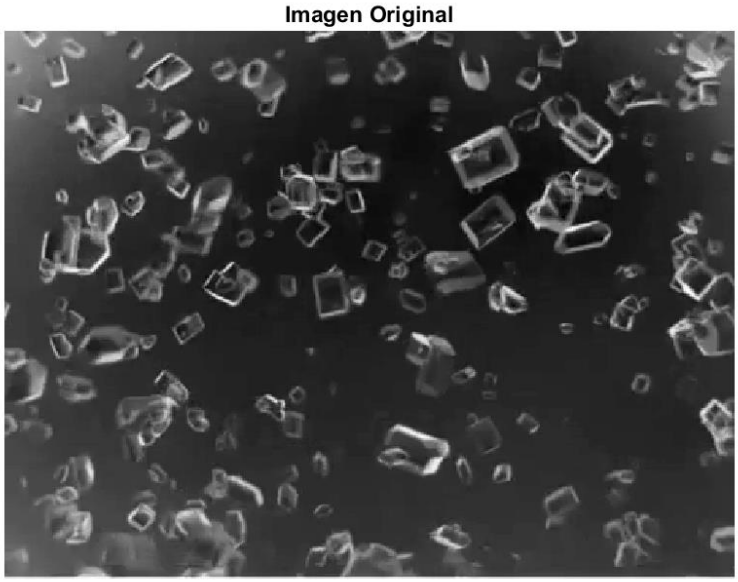


Ilustración 3-4 Imagen del vídeo antes de la operación morfológica.

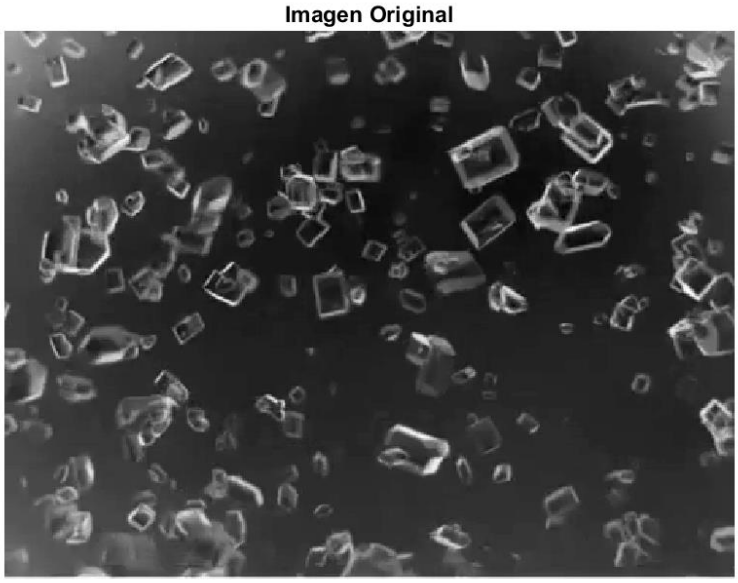


Ilustración 3-5 Imagen del vídeo después de la operación morfológica.

Las imágenes han sido elegidas de manera aleatoria dentro de las secuencias. En ella se puede ver cómo a cambio de un pequeño emborronamiento causado por la aplicación del *opening* se tiene un fondo bastante más limpio y nítido, evitando así falsos resultados futuros.

Resumiendo, conceptos, en este apartado se decide aplicar una apertura a las imágenes de forma previa a la aplicación de otros tratamientos. Esta apertura se hará con máscaras de tamaño 3x3 en las imágenes del vídeo y de 5x5 en las del escenario ya que por su diferencia de tamaño habrá que ajustar los tamaños de las máscaras.

3.2.2.2 **Mejora de contraste**

En líneas generales, las imágenes tanto procedentes del escenario de pruebas como las procedentes del vídeo del tacho, tienen un histograma con poco rango de dispersión de los niveles de gris de la imagen. En él se puede apreciar una mayor densidad de datos los extremos (debida a la presencia de granos sobre un fondo), pero con algunos de sus píxeles situados en niveles intermedios a estas masas de puntos. Este hecho da lugar a confusiones sobre el tratamiento de esos píxeles, por clasificarlo como grano o como fondo.

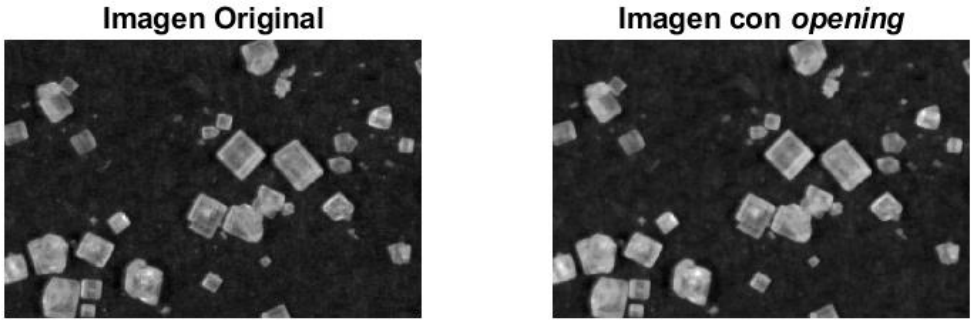


Ilustración 3-6 Detalle en la operación de opening sobre la imagen del escenario.

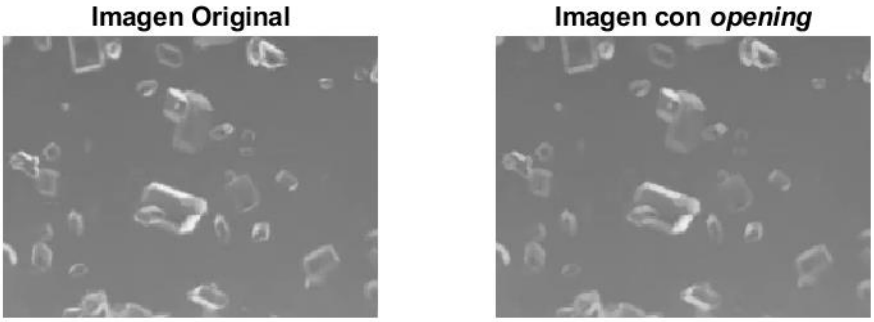


Ilustración 3-7 Detalle en la operación de opening sobre la imagen del vídeo.

Para evitar este tipo de confusiones, bien causadas por pequeñas perturbaciones en el fondo de la imagen o bien porque el contorno del grano no sea lo suficiente oscuro para ser tratado como tal, se emplea el uso de la función de Matlab *imadjust.* Esta función se aplica a la imagen con la intención de separar estas dos densidades del histograma, para así crear un contraste más pronunciado de los granos sobre el fondo. De esta manera se podrá obtener una segmentación más adecuada, ya que no se confundirán los píxeles propios del fondo con los que forman los granos. Esta operación recibe el nombre de *contrast stretching* y está orientada a imágenes esencialmente oscuras o claras, caso en el que se encuentran las imágenes con las que se trabaja, haciendo que en estas se produzca un aumento significativo de contraste que haga que sea más sencillo poder aplicar técnicas de visión artificial sobre ellas.

A continuación, se muestran los histogramas con sus respectivas imágenes antes y después de la aplicación de esta mejora, para ambos casos. Los resultados son:

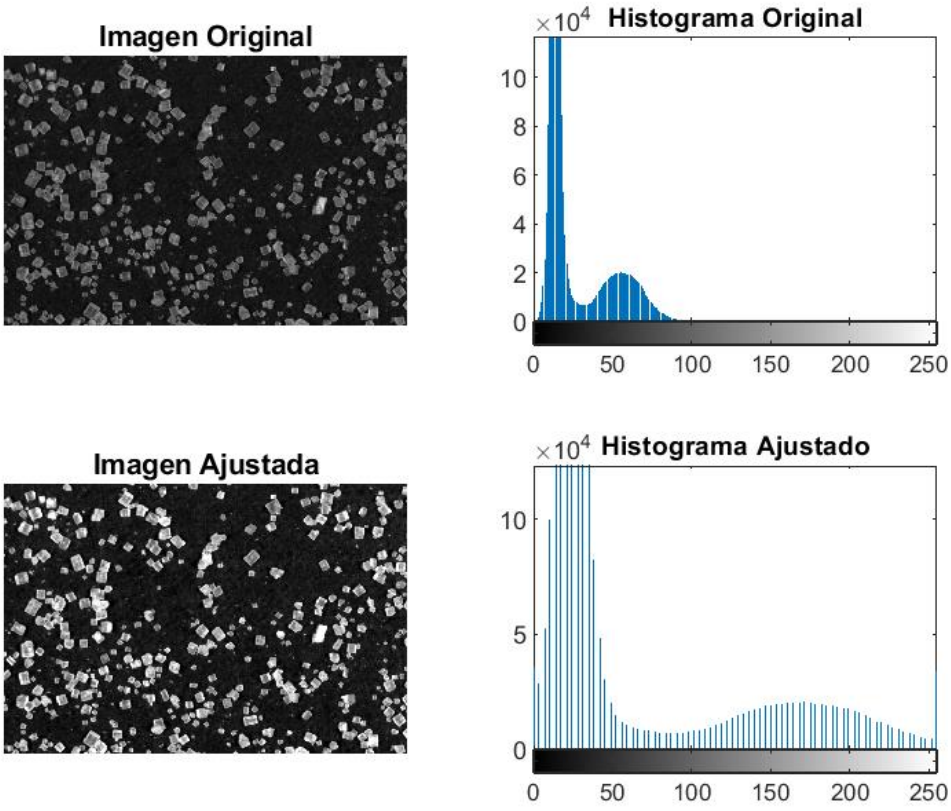


Ilustración 3-8 Muestra y comparación de los resultados tras la aplicación de la mejora sobre imágenes del escenario de pruebas.

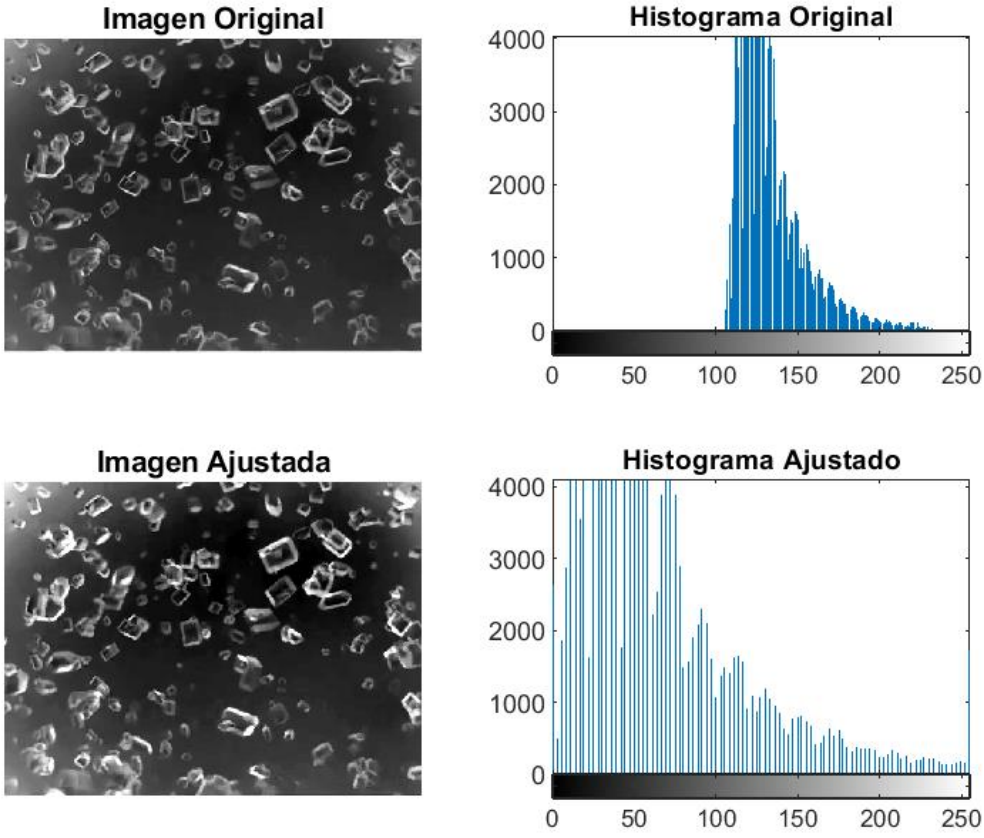


Ilustración 3-9 Muestra y comparación de los resultados tras la aplicación de la mejora sobre imágenes del vídeo.

De estas imágenes se puede sacar la conclusión de que en ambos escenarios la mejora aplicada funciona, consigue separar en términos del histograma las masas de píxeles para poder aplicar sobre ellos posteriormente algoritmos de umbralización y segmentación para que posteriormente se puedan obtener mejores resultados.

3.2.3 Algoritmo de binarización y filtrado preliminar de los granos

Este algoritmo, tal y como se comentó en apartados anteriores, fue propuesto en el artículo citado, con Faria como uno de los autores principales [14]. En él podemos encontrar un esquema básico de cómo tratar las imágenes con los granos de sacarosa de forma que se pueda trabajar mejor con la imagen en términos estadísticos. En este trabajo, se seguirán los pasos descritos como procesamiento previo al algoritmo que determinará si los granos obtenidos se corresponden a un único grano o a un conglomerado de ellos.

A continuación, se procede a detallar este proceso, adjuntando en todos los puntos de este, imágenes de los pasos y sus resultados. Con la finalidad de hacer más intuitivo el código del trabajo, esta sucesión de pasos se realiza en una función aparte, la cual es llamada desde el programa principal. Sus argumentos serán en entrada la imagen de partida, la mejorada en los pasos anteriores, y como salida, la imagen resultante tras todo el proceso detallado en el artículo encabezado por Faria.

Tal y como se comentó en el apartado anterior, había que tener en cuenta que las imágenes, debido a la naturaleza de su obtención, se encontraban con fondo oscuro frente a objeto claro y viceversa. En el apartado actual, se trabajará con las imágenes del escenario previo invertidas, para que la similitud entre ambos casos sea la mayor posible. Para llevar a cabo esta inversión, se usa simplemente una orden antes de la llamada de dicha función en la que se permutan los valores de la imagen en escala de grises.

a) **Binarización de la imagen:** este paso es esencial a la hora de trabajar en términos de visión artificial, pues en la mayoría de las aplicaciones, no se trabaja con la imagen ni en color, ni en escala de grises, sino que habrá que aplicar el algoritmo de binarización más adecuado para poder obtener de manera separada el fondo de la imagen de los granos. Con el fin de mejorar esta separación se ha procedido a los pasos previos, descritos en los apartados 3.2.2.1 y 3.2.2.2. Gracias a estos pasos, la imagen ha quedado limpia de impurezas en el fondo o incluso en los granos y más contrastada, para que así sea más sencillo para el programa reconocer el contorno del grano. De todos los posibles métodos de umbralización, se escoge la umbralización de Otsu por sus óptimos resultados con bajos niveles de consumo de recursos y por la conveniencia que aporta este método al tratarse de un umbral calculado con independencia de las imágenes de la secuencia. Así pues, a pesar de que las imágenes del vídeo sean tomadas bajo las mismas condiciones, la iluminación podría variar, haciendo que el uso de un umbral fijo no sea apropiado y pueda dar lugar a errores en la binarización.

   Esta umbralización se aplica sobre la imagen en escala de grises obtenida en el apartado anterior, la cual estaba optimizada para este paso. A través del programa Matlab, esta umbralización se consigue con la función *graythresh* para obtener el umbral comprendido entre 0 y 255, calculado para cada una de las imágenes, usándose luego para aplicarlo en la orden *imbinarize* como parámetro de entrada.

   A pesar de que la umbralización por Otsu es una de las mejores técnicas de binarización en el procesamiento de imágenes, debido a la características descritas de la estructura del grano de azúcar y en el caso de las imágenes del video real, este umbral se modificará un cierto porcentaje en cada caso para obtener los resultados deseados.

   Este ajuste consiste en bajar el umbral obtenido por Otsu un 35%, lo que hará que los granos queden mejor definidos en la binarización. Debido a la naturaleza de la imagen como ya se ha explicado, que se halla retroiluminada, no se espera binarizarla en un principio incluyendo su centro de la estructura, sino que primero se reconocerán sus aristas y en pasos posteriores se obtendrá el resto de su estructura. Como desventaja, esta bajada del umbral causará un oscurecimiento de las esquinas de la imagen, ya que en ellas no hay la misma luminosidad que en el medio y se binariza como grano también. Estas esquinas se descartarán de ser granos en el pasos a continuación.

Se muestran las imágenes de cómo quedarían binarizadas, y en el caso del vídeo, la diferencia entre el uso del umbral del Otsu tal y como lo calcula el algoritmo y el algoritmo de Otsu ajustado.

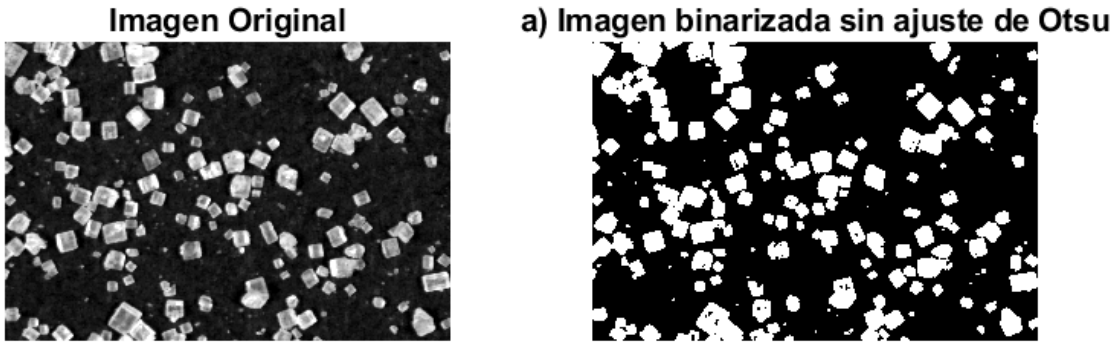


Ilustración 3-10 Imagen del escenario binarizada con Otsu.

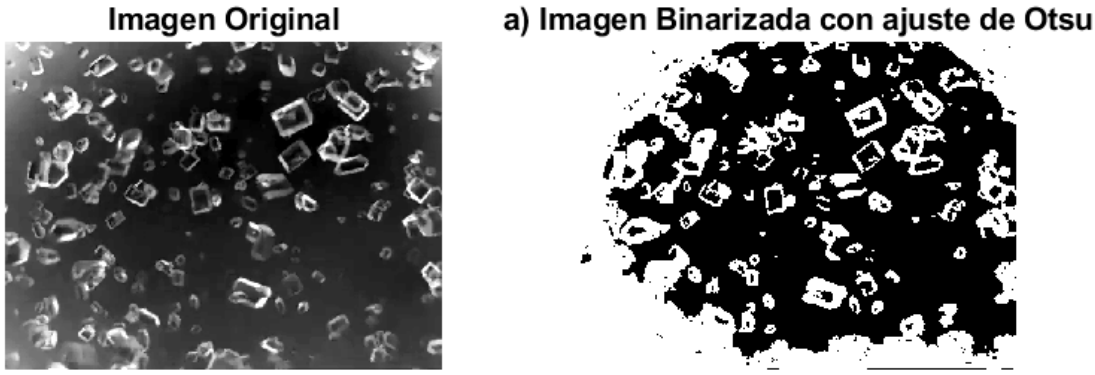


Ilustración 3-11 Imagen del vídeo binarizada con Otsu sin ajustar.

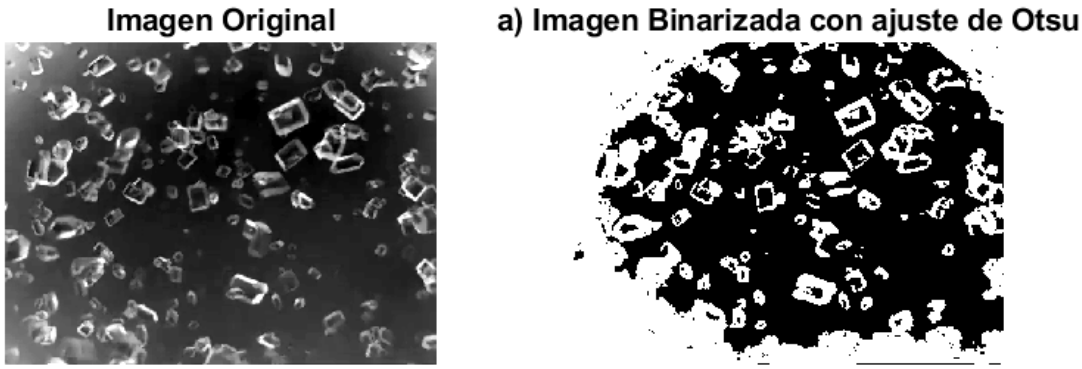


Ilustración 3-12 Imagen del escenario binarizada con Otsu ajustada al -35%.

En las imágenes adjuntadas se puede apreciar cómo en el caso del escenario no se hace necesario el ajuste del umbral de uso ya que el resultado que se obtiene es suficiente para ser procesado posteriormente. Sin embargo, en el caso de las imágenes del vídeo, si se usa el umbral dado por Otsu, apenas se podrán identificar los bordes del grano, lo que puede desencadenar desbordamientos y reconocimientos de regiones erróneas en los siguientes pasos.

Por este motivo, se opta por el uso de las imágenes del vídeo binarizadas con un umbral de Otsu disminuido en un 35%.

b) Relleno de huecos: uno de los problemas que presenta una umbralización es la existencia de pequeñas perturbaciones, aperturas o agujeros dentro de los objetos que se pretende segmentar, ya que, en su mayoría, no son objetos totalmente planos en su interior. Este factor con los granos de sacarosa en ambas situaciones se ve incrementado por su naturaleza cúbica, teniendo densidades y tonos diferentes en sus aristas respecto de sus caras. Esto afectará a la hora de umbralizar, dando como resultado regiones que se encuentran huecas en su interior (correspondientes a las caras de los granos). Como solución a ello, se usa la orden de Matlab *imfill* con la opción de *'holes'* la cual será capaz de reconocer las regiones y sus huecos internos y rellenarlos, o lo que es lo mismo, hacer que el valor de su interior, como el de sus bordes, sea '1'. Esta función no se aplica sobre la imagen binaria como tal, sino que se le realiza la operación de imagen gradiente a la binaria, de modo que se obtiene una imagen binaria con los contornos de los granos presentes en la imagen.

Con esta acción se consigue que, si la orden *imfill* no actúa de forma de forma correcta, se mantenga el contorno, y no el grano con las perturbaciones en su interior que posteriormente será reconocido de forma errónea. Estos granos que no hayan sido reconocidos como tal después del relleno, serán eliminados con el paso siguiente por tener un contorno menor que la máscara que se aplica.

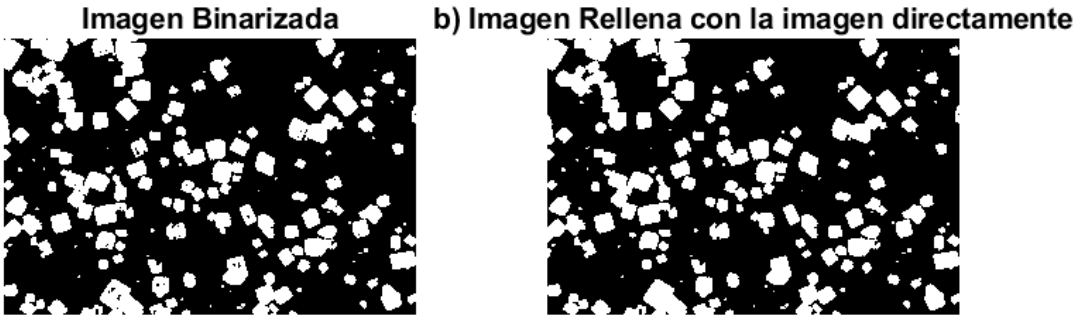


Ilustración 3-13 Imagen del escenario con el filling a través de la imagen binarizada directamente.

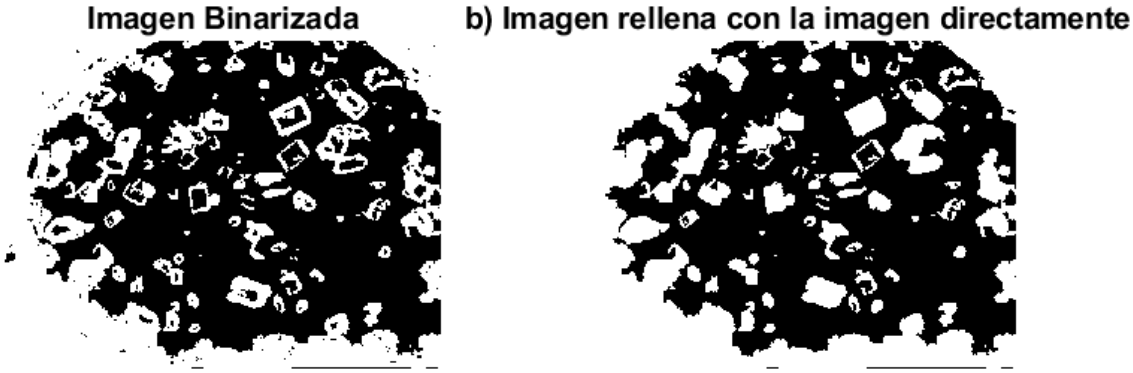


Ilustración 3-14 Imagen del vídeo con el filling a través de la imagen binarizada directamente.

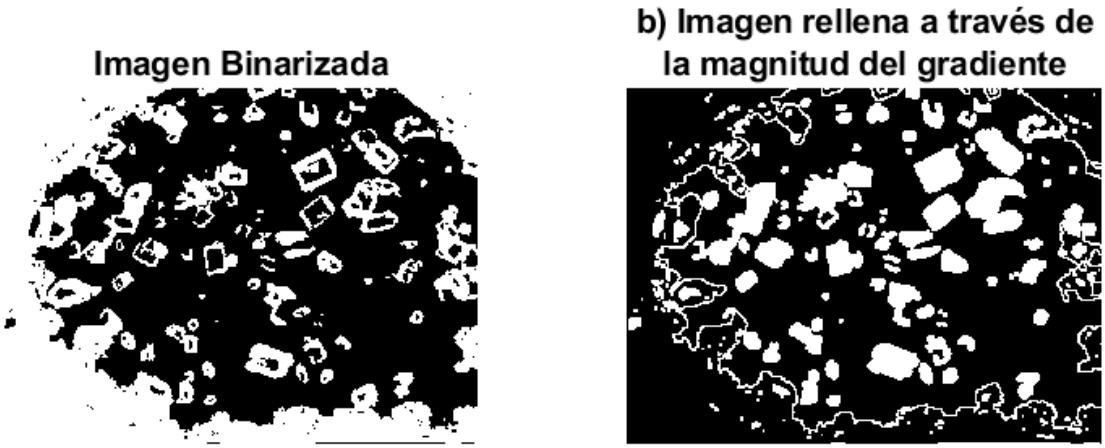


Ilustración 3-15 Imagen del vídeo con el filling a través de la magnitud del gradiente.

De estas imágenes se concluye que, en el caso del escenario, no es necesario actuar a través del gradiente, ya que sus granos se rellenan correctamente a través de la aplicación del algoritmo de relleno sobre la imagen sin tratar. Por el contrario, en el caso del vídeo, se puede apreciar que los resultados al aplicar el relleno sobre la imagen gradiente da mejores resultados que hacerlo sobre la imagen sin procesar. Por el momento, se tienen que los elementos situados en los bordes y esquinas aparecen como líneas del contorno, pero esto será el comportamiento óptimo para futuros pasos.

Por estos motivos, se elige la imagen rellenada por la binaria sin tratar en el caso de las imágenes del escenario y las imágenes rellenadas a través de la magnitud del gradiente en el caso del escenario, pues se obtienen los granos situados en el centro y alrededores de la imagen con un tamaño más aproximado al real que si no se aplica este tratamiento.

c) **Eliminación de ruido:** es posible que a la hora de umbralizar se obtengan pequeñas perturbaciones, más sobre el fondo que sobre los propios granos (combatido en el paso anterior con el relleno de los huecos). A pesar de que en los apartados 3.2.2.1 y 3.2.2.2 se ha actuado de manera preventiva a este hecho para reducir el número y tamaño de estas perturbaciones, siempre quedarán algunas de ellas presentes. Por ello, se procede a actuar sobre ellas una vez binarizada la imagen. Se hará a través de la apertura nuevamente de la imagen con una máscara de pequeño tamaño, la cual elimine píxeles que no han debido ser contados como '1', pero que a su vez no afecte demasiado al tamaño de los granos de la imagen. Esta apertura fue usada antes para la eliminación de impurezas en la imagen en escala de grises, ahora se usa con la intención de eliminar puntos espurios de pequeño tamaño que no son granos y reconocidos como ellos.

En este caso, se ha comprobado que para las imágenes del escenario es suficiente con la aplicación de un opening de tamaño de máscara 3x3 pixeles ya que no se encuentras más problemas que estos pequeños granos pertenecientes a impurezas del fondo. Sin embargo, para las imágenes del vídeo sí se necesita una máscara superior debido al comportamiento de la magnitud del gradiente sobre las sombras de las esquinas. Con una máscara de 3x3 no es suficiente, se necesita una de 5x5 para poder eliminar correctamente estas líneas no rellenas y además las pequeñas impurezas reconocidas como granos. Con esta apertura, se está optando por la eliminación previa de granos que no van a arrojar resultados fiables. Estos granos que se pierden en las esquinas de la imagen del vídeo debido a su mala iluminación en las imágenes no podrían ser reconocidos con su tamaño real, por ello se prefiere su eliminación del cómputo final.

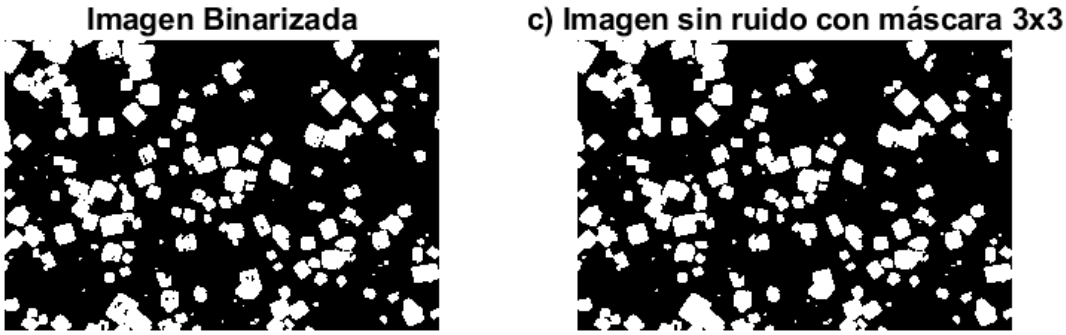


Ilustración 3-16 Imagen del escenario eliminando el ruido con máscara de tamaño 3x3.

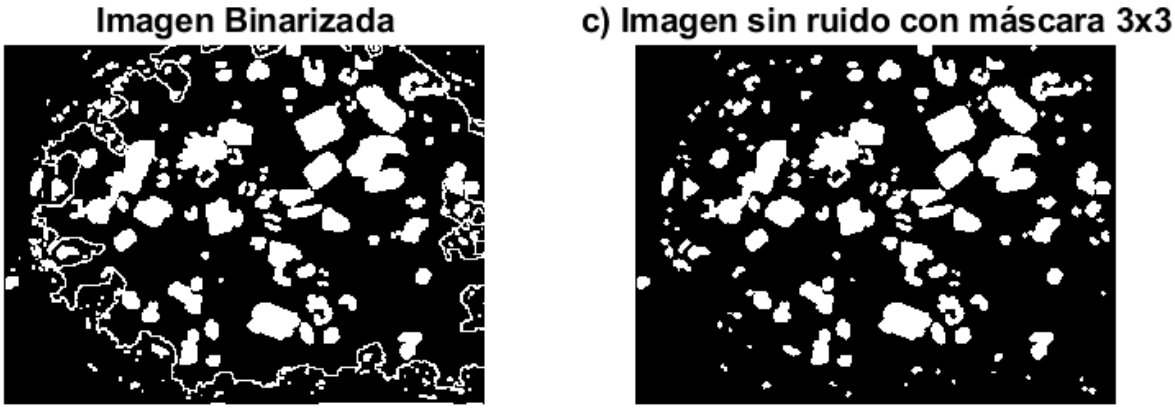


Ilustración 3-17 Imagen del vídeo eliminando el ruido con máscara de tamaño 3x3.

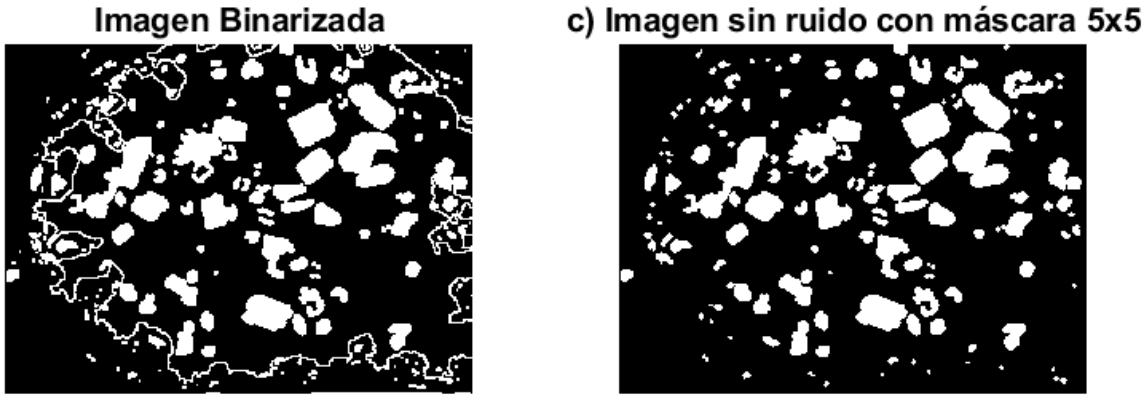


Ilustración 3-18 Imagen del vídeo eliminando el ruido con máscara de tamaño 5x5.

Expuestas las diferentes alternativas de aplicación de la eliminación de ruido, se considera el uso de una máscara 3x3 en el caso del escenario y de 5x5 para conseguir los resultados esperado en las imágenes del vídeo.

Esta decisión viene motivada por la desaparición con las mismas de los pequeños granos producto de imperfecciones, a la vez que errores en la aplicación del algoritmo de relleno.

Este fenómeno ocurre porque el algoritmo *imfill* funciona mejor a través de la magnitud de la imagen gradiente en imágenes donde después de la binarización no quedan bien definidos sus contornos, pues en estos casos se tienen aperturas en los elementos que *imfill* no reconoce como agujeros (*'holes'*) y que por ello no rellena, quedando el grano abierto y perdiendo tamaño que se debería tener en cuenta para futuros cálculos y estadísticas.

Como se puede apreciar, el resultado en la imagen del vídeo real no es del todo óptimo debido a que la naturaleza del vídeo es la retroiluminación desde un punto y por ello se acentúa la cualidad traslúcida del grano de azúcar, haciendo que lo que se puede reconocer de él son sus aristas y no sus caras tal y como pasaba en las imágenes del escenario con el que se ha practicado. A pesar de ello, con la aplicación de los pasos descritos anteriormente se pueden conseguir unos resultados en la segmentación aceptables y fieles a la realidad de la imagen a pesar de sufrir una cierta pérdida de números de granos de la imagen.

d) Eliminación de granos en los bordes de la imagen: este apartado, a pesar de estar descrito en el artículo de Faria, se va a dejar para un paso posterior, una vez que se aplique el algoritmo de separación de regiones Watershed. Esta decisión viene incentivada por el hecho de que al eliminar elementos que colindan con el borde de la imagen, si estos se encuentran en un estado de aglomeración, se acabarán eliminando más de un grano, es decir, se acabará perdiendo información que puede ser importante. Por este hecho se postpone su aplicación al apartado 3.2.4

e) Reconocimiento de elementos de la imagen: de la misma manera que con la eliminación de elementos en los bordes, este trabajo tiene unas necesidades diferentes a las que tienen las imágenes del artículo de Faria. El reconocimiento de regiones y la obtención de los parámetros de interés se postpone igualmente hasta después de la aplicación de Watershed y de su posterior eliminación de granos en los bordes.

3.2.4 Algoritmo Watershed para la segmentación de regiones

Tal y como se comentó en apartados previos, el uso de este algoritmo viene incentivado por la naturaleza de las imágenes, tanto por su forma de adquisición como por su disposición en el espacio. En ellas, se puede observar cómo en algunas ocasiones, los granos se encuentran suspendidos en la solución de manera tan próxima, que tras aplicarle la binarización y algunas mejoras, tienden a unirse granos en su reconocimiento. Otro factor que hace que haya una elevada cantidad de errores en el reconocimiento de regiones es el solape en los diferentes planos de profundidad de los granos.

Con el objeto de evitar estos errores, se usan máscaras de un tamaño muy pequeño en los algoritmos de mejora de la imagen y se toman las imágenes dentro de un tubo con la menor profundidad posible, como queda descrito en el apartado 2.4 cuando se habla de *Flow Cell* y su profundidad de 2mm.

La elección del algoritmo Watershed entre otros muchos algoritmos de separación de regiones viene dada por su robustez y buenos resultados en imágenes como las que presenta el trabajo. Cabe destacar, que durante su estudio en el ámbito del trabajo se han tenido algunos problemas con su uso en el programa Matlab, pero se han encontrado soluciones de suma calidad ante ellos.

3.2.4.1 **Descripción del algoritmo Watershed.**

El algoritmo Watershed pertenece a la familia de algoritmos de separación de regiones, siendo uno de los métodos mejor reconocido y usado en el estudio de las imágenes. Su significado en español se traduce como "cuenca", sirviendo de símil con ejemplos hidráulicos, lo cual es una práctica muy recurrente a la hora de explicar algoritmos no solo en el ámbito de la imagen, sino también en la aplicación de algoritmos en general.

Entrando en materia, su funcionamiento consiste en "inundar" las regiones de la imagen con mínimos locales. Estos mínimos pueden ser obtenidos bien por los niveles de gris de la propia imagen en escala de grises o bien por la imagen gradiente de esta. La decisión se toma dependiendo del tipo de imagen, tal y como se refleja en el libro de González y Woods [21], referente principal para la búsqueda de información de técnicas del procesamiento de imágenes. La inundación se lleva a cabo desde los mínimos locales, convirtiéndolos en fuentes de agua, que al ir inundándose convergirán en las sucesiones de puntos llamados barreras, es decir, las regiones de separación de los elementos de la imagen.

Este método trabaja internamente a través de la imagen de distancias. Esta imagen de distancias trata de una imagen en escala de grises compuesta por la distancia al fondo de cada punto de la imagen. Por ello, cada punto del fondo se considerará como 0 e irá incrementando su nivel de gris a medida que se acerque al centroide del elemento. Por ello, el centroide de cada elemento será el píxel con más brillo del elemento. A partir de ahí, se podrá aplicar fácilmente el algoritmo Watershed, con la técnica de inundación de los mínimos locales.

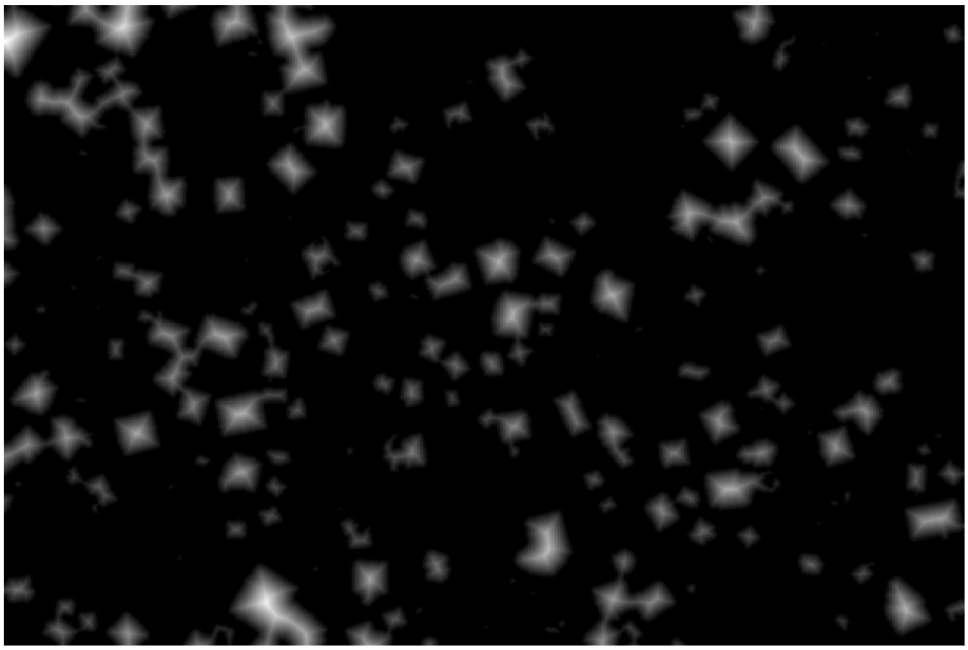


Ilustración 3-19 Imagen de distancias la imagen del escenario.

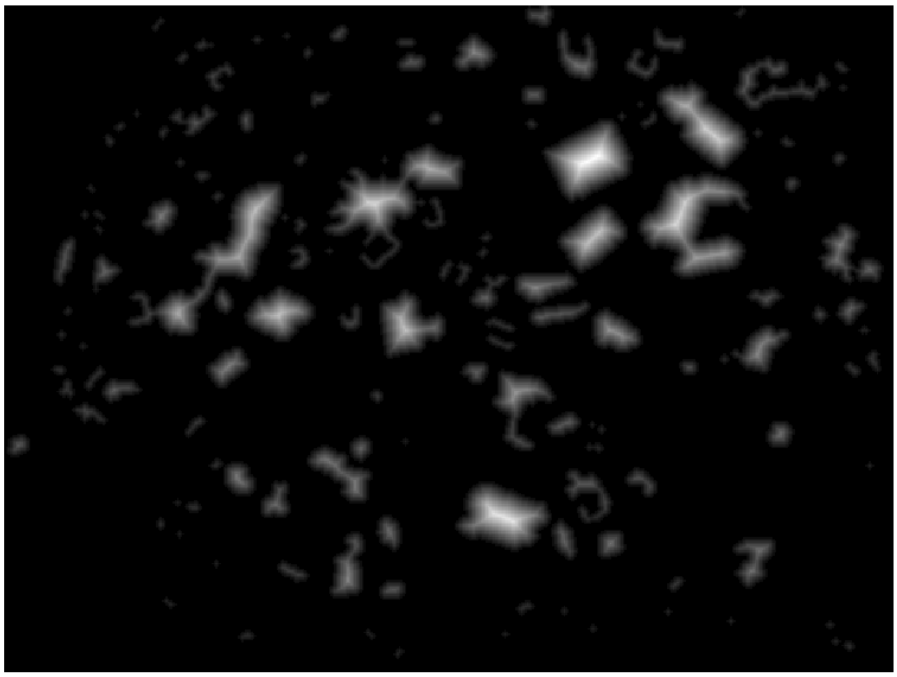


Ilustración 3-20 Imagen de distancias la imagen del vídeo

En el caso de este trabajo se opta por trabajar con la imagen binaria como entrada del algoritmo por los motivos que se ha visto previamente. Al aplicarle la imagen de distancias a la imagen binaria con aglomeraciones, Watershed será capaz de segmentar correctamente los elementos que se encontraban reconocidos como uno mismo en la primera imagen binaria. De este algoritmo se obtiene una imagen en escala de grises cuyos elementos se encuentran etiquetados de manera única, de manera que cada elemento se encuentra con los pixeles que lo componen a un nivel de gris diferente comprendido entre 1 y el número de elementos de la imagen. Como práctica generalizada, el fondo se encuentra etiquetado como elemento de valor 0.

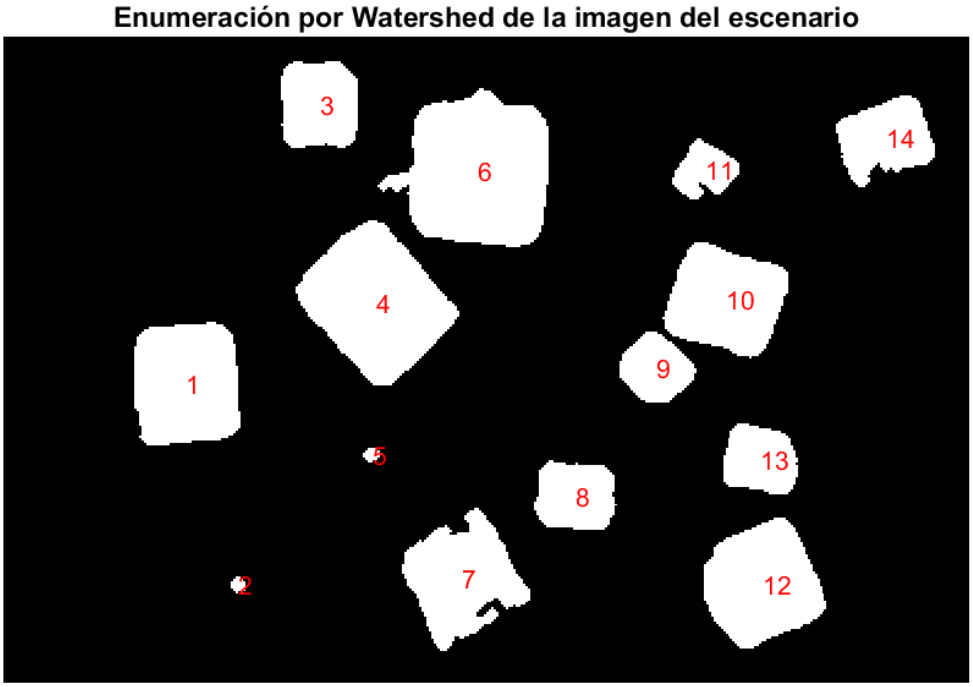


Ilustración 3-21 Resultado de aplicar Watershed a una imagen del escenario.

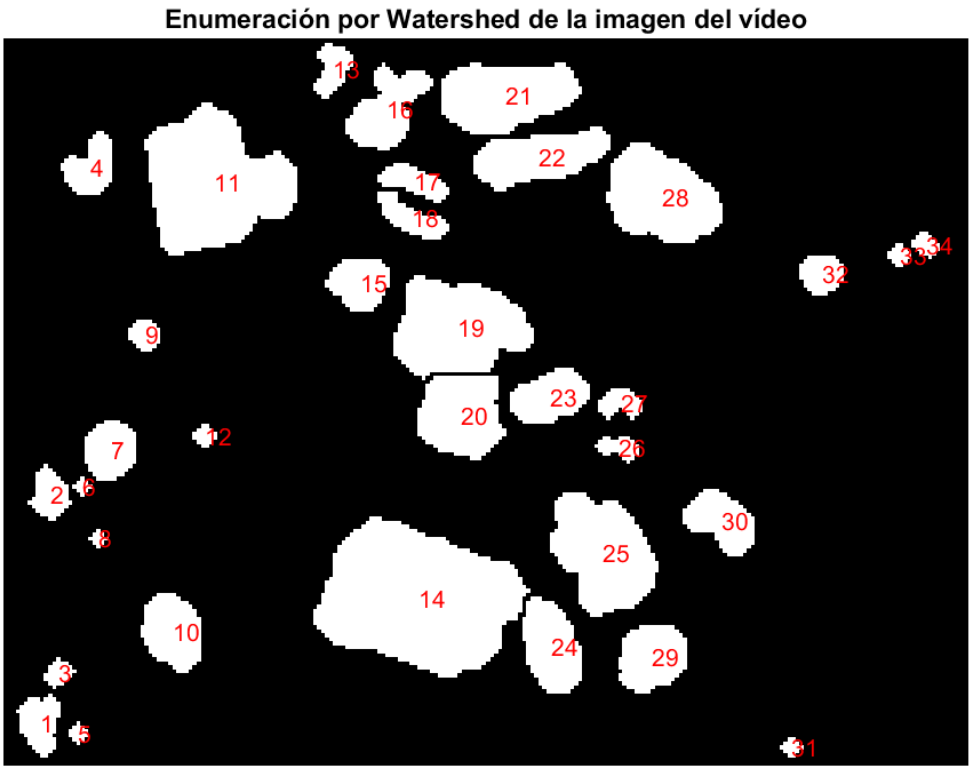


Ilustración 3-22 Resultado de aplicar Watershed a una imagen del vídeo.

Resumiendo, Watershed se trata de un algoritmo de separación de regiones que a través de una imagen en escala de grises que describa de forma topográfica las regiones a separar (imagen de distancias en el caso del trabajo) se obtiene como resultado una imagen cuyos elementos ya separados se encuentran etiquetados desde el número 1 en adelante.

3.2.4.2 **Watershed en Matlab**

Tras la consulta en la documentación de MathWorks se concluye que Matlab tiene implementada un método que aplica directamente el algoritmo Watershed a una imagen pasada como parámetro. Se trata del método *watershed* que desarrolla internamente el algoritmo y devuelve como salida una imagen etiquetada tal y como se ha comentado en el apartado anterior.

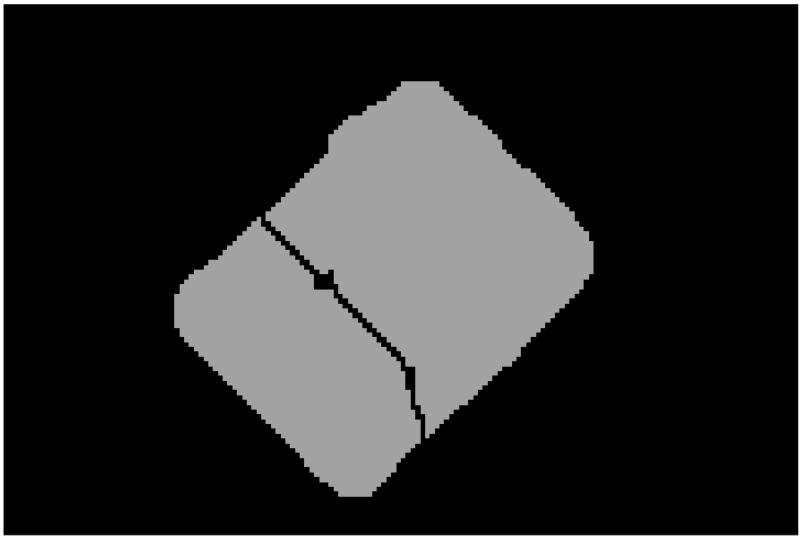


Ilustración 3-23 Detalle de la sobre-segmentación causada tras la aplicación de Watershed en Matlab.

Tras multitud de pruebas con la citada función implementada en Matlab, los resultados obtenidos no son los esperados ni los descritos en la documentación. El problema que se da de forma repetitiva es la sobre-segmentación de regiones. Este problema puede tener su origen en la naturaleza de las imágenes y por ello en la imagen pasada como argumento a la función.

Este error puede deberse a la estructura irregular de los granos de sacarosa que, al no tener contornos continuos y rectos, se producen errores en la aplicación interna del algoritmo. Para ello, se recurre al libro de González y Woods [21], que propone 3 formas de aplicar Watershed evitando la sobre-segmentación de sus elementos.

A continuación, se muestran dos ejemplos de la sobre-segmentación sobre imágenes del escenario y del vídeo causadas por la aplicación de Watershed en Matlab:

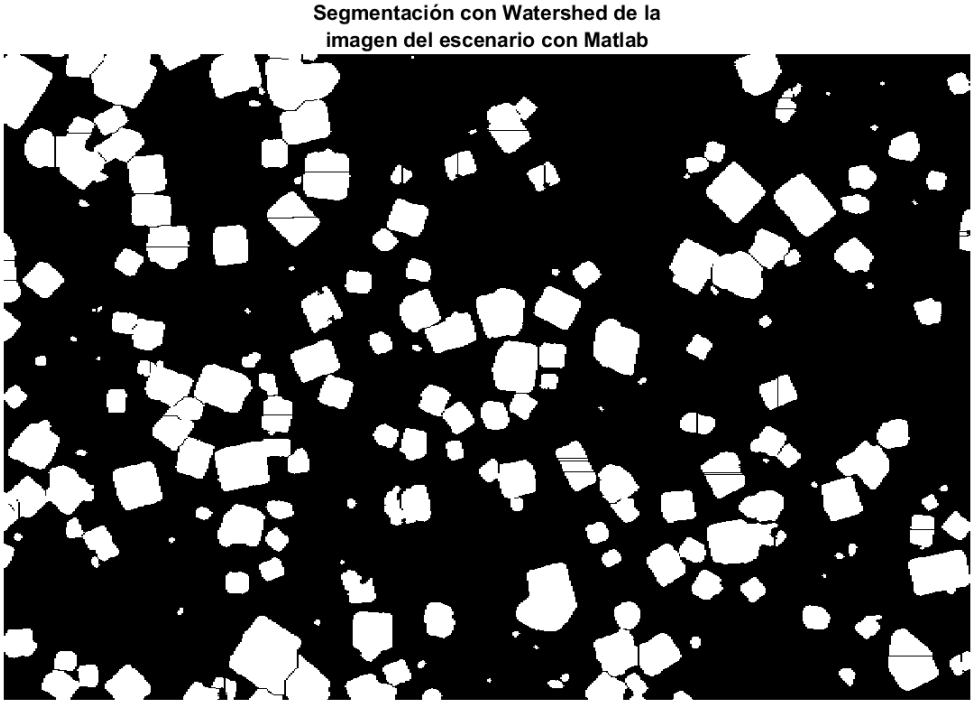


Ilustración 3-24 Imagen del escenario como ejemplo de la sobre-segmentación.

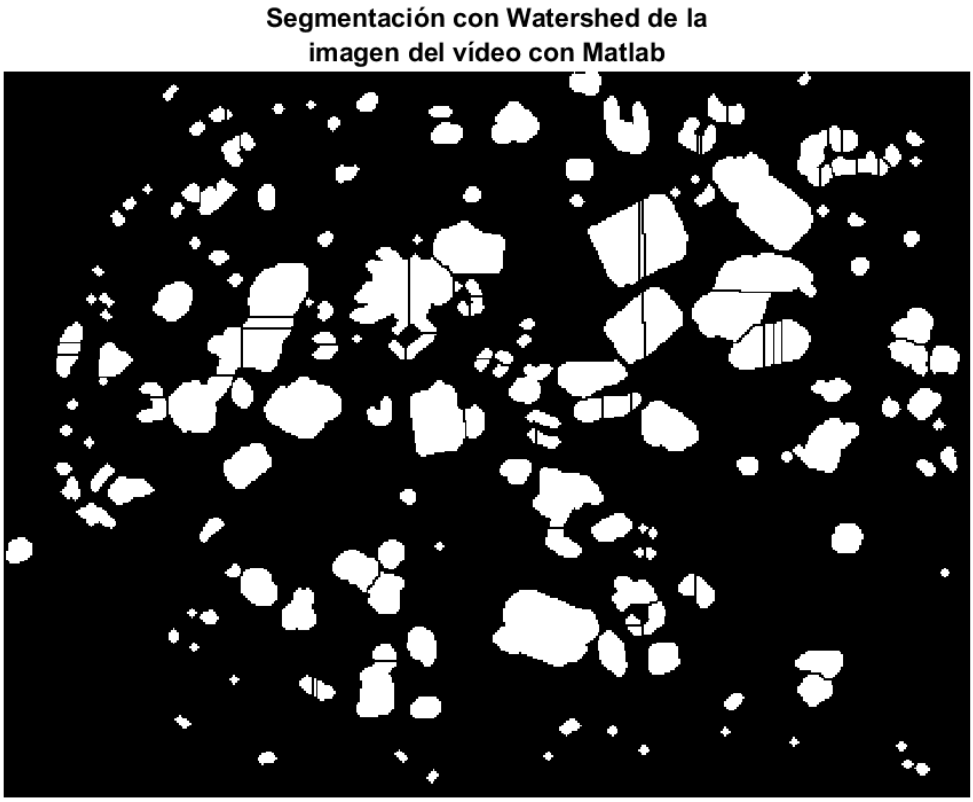


Ilustración 3-25 Imagen del vídeo como ejemplo de la sobre-segmentación.

A la vista de los resultados obtenidos aplicando el algoritmo Watershed tal y como se recomienda en la documentación de MathWorks, se estudia la posibilidad de hacerlo siguiendo las pautas descritas en el libro de González y Woods para prevenir la sobre-segmentación de las imágenes resultantes. En este libro se encuentran descritas tres estrategias que se han implementado y cuyos resultados se muestran a continuación.

a) Watershed usando la transformada de la distancia

Este es el ejemplo básico de uso de Watershed. En él se usa la imagen de distancias como parámetro de la función *watershed*. A diferencia de la literatura, en este trabajo se empleará la propiedad *'cityblock'* como forma de obtener la imagen de distancias, puesto que es la más recomendada para imagen con elementos de formas cúbicas.

Código para aplicar Watershed básico en Matlab

```
dist = bwdist(~image,'cityblock');
Label = watershed(-dist);
Label(~image) = 0;
```

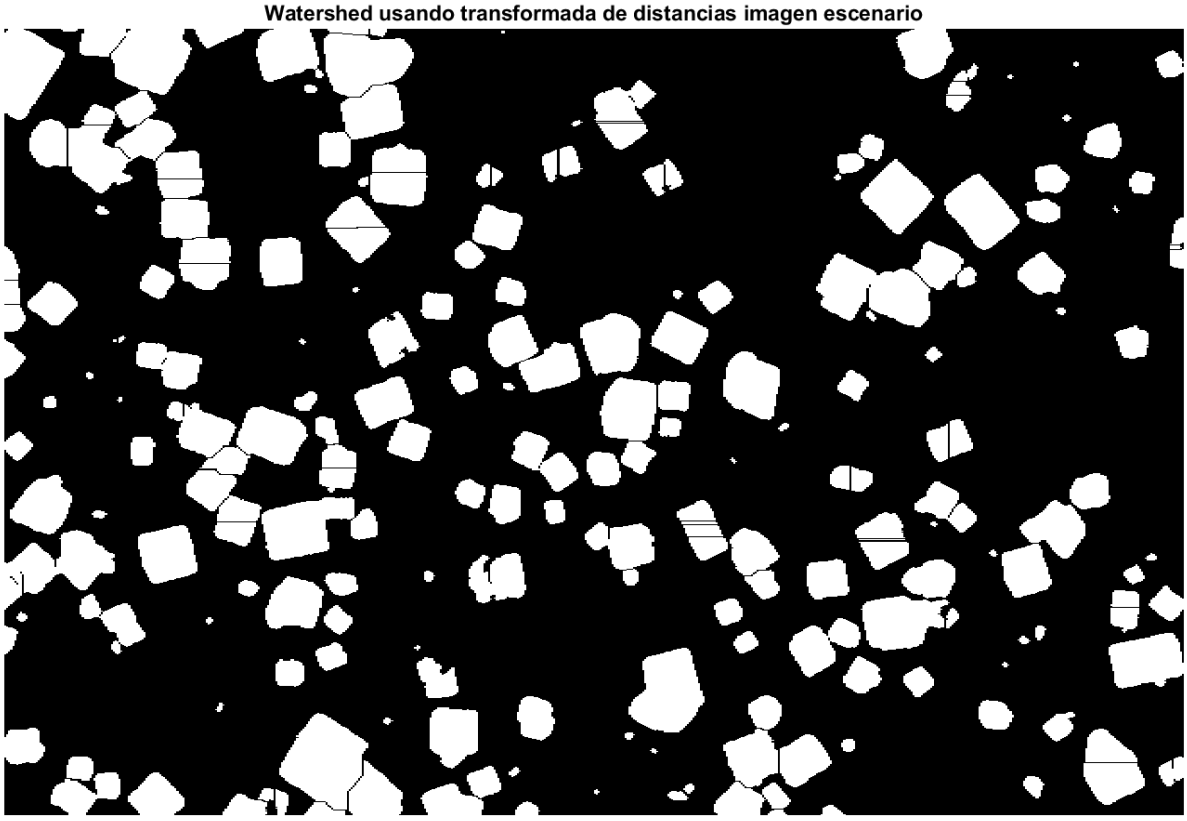


Ilustración 3-26 Imagen del escenario aplicando la primera propuesta del libro (imagen de distancias).

Con este código se podrá obtener la imagen previamente tratada, segmentada según Watershed. El resultado de aplicar esta técnica a cada una de las imágenes es el siguiente:

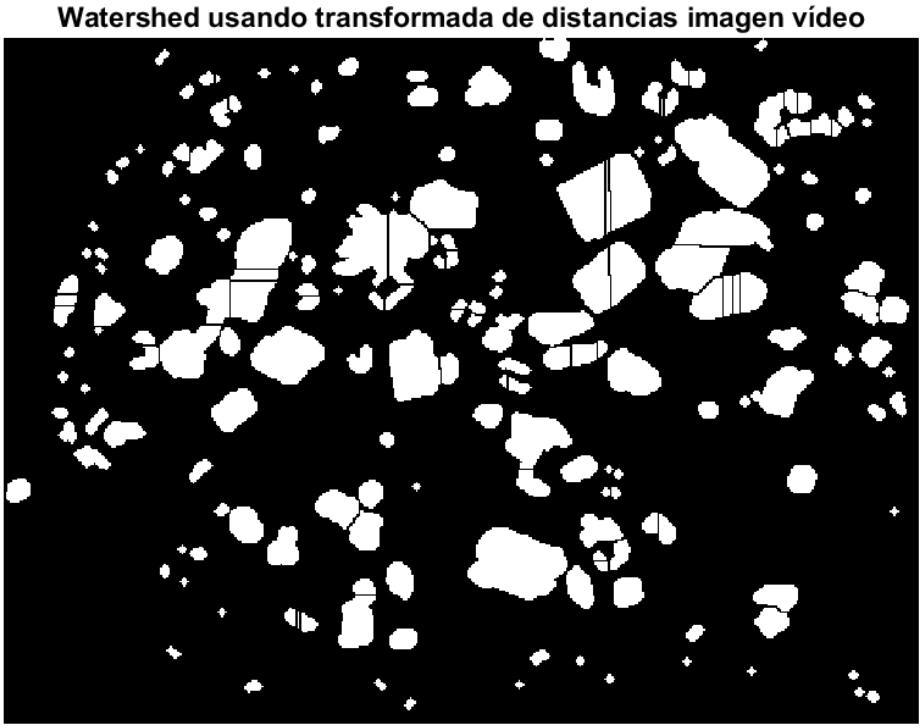


Ilustración 3-27 Imagen del vídeo aplicando la primera propuesta del libro (imagen de distancias).

b) Watershed usando gradientes

En esta técnica propuesta se usa la imagen gradiente para aplicar el algoritmo Watershed. La intención de esta aplicación es reducir el número de sobre-segmentaciones en la imagen mediante el uso de los gradientes para determinar dónde se encuentran las separaciones correctas de los elementos de la imagen. Esta técnica se implementa entre otros, con el siguiente trozo de código:

Código para aplicar Watershed usando gradientes en Matlab

```
h = fspecial('sobel');
g = sqrt (imfilter(double(image),h,'replicate').^2 ...
+ imfilter(double(image),h',replicate').^2);

p = ones(3,3);
g2 = imclose(imopen(g,p),p);

Label = watershed(g2);
wr2 = Label ==0;
f = esc;
f(wr2) = 0;
```

El resultado de aplicar esta práctica a las imágenes de ambos casos es el siguiente:

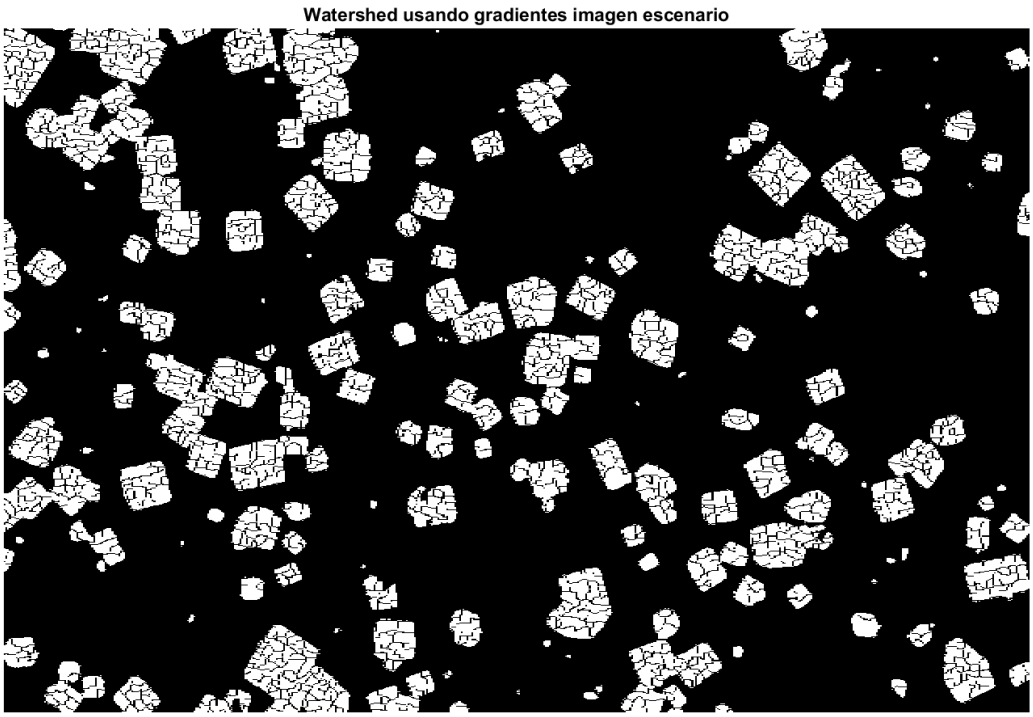


Ilustración 3-28 Imagen del escenario aplicando la segunda propuesta del libro (gradientes).

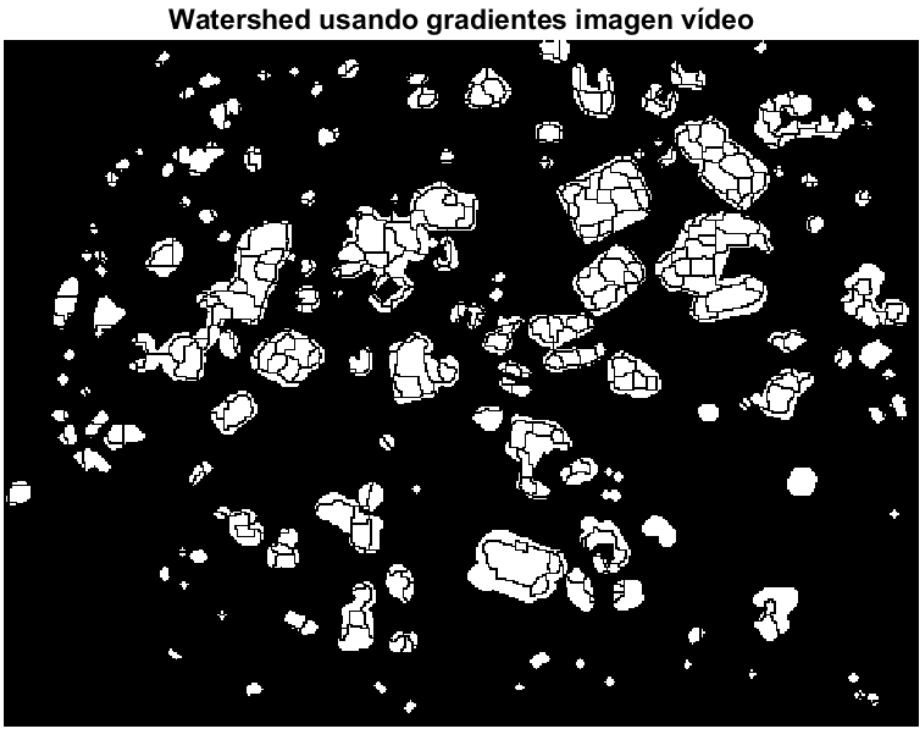


Ilustración 3-29 Imagen del vídeo aplicando la segunda propuesta del libro (gradientes).

Como se puede apreciar, a pesar de usar una técnica recomendada para evitar la sobre-segmentación, en el caso que abarca el presente trabajo, no se obtienen los resultados esperados.

c) Watershed controlado por marcadores

Esta es la última propuesta del libro, presentada como la solución definitiva de Matlab para la correcta segmentación y abolición de la sobre-segmentación del algoritmo Watershed. Esta técnica es la más compleja y difícil de aplicar y ajustar sobre las imágenes puesto que basa su aplicación en restringir el número de regiones segmentadas permitidas, evitando de este modo la sobre-segmentación. Este hecho se logra a través del uso de marcadores de dos tipos: internos, que indican donde se encuentra situado el interior del objeto a segmentar, y externos, contenidos en el fondo de la imagen, delimitando las regiones. Se muestran a continuación unas ilustraciones procedentes del libro Digital Image Processing [21] que permiten ver con claridad estos conceptos.

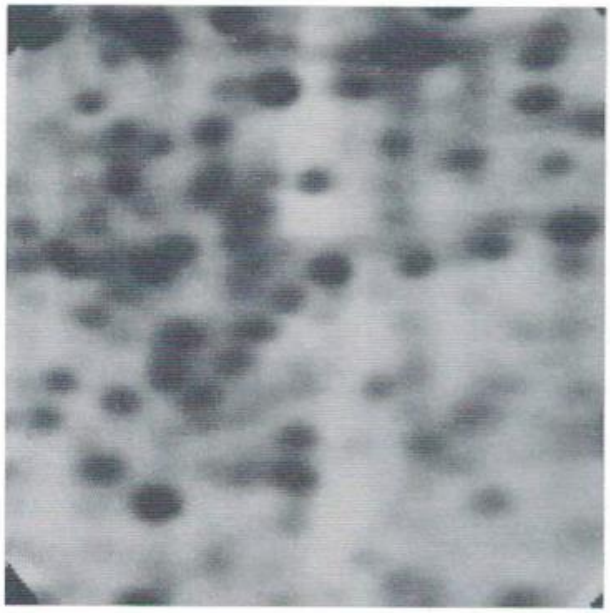


Ilustración 3-30 Imagen original del ejemplo (Figure 11.28 (a))

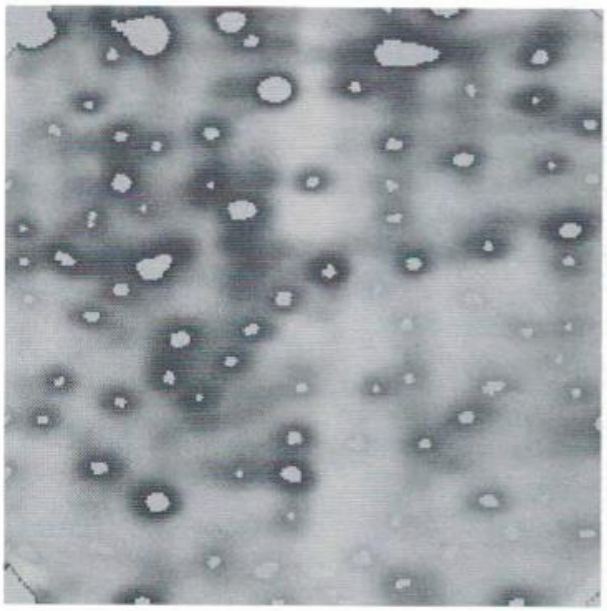


Ilustración 3-31 Marcadores internos en la imagen original (Figure 11.28 (d))

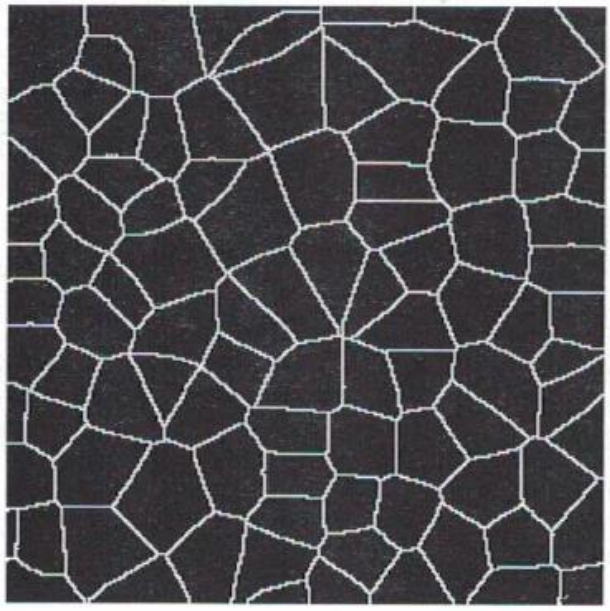


Ilustración 3-32 Marcadores externos

(Figure 11.28 (e))

Una vez conocidos los conceptos de los marcadores, tiene gran importancia su ajuste para el caso concreto que concierne al trabajo, pues los marcadores internos deberían ajustarse a cada una de las regiones (ajuste del valor de h descrito a continuación). El siguiente paso es modificar la imagen de gradientes con ellos para lograr abolir la sobre-segmentación.

Para ello, ha sido necesario los siguientes pasos descrito a través del código en Matlab:

Código para aplicar Watershed controlado por marcadores en Matlab

```
g = sqrt (imfilter(double(image),h,'replicate').^2 ...
+ imfilter(double(image),h','replicate').^2);


im = imextendedmin (image, h);

dist = bwdist(image,'cityblock');
Lim = watershed(dist);

em = Lim == 0;

g2 = imimposemin(g, im|em);

Label = watershed(g2);
f = image_bin;
f(Label==0) = 0;
```

Cabe destacar, que para cada uno de los tipos de imágenes ha sido necesario obtener un nivel de la variable *h* que se trata del umbral alto de la imagen. Estos valores han sido obtenidos de forma práctica mediante la ejecución consecutiva del código con diferentes valores posibles de *h*. Los valores tras estas pruebas han sido fijados en 30 para las imágenes del escenario y 95 para las imágenes del vídeo. Esta variación se debe de nuevo a la diferencia presente entre ambos tipos de imágenes.

Con su ejecución, se obtienen las siguientes imágenes de prueba:

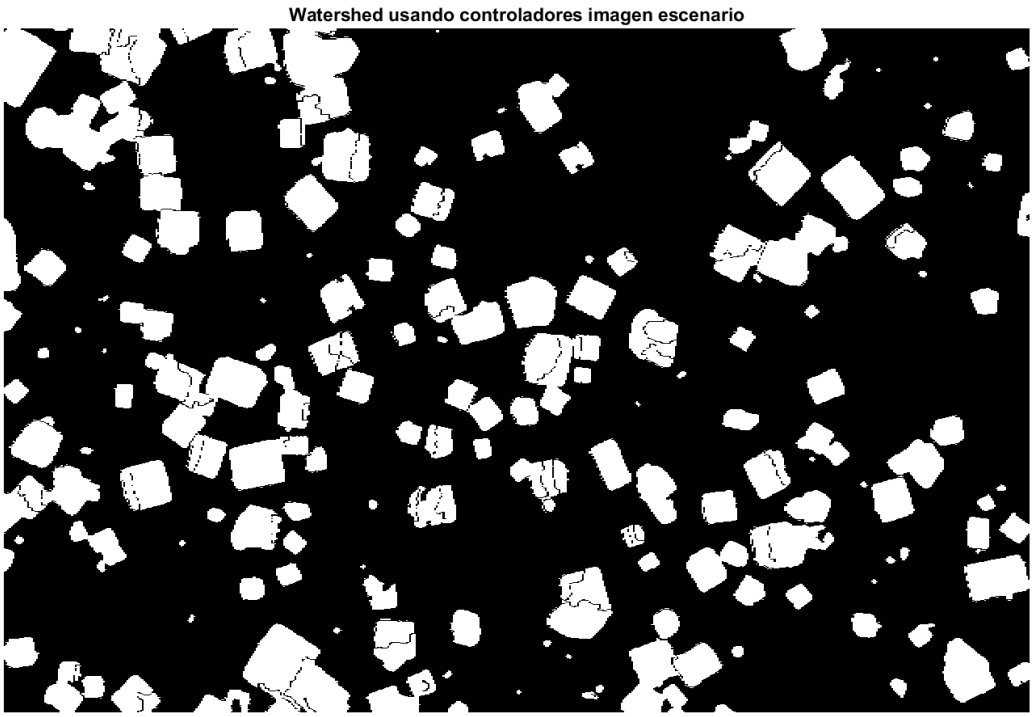


Ilustración 3-33 Imagen del escenario aplicando la tercera propuesta del libro (marcadores).

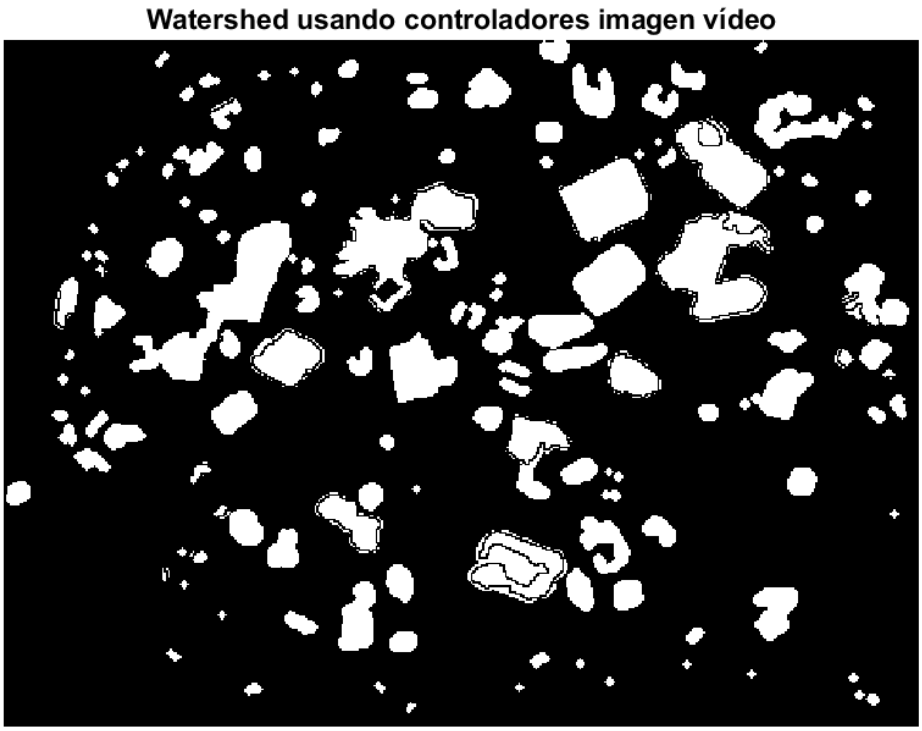


Ilustración 3-34 Imagen del vídeo aplicando la tercera propuesta del libro (marcadores).

A pesar de ser los métodos recomendados por autores con gran peso en el procesamiento de la imagen en Matlab, se puede apreciar que este algoritmo no da resultado en el tipo de imágenes presentes en el trabajo. Por ello se opta por buscar una solución fuera del programa Matlab, de forma que aporte resultados óptimos para la separación de granos de sacarosa aglomerados.

3.2.4.3 **Solución optada: Watershed a través de ImageJ**

Debido a la imposibilidad de segmentar correctamente las imágenes con el algoritmo Watershed que ofrece el programa Matlab se opta por el uso del algoritmo a través del programa especializado en imágenes médicos *ImageJ* (en adelante, IJ). Se conoce de esta solución por parte de la alumna debido a haberla puesto en práctica una asignatura del grado. Durante el desarrollo de la asignatura se usaron algoritmos de segmentación avanzados, de entre los cuales destaca con fuerza Watershed.

Gracias a la calidad de los resultados que se obtienen, se prueba a exportar este algoritmo exacto, mejorado con algunas técnicas sobre las imágenes de distancias y sus tipos de datos, al programa sobre el que se está trabajando: Matlab.

Al tratarse IJ de un programa de código abierto, se puede obtener fácilmente el paquete de funciones usadas para este algoritmos, programadas en lenguaje Java, desde su página principal [22]. Para su uso en Matlab se necesita usar el comando *javaaddpath* especificando la ruta donde se encuentra el paquete descargado como '*ij.jar*'. Además del paquete de funciones en Java usadas por IJ, se necesita también una función de Matlab encargada de transformar las imágenes tal y como las trabaja Matlab como matrices numéricas a un tipo de dato con el que trabaja IJ internamente, *Image Plus*. Esta función no se incluye en los paquetes estándares de Matlab, pero al tratarse también de una función de código abierto está disponible su descargar a través de la página *GitHub* [23].

Una vez preparado el entorno para aplicar el algoritmo, se procede a hacer las mismas pruebas que se han visto en apartados anteriores. Los resultados que se obtienen son los siguientes:

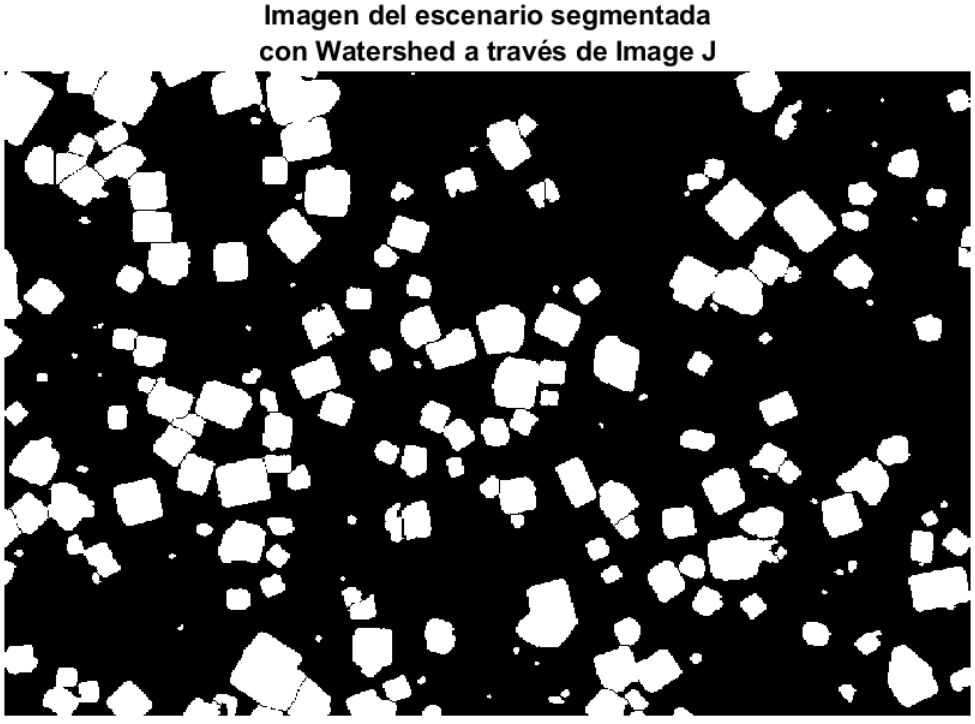


Ilustración 3-35 Imagen del escenario al aplicarle Watershed con Image J.

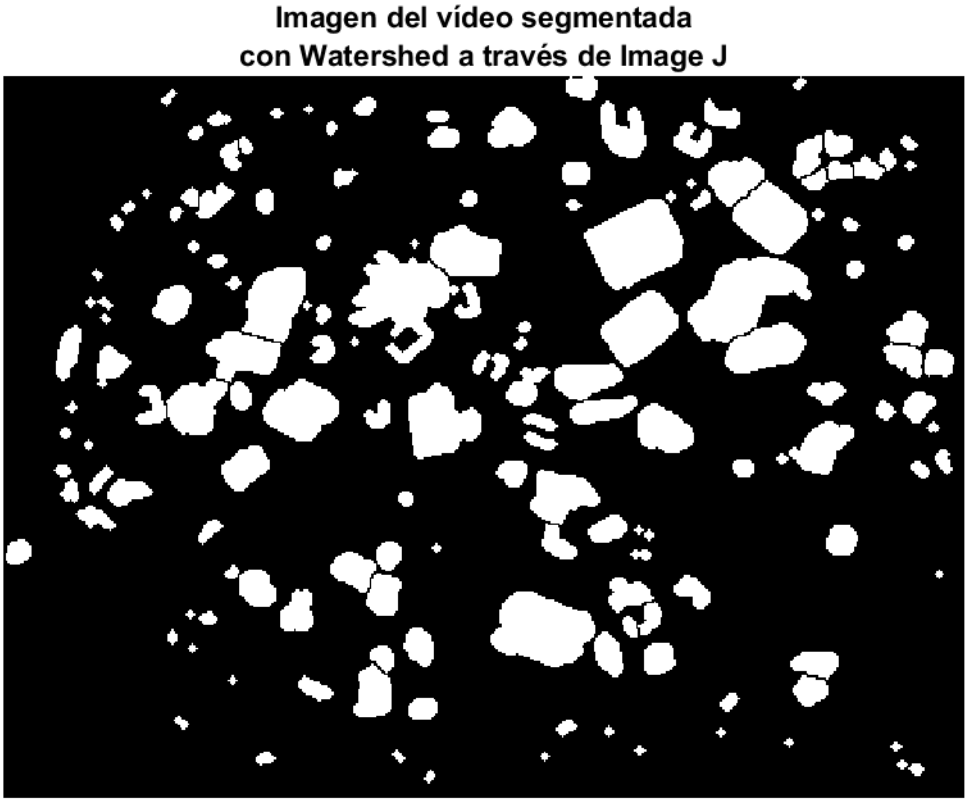


Ilustración 3-36 Imagen del vídeo al aplicarle Watershed con Image J.

Tal y como se puede apreciar, no se produce sobre-segmentación en ninguno de los casos estudiados. Este es el motivo fundamental por el que se elige la aplicación del algoritmo Watershed a través del paquete usado en Image J. Los resultados son los esperados, por lo que se continúa con la mejora de la imagen para su posterior estudio, a partir del procedimiento ya descrito.

3.2.5 Continuación del algoritmo propuesto por Faria para el reconocimiento de regiones

En este apartado se continúa con los pasos de Faria descritos en su artículo. Se recuerda que se interrumpió en el paso d) puesto que antes de él se debía hacer una segmentación adecuada a la naturaleza de las imágenes. Por ello, se sigue y repite el epígrafe:

d) Eliminación de granos en los bordes de la imagen: el siguiente paso en el tratamiento de las imágenes será tal y como propone Faria eliminar los elementos situados en los bordes de la imagen. Este paso viene motivado por la posibilidad de encontrar granos entrecortados con el borde de la imagen, por no haberlos podido captar el sensor. De esta forma, con la eliminación de los granos posiblemente cortados, los datos arrojados por el reconocimiento de estos serán más fieles a la realidad, pues no se cuentan con granos que no se visualizan a su tamaño adecuado.

Para la aplicación de esta técnica Matlab cuenta con la función *imclearborder* que automáticamente detecta los elementos en contacto con los bordes de la imagen y los elimina de esta para no ser reconocidos posteriormente.

El resultado de su aplicación es el siguiente:

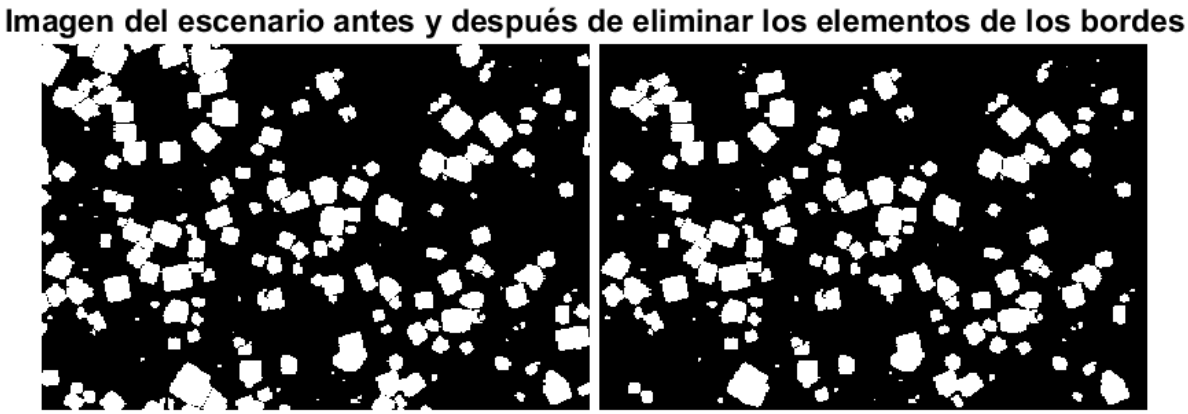


Ilustración 3-37 Imagen del escenario tras aplicar la eliminación de bordes.

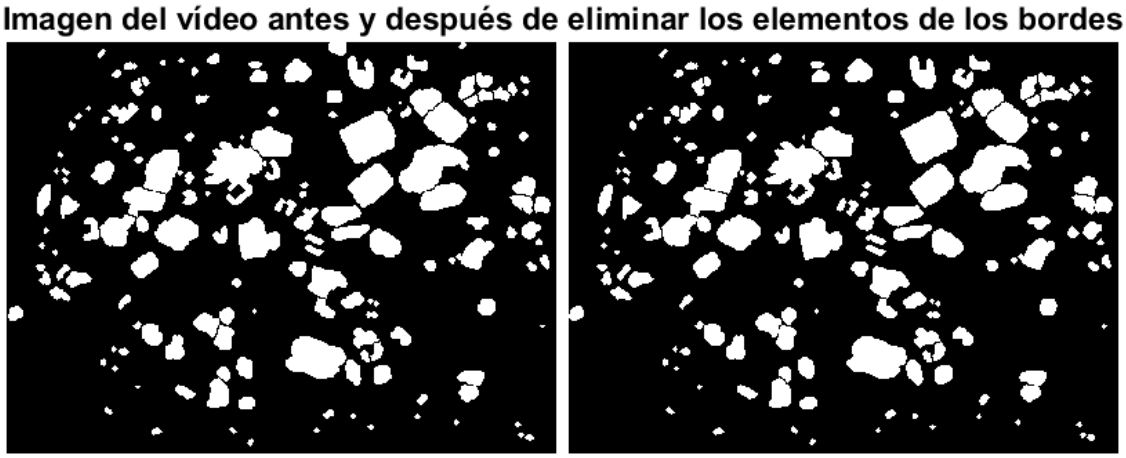


Ilustración 3-38 Imagen del vídeo tras aplicar la eliminación de bordes.

Como se puede apreciar, en las imágenes del vídeo no se puede percibir apenas el cambio al eliminar los elementos de los bordes porque este fenómeno apenas se da en estas imágenes. Esto se debe a que con la aplicación de la segunda apertura con radio 5px sobre la imagen de la magnitud del gradiente, los elementos de los bordes que se encontraban en las sobras de la imagen ya fueron descartados. A pesar de ello, se mantiene el uso de esta sentencia para evitar futuros casos y por consecuencia, la obtención de datos que no se ajusten con la realidad de la imagen.

e) **Reconocimiento de elementos de la imagen:** una vez que se tienen los granos que se van a estudiar en la imagen, el paso restante con ella es transformarla desde el espacio de las imágenes en dos dimensiones al espacio de los vectores de características. Esta es la práctica más usada en *Machine Learning*. En este paso, la decisión más importante es saber elegir adecuadamente las características que se quieren estudiar y qué obtener de ellas, para así poder llegar a las conclusiones que se necesitan en el presente trabajo.

   Estas características se podrán extraer fácilmente con la orden de Matlab *regionprops* la cual contiene una amplia variedad de parámetros, las características que se podrán calcular de cada una de las regiones de la imagen.

Estas propiedades leídas de la imagen a través de *regionprops* se detallarán en el siguiente apartado donde se explicará la motivación y utilidad de cara una de las características de la imagen.

Gracias a *regionprops* se podrá pasar de una imagen en escala de grises cuyos elementos se encuentran en diferentes niveles de gris para identificarse como elemento independiente a una variable tipo *struct* donde en cada nivel se tendrán los diferentes valores de las propiedades exigidas a en su invocación. Por ello, será una estructura con tantos valores como granos haya en la imagen y de esta forma se podrá estudiar cada elemento por separado y decidir si se incluye o no al cómputo general, en el caso de que se acontezcan errores en el procesamiento previo de la imagen.

3.3 Retos importantes en la aplicación de la visión artificial

Una vez implementado el algoritmo necesario para el procesado de las imágenes el aspecto que más preocupa y sobre el que más se trabaja es la segmentación. Como se ha expuesto, ha sido uno de los pasos sobre los que más se ha trabajado puesto que la naturaleza de las imágenes hace que se produzca la sobre-segmentación en los granos, o por el contrario no se reconozcan como aglomerados. Con la aplicación del algoritmo Watershed a través del método usado en el programa de código abierto Image J, este problema queda solucionado de forma óptima, ya que consigue segmentar correctamente la amplia mayoría de granos de la imagen.

Por este motivo, el reto que se plantea es conseguir que la imagen se binarice correctamente reconociendo los granos sobre el fondo, a pesar de que estos se encuentren aglomerados. En este último caso, se contará con el algoritmo citado anteriormente junto con el proceso de preprocesamiento de las imágenes que se aplica antes sobre ellas.

Con todo esto, se ajustarán los parámetros en estos pasos para evitar en la amplia mayoría de casos que se produzcan este tipo de errores, pero que, si se producen, el programa sea capaz de identificar que se han producido y pueda actuar descartando la imagen procesada erróneamente.

4 ANÁLISIS DE LOS DATOS EXTRAÍDOS DE CADA GRANO Y SU PROCESAMIENTO POR IMAGEN

Este capítulo está destinado al estudio de las imágenes con la finalidad de obtener los datos que se necesitan de ellas, lo que en términos del procesamiento de imágenes se denomina pasar del dominio de las imágenes al espacio de las características. De esta forma se pasa de tener la información como un dato bidimensional del que no se puede hacer más que interpretarla visualmente a tener la información controlada por los parámetros más adecuados para obtener la información que se desea finalmente: saber si el proceso se encuentra en el punto óptimo para su detención o si por el contrario debería seguir.

Este análisis constará de dos partes fundamentalmente. Una primera que analizará los parámetros obtenidos de cada imagen por separado. En él se calculará cada parámetro de la imagen directamente y se decidirá para cada grano si se va a considerar en el cómputo o no y si la imagen en sí se computará o por el contrario se ha sufrido algún error en la segmentación y se descarta por no arrojar resultados fiables.

La segunda parte del análisis se hará sobre el conjunto de imágenes analizadas hasta el momento, pues del presente software se espera ser ejecutado en tiempo real, y no tal y como se está probando sobre un vídeo ya grabado. De esta segunda parte se pretende obtener en qué punto se encuentra el proceso, así como tomar la decisión de seguir leyendo imágenes del proceso, darlo por finalizado o detenerlo tras detectar un comportamiento anómalo en el crecimiento de los granos.

La estructura del capítulo será la siguiente:

1) Extracción de características de cada elemento de la imagen
2) Comprobaciones realizadas sobre las características extraídas
3) Análisis de los datos de cada imagen
4) Análisis del conjunto de las imágenes

4.1 Extracción de características de los granos de la imagen

En este apartado se pretenden asentar la elección de cada característica que se va a obtener de cada uno de los elementos presentes en la imagen. En él se limita a exponer cada una de las características, definirlas y describir su forma de obtención, pues en el siguiente apartado 4.2 quedará justificado su uso a través de las magnitudes que se van a calcular con ellos para identificar y decidir sobre la imagen.

El punto de partida de la extracción de características es el de tener la imagen completamente preparada y su segmentación realizada para poder hacer los estudios estadísticos necesarios sobre ella. Con esta finalidad, el primer paso será extraer una serie de parámetros de cada uno de los granos de la imagen. Para ello se recurre a una función de Matlab nombrada con anterioridad, capaz de obtenerlos con un coste computacional bastante bajo para la magnitud de sus cálculos. Se trata de *regionprops*, la cual posee una serie de atributos con los que se pueden calcular los diferentes parámetros necesarios para el análisis.

Los parámetros que se van a obtener de las imágenes directamente serán:

a) Imagen silueta del grano: este parámetro se extraerá con la mera finalidad de tener una idea de cómo es la forma de cada grano que se ha segmentado. Se trata de una imagen en cuyo interior se encuentra inscrito el grano, por ello su tamaño será variable en las diferentes muestras que se van a mostrar posteriormente.
Esta imagen binaria se obtendrá a través de la función de la propiedad de *'Image'* y se procesará únicamente en los scripts dedicados a la obtención de gráficas para la presente memoria.

b) Área de grano: este parámetro se extraerá con la finalidad de saber en qué punto se encuentra el proceso de la cristalización del azúcar, pues como se ha argumentado en el comienzo del trabajo, el volumen del grano crece desde que se introduce hasta que se da por concluido este proceso. A través del área del grano, se podrá saber si es momento de para el proceso, si ha habido algún error o si se tiene que seguir con él.

Sin embargo, no será con este parámetro con el que se trabajará, sino que será con el área convexa, parámetro explicado y argumentado en los próximos epígrafes.

Este parámetro del área del grano se obtendrá como un número escalar que indica cuantos píxeles contiene la región que se ha identificado. Se podrá leer gracias a la propiedad '*Area'* y será usada dentro de la función destinada a la obtención de parámetros y estadísticas de la imagen.

c) Imagen silueta convexa del grano: junto con la imagen de la silueta del grano, esta propiedad también cumple la mera labor de aportar una idea del porqué se ha decidido trabajar con el área convexa en lugar de con el área tal cual se obtiene a través de '*Area'* e '*Image'*. En apartados posteriores se podrá ver representada las diferencias existentes entre la imagen del grano inscrito tal y como se obtiene tras la segmentación y esta imagen dada a través de su forma convexa.

Esta imagen binaria se obtendrá a través de la función de la propiedad de *'ConvexImage'* y se procesará únicamente en los scripts dedicados a la obtención de gráficas para la presente memoria.

d) Área convexa del grano: tal y como se comentó en el apartado b) será esta la propiedad con la que se trabaje primordialmente por la naturaleza de la segmentación de los granos. En la amplia mayoría de casos que se observa, gracias a la utilización de este área en lugar de su área original, el grano se identifica mejor con su forma convexa, es decir, se asemeja más al original captado por la cámara. Por este motivo, se decide tomar como datos para el seguimiento el área convexa, que no es más que el número de píxeles que forman la imagen del área convexa.

En las siguientes imágenes se puede apreciar de forma gráfica por qué se ha decidido trabajar con el área convexa en lugar que con la normal.

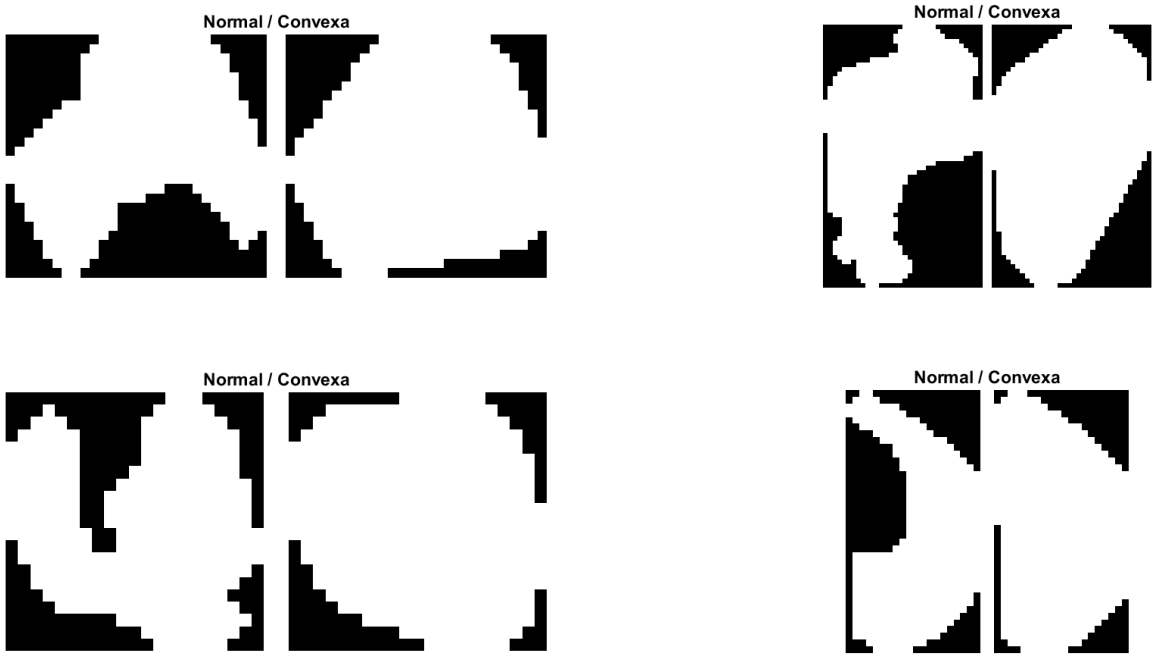


Ilustración 4-1 Gráficas de granos según su área reconocida y su convexa.

Este parámetro será estudiado para el seguimiento del proceso como ya se ha explicado, pero también tendrá un papel fundamental en la decisión de si el grano debe ser descartado por deformidad o no, ya que es con él con lo que se calculará la solidez de la región, parámetro explicado en un epígrafe posterior.

Este parámetro del área del grano se obtendrá como un número escalar que indica cuantos píxeles contiene la región que se ha identificado. Se podrá leer gracias a la propiedad '*ConvexArea'* y será usada tanto dentro de la función destinada a la obtención de parámetros y estadísticas de la imagen, como para la generación de gráficas para poder saber de forma visual en qué punto de crecimiento se encuentra el proceso.

e) Diámetro Equivalente: consiste en un parámetro que relaciona la región reconocida con un círculo y de esta forma calcula lo que sería su diámetro en el caso de que la región fuera de forma circular con idéntica área. Este parámetro aportará una estandarización de las regiones a pesar de sus diferentes formas geométricas, haciendo más sencillo su estudio en conjunto.

La finalidad dentro del estudio trata de saber cómo se asemeja cada una de las regiones reconocidas a un cuadrado o rectángulo. Esta estrategia es la que se sigue en el artículo de Farias ya citado [14] para reconocer errores, ya que no se espera que el grano tenga una forma rectangular muy pronunciada, sino que se espera de él una forma cuadrada o circular (debido al procesamiento de la imagen y a la consideración de su área convexa).

Por este motivo, este parámetro será decisivo junto con el máximo diámetro de Feret para determinar si ha de ser un grano considerado en el cómputo total o por el contrario habría que descartarlo. Esta relación se describirá tras presentar dicho parámetro a continuación.

El diámetro equivalente se obtiene de cada región a través de la propiedad *'EquivDiameter'* y consiste en el número de píxeles que componen el diámetro del círculo con la misma área que la región reconocida.

f) Diámetro máximo de Feret: como se ha adelantado, la motivación por el uso de este parámetro viene dada por su interés a la hora de estudiar la forma geométrica aproximada del grano. Este diámetro forma parte de una serie de diámetros que se definen en función del ángulo, quedando descrita perfectamente la forma del objeto que se estudia. El diámetro máximo es el mayor de ellos, obtenido generalmente junto con su ángulo de ocurrencia, parámetro que no va a interesar en el presente trabajo. Con este parámetro se podrá tener una idea bastante aproximada de cuál será el calibre de cada uno de los granos en cada momento del proceso.

Tal y como propone Farias [14], el estudio de este parámetro junto con el diámetro equivalente puede arrojar información acerca de la forma del grano gracias a la relación Diámetro Máximo de Feret / Diámetro equivalente (F_{max}/D_{eq} en adelante). Más adelante se detallarán las consideraciones adoptadas sobre esta relación y la implicación que tiene en el descarte de granos de la imagen.

El diámetro máximo de Feret se obtiene de cada región a través de la propiedad '*MaxFeretProperties'* y posteriormente accediendo a él en el campo de la estructura devuelta como '*MaxFeretDiameter'*. Consiste nuevamente en el número de píxeles que componen el diámetro del diámetro máximo presente en la región.

g) Solidez: consiste en la relación existente entre el área reconocida de la imagen binaria pasada como argumento de entrada, y el su área convexa tal y como se ha explicado su respectivo apartado. Esta relación arrojará como resultado un número decimal que puede ser indicador de si se ha cometido algún error en la segmentación y si ese grano debe ser descartado o no en el estudio.

Este parámetro se obtendrá a través de la propiedad *'Solidity'*, devolviendo de este modo la relación citada en tipo *double* de cada región.

Estos serán todos los datos que se extraerán de cada imagen leída y segmentada. El siguiente paso consiste en interpretarlos, pero para ello habrá que cerciorarse de que se están leyendo las imágenes correctamente. Pueden darse casos de errores tanto en la lectura, como en la binarización o en la segmentación, por ello, se necesita un paso adicional a este proceso de extracción de características.

4.2 Comprobaciones de los parámetros extraídos de la imagen

Este apartado está destinado al estudio de cada parámetro extraído de cada región de la imagen con intención de concluir si la región que se está interpretando como grano realmente lo es, o bien identificar si se ha producido un error en la binarización de la imagen y por ello habrá que descartar la inclusión de la imagen en el estudio.

La motivación de esta eliminación tanto de elementos de la imagen como de la imagen en sí no es otra que la de prevenir errores en la interpretación y por ello la de tomar decisiones erróneas sobre el proceso. De esta forma se decide que es mejor prescindir de ciertos datos de entrada antes que obtener datos de salida que no se ajustan a la realidad del proceso.

Una vez definidos todos los parámetros que se van a obtener de cada imagen, se procede a describir las comprobaciones que se realizan sobre ellos con el fin de descartar o aceptar la región analizada y la imagen. Se presentan de forma esquemática:

1) Comprobación de la solidez
2) Comprobación de la relación de Farias
3) Comprobación del área
4) Comprobación del valor máximo del Test de Kolmogorov

A continuación, se detallará cada una de las comprobaciones, especificando la motivación de su uso, forma de actuar y valores límite.

4.2.1 Comprobación de la solidez

La solidez consiste en la relación existente entre el área leída de la imagen segmentada y el área convexa calculada por la función *regionprops* tal y como se detalló en el epígrafe g) del apartado 4.1.

Se entenderá que se ha producido un error en el reconocimiento del grano si esta relación entre las diferentes áreas sobrepasa al mínimo impuesto en una tercera parte. La interpretación de este mínimo es que, si el área convexa calculada se hace mayor más de tres veces que el área extraída, se va a considerar que el grano no ha sido reconocido correctamente. Este descarte de grano se llevará a cabo a través de una bandera que indique en cada iteración del bucle encargado de recorrer la estructura donde han sido extraídos en bruto las características de los granos, si el grano ha sido descartado por exceder los valores límite impuestos. El código correspondiente a esta comprobación tendrá la siguiente estructura:

Código para comprobar la solidez en Matlab

```
granoDescartado = 0;
…
solidezMinima = 0.65;
if (solidezGrano < solidezMinima)
  numeroDescartes = numeroDescartes + 1;
  granoDescartado = 1;
end
```

A su vez, se tendrá adaptado el código para que en caso de que el elemento se descarte, sus características no sean almacenadas en la variable destinada a este fin.

Este valor límite ha sido estudiado a través de la observación del comportamiento de los granos a través de diferentes valores de su solidez, determinando que, estando este parámetro a menos de 0.65, ya se incurre en ser una región nada semejante a un grano y por ello debe ser descartado del estudio.

Se procede a mostrar algunos ejemplos de las regiones reconocidas, determinando en cada caso su solidez, área segmentada y área convexa estimada. Estos parámetros mostrados son los extraídos previamente con las propiedades *'Solidity'*, *'Image'* y *'ConvexImage'* respectivamente.

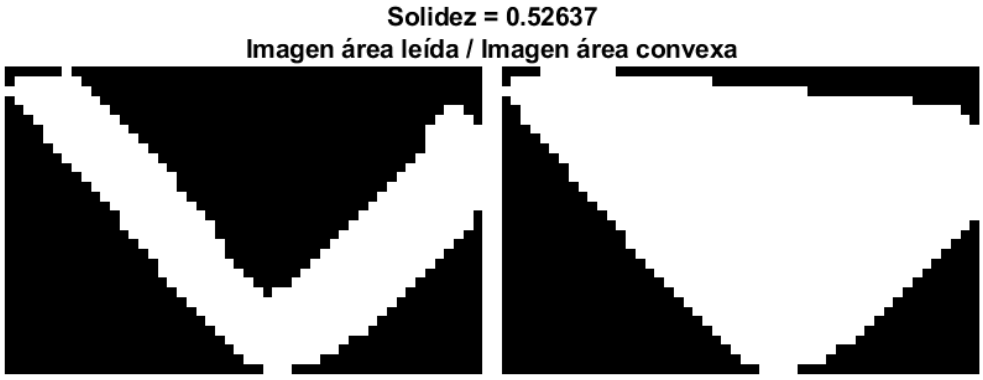


Ilustración 4-2 Grano con una solidez muy inferior al mínimo

Aquí se puede apreciar que, con la solidez muy baja, se pierde el concepto de grano como tal y lo que se convierte en área convexa no se asemeja a la forma que debiera tener el grano. Sin embargo, tras un exhaustivo estudio, se llega a la conclusión de que con una solidez superior a 0.65 ya se obtienen muestras que con su forma convexa se obtiene la silueta del grano aceptable.

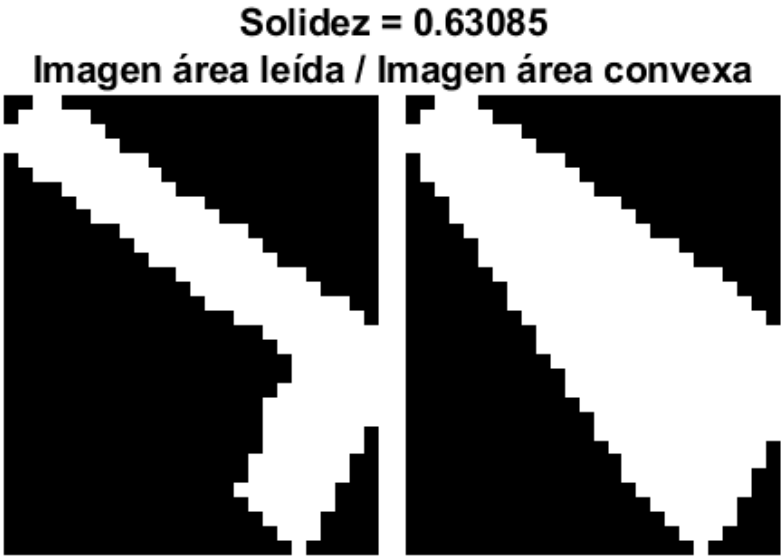


Ilustración 4-3 Grano con una solidez por debajo del mínimo

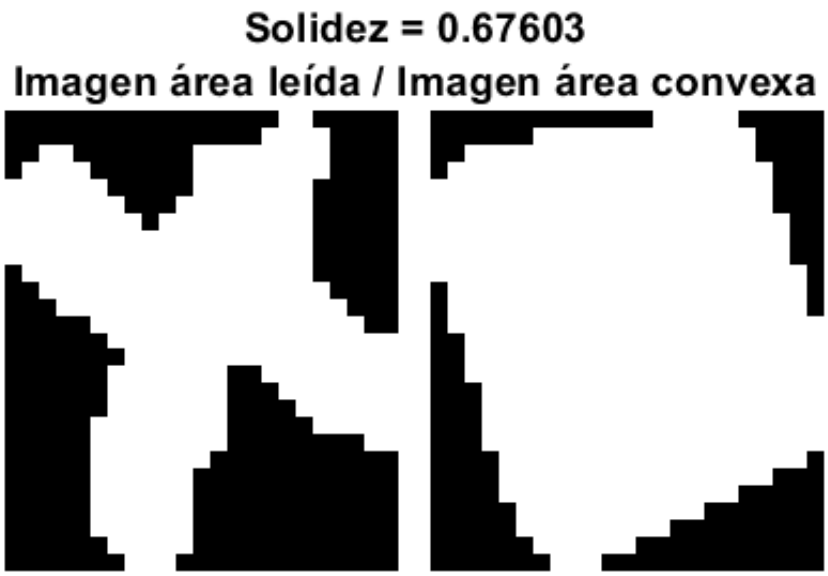


Ilustración 4-4 Un grano con la solidez justo por encima del mínimo.

Se puede observar en las imágenes que el grano se va asemejando a su forma original a medida que se acerca su solidez al mínimo que se ha impuesto.

4.2.2 Comprobación de la relación de Farias

La siguiente comprobación tal y como se adelantó en apartados previos, será la de la relación establecida por el autor Farias en su artículo sobre el estudio de la cuantificación de la morfología de los cristales de sacarosa a través del análisis de imagen [14]. En él se establece que la relación de aspecto siguiente:

$$Aspect\ Ratio\ =\ F_{max}/D_{eq}$$

Con esta relación se podrá determinar la circularidad del cristal de una forma más aproximada que con otros cálculos de esta. Por ello, se descarta el uso de otros parámetros como son las longitudes de los ejes de la elipse o bien directamente la circularidad o excentricidad dadas por Matlab para el estudio de este parámetro. Se considera que las citadas alternativas son más adecuadas para el estudio de la geometría de elementos circulares, y la geometría principal que se esperará del proceso de cristalización del azúcar es cuadrada o rectangular.

Para el estudio de estructura a través de la relación de aspecto, dicho artículo estipula que los siguientes valores de la relación, determinará su geometría de la siguiente forma:

$$F_{max}/D_{eq}\ =\ 1\ \rightarrow\ elemento\ circular$$

$$F_{max}/D_{eq}\ =\ \sqrt{\pi/2}\ \rightarrow\ elemento\ cuadrado$$

$$F_{max}/D_{eq}\ >\ \sqrt{\pi/2}\ \rightarrow\ elemento\ rectangular$$

Gracias a esta relación se determinará como valor deseado $\sqrt{\pi/2}$ y se considerarán como descartados aquellos granos que superen en más de 1.65 su valor. Por ello, para la comprobación de esta relación se tendrá un código como el siguiente:

Código para comprobar la relación de aspecto en Matlab

```
granoDescartado = 0;
…
relacionCuadrado = sqrt(pi/2);
relacionMaxima = 1.65*relacionCuadrado;
if (relacionGrano < relacionMaxima)
  numeroDescartes = numeroDescartes + 1;
  granoDescartado = 1;
end
```

Al igual que con la comprobación anterior, se tendrá adaptado el código para que en el caso que se descarte el grano por alguna de estas dos comprobaciones, sus parámetros no sean almacenados junto con los parámetros de los granos que sí que han sido aceptados.

El valor límite se ha establecido en 1.65 veces superior al esperado de una geometría cuadrada puesto a que se aceptarán granos con geometrías rectangulares no demasiado pronunciadas, es decir, cuyos lados mínimos y máximos no sean excesivamente diferentes.

Se procede a mostrar algunos ejemplos de las regiones reconocidas, determinando en cada caso su forma a través de la propiedad extraída con *'Image'*, *'ConvexArea'* y su relación de aspecto a través de F_{max}/D_{eq} con las propiedades *'MaxFeretDiameter'* y *'EquivDiameter'* respectivamente.

Ilustración 4-5 Grano con una relación de aspecto muy superior a la permitida.

Aquí se puede apreciar que, para una relación de aspecto bastante elevada respecto del valor citado para su geometría cuadrada, se tiene un grano que se consideraría erróneo. Ajustando este parámetro, se concluye en fijarlo para una relación con respecto a la proporción del aspecto cuadrado 1.65 veces. Por ello se pueden pasan a aceptar granos con las siguientes geometrías:

Ilustración 4-6 Grano con una relación de aspecto ligeramente superior a la permitida.

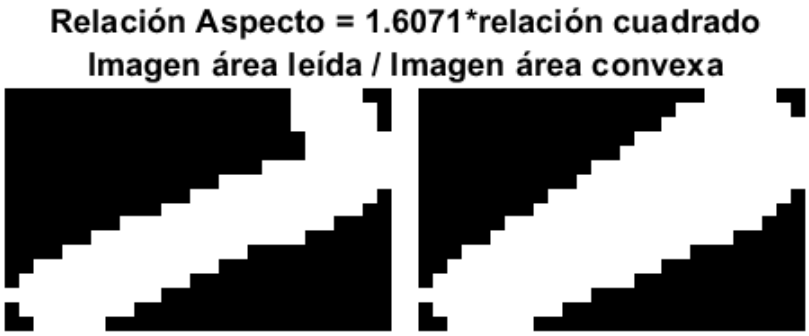


Ilustración 4-7 Grano con una relación de aspecto dentro de la permitida.

Debido a los resultados que se obtienen en las ilustraciones anteriores, se argumenta la fijación del máximo de este parámetro en 1.65.

4.2.3 Comprobación del área

Esta será la tercera comprobación realizada sobre los parámetros extraídos de la imagen y una de las más importantes, pues es capaz de detectar si se ha producido un error en la binarización o en la segmentación. La motivación de su aplicación es la de detectar este tipo de errores pues al producirse, las regiones reconocidas por granos serán gran parte de la imagen, frente a un correcto reconocimiento que supondría que un grano ocupará una parte prácticamente despreciable del total de área.

En este caso, al fallar en esta comprobación, en lugar de descartar tan solo el elemento que se estudia se procederá a descartar la imagen por completo, ya que si se produce el fallo se ha producido en la binarización, reconociendo esta por grano gran parte del fondo de la imagen.

Para la implementación de esta comprobación se ha establecido como área máxima aceptada una treintava parte del total de área de la imagen, ya que se pueden producir desbordamientos de las regiones no solo del fondo entero, sino también de un grupo de elementos reconocidos como uno solo. En este caso, el indicador que se utilice no será la que descarte el elemento, sino una diferente encargada de que no se almacene ninguno de los datos leídos de esa imagen fallida. Se podrá realizar dicha comprobación gracias a un código como el siguiente:

Código para comprobar el área en Matlab

```
imagenDescartada = 0;
…
[M,N] = size(imagenOriginal);
areaTotal = M*N;
areaMaxima = areaTotal/30;
if (areaGrano > relacionMaxima)
  imagenDescartada = 1;
  break;
end
```

Como se puede apreciar, se recurre al uso de la sentencia `break`, que actuará en caso de que se haya producido el caso de estudio saliendo de la ejecución del bucle encargado de recorrer cada uno de los elementos reconocidos en la imagen para ahorrar tiempo de procesamiento. En el caso de que la imagen sea descartada, los datos que se pasarán no serán los leídos, por ello la bandera usada para descartar la imagen será la que determine qué datos se van a transmitir. Esta estructura será la que se detallará en el siguiente apartado.

El valor del área máxima respecto al total de la imagen ha sido obtenido a través del estudio del comportamiento de la binarización y segmentación de las imágenes, con especial importancia en los momentos iniciales y finales del vídeo, que se tiende a fallar en el reconocimiento de regiones por parte de la segmentación, por encontrarse en el primer caso enfocado únicamente un grano de gran tamaño y en el segundo, por encontrarse los granos altamente aglomerados y juntos.

Se procede a mostrar algunos ejemplos de las regiones reconocidas, determinando en cada caso su forma a través de la propiedad extraída con *'ConvexArea'* representando la imagen completa con su relación con el área total de la imagen.

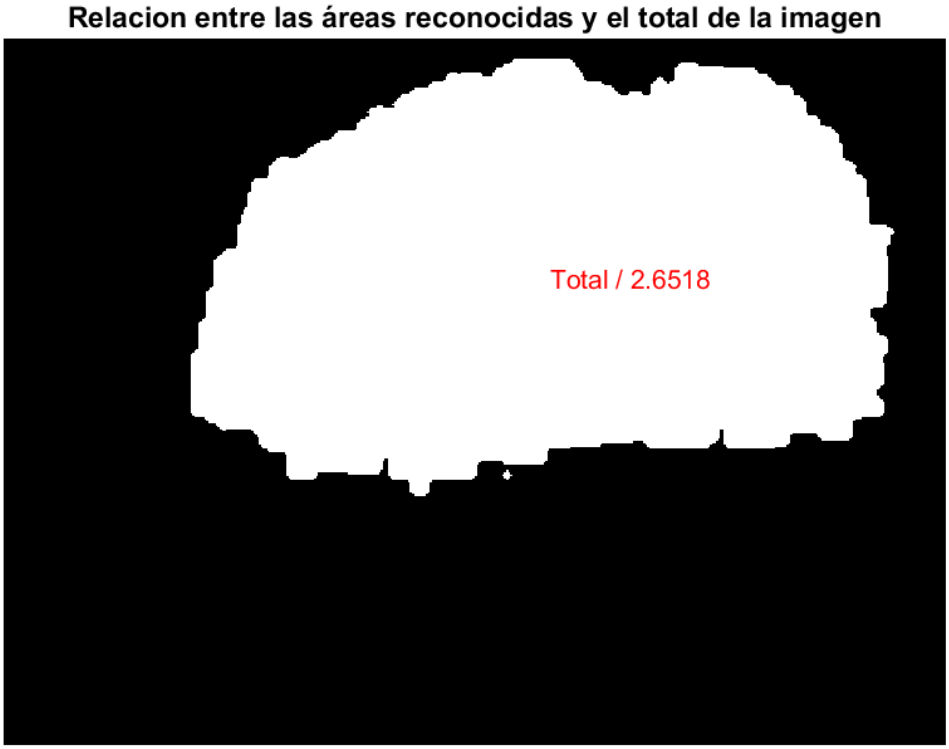


Ilustración 4-8 Ejemplo de desbordamiento de la binarización

En esta primera imagen se puede apreciar el caso que se da al principio del vídeo, en el cual se comete un error en la binarización de la imagen y se reconoce el fondo como grano, teniendo un área de menos de la mitad del área total de la imagen.

Para ajustar el parámetro se han ido estudiando este tipo de imágenes, llegando a la conclusión de que fijar el área máxima aceptada a 30 veces más pequeña que su total puede filtrar los casos de error en la binarización y segmentación sin afectar a los granos como tal.

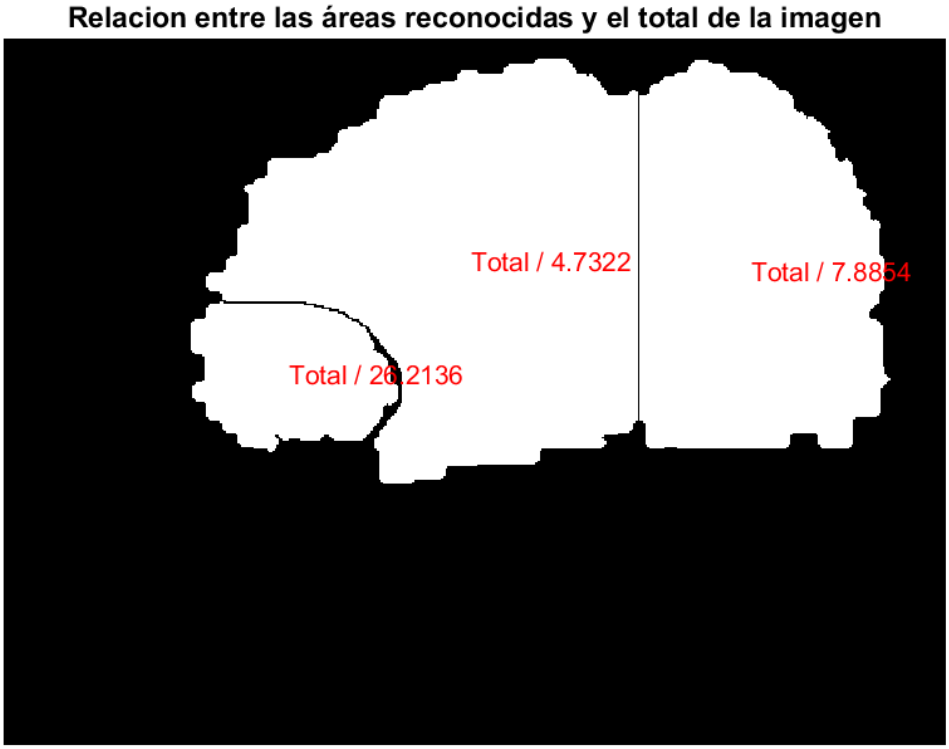


Ilustración 4-9 Ejemplo de desbordamientos de la binarización y posterior error en la segmentación.

4.2.4 Comprobación de los valores máximos del Test de Kolmogorov

El Test de Kolmogorov-Smirnov será un estudio realizado sobre los valores del calibre de los granos de la imagen que se explicará en el apartado 4.4.1 con mayor grado de detalle. Esta comprobación en este punto del algoritmo se realiza con la intención de descartar imágenes que distan de ser distribuciones gaussianas tal y como se espera de la distribución de los calibres del grano en la imagen durante su proceso de cristalización.

Este valor máximo del test se obtiene en términos relativos a cada imagen. Tras el estudio de los valores obtenidos en las diferentes imágenes, se fija el umbral máximo en 50%, haciendo referencia a ese término relativo. De la misma manera, se descartará una imagen si su valor es '*NaN*' (Not a Number) porque se haya producido un error a la hora de ejecutar el test, bien por errores en la segmentación o bien por que la distribución de los calibres de la imagen sea tan diferente a una gaussiana que no sea capaz de asemejarla.

4.3 Estructura de los datos extraídos de la imagen

Una vez pasadas las comprobaciones presentadas, los datos tendrán que ser almacenados en una variable general donde estén todos estos datos leídos de cada una de las imágenes que se vayan procesando. La idea principal de esta estructura es poder acceder a ellos tanto obtener parámetros de la imagen actual que se está procesando en el momento como para tener una visión global de las imágenes anteriores leídas y así poder saber de otra forma si el proceso está funcionando como se esperaba o si por el contrario se detectan irregularidades que puedan llegar a provocar el aborto de este.

4.3.1 Necesidades de la estructura

El procesamiento de imágenes a menudo arroja datos con una considerable complejidad a la hora de almacenarlos, los requisitos más relevantes para el presente trabajo son los siguientes:

Requisito 1: Ejecución en tiempo real

El objetivo principal del trabajo es que pueda seguir el proceso de la cristalización en tiempo real, por ello, no se tendrá de antemano el número total de elementos a almacenar. Este aspecto hará que su tamaño sea dinámico y que sea la propia ejecución del programa la que vaya creando su dimensión.

Requisito 2: Diferente número de elementos estudiados en cada imagen

Otra característica para tener en cuenta y que es la más restrictiva a la hora de decidir sobre el tipo de dato es que no se va a tener el mismo número de granos en una imagen que en la siguiente. Por este motivo queda descartado el tipo de dato más usado y sencillo de usar en Matlab, las matrices multidimensionales, pues estas requieren que se tenga en mismo número de elementos por filas que por columnas. Se baraja la opción del relleno con ceros y la redimensión de esta a medida que se necesite, pero se descarta por su elevado peso computacional y dificultad.

Requisito 3: Etiquetado y acceso

Por último, se considera útil la característica de poder tener una cabecera en el tipo de variable por dos motivos: saber en todo momento de qué dato se trata cada dato de la variable y poder acceder a ellos sin necesidad de saber en qué índice se ha puesto de la variable. Como consecuencia de lo expuesto anteriormente, se descarta el tipo de dato *cell* que ofrece Matlab que, a pesar de solucionar el requisito de la diferencia de tamaños de los datos entre imágenes, no ofrece la etiquetación de los campos, sino que se accede a ellos por índices de una forma ordenada.

4.3.2 Elección del tipo de dato y forma de acceso

Se barajan tres tipos de datos fundamentales para almacenar las características extraídas de la imagen:

a) **Matrices multidimensionales:** quedan descartadas por el segundo requisito debido a que en su uso deben quedar con el mismo número de elementos las matrices almacenadas en cada plano de la matriz. Como este requisito no se va a poder respetar y la alternativa del relleno con ceros no es viable a la hora de implementarlo en el código y ofrecer una rápida respuesta tal y como exige el problema, quedan descartadas las matrices multidimensionales como estructura para el almacenamiento de datos.

b) **Array de celdas (*cell*):** este tipo de datos resuelve el problema presentado en el requisito 2º, puesto que en cada una de sus celdas se puede almacenar una matriz con un tamaño que no tiene por qué corresponderse con el tamaño de sus celdas vecinas. Sin embargo, los arrays de celdas no resuelve el tercer requisito, pues los datos en ellos se tienen que almacenar por orden en el índice correcto. Esta característica hace que no se trate del tipo de dato óptimo para el trabajo, pues tanto para leer los datos como para escribirlos se requerirá el conocimiento del lugar exacto en el que se han escrito, pudiendo provocar confusiones y errores en su tratamiento.

c) **Array de estructuras (*struct*):** tras un estudio exhaustivo de los tipos de datos que ofrece Matlab, se concluye que este tipo de datos es el adecuado para la aplicación que se requiere. Entre sus características más destacadas se encuentra sus acceso a través del nombre del campo, independientemente del orden en el que se haya creado. Gracias a estos campos, se podrán nombrar de forma intuitiva para poder acceder a ellos sin complicación.

   Sin embargo, para su lectura en bloque, por ejemplo, para leer los números de granos descartados en todas las imágenes, no se ofrece por parte de Matlab ninguna opción directa. Por el contrario, habrá que crear un bucle que recorra todos los campos de granos descartados, para cada una de la imágenes leídas hasta el momento.

4.3.3 Organización de los campos

Tras la elección del array de estructura como tipo de dato preferente para el almacenamiento de los parámetros extraídos de las imágenes, se procede a desglosar su estructura interna y así saber cómo se van a almacenar los datos en ellas.

Su tamaño será dinámico, teniendo tantas estructuras dentro de ella como imágenes se hayan procesado. Por el contrario, sus campos serán estáticos, pues una vez se ejecuta la extracción de características de la primera imagen, quedan determinados para el resto de la ejecución del programa. Estos campos se definen bajo los siguientes nombres, siendo indiferente para su acceso el orden de su creación.

a) Estado: se trata de un valor entero (en Matlab tratado como *double*) que podrá ser 0 o 1.

- El estado a 0 indica que no se ha producido error en la extracción de características de la imagen, y por ello los datos almacenados a continuación (a excepción del total de granos) deben ser almacenados y considerados en futuros estudios.
- El estado a 1 indica que se ha producido un error en la binarización y segmentación de la imagen, y que por ello esta imagen no debe ser considerada en el cómputo total estadístico.

Por ello, los datos puestos en el resto de los campos no son los esperados ni los que se deben tratar. Se decide poder a -1 los campos de las estructuras correspondientes a las imágenes clasificadas como erróneas para identificarlas fácilmente del resto en el caso de que se requiriera.

b) TotalGranos: será un número entero (en Matlab tratado como *double*) conteniendo el número total de elementos reconocidos por la función *regionprops*. Este será el único campo que independientemente del valor del estado, contendrá un valor real sobre la imagen.

c) Aceptados: nuevamente se trata de un número entero (en Matlab tratado como *double*) que indicará el número de elementos de los reconocidos en primera instancia que han pasado las comprobaciones y por tanto se tendrán en cuenta para la obtención de decisiones y estadísticas. Este número se irá aumentando en una unidad cada vez que un elemento pase las comprobaciones, a su vez servirá de índice para los vectores de datos que se incluirán en los diferentes campos. De esta forma, habrá tantos datos como elementos aceptados y no tantos como elementos reconocidos desde un principio en la imagen segmentada.

d) Descartados: en este caso, se trata del número entero (en Matlab tratado como *double*) de granos que por no cumplir la comprobación de su solidez o de la relación de Farias, han quedado descartados del cómputo total de granos.

e) Areas: es un campo compuesto por un vector de números tipo *double* donde se encuentran almacenadas las áreas de cada elemento aceptado como grano de la imagen. Como se dijo en su correspondiente apartado, este número representa el número de píxeles que forman el grano de azúcar.

f) Solidez: se compone por un vector de número tipo *double* donde se almacenan estas relaciones del área y el área convexa de cada uno de los granos de la imagen.

g) RelacionFarias: se trata de un campo destinado a almacenar las relaciones de aspecto tal y como se ha detallado en formato vector de números *double*. A cada grano de la imagen se le calcula este parámetro y si tras ser comprobado, se acepta, se almacena en este campo en su respectiva celda.

h) Feret: se corresponde con un vector de tipos *double* donde se guardan los máximos diámetros de Feret que posteriormente se usarán como calibre del grano, puesto que como no hay una magnitud que se considere directamente como calibre, se considera que es este diámetro el que más se le asemeja.

i) TestKolmogorov: se corresponde con un valor tipo *double* que se corresponde con el resultado del test de Kolmogorov. Este test se detallará más adelante y se argumentará su uso en el trabajo. Dentro de su grupo no se usará como factor que determine si la imagen debe o no debe ser descartada, ya que se trata de una medida de la fidelidad de la distribución de la imagen con una función gaussiana.

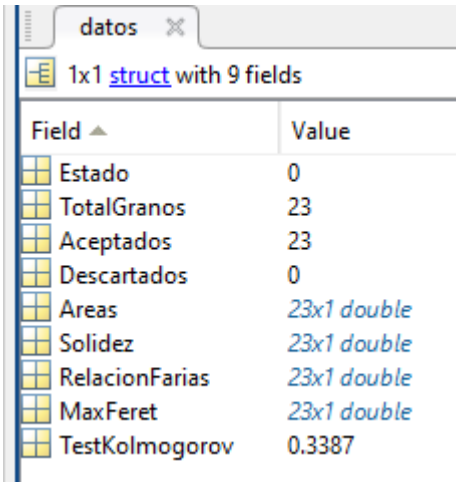


Ilustración 4-10 Captura de la forma que tendrá la estructura en la ejecución del programa Matlab.

4.4 Validación estadística de las características de cada imagen

Una vez que se obtienen las características de cada imagen, se comprueban y se presentan en una estructura de datos, se procede a trabajar con estos datos para saber si el proceso se encuentra en el punto que debe o si, por el contrario, se está produciendo algún error en él.

Para tomar estas decisiones se tendrán los siguientes estudios estadísticos sobre los datos de cada imagen como tal en conjunto con el resto de los datos ya obtenidos de imágenes anteriores. De esta forma, se podrá tener fácilmente una visión global de lo que se lleva de proceso gracias al almacenamiento de datos de las imágenes, y no las imágenes como tal.

Gracias a este cambio de dominio, del de las imágenes con gran cantidad de peso en datos de cada píxel, al dominio de los datos de las imágenes que más interesan en forma de estructuras, se tiene una gran facilidad para trabajar con todas las imágenes leídas hasta el momento, ya que se puede acceder a los datos de interés de forma ordenada y rápida ya que no tiene gran peso como una imagen completa.

4.4.1 Relación de las magnitudes obtenidas de la imagen con las reales

Este apartado precede a los estudios realizados sobres las imágenes ya que en ellos se hará uso de un significante cambio de escala de los datos leídos. Este cambio consiste en dos aspectos:

a) **Medida temporal:** del vídeo, cada vez que se lee una imagen y esta es aceptada para su tratamiento (cumple las expectativas de error de sus magnitudes) el tiempo en el que el frame es leído se almacena en una variable tipo array que asocia inequívocamente el dato leído con la unidad temporal (en segundos) en la que ha sido leído.

b) **Medida del calibre:** cabe destacar que las medidas de los calibres de una imagen, tal y como se ha comentado, se lee en la unidad de píxeles, lo cual no arroja interés en el estudio pues es una medida relativa a la posición de la cámara respecto de los granos y de las características de esta. Por ello, se hace una asociación lineal entre el número de píxeles y los micrómetros en la realidad que estos representan. Esta asociación se hace de forma empírica a través del estudio de la imágenes en diferentes puntos del proceso, sabiendo cuál es su magnitud real.

En cada estudio, se hará uso de la función implementada para este fin de nombre *cambia_escala.m* que trabajará con dos parámetros de entrada: el array de datos a convertir y los límites en el eje X y el eje Y de las magnitudes. Como único parámetro de salida se tiene el array de datos convertidos de la escala de partida a la indicada a través de los valores de los límites de entrada.

En el caso de querer cambiar el calibre, se le pararía como parámetros la variable con los calibres leídos en píxeles y los límites inferior y superior de píxeles de los calibres (11px y 44px respectivamente) y los valores mínimos y máximos en micrómetros del calibre (150µm y 600µm respectivamente).

4.4.2 Estudio de los valores del test de Kolmogorov

La motivación de esta comprobación es la de saber cómo de semejante es la distribución de granos de una imagen a una función gaussiana. Esta comparación se hace puesto que, dentro de una imagen, se espera que haya una mayoría de granos que tengan valores similares al calibre medio y que una pequeña de ellos, tengan valores por debajo y por encima de esta magnitud, obteniendo por lo tanto una distribución altamente similar a la descrita por una función gaussiana.

Por esta razón se elige trabajar con el test conocido por el nombre de *'Prueba de Kolmogorov-Smirnov'* que, en lugar de comparar la distribución con una gaussiana como tal, punto a punto, lo hace a través de las desviaciones en la función de distribución acumulada (Cumulative Distribution Function o CDF) para obtener resultados más precisos que de la forma citada.

4.4.2.1 Interpretación de los datos

Puesto que no se dispone de datos precisos de cómo se espera en cada instante del proceso que sea la distribución de granos esperada, se trabaja haciendo el test para valores de la distribución dada frente a la distribución ideal dada por una gaussiana con la media y desviación calculada de los datos a estudiar. En un futuro, esta comprobación podría verse mejorada por la inclusión de datos reales de la distribución a lo largo del tiempo que se espera del proceso de cristalización del azúcar.

Una vez conocidos cuáles serán los datos que se van a comparar y por ello, a los que se les va a hacer la prueba citada, se procede a detallar el proceso al que se someterán.

4.4.2.2 Pasos seguidos por el test

Para este test, como bien se ha comentado previamente, los datos que se necesitarán serán los calibres de los granos de cada imagen para poder ser estudiados. De ellos, lo que se obtendrá en primer lugar será su media y desviación típica, al igual que en la comprobación gráfica anterior. A su vez, se obtiene el mínimo y máximo valor de esta magnitud para establecer los límites en el siguiente paso.

Se construye una función gaussiana, en adelante la gaussiana teórica, con la media y desviación que se han obtenido de los datos aportados, para los puntos comprendidos entre los valores límites de los datos.

A continuación, se obtiene el histograma de la distribución teórica con un ancho del bin[5] adecuado a la secuencia de datos a estudiar, que no será ni lo suficiente pequeño para obtener poca cantidad de granos por bin ni lo suficiente ancho para no obtener una visión correcta de los datos. El siguiente paso consiste en obtener los datos de la función teórica como una sucesión de sumas acumuladas de sus valores, para ello habrá que multiplicar cada uno de sus valores por el área total descrita ya que la gaussiana se encuentra normalizada.

Por último, la comprobación que realizará el test propuesto será calcular la diferencia en los puntos situados en el centro de cada bin entre el valor arrojado por la función gaussiana teórica y los datos reales de la imagen. El resultado del test será el máximo valor obtenido entre todas estas diferencias. Este valor se encuentra normalizado por el valor total de la integral, por ello los datos a lo largo de las imágenes no tendrán los mismo valores de referencia, sino que la referencia será la de su propio entorno de trabajo: la propia imagen.

[5] Se denomina 'bin' a cada una de las agrupaciones de datos de una gráfica, especialmente las de barras como es el caso del histograma de una imagen. Los datos se agrupan con la intención de tener una visión más clara de la distribución de los datos aportados cuando hay una cantidad considerable de datos a estudiar.

4.4.2.3 **Interpretación de los resultados en cada imagen**

Los resultados que se esperan obtener, tal y como se ha citado, será un número normalizado en cada imagen, pero se puede comprender mejor este test a través de la siguiente gráfica donde se ven el diagrama de barras obtenido de la gaussiana teórica y los valores en los centros de cada bin de los calibres de la imagen. Se muestra esta gráfica para una imagen aleatoria del vídeo real.

La primera gráfica que se muestra se corresponde con la superposición de la gaussiana teórica y los datos sin aplicarles la suma acumulada. De esta forma quedaría:

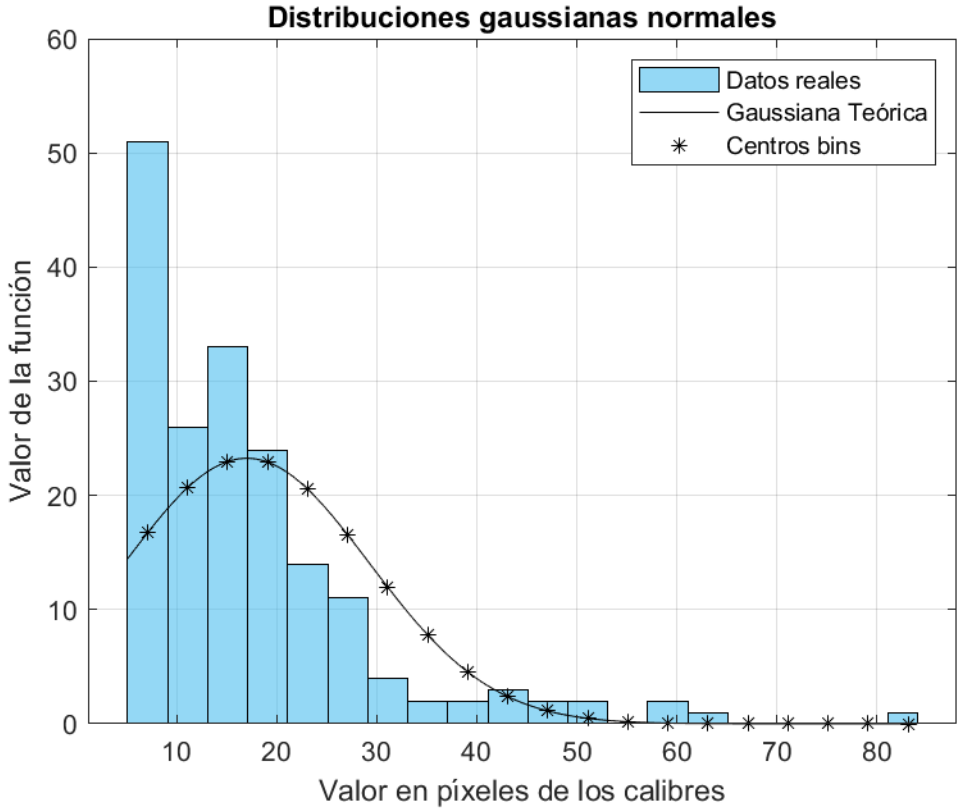


Ilustración 4-11 Gráfica de la distribución gaussiana teórica con la superposición de los datos obtenidos de una imagen.

La siguiente imagen muestra estas mismas distribuciones, pero en su forma acumulada, es decir, cada barra y cada punto mostrará los valores que se van acumulando a cada valor del calibre de los granos.

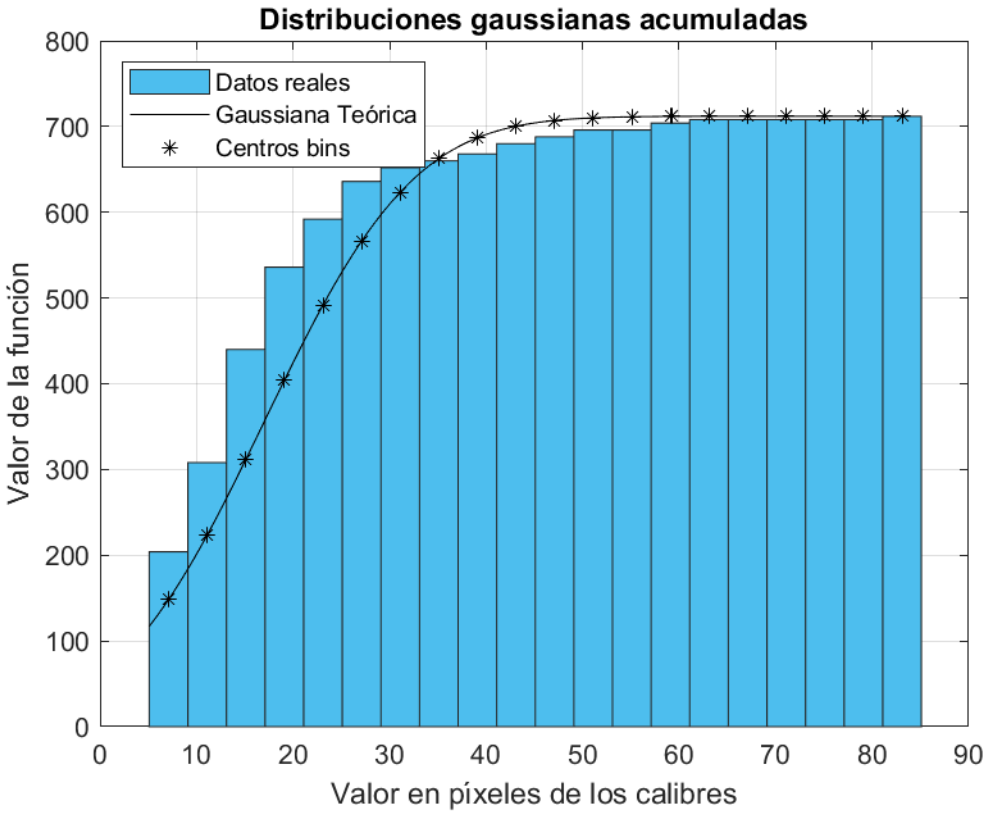


Ilustración 4-12 Gráfica de la distribución gaussiana teórica con la superposición de los datos obtenidos de una imagen, ambas en su forma acumulada.

Por último, se muestra la misma gráfica tras aplicarle la prueba de Kolmogorov-Smirnov y localizar la distancia máxima entre ambas distribuciones, marcada bajo el nombre de '*distKS*' con su valor normalizado con la integral total del área de la gráfica.

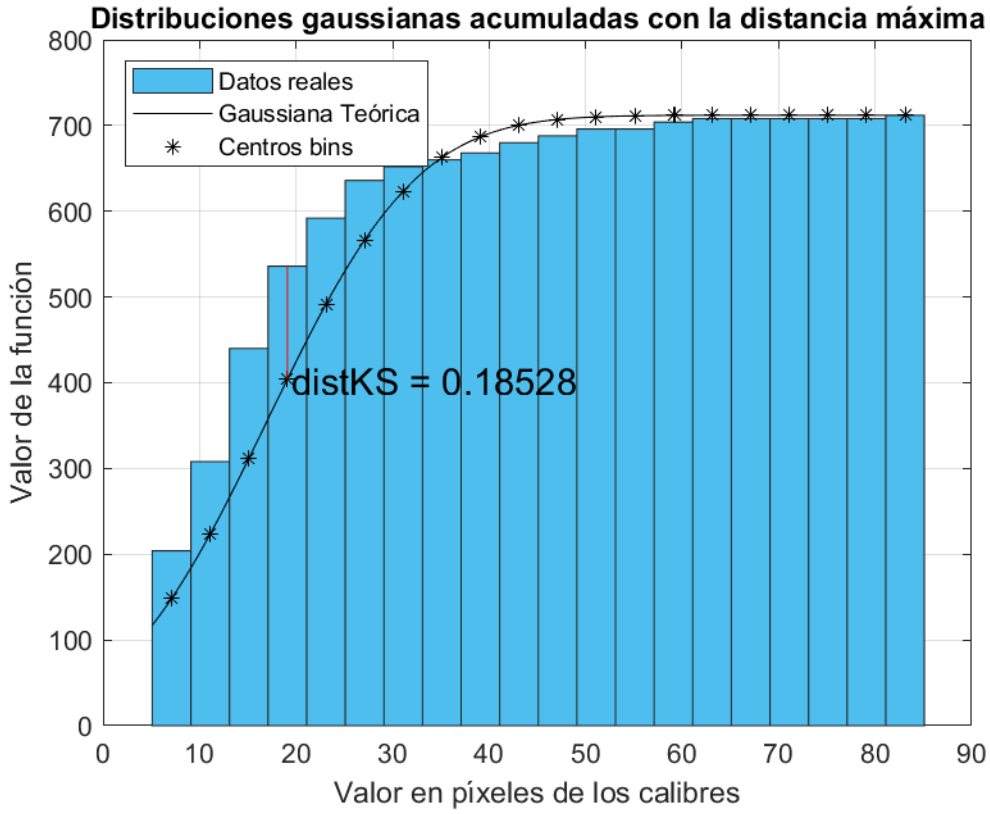


Ilustración 4-13 Gráfica de la distribución gaussiana teórica con la superposición de los datos obtenidos de una imagen, ambas en su forma acumulada, indicando la distancia máxima en el test

4.4.2.4 Uso del test para la detección de outliers

El primer estudio estadístico que se le va a realizar a los datos leídos de las imágenes será el de los valores obtenidos a través del test del Kolmogorov. Para ello se va a establecer un valor de umbral máximo para los valores del test con el objetivo de tener una idea de si esos valores que se han obtenido indican que la imagen está siguiendo el curso esperado, es decir, que la distribución de los calibres de los granos de la imagen se puede interpretar como una distribución gaussiana de media el calibre esperado en el punto del proceso al que corresponde.

Cabe recordar, que los valores arrojados por el test de Kolmogorov que se almacenan como datos en la variable de salida del proceso, son los valores máximos del test en cada imagen de forma relativa a su tamaño, lo cual unifica el estudio independientemente de cuál sea el calibre esperado sobre el que se está trabajando.

Este umbral se fijará con valor 0.3, o lo que es lo mismo, aceptar que el valor máximo permitido máxima distancia del test de Kolmogorov será del 30% del tamaño del calibre. La gráfica que se obtiene de este proceso sería la siguiente:

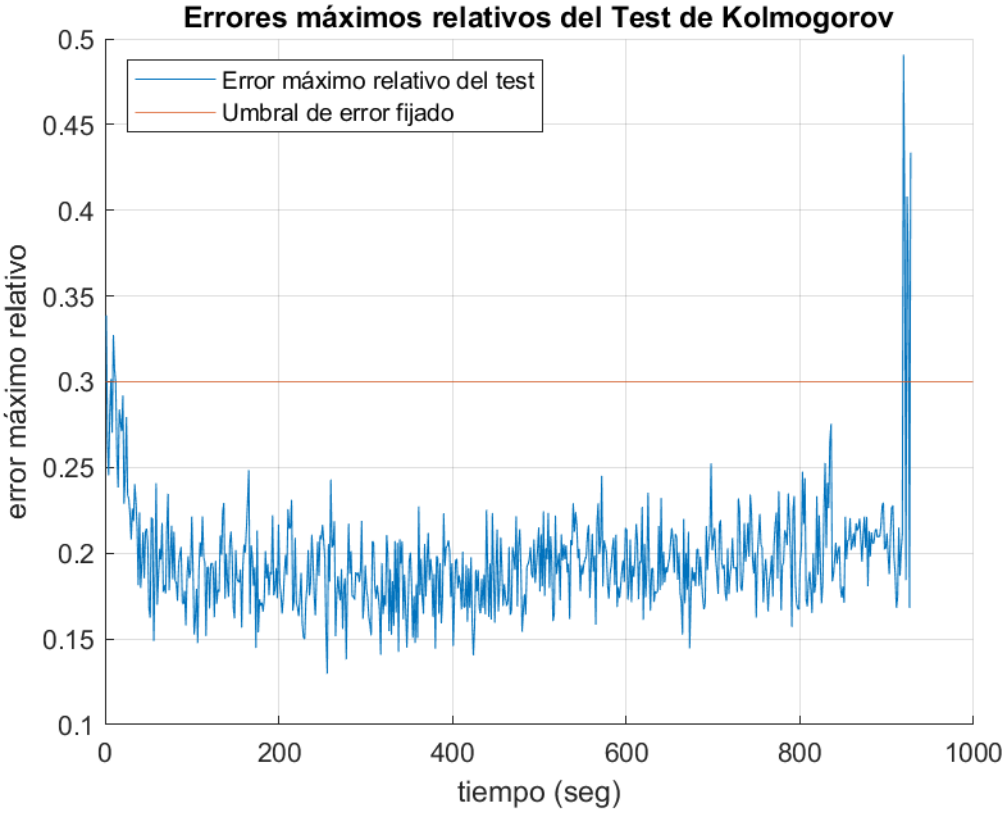


Ilustración 4-14 Errores máximos relativos del test de Kolmogorov

Se puede apreciar como esta distancia máxima entre la distribución gaussiana esperada y la distribución presente en la imagen aumenta en los extremos del proceso, manteniéndose prácticamente constante en los puntos intermedios de este.

4.4.2.5 **Estudio dinámico del error cuadrático medio**

Como medida alternativa del error en el proceso se propone el estudio del error cuadrático medio sobre los valores obtenidos hasta el momento por el test de Kolmogorov para poder ver con más claridad si en el proceso se está acumulando un error que sea preocupante. Como se desea que este estudio pueda ejecutarse en cualquier punto del proceso e indique el error hasta el momento, se hará en cada punto sobre el número de muestras total estudiadas, no sobre el total del proceso, pues se desconoce. Teniendo en consideración este hecho, de esta magnitud de error se espera su comportamiento decreciente, pues a medida que avance el proceso, el número total de muestras sobre el que se pondera será mayor y esto actuará de forma inversa a la magnitud del error.

Para obtener los valores de este error se aplica la siguiente fórmula:

$$ECM = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y} - Y_i)^2$$

Donde el ECM esperado, $\hat{Y}$, será 0 y el vector de datos Y_i el vector con las distancias máximas del test de Kolmogorov.

Dada esta magnitud, se procede a representar el error acumulado de todas las imágenes junto al umbral que se ha estimado con un valor de 0.1 unidades.

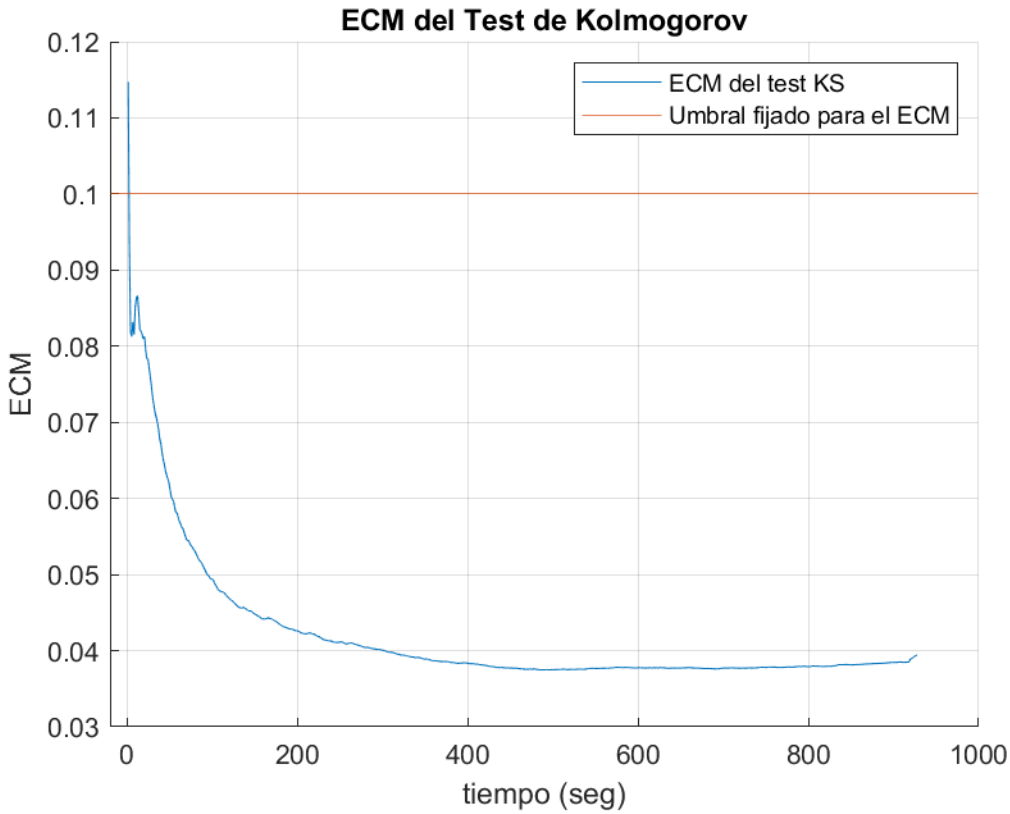
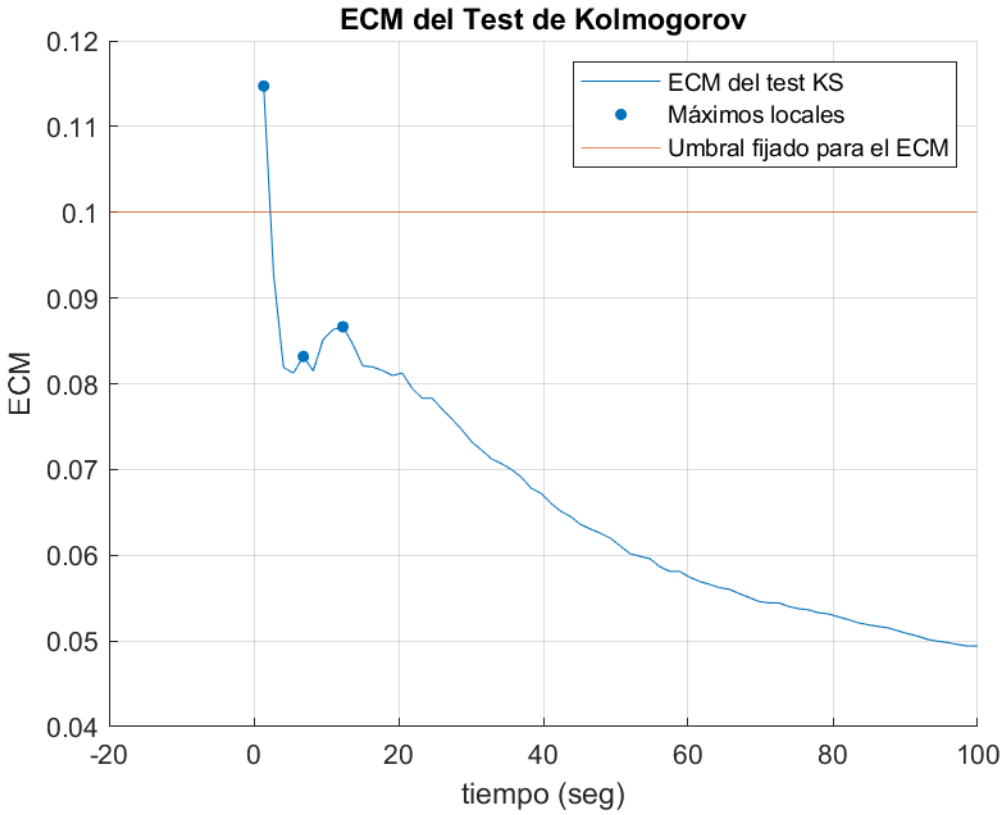


Ilustración 4-15 ECM acumulado a lo largo del proceso.

Con la intención de indicar en qué puntos del proceso se acontecen errores relativos al test realizado que destacan sobre el resto y apreciar que en ellos no se obtienen magnitudes excesivamente elevadas, se muestra la siguiente gráfica donde aparecen marcados a través de puntos los máximos locales más relevantes de la primera parte del proceso. Se centra en este primer tramo debido a que es la región del proceso donde más errores se producen, de forma que es donde únicamente se van a dar estos máximos locales.



.

Ilustración 4-16 Detalle sobre el ECM.

4.4.3 Calibre medio y desviación típica

Se actúa sobre el calibre del grano pues se trata de la magnitud más usada en la industria en términos de medición de granos. En el presente trabajo, como la magnitud calibre no se puede definir exactamente dentro del procesamiento de imágenes, se trabaja con el diámetro máximo de Feret, que no será más que la mayor longitud en píxeles existente en el grano de un extremo a otro. Cabe recordar que este parámetro fue obtenido de cada uno de los granos de la imagen a través de la función *regionprops* con su propiedad *'MaxFeretProperties'* y se encuentra dentro de la estructura de datos en el campo *MaxFeret.*

4.4.3.1 Interpretación de los datos

El objetivo principal de este estudio se basa en mantener una visión general del proceso a través de la magnitud de su calibre. Este estudio tendrá dos componentes, la media del calibre de cada imagen obtenida a través de todos los calibres de los granos de la imagen y, por otro lado, la desviación típica de estos en el conjunto de muestras estudiado.

Con estos dos datos se podrá saber si el proceso está siguiendo el comportamiento esperado, pues se tendrá como referencia para él una curva obtenida de forma experimental a través de los datos obtenidos de procesos estudiados con anterioridad.

A continuación, se detallan las magnitudes que se van a emplear en el estudio:

a) **Calibre medio de los granos de la imagen:** se trata de una magnitud, tal y como se ha dicho previamente, obtenida a través de la media de los diámetros máximos de Feret. Estos diámetros quedan descritos en el apartado Extracción de características de los granos de la imagen4.1 de este capítulo, con la novedad de que en vez de con todos sus valores, se trabaja con el valor medios de ellos en cada imagen. Con esta magnitud se podrá estudiar el tamaño en media de los granos de la imagen para tener una idea general de si el proceso se está efectuando correctamente o no.

b) **Desviación del calibre de los granos de la imagen:** esta magnitud se corresponde al cálculo de la desviación típica estándar de los diámetros máximos de Feret de cada una de las imágenes estudiadas. Esta magnitud se emplea con el fin de averiguar si dentro de cada una de las imágenes se están dando casos de granos de tamaños muy diferentes entre ellos, hecho que arrojaría valores elevados a la desviación y que se estudiará a través de esta magnitud.

4.4.3.2 Obtención de los valores teóricos

Una vez presentadas las magnitudes con las que va a contar este estudio estadístico se procede a detallar el proceso a través del cual se obtienen las magnitudes teóricas con las que se irá comparando el proceso de la cristalización a lo largo del tiempo. Se tendrán dos curvas teóricas obtenidas por el estudio de los valores esperados en cada punto del proceso.

a) **Curva teórica de la evolución del calibre medio:** esta es una curva hipotética que representa cuál debería ser la evolución que siga la magnitud del calibre a medida que el proceso se ejecuta. En el sistema real, se obtendría del análisis detallado previo que se hace del proceso. En este trabajo se adopta una curva que emule las evoluciones encontradas en trabajos publicados por otros autores que sí reflejan evoluciones reales. Esta curva se calcula a través de la expresión matemática: 150 + K*(1-exp(-tiempos/tau));

$$calibre\ inicial\ +\ diferencia\ de\ calibres\ \times\ (1 - e^{-t/\tau})$$

Donde se tiene que el calibre inicial son 150um y el final 600um y por ello la *diferencia de calibres* 450um. La *t* representa al vector de tiempos y τ una constante de valor la tercera parte del tiempo total del proceso.

Estos valores han sido obtenidos tras el estudio de la evolución de los calibres medios en el proceso, con el fin de establecer un modelo matemático que se asemeje a la realidad.

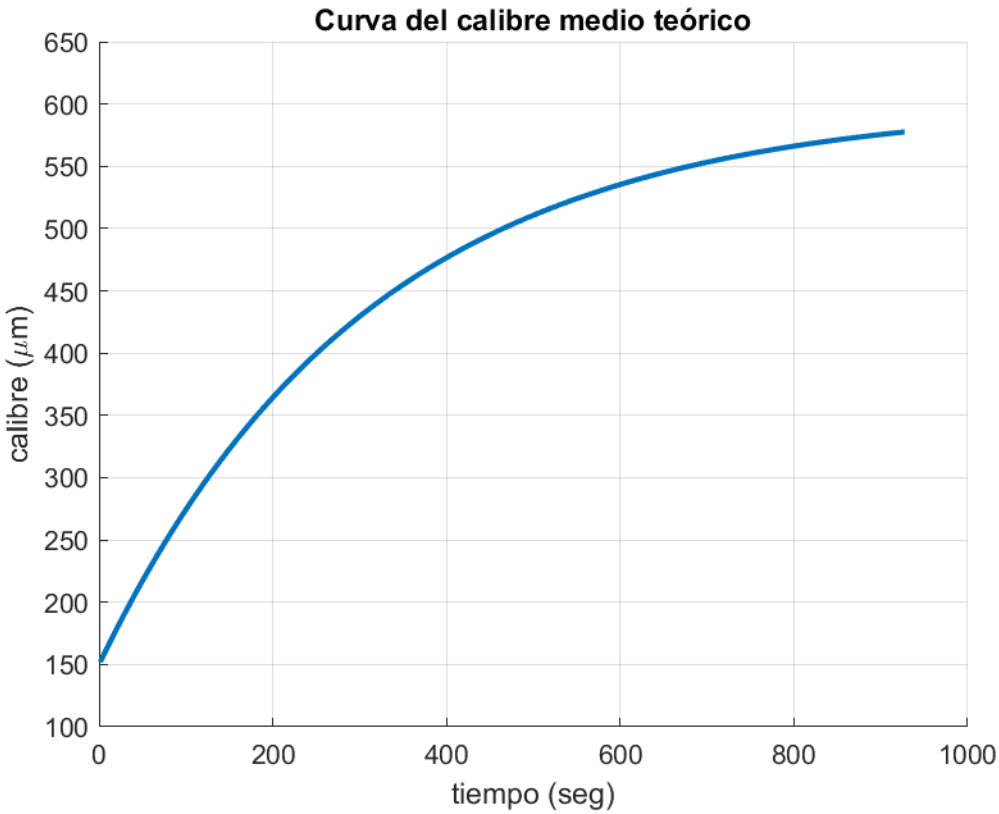


Ilustración 4-17 Calibre medio teórico.

b) **Curva teórica de la desviación del calibre:** para ajustar al máximo esta curva se obtienen de forma empírica tres valores de la desviación típica estándar esperada para el proceso a través del estudio de las imágenes del proceso. Estos tres puntos se sitúan al principio, mitad y final del proceso, pues se entiende que en él no se va a tener un valor de la desviación constante. La curva se obtiene haciendo crecer linealmente la magnitud de la desviación desde un punto a su siguiente. Se aprecia en este estudio, que tal y como cabía esperar, la desviación crece a medida que se desarrolla el proceso, pues a medida que crecen los granos, hay más disparidad de calibres en la imagen.

La siguiente gráfica muestra cómo quedaría esta curva tras realizar la interpolación.

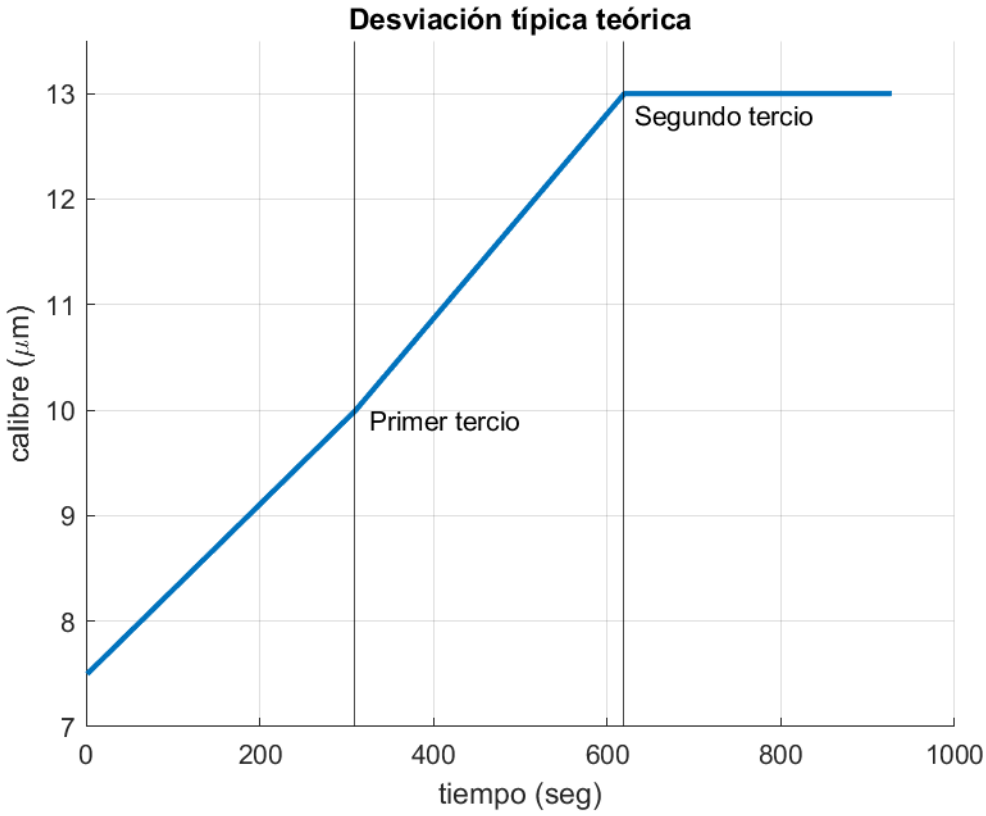


Ilustración 4-18 Desviación típica teórica a aplicar.

Tras calcular estos valores, se le aplica al calibre medio de forma que, en términos de gaussianas, sus colas alberguen hasta el 95% de los valores. Esto equivale a aplicarle 1.64 veces la desviación típica al calibre medio para recoger este porcentaje de valores, quedando la desviación típica de la siguiente forma:

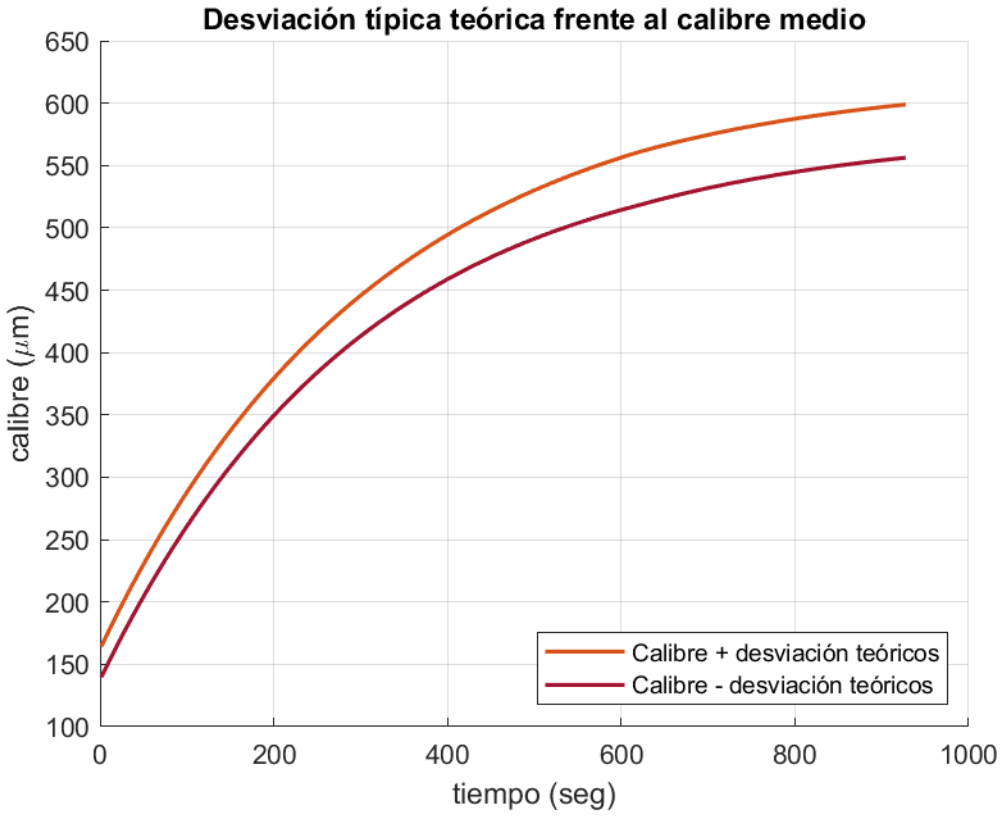


Ilustración 4-19 Desviación típica aplicada al calibre medio

En representación junto con la evolución del calibre medio se obtendría:

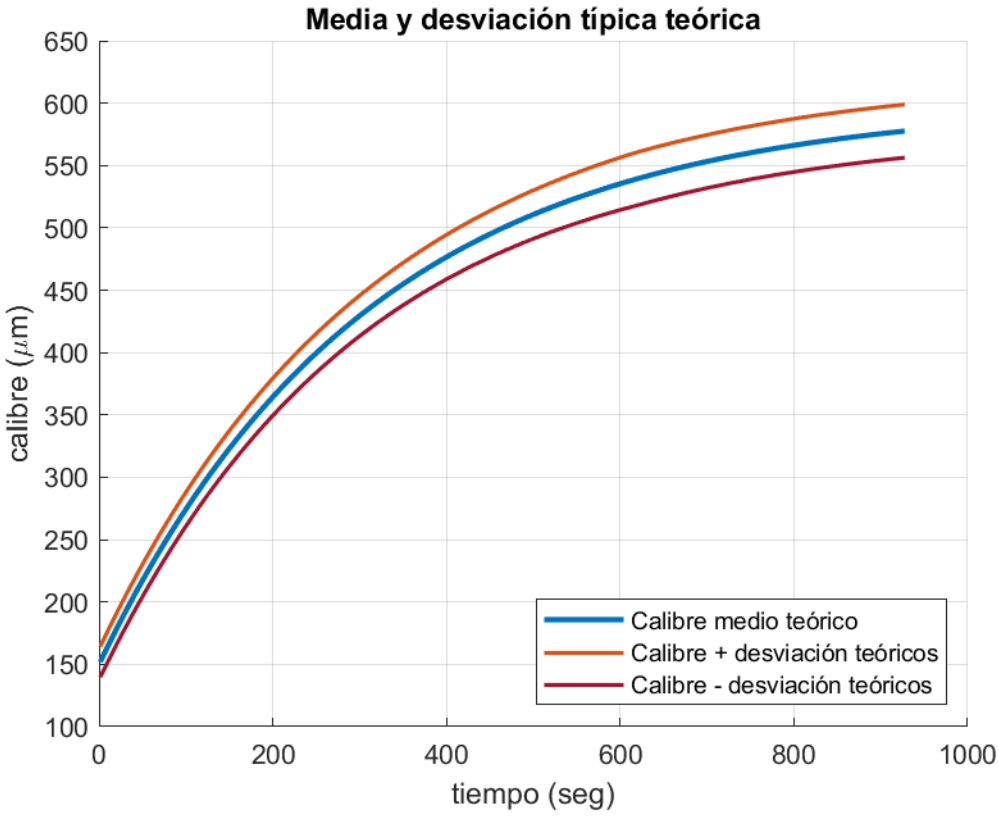


Ilustración 4-20 Media y desviación teórica

4.4.3.3 Obtención de valores experimentales

Una vez establecidas las curvas que se van a usar como referencia, la media y desviación típica del calibre teóricas, el siguiente paso consiste en procesar los datos obtenidos de las imágenes de forma que se pueda interpretar si el proceso está cumpliendo las expectativas.

Para ello, los datos leídos de las imágenes, concretamente los valores del diámetro máximo de Feret, serán transformados través de las funciones de Matlab *polyfit* y *polyval* usadas para obtener un polinomio del grado indicado que represente la curvatura de los datos indicados, en este caso, los calibres medios leídos de las imágenes. Su funcionamiento pasa por construir un polinomio de un determinado grado, con *polyfit* y después evaluar los datos experimentales con él a través de la función *polyval*.

A continuación, se muestran los datos ajustados a través de diferentes grados del polinomio:

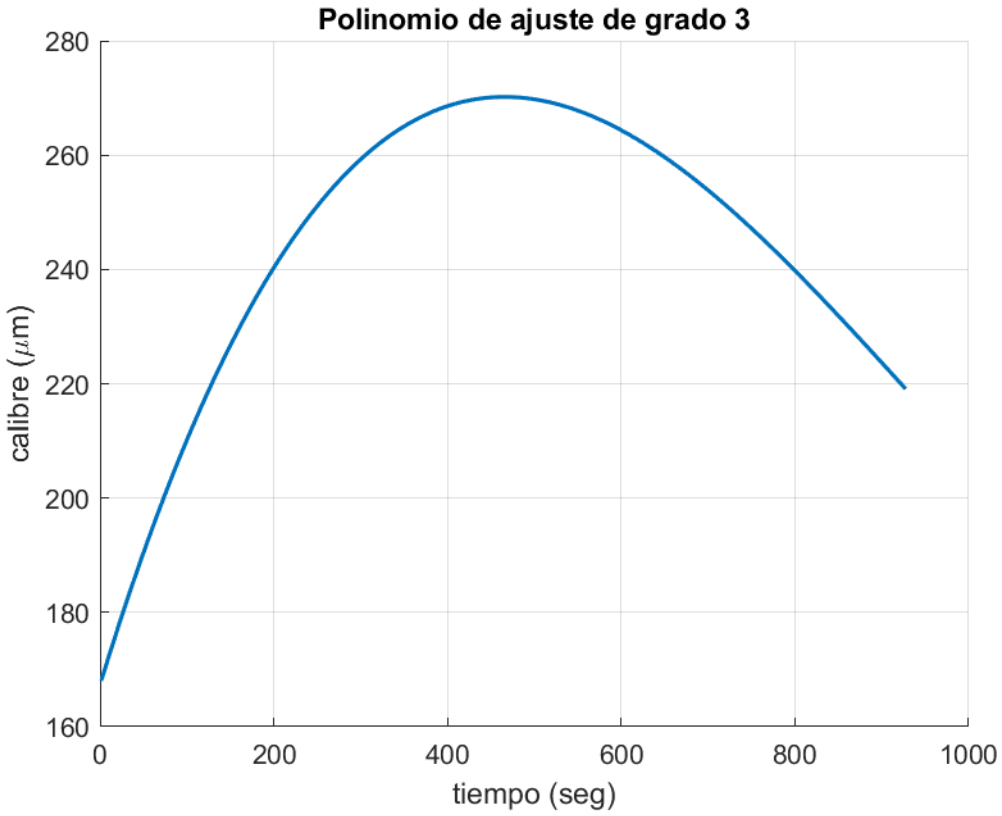


Ilustración 4-21 Polinomio de ajuste de 3er grado.

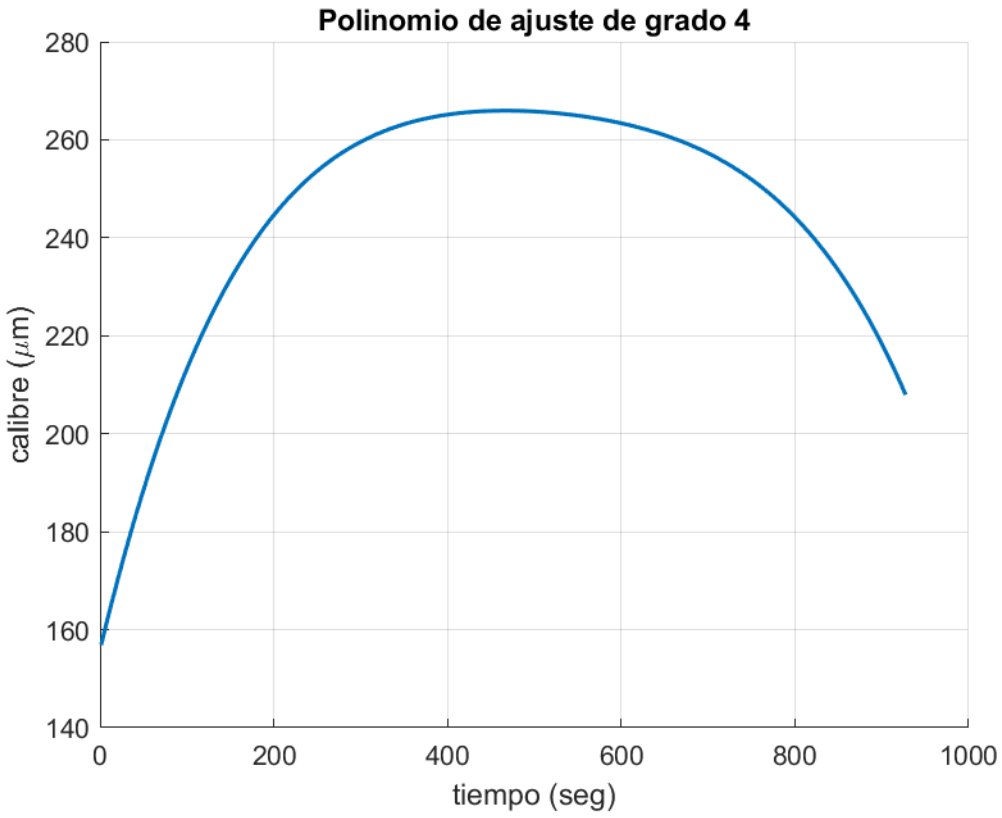


Ilustración 4-22 Polinomio de ajuste de 4o grado.

Debido a que se obtiene un mayor grado de detalle y que no es necesario tener más, se elige el polinomio de grado 4 para el ajuste futuro de los calibre medios experimentales.

El siguiente paso, es a través de los datos leídos, ajustarlos según sus valores teóricos medios y el polinomio de ajuste. Esto se consigue con la siguiente expresión:

$$datos\ exp\ ajustados\ =\ datos\ exp\ +\ (datos\ teóricos\ -\ polinomio\ de\ ajuste)$$

En la siguiente gráfica se muestra cómo una nube de datos experimentales se ajusta según este polinomio descrito:

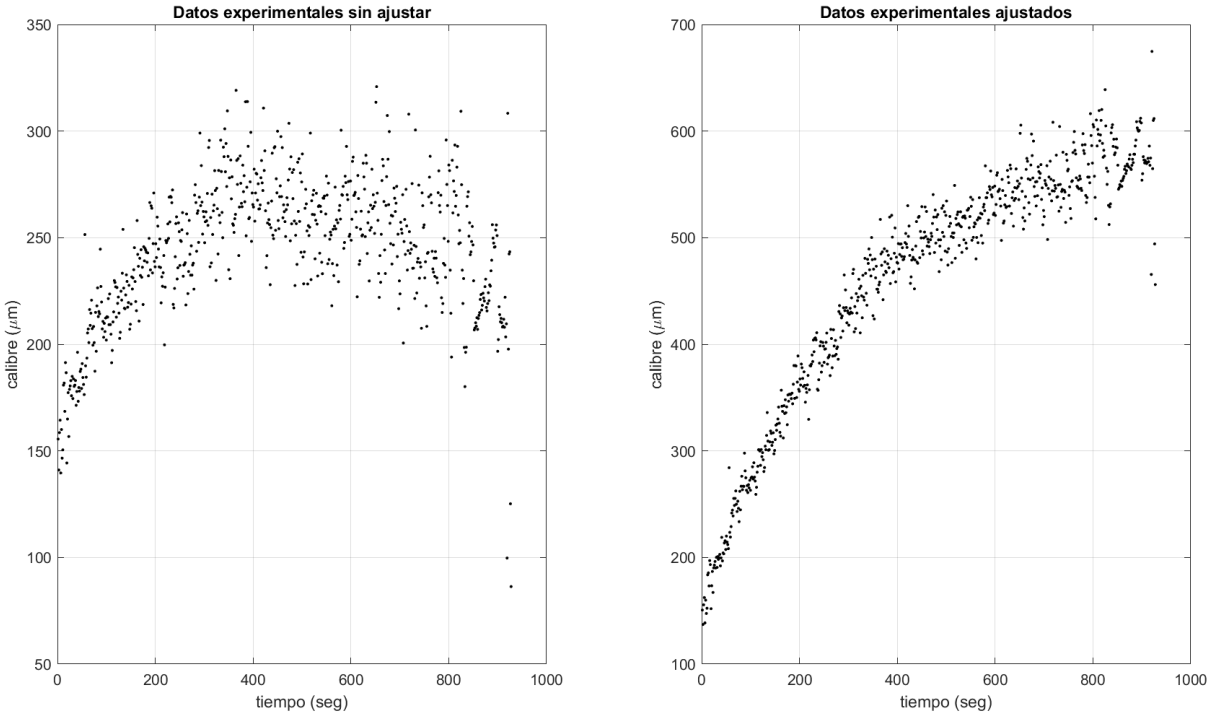


Ilustración 4-23 Datos experimentales antes y después del ajuste.

Para tener una visión más apropiada del desarrollo del proceso, estos datos se contrastarán con la media y la desviación de este en cada punto, tal y como queda reflejado en la siguiente gráfica donde se superponen la nube de datos experimentales con las curvas teóricas obtenidas expuestas con anterioridad.

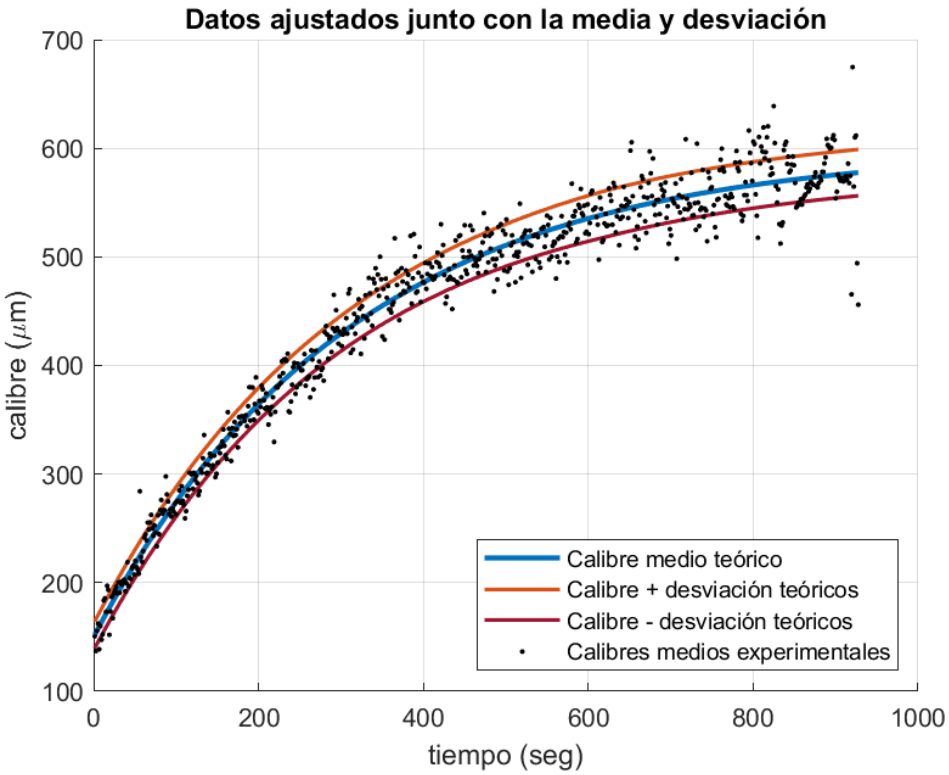


Ilustración 4-24 Datos experimentales frente a la media y desviación teóricas.

4.4.3.4 **Estudio sobre la coincidencia gráfica teórica y experimental**

En la gráfica anterior se puede apreciar cómo hay puntos que quedan fuera de los límites propuestos por las curvas de la desviación típica estándar propuestas anteriormente. Resulta interesante proponer esta pertenencia al rango descrito como medida de error.

Esta pertenencia no se va a hacer de forma estricta al rango descrito por la desviación típica estándar como tal, sino que se va a hacer con una cierta holgura establecida por un coeficiente. Este coeficiente se aplica a la desviación para establecer dos bandas, las cuales se usarán para conocer si el dato experimental se sale del valor esperado o no.

El cálculo de estas bandas se hará por la siguiente fórmula:

$$banda\ error\ =\ calibre\ teórico\ \pm\ coefError\ \times\ coefPercentil\ \times\ desviación$$

Este coeficiente se fija con el valor 2 para tener una banda lo suficiente amplia para no contar como erróneas imágenes que se salen por poco de lo esperado, ni pasar por alto otras que sí que lo hagan.

De esta manera, se trabajará con una gráfica de pertenencia al rango con la siguiente forma, viéndose respecto a estos datos experimentales:

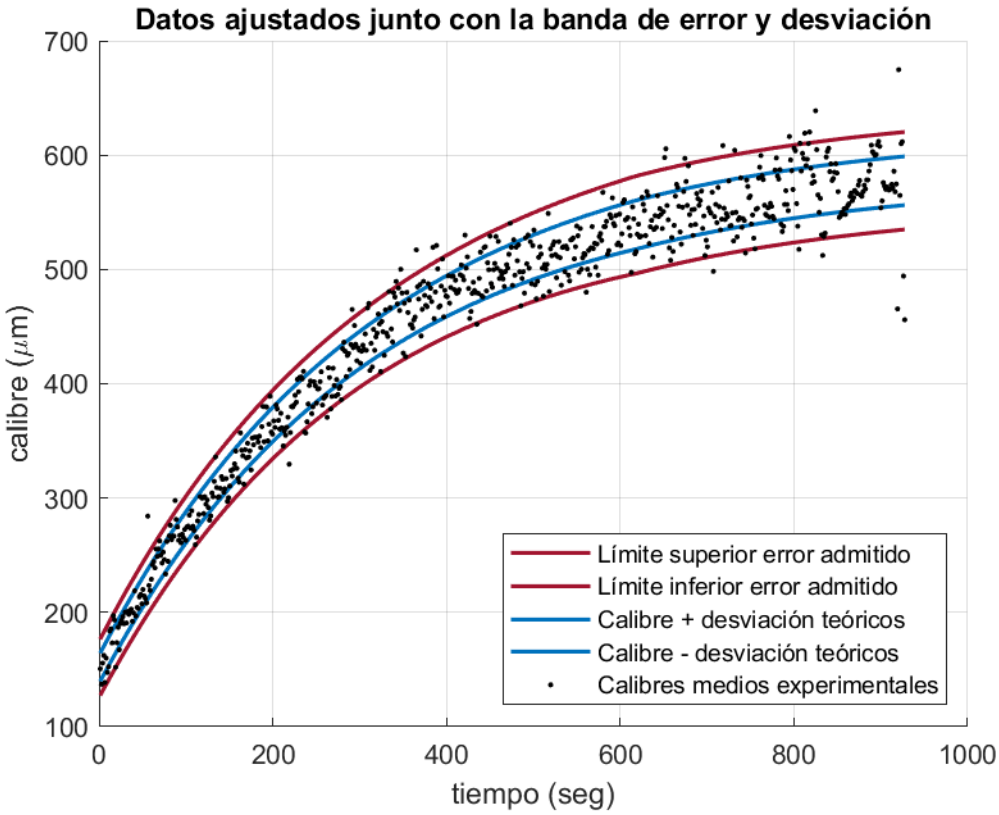


Ilustración 4-25 Datos experimentales junto con las bandas de pertenencia correcta.

Por último, en este estudio se va a tener la medida del error respecto a esta banda propuesta. Se tendrá un número de imágenes erróneas y este número frente al total para tener una idea del porcentaje de erróneas frente a las procesadas.

Esta medida del error se podrá visualizar con mayor facilidad a través de la siguiente gráfica, donde a su vez se podrá apreciar cómo evoluciona y de la manera que aumenta a lo largo del proceso, teniendo así una idea más global de en qué parte del proceso se están dando complicaciones y poder actuar correctamente.

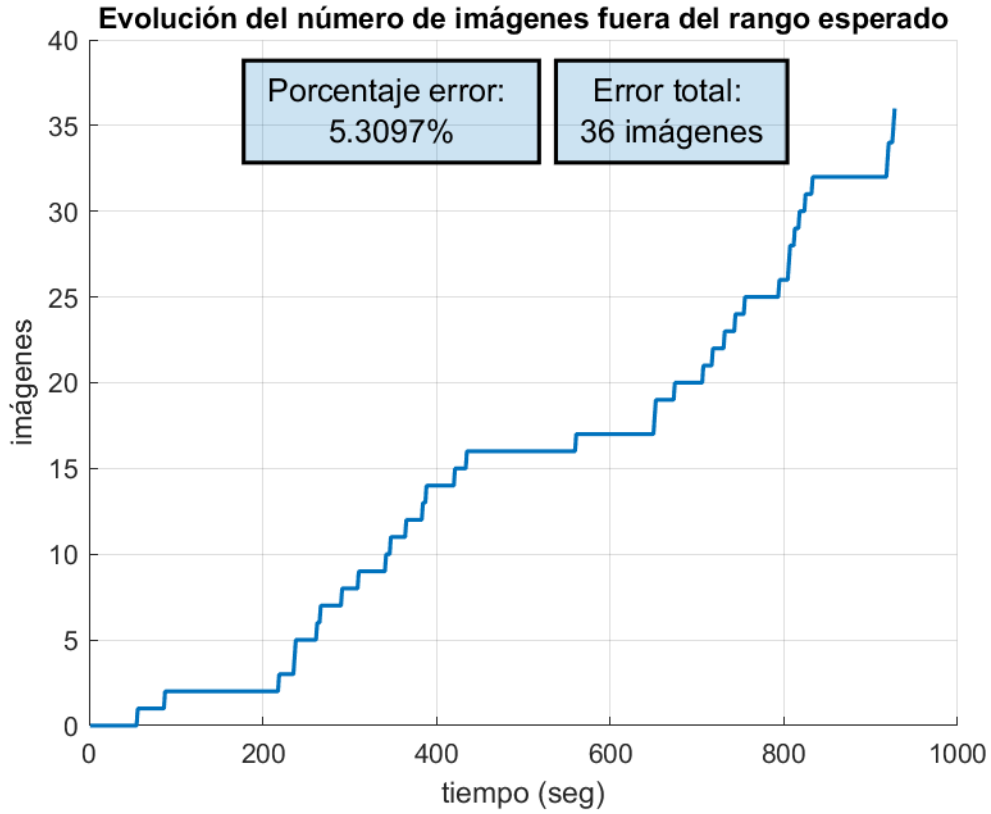


Ilustración 4-26 Imágenes fuera del rango esperado.

4.5 Monitorización de la evolución temporal del crecimiento del grano

Una vez vistos todos los estudios que se van a realizar sobre las imágenes del proceso, se procede a mostrar la forma más genérica de ver cómo evoluciona el proceso. Para ello se elige variable de estudio para el seguimiento del crecimiento del grano el calibre medio, magnitud ya estudiada previamente. El hecho de ir mostrando el valor de esta imagen en el proceso viene motivado por ser la variable que mejor representa el calibre del grano en cada punto.

Estas gráficas se irán obteniendo de forma dinámica, es decir, se irá dibujando a medida que se lean y procesen las imágenes del vídeo con el fin de poder mostrar en tiempo real estado en el que se encuentra el proceso.

De la misma forma que se justificó el uso del ajuste polinómico en el estudio de la media y la desviación típica del calibre en el apartado 4.4.3.3 se aplicará un ajuste polinómico a los datos de la monitorización de la evolución temporal del crecimiento del grano.

5 APLICACIÓN FINAL DEL ALGORITMO SOBRE LAS IMÁGENES

El objetivo principal de este capítulo consiste en aplicar el algoritmo de procesamiento de imágenes propuesto a los datos que se tienen del proceso. Desgraciadamente se tiene un único vídeo del proceso de cristalización del azúcar y por este motivo, el algoritmo quedará como una propuesta con vistas a ser mejorado en un futuro ante un mayor input de datos de entrenamiento, ajustando así los parámetros pertinentes para obtener una mejor monitorización.

Con los datos que se tienen en la actualidad, el capítulo se estructurará de la siguiente forma:

1) Recapitulación del algoritmo que se va a aplicar
2) Conclusiones de los datos obtenidos a través de los diferentes estudios.

5.1 Procedimiento que se va a aplicar

Con intención de esquematizar y aclarar los puntos clave del algoritmo se tiene la siguiente tabla con los pasos que se van a aplicar sobre el vídeo a estudiar. Cabe recordar que en este estudio se tendrá un vídeo grabado, en diferido, con un principio y un final, sin embargo, el objetivo futuro de este algoritmo es ser aplicado a un flujo continuo de imágenes sin saber cuál es el final de dicho proceso.

Por estos motivos, se tiene la siguiente lista de acciones a aplicar con un elevado grado de abstracción:

Tabla 5-1. Pasos del algoritmo.

Paso	Método Matlab	Script
Lectura del flujo de imágenes de entrada	*VideoReader*	*main_video.m*
Selección y lectura de la imagen	*readFrame*	*main_video.m*
Conversión a escala de grises	*rgb2gray*	*main_video.m*
Inversión de la imagen	*imcomplement*	*prep_video.m*
Apertura en escala de grises	*imopen*	*prep_video.m*
Mejora de contraste	*imadjust*	*prep_video.m*
Binarización por Otsu	*graythresh, imbinarize*	*prep_video.m*
Relleno de huecos	*imfill*	*prep_video.m*
Eliminación de ruidos e impurezas	*imopen*	*prep_video.m*
Conversión a *ImagePlus*	*copyImagePlus*	*java_watershed.m*
Umbralización en *ImagePlus*	*IJ.run("Convert to Mask")*	*java_watershed.m*
Aplicación de Watershed	*IJ.run("Watershed")*	*java_watershed.m*
Conversión a matriz de Matlab	*IJ.getFloatArray*	*java_watershed.m*
Rotación de la matriz	*imrotate, flip*	*java_watershed.m*
Eliminación de granos en bordes	*imclearborder*	*main_video.m*

Extracción de características	*regionprops*	*carac_imag.m*
Descarte de imágenes por límites	-	*carac_imag.m*
Obtención de la estructura de datos	*struct*	*carac_imag.m*
Descarte por test de Kolmogorov	-	*main_video.m*
Obtención de gráficas para estudio	*plot*	*main_video.m*

Una vez expuestos los pasos, el vídeo será procesado y estudiado a través de la obtención de sus datos más característicos, los cuales se expusieron en el apartado 4.3.3.

5.2 Conclusiones sobre los resultados obtenidos

En este apartado se procede a analizar a través de gráficas trazadas mediante los datos obtenidos de las imágenes analizadas si este proceso sigue la evolución esperada o si, por el contrario, se detecta algún problema en su desarrollo que indique que lo más adecuado sea pararlo.

5.2.1 Imágenes procesadas

Tal y como se ha comentado anteriormente, en el presente estudio se trabaja a través de un vídeo ya grabado del proceso de cristalización de los granos de azúcar. Por este motivo, es sencillo saber con cuántos frames se cuenta de forma previa al estudio. En el caso de que se tuviera un vídeo en tiempo real se tiene una variable que actúa como contador dentro de la función *main_video.m* encargada de ir recorriendo los frames del vídeo con la cual se podrá tener el número de frames que se procesan.

Puesto a que la tasa de frames por segundo (fps) del vídeo con el que se trabaja es bastante reducida, tan solo 6fps, se opta por aplicar una tasa de reducción de frames leídos y procesados de 5, es decir, de cada 5 frames que se lean, solo se procesará un frame. Este hecho lleva a tener una tasa de lectura final aproximada de 1,2fps.

De esta forma y para este caso, se tiene que el número total de frames que se van a procesar de este vídeo son 1113 imágenes a lo largo del proceso.

Al ejecutar el algoritmo, la variable de salida con los datos obtenidos del proceso cuenta con 678 entradas, de las 1113 procesadas en un principio. Esta reducción se debe a lo comentado anteriormente, la criba de imágenes que no cumplen con las expectativas del proceso y que por ello serán descartadas, pues cuentan con errores en su binarización, segmentación o procesamiento.

En la siguiente tabla se puede apreciar con mayor claridad la intencionalidad de este apartado en el estudio:

Tabla 5-2. Porcentajes imágenes procesadas.

Tipo de dato	Total
Duración del vídeo	927 segundos
Frames por segundo del vídeo	6 fps
N.º Imágenes del vídeo	5565 imágenes
Tasa de lectura	5
Frames por segundo final	1,2 fps
N.º Imágenes procesadas	1113 imágenes
N.º Imágenes aceptadas	678 imágenes
Porcentaje de imágenes aceptadas	60,916 %

Se puede observar que, a pesar de tener una tasa de reducción bastante elevada, se descartan casi el 40% de las imágenes que se leen. Este factor no debería ser preocupante en un principio, ya que se opta por el descarte en lugar de tener datos de salida que no sean reales debido a problemas en la interpretación de estas imágenes.

5.2.2 Tiempo de ejecución

Gracias a que se tiene un vídeo ya grabado del proceso, se podrá tener una idea de cómo de rápido actúa este algoritmo frente al flujo de entrada de datos. Como se ha indicado en el apartado anterior, el vídeo cuenta con 927 segundos de imágenes, por lo que se espera que el algoritmo sea capaz de actuar en el mismo o menor tiempo de duración del vídeo para poder hablar de un algoritmo que actúa en tiempo real, de lo contrario, se tendría un algoritmo incapaz de seguir el ritmo del flujo de entrada.

Para tener constancia de este tiempo de procesamiento del vídeo al completo se tienen las funciones de Matlab *tic* y *toc* las cuales actúan almacenando en una variable el tiempo real transcurrido entre la ejecución de ambas. Se usan al iniciar el proceso y al finalizarlo tras lanzar la ejecución de las gráficas para tener constancia de este tiempo de procesamiento.

Este tiempo en las diferentes ejecuciones del algoritmo es fluctuante, ya que se trata de un tiempo relativo en gran parte a la velocidad de procesamiento de la máquina en la que se ejecuta en cuestión. Se tiene que, en media, con un procesador Intel Core i-5 de 7th, el tiempo de procesamiento del algoritmo para un vídeo de 927 segundos es de 700 segundos, lo cual está por debajo, dejando margen, del tiempo máximo requerido.

Como se comentó al comienzo de este capítulo, el hecho de tener un vídeo grabado, no será la forma final que se busca con este algoritmo, sino que se contará con un flujo continuo de imágenes de entrada sin tener constancia del tiempo final. Por este motivo, las funciones usadas para la lectura de los frames no serán las mismas, hecho que puede ocasionar bien retrasos en el tiempo de procesamiento, o, por el contrario, puede acelerar este tiempo al ser más rápidas en ejecución que los métodos de la clase citada para esta lectura *VideoReader*.

5.2.3 Estudio de Kolmogorov-Smirnov

En este apartado se procede a estudiar los datos que provienen del estudio a través del test de Kolmogorov realizado a los datos del calibre de la imagen siguiendo las pautas descritas en el apartado 4.4.2.

Este test estudiará las diferencias máximas en cada imagen entre la distribución gaussiana esperada en cada punto de proceso con la que se tiene en la imagen que se estudia. Estas distancias no se estudiarán de forma aislada, es decir, imagen por imagen, sino que se tendrá en cuenta los datos del test en imágenes anteriores.

De esta forma, se trabaja en este apartado con dos tipos de datos:

1) Diferencias máximas del test de Kolmogorov-Smirnov
2) Error Cuadrático Medio acumulado

Con el objetivo de detectar outliers, se han establecido los umbrales citados en 4.4.2.4 a través de los cuales de nuevo se podrá descartar la imagen o no. Podrá verse su aplicación en la ejecución final del algoritmo en el apartado 5.2.6.

5.2.4 Estudio del calibre medio y su desviación

En este apartado se realizará el estudio sobre los valores del calibre medio de los granos de la imagen. Para ello, se van a obtener dos gráficas, una primera donde se podrá apreciar estos valores citados, y otra segunda donde se obtendrá el número de imágenes totales y en proporción que se considerarán erróneas.

5.2.4.1 Gráfica del calibre medio frente a la banda de error permitida

En ella podrán verse dos curvas diferentes y los valores de los datos experimentales tratados.

Los datos experimentales no aparecen tal cual se leen, sino que, tal y como se expuso en el apartado 4.4.3.3 estos datos han sido ajustados a través de un polinomio de 4º grado que los ajusta a los datos esperados para poder estudiar si estos arrojan algún error no esperado, o si por el contrario el proceso está siguiendo la evolución esperada.

La siguiente curva que se aprecia es la formada por la desviación típica estándar esperada en cada punto del proceso, de la cual se obtiene la tercera y última, denominada banda de error. Esta ha sido obtenida a través de sumar o restar 2 veces el valor de la desviación en cada punto, pues el simple valor de la desviación suponía una banda muy restrictiva para estos valores experimentales. Como en el apartado anterior, se podrá ver gráficamente en el apartado de la ejecución final 5.2.6.

5.2.4.2 Gráfica imágenes erróneas detectadas a través del estudio del calibre

En esta gráfica, se han considerado como erróneas aquellas imágenes cuyo valor experimental del calibre medio quedara fuera de la banda formada por el valor del calibre medio esperado ± 2 veces el valor de la desviación típica estándar esperada en cada punto del proceso, como se hace referencia en el apartado 4.4.3.4.

Una vez definido el concepto de imagen errónea, se obtiene la siguiente gráfica donde quedan reflejadas qué imágenes en qué puntos del proceso se han considerado erróneas y cómo evoluciona este error en el tiempo de forma acumulada. Podrá verse junto con el resto de las gráficas en el apartado de la ejecución final 5.2.6.

5.2.4.3 Observaciones sobre las gráficas

Como se puede apreciar, ambas gráficas están directamente relacionadas, pues la finalidad con la que se obtienen no es otra que la de detectar valores outliers del calibre medio de los granos en cada imagen. Destaca de ellas que estos outliers se dan fundamentalmente en el inicio y en el final del proceso, pues son los tiempos sonde más difieren los valores experimentales de los esperados ya que el grano es más difícil de ser detectado debido a su pequeño tamaño y su aglomeración respectivamente.

5.2.5 Estudio del calibre

El último estudio que se le hace al proceso de cristalización es el del seguimiento del calibre medio sin ser ajustado previamente. Por ello, cabe destacar que estos datos tendrán cambios más abruptos en pequeños lapsus de tiempo. Con esta gráfica se pretende conocer cómo evoluciona el estudio con el paso del tiempo a medida que se leen y procesan las imágenes recibidas, o en este caso, leídas del propio vídeo de estudio.

5.2.6 Ejecución final

Las gráficas mostradas dentro de este apartado 5.2 son las que forman parte de una sola, la cual es la que realmente se verá al ejecutar el algoritmo. En ella, podrán verse estas 5 gráficas en una misma pantalla, facilitando así la interpretación en conjunto de ellas.

La idea fundamental de esta obtención de gráficas no es otra que la de poder seguir el proceso en tiempo real, y por ello, los scripts creados para su representación están diseñados para mostrar los datos obtenidos hasta el momento y poderlos representar en cualquier punto del proceso.

Para evitar la sobrecarga de gráficas y reducir su tiempo de refresco para poder interpretarlas correctamente, se lanzarán estas gráficas cada cierto tiempo, para saber cómo se está siguiendo el proceso.

Por defecto, se refresca esta gráfica cada vez que se procesen 50 imágenes, pero en el caso de que se detecte alguna imagen errónea a través del estudio de su calibre medio, también se mostrará para poder analizar el problema y parar el proceso si así se considera.

Algunas de las gráficas que se obtienen durante la ejecución del algoritmo son las siguientes:

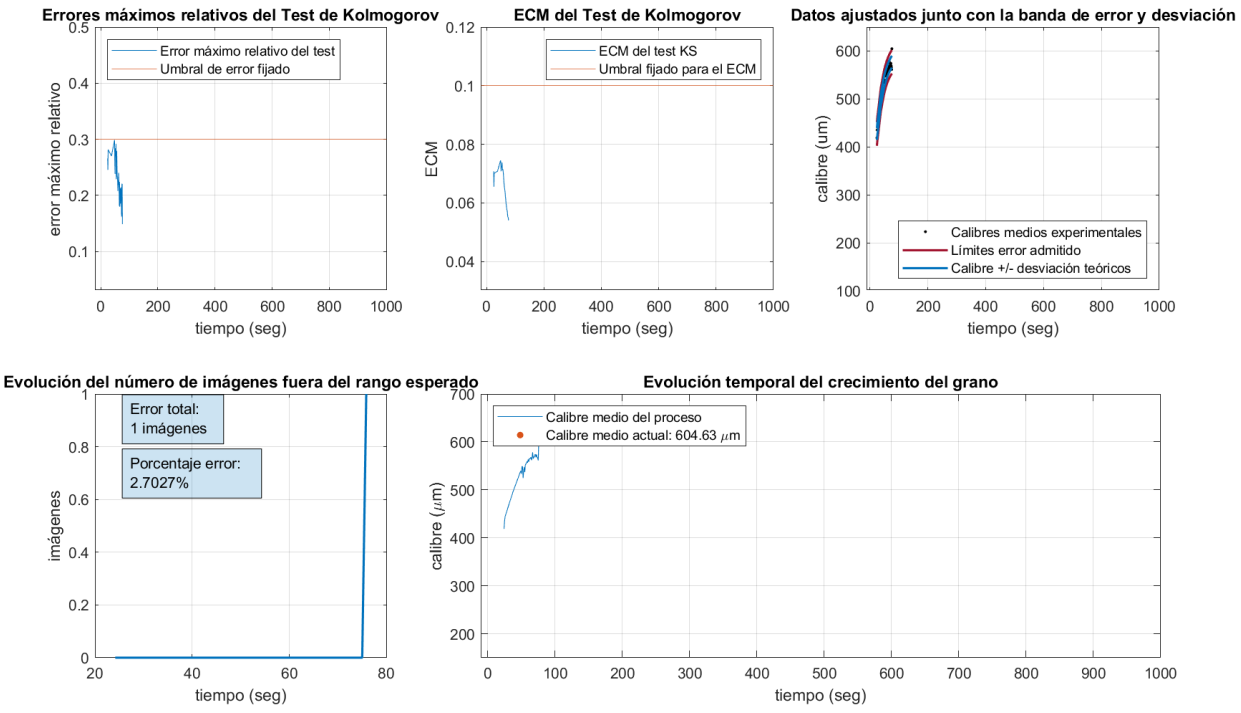


Ilustración 5-1 Gráfica extraída en un punto próximo al comienzo del proceso.

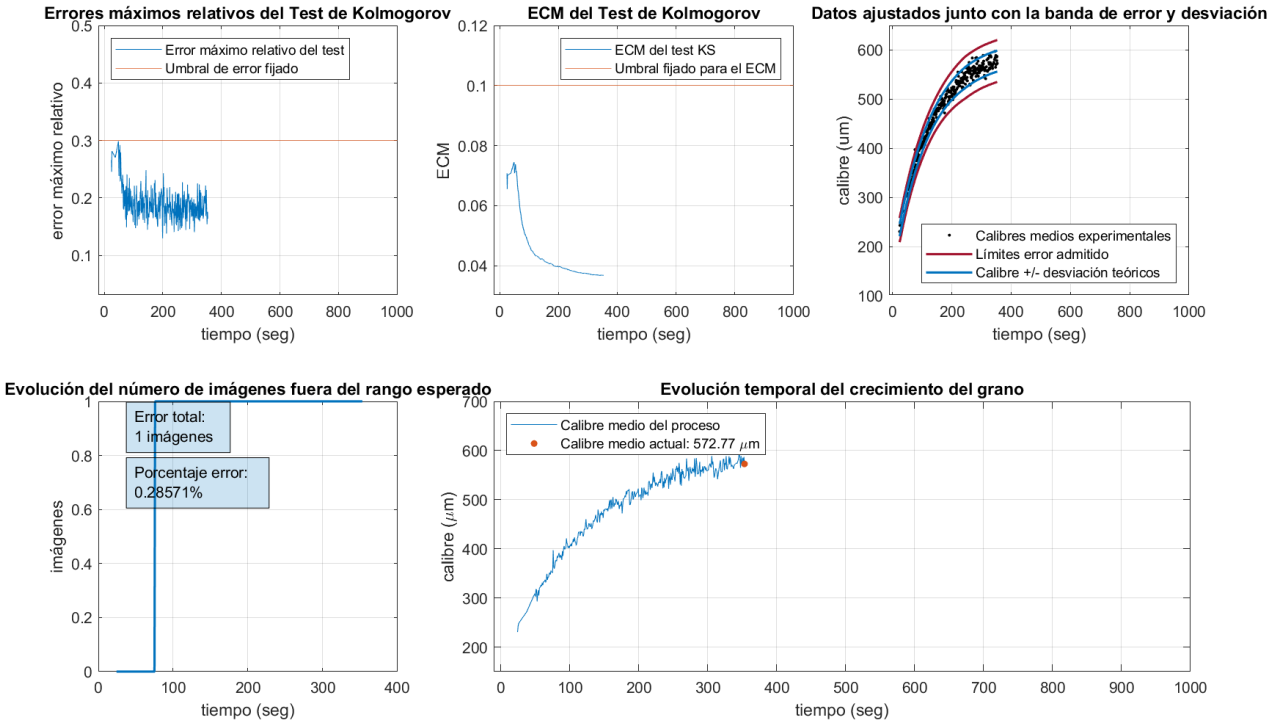


Ilustración 5-2 Gráfica extraída en un punto próximo a la mitad del proceso.

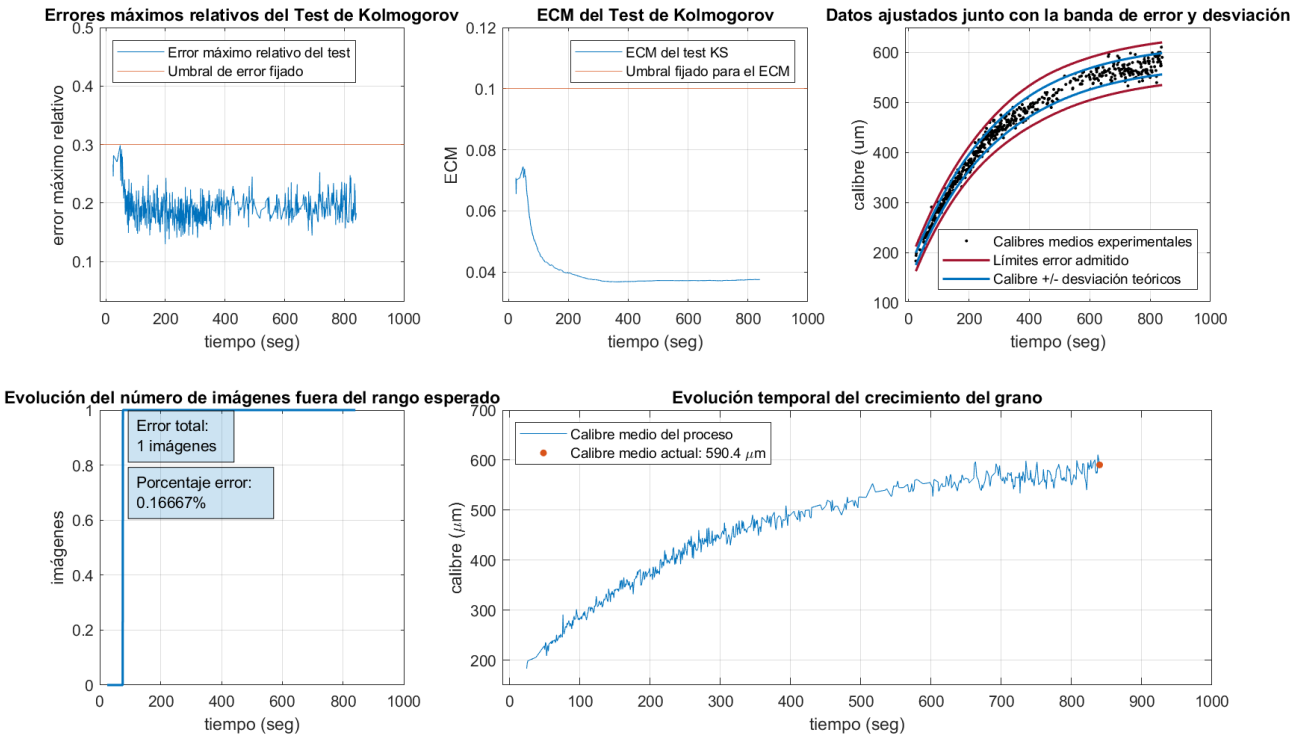


Ilustración 5-3 Gráfica extraída en un punto próximo al final del proceso.

Por último, la gráfica en el punto final del proceso correspondiente a la extraída tras leer el frame final del vídeo es la siguiente:

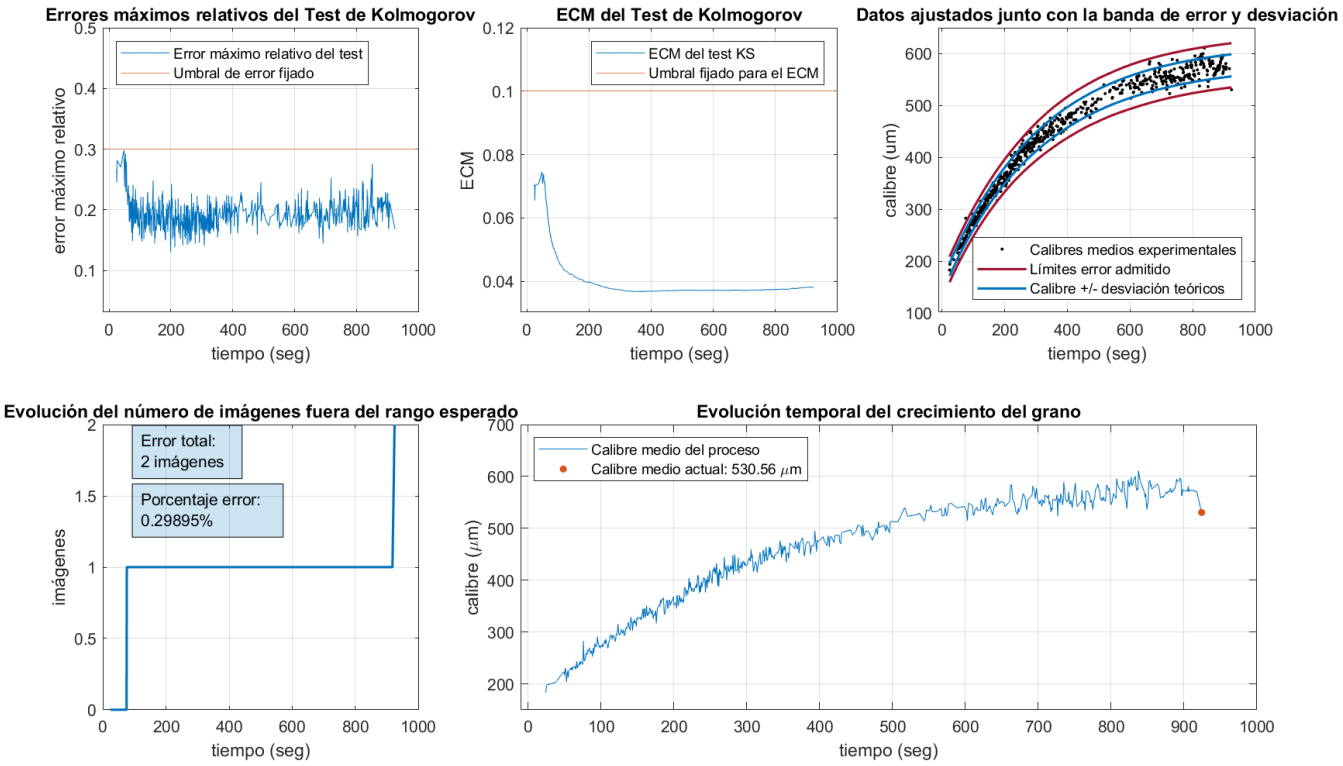


Ilustración 5-4 Gráfica extraída tras procesar el último frame del proceso.

5.2.7 Conclusiones y trabajos futuros

En este apartado se procede a retratar las principales ventajas, inconvenientes y comentarios sobre mejoras que se le pueden aplicar al algoritmo presentado. Con el fin de quedar reflejados lo mejor posible, se distribuyen en los siguientes epígrafes:

1) Entrada del proceso
2) Ajuste del algoritmo al input adecuado
3) Señal de parada del proceso

A continuación, se detallan cada uno de estos puntos mencionados:

5.2.7.1 Mejora de los datos de entrada

La parte fundamental del algoritmo, como lo es en la mayoría de los casos en los que se trabajan con datos experimentales, es la entrada a este. En el caso que atañe este trabajo, se tiene un único dato de entrada en forma de vídeo ya grabado sobre un supuesto proceso de cristalización.

Al visualizar el vídeo, se puede apreciar que la evolución del crecimiento de los granos no es constante ni se asemeja a la perfección al comportamiento esperado, sino que, por el contrario, en el vídeo se pueden apreciar saltos en este proceso de un frame al siguiente, incoherencias en sus tamaños y otras irregularidades tal y como se puede apreciar en las siguientes capturas de dos frames consecutivos:

Ilustración 5-5 Captura del vídeo en un frame.

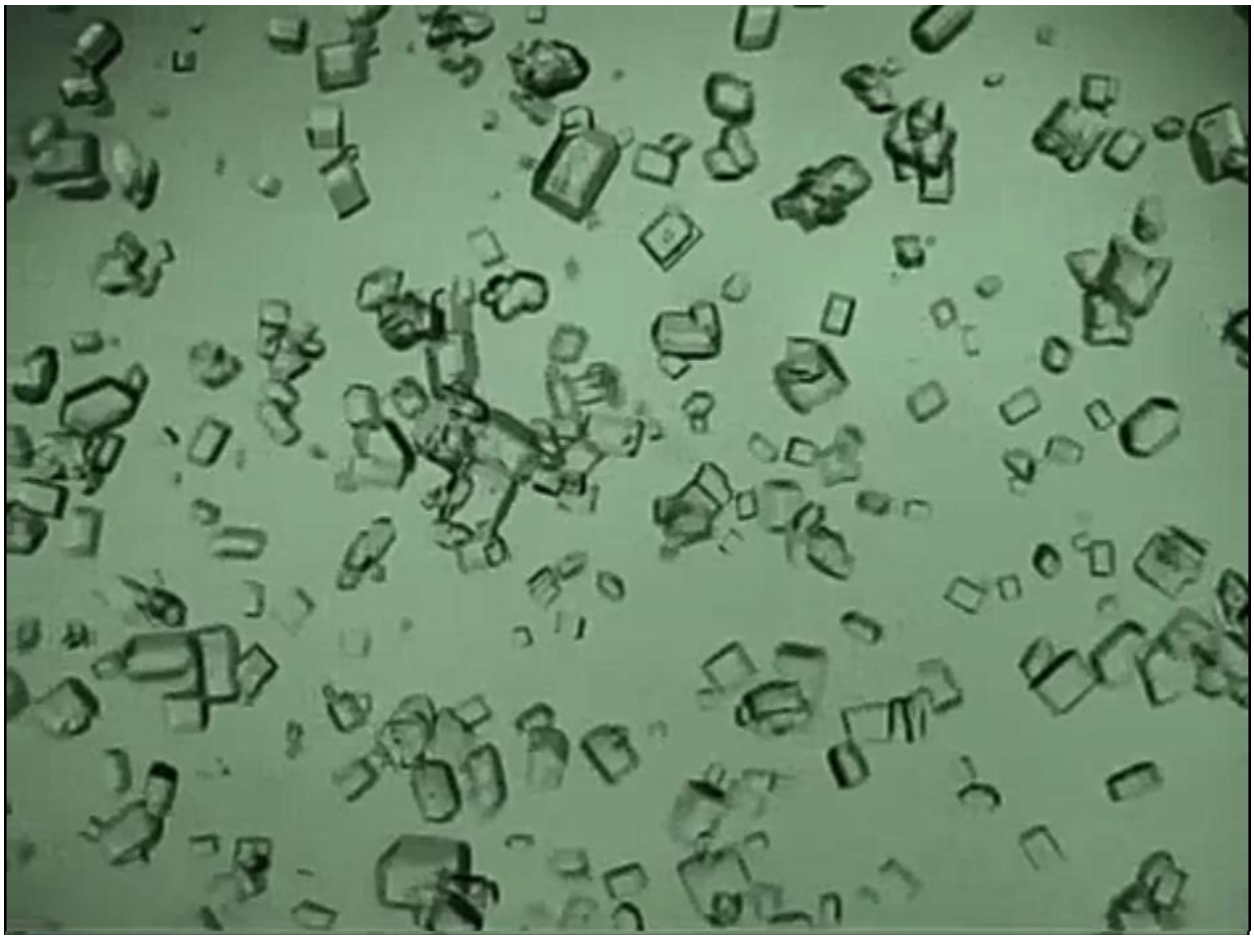


Ilustración 5-6 Captura del vídeo en un frame muy próximo.

Estas, sin embargo, se dan con mayor fuerza en el comienzo y final del vídeo, tal y como queda reflejado en el estudio de las imágenes erróneas anteriormente presentado, por lo que se puede entender que estos errores se deben a un error en la grabación del vídeo y no un fallo presentado por el algoritmo.

Se adjunta unas capturas del vídeo en estos puntos citados para comprender mejor el problema:

Ilustración 5-7 Captura de pantalla de un frame del comienzo del proceso.

En esa imagen se puede apreciar, que el único grano con el que cuenta el frame, se encuentra localizado en uno de los bordes de la pantalla por lo que, en el caso de ser reconocido correctamente, será descartado por la función de limpieza de bordes. Este hecho provoca por una parte que no se cuente con suficientes granos para obtener valores medios o estadísticos adecuado y además el peligro que supone que, al no haber granos en la escena, sea el fondo el que esté en riesgo de ser segmentado debido al problema ya comentado de la retroiluminación centrada, la cual actúa oscureciendo los bordes y esquinas de la imagen.

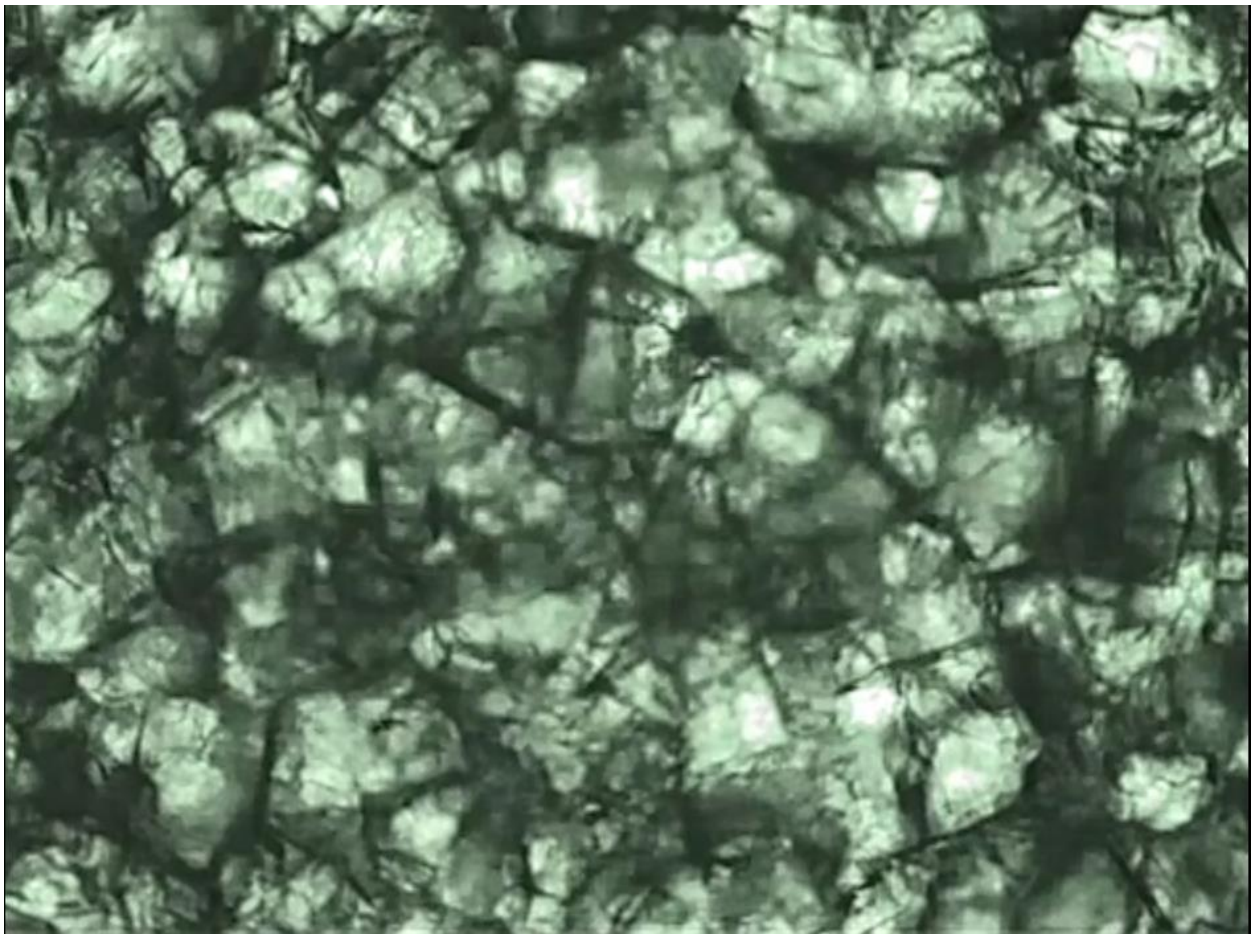


Ilustración 5-8 Captura de pantalla de un frame del final del proceso.

Como se puede apreciar en la imagen, este algoritmo sufre de desbordamientos en la segmentación de imágenes a medida que se desarrolla el proceso debido a que los granos de sacarosa que acontecen en las imágenes, cuando el proceso se va desarrollando, se van aglomerando, haciendo que sea cada vez más difícil para el algoritmo reconocer correctamente las regiones de la imagen, es decir, los granos de ella.

Por lo motivos citados, se hace vital una mejora en la captación de las imágenes que pase por tratar los siguientes aspectos fundamentales para evitar errores en su lectura durante el proceso:

1) **Iluminación:** mejorar la retroiluminación de las imágenes de forma que esta se dé con la misma intensidad en todos los puntos de la imagen para que no se encuentren los bordes y esquinas más oscuros y por ellos peligren de ser segmentados como granos de azúcar.
2) **Número de granos:** asegurar que durante el proceso haya un mismo número de granos, suficiente para obtener datos estadísticos con cierta fiabilidad, pero no tantos como para aglomerarse hasta el punto de no poder ser reconocidos ni por el algoritmo ni tampoco siquiera por el ojo humano en su inspección.
3) **Continuidad del flujo:** un factor muy importante para tener en cuenta de cara al flujo de entrada de imágenes para su posterior estudio es que estas sean continuas en el tiempo y que los granos no sufran alteraciones de tamaño de un frame a su siguiente tal y como se ha mostrado anteriormente.

5.2.7.2 **Reajuste del algoritmo**

Como ya se ha mencionado, esta versión del algoritmo queda pendiente de ajuste ante nuevos datos más adecuados de entrada, de forma que estos datos puedan ser procesados con la máxima fidelidad posible al igual que se ha hecho ante los datos de entrada (el vídeo) del que se dispone.

En el caso de estar ante un flujo de entrada, los pasos a seguir para obtener el mayor rendimiento de este serían ajustar algunos parámetros definidos para las imágenes de las que se disponía hasta el momento. Estos parámetros serían tales como tamaños de máscara, tasa de lectura de imágenes, tamaños mínimos y máximos de las escalas de los granos, umbral de binarización, valores mínimos y máximos de los parámetros estudiados de los granos de la imagen…

5.2.7.3 **Señal de parada**

Como se indicó al comienzo del trabajo, la intención fundamental de este es el de servirle de apoyo a los operarios de la industria azucarera encargada de monitorizar y evaluar el proceso de cristalización del azúcar. Para ello es fundamental que este algoritmo propuesto tenga la capacidad de bien alertar al operario o bien ser capaz de poder detener el proceso siendo conectado a los módulos adecuados en la industria. Esta decisión se tomaría en base al error estudiado en cada punto del desarrollo del proceso, estableciendo los umbrales adecuados para esta decisión.

Para una mejor comprensión del proceso de decisión que se espera del algoritmo se adjunta un diagrama de flujo del comportamiento que este tendría llegado el momento de implementarlo.

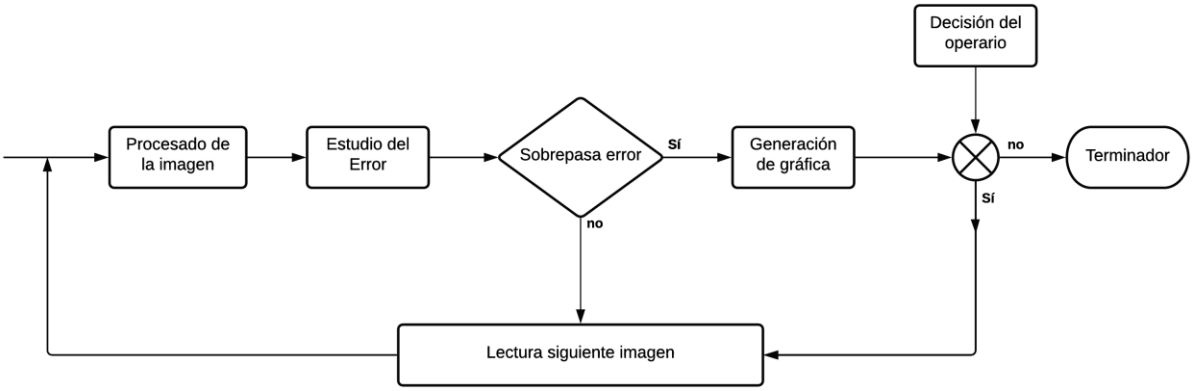


Ilustración 5-9 Diagrama de flujo del comportamiento deseado del algoritmo.

ANEXO A: CÓDIGO PARA LA EJECUCIÓN DEL TRABAJO

Función principal

main_video.m

PRUEBA COMPLETA CON IMÁGENES EXTRAÍDAS DEL VÍDEO

```
function [datos, tReal] = main_video ()
% datos = estructura con los datos extraídos del vídeo leído
% tReal = vector con los tiempos donde han sido leídos los datos


clear;
close all;

% Para contar el tiempo que tarda en ejecutarse el algoritmo sobre el vídeo completo
tic;


% Declaraciones


% Para evitar que aparezcan alertas producidas por Java
warning('off','all')
rmpath('folderthatisnotonpath')
javaaddpath('ij.jar')


% Rutas de donde se leerán y almacenarán las diferentes imágenes
% ruta_seg = 'imagenesSegmentadas/seg_';
ruta_vid = 'video/';
ruta_seg = 'imagenesSegmentadas/';
seg_name = 'seg_';


ruta_read = 'video.mp4';
ruta_wr = [ruta_vid, ruta_seg];
```

```
% Elimina los elementos que pudiera haber y se asegura de que haya carpeta
% donde guardar las imágenes segmentadas
if isfolder(ruta_wr)
    rmdir(ruta_wr,'s')
end

mkdir(ruta_wr)



% Variable índice para ir almacenando los datos
numImProc = 0;

% Variable contador de gráficas impresas
graficasImp = 0;

% Tasa de reducción del FrameRate
tasa = 5;

% Lectura del objeto vídeo
vidObj = VideoReader(ruta_read);

% Número de imágenes que se van a leer en un principio del vídeo.
% Relacionado con el % de frames que se leen
Nimag = fix(vidObj.NumberOfFrames/tasa);

% Nueva lectura del vídeo, tras usar el atributo NumberOfFrames hay que
% volver a declarar el objeto para leer de él
vidObj = VideoReader(ruta_read);

% Valriable para almacenar los valores obtenidos del test de Kolmogorov en
% cada imagen.
vectorKolmo = zeros(1,Nimag);
```

```
% Variable para almacenar los tiempos reales del vídeo donde se han leído
% los frames procesados
tReal = zeros(1,Nimag);



% Los granos comienzan a crecer de 11px al principio hasta 66px al final
pxLim = [11,66];

% Los granos comienzan a crecer de 150um al principio hasta 600um al final
calLim = [150,600];



i = 1;
while hasFrame(vidObj)



    % Lectura de la imagen
    im = readFrame(vidObj);

    if ~mod(i,tasa)

        % La imagen leída se encuentra en RGB debido al método de lectura,
        % se pasa a escala de grises para trabajar con ella.
        im = rgb2gray(im);

        % Preprocesamiento de la imagen
        pre = prep_video(im);

        % Aplicación de Watershed a través de IJ
        seg = java_watershed(pre);

        % Último paso del preprocesamiento
        cl = imclearborder(seg);
```

```
        % Para guardar las imágenes segmentadas
%         imwrite(cl, [ruta_wr, seg_name, 'im', int2str(i/tasa),'.png']);

        % Obtención de datos de la imagen
        datosAux = carac_imag(cl);

        % Si la imagen leída se ha decidido ser procesada en el campo estado
        % habrá un 0
        if ~datosAux.Estado
            % Estudio previo a la estadística de los valores del test
            vectorKolmo(numImProc + 1) = datosAux.TestKolmogorov;
            isErrKS = est_KS(vectorKolmo(1:numImProc+1),tReal(1:numImProc+1),false);
            if not(isErrKS)

                % Aumento del número de imágenes procesadas
                numImProc = numImProc + 1;


                % Almacenaje del tiempo real del frame en el vídeo
                tReal(numImProc) = vidObj.CurrentTime;


                % Almacenamiento de los datos en una estructura con los campos
                % determinados dentro de la función caracImag
                datos(numImProc) = datosAux;

                isErrMedio  =  est_mediosDesv(datos,  tReal(1:numImProc),  [pxLim,
calLim], false);

                if (mod(numImProc,50)==0) || isErrMedio
                    graficasImp = graficasImp + 1;
                    fig = figure(1);
                    est_KS(vectorKolmo(1:numImProc),tReal(1:numImProc),true);
                    est_mediosDesv(datos,  tReal(1:numImProc),  [pxLim,  calLim],
true);
                    ev_temp_cal(datos, tReal(1:numImProc), [pxLim,calLim])
                    saveas(fig,
['resultadosEjecucion/graficaFinal_',num2str(graficasImp),'.png'])
                end
```

```
            end
        end

    end
    i = i + 1;

end



% Vector con los tiempos en los que se procesa cada imagen
tReal(numImProc + 1:end) = [];
vectorKolmo(numImProc + 1:end) = [];



fig = figure(1);
est_KS(vectorKolmo,tReal, true)
est_mediosDesv(datos,tReal,[pxLim,calLim], true);
ev_temp_cal(datos, tReal, [pxLim,calLim])
saveas(fig, 'resultadosEjecucion/graficaFinal.png')

toc;
end
```

Función para el preprocesamiento

prep_video.m

Función para obtener la imagen preprocesada

original = imagen binaria a tratar

```
function [mej] = prep_video (original)



% rMask: tamaño de las máscaras para las aperturas
% rMask1: máscara para la apertura en escala de grises al empezar el
% preprocesamiento
% rMask2:  máscara para la apertura binaria para eliminación de ruido final


rMask1 = 1;
rMask2 = 2;
alfa = 0.65;
% Inversión para trabajar con el mismo tipo de imágenes
original = imcomplement(original);


% Apertura en escala de grises
mask1 = strel('disk', rMask1);
cont = imopen(original, mask1);


% Mejora de Contraste
h = imadjust(cont);


% a) Binarización por Otsu
T = graythresh(h);
a = imbinarize(h,T*alfa);




% b) Relleno de Huecos
[Gmag, ~] = imgradient(a);


b = imfill(Gmag>0, 'holes');




% c) Eliminación de Ruido e Impurezas
mask2 = strel('disk',rMask2);
mej = imopen(b, mask2);
end
```

Función para la segmentación

java_watershed.m

Función para obtener la imagen segmentada

original = imagen binaria a segmentar

segmentada = imagen binaria segmentada por Watershed

```
function [segmentada] = java_watershed (original)




% Transformación a tipo UINT8
bw = uint8(original*255);


% Conversión al formato con el que trabaja java: ImagePlus
imp = copytoImagePlus(bw);


% Umbralizar la imagen para hacerla binaria
imp.getProcessor().setThreshold(255,255,0)
ij.IJ.run(imp, "Convert to Mask", "")


% Aplicar el algoritmo Watershed
ij.IJ.run(imp, "Watershed", "")


% De formato ImagePlus a formato Matlab
im_pro = imp.getProcessor();
im_arr = im_pro.getFloatArray()>0;


% La imagen ante esta forma de obtenerla en formato Matlab, se rota 90
% grados, por lo que se rectifica
segmentada = flip(imrotate(im_arr,90))>0;


end
```

Función para la obtención de características de la imagen

carac_imag.m

Función para obtener las características de los granos de la imagen

```
% Devuelve los datos de la imagen como una estructura de datos de la
% siguiente manera:


% DatosImagen.Estado            = {0,1}     // Si la imagen ha sido descartada
% DatosImagen.TotalGranos       = int       // Número de granos
% DatosImagen.Aceptados         = int       // Número de granos procesados
% DatosImagen.Descartados       = int       // Número de granos descartados
% DatosImagen.Areas             = double[]  // Valores de áreas
% DatosImagen.Solidez           = double[]  // Valores de solidez
% DatosImagen.RelacionFarias    = double[]  // Valores de la relación de Farias
% DatosImagen.MaxFeret          = double[]  // Valores del diámetro de Feret
% DatosImagen.TestKolmogorov    = double    // Valores del Test de Kolmogorov


function DatosImagen = carac_imag (original)


% Extracción de características en bruto de la imagen binaria segmentada
rp = regionprops(original, 'Solidity', 'Area', 'ConvexArea', ...
    'EquivDiameter', 'MaxFeretProperties');


% Número de granos/regiones reconocidos en la imagen
Ngranos = length(rp);

% Valores límite de algunas características que determinará si el grano o
% la imagen será procesado


% Solidez = Área/ÁreaConvexa
SolMin = 0.65;
```

```
% Relación de Farias = Fmáx/Deq
% Rel = 1 --> región circular
% Rel = sqrt(pi/2) --> región cuadrada
% Rel > sqrt(pi/2) --> región rectangular
relCuad = sqrt(pi/2);



% Valor máximo de la relación para considerar que se ha cometido un error
% en la segmentación
relMax = 1.65*relCuad;

% Tamaño total en píxeles de la imagen
[M,N] = size(original);
% Porcentaje del área del total para esta consideración
coef = 1/30;
% Área máxima permitida según total
areaMax = coef*N*M;

% Valor máximo permitido para el máximo del test de Kolmogorov
kolmoMax = 0.5;



% Variables donde se almacenarán algunos datos de la imagen

% Número de granos descartados
Ndesc = 0;
% Número de granos aceptados
Nacep = 0;
% Número de imágenes descartadas
ImagDesc = 0;
% Área de cada grano
Areas = zeros(Ngranos,1);
% Diámetro Máximo de Feret de cada grano
Feret = zeros(Ngranos,1);
% Relación de Farias de cada grano
RelFarias = zeros(Ngranos,1);
```

```
% Solidez de cada grano
Solidez = zeros(Ngranos,1);


if Ngranos == 0
    ImagDesc = 1;
else


    % Bucle para recorrer cada uno de los elementos de la imagen
    % En él se tienen variables auxiliares para trabajar de forma más cómoda
    % con los parámetros obtenidos del elemento
    for i = 1:Ngranos
        % Variable auxiliar para la solidez
        solAux = rp(i).Solidity;
        % Variable auxiliar para el diámetro máximo de Feret
        ferAux = rp(i).MaxFeretDiameter;
        % Variable auxiliar para la relación de Farias
        relAux = ferAux/rp(i).EquivDiameter;
        % Variable auxiliar para el área del grano
        areaAux = rp(i).ConvexArea;


        % Bandera para identificar si el grano se ha descartado
        % GranoDesc = 0 --> Grano se acepta, ha superado los requisitos
        % GranoDesc = 1 --> Grano se descarta por algún motivo comprobado
        GranoDesc = 0;


        % Doble comprobación: límites de la solidez y de la relación de Farias
        if (solAux < SolMin) || (relAux > relMax)
            % Si se considera fallo, se desprecia el grano y
            % aumenta el número de granos descartados
            Ndesc = Ndesc + 1;
            % Se activa la bandera de grano descartado
            GranoDesc = 1;
        end


        % Comprobación del tamaño del grano. Al darse este caso se decide
        % descartar la imagen entera pues se considera fallida la segmentación
```

```
        if areaAux > areaMax
            % Se activa la bandera de imagen descartada
            ImagDesc = 1;
            % Se sale del bucle for para dejar de procesar los elementos
            % restantes
            break;
        end

        % Si no se ha producido ningún error, se procede a guardar los
        % parámetros extraídos de la región
        if ~GranoDesc
            % Aumenta en una unidad el número de elementos aceptados, a su vez,
            % esta variable servirá de índice para ir almacenando los elementos
            % de la imagen en orden, pues si se desprecian elementos, habrá
            % celdas sin datos.
            Nacep = Nacep + 1;
            % Se guarda el valor del área
            Areas(Nacep) = areaAux;
            % Se guarda el valor de la solidez
            Solidez(Nacep) = solAux;
            % Se guarda el valor de la relación de Farias
            RelFarias(Nacep) = relAux;
            % Se guarda el valor del diámetro máximo de Feret
            Feret(Nacep) = ferAux;

        end

    end


    % En el caso de que se hayan descartado granos, quedarán Ngranos - Ndesc
    % celdas al final de cada una de las variables donde se almacenan listas de
    % parámetros sin datos. Para ello se eliminan las celdas siguientes al
    % número de elementos aceptados
    % En el caso Ngranos = Nacep, estas operaciones no tendrán sentido, así que
    % no actuarán
    Areas(Nacep+1:end) = [];
```

```
    Feret(Nacep+1:end) = [];
    Solidez(Nacep+1:end) = [];
    RelFarias(Nacep+1:end) = [];

end

% Comprobación del resultado del test de Kolmogorov-Smirnov. Si el
% valor del test supera el 50% de error relativo o se obtiene un
% número erróneo (NaN) la imagen quedará descartada.
if ~ImagDesc
    vKolmo = test_kolmo(Feret);
    if isnan(vKolmo) || vKolmo > kolmoMax
        ImagDesc = 1;
    end
end

% Se opta por la exportación de datos en formato estructura por su
% versatilidad en tipos de datos y tamaños, pues no todas las imágenes
% tendrán el mismo número de elementos y por ello número de datos

% Independientemente de si la imagen ha sido o no descartada, se almacena
% el estado de la bandera de descarte de imagen y el número de granos que
% se ha reconocido en ella para su futuro tratamiento.
DatosImagen.Estado = ImagDesc;
DatosImagen.TotalGranos = Ngranos;

% Si la imagen se entiende por descartada, se guarda en los diferentes
% campos números negativos para ser fácilmente identificable a posteriori
ERROR = -1;
if ImagDesc
    DatosImagen.Aceptados = ERROR;
    DatosImagen.Descartados = ERROR;
    DatosImagen.Areas = ERROR;
    DatosImagen.Solidez = ERROR;
    DatosImagen.RelacionFarias = ERROR;
    DatosImagen.MaxFeret = ERROR;
    DatosImagen.TestKolmogorov = ERROR;
```

```
else
    % Si la imagen se entiende por válida, se almacenan los siguientes
    % parámetros obtenidos de ella en cada uno de sus campos
    DatosImagen.Aceptados = Nacep;
    DatosImagen.Descartados = Ndesc;
    DatosImagen.Areas = Areas;
    DatosImagen.Solidez = Solidez;
    DatosImagen.RelacionFarias = RelFarias;
    DatosImagen.MaxFeret = Feret;
    DatosImagen.TestKolmogorov = vKolmo;

end


end
```

Función para el test de Kolmogorov-Smirnov

test_kolmo.m

Función para obtener los valores del test de Kolmogorov-Smirnov

```
function distKS = test_kolmo (diametros)
% diametros = vector con los valores de los diámetros de los granos
% distKS = valor máximo de la distancia en el test


rMin = min(diametros);
rMax = max(diametros);


muExper    = mean(diametros);
sigmaExper = std(diametros);


coefDesv = 1.0;
muTeor = muExper*coefDesv;
sigmaTeor = sigmaExper*coefDesv;


anchoBin = 4;


fig = figure(100);
h = histogram (diametros, 'BinWidth', anchoBin,'BinLimits', [rMin,rMax]);
hAlturas = h.Values';


% Obtenemos el área del histograma:
areaHist = sum(h.Values)*anchoBin;


% Podemos muestrear la función teórica para que tenga el mismo número de muestras que
la experimental:
rCentrosBins = h.BinEdges(1:end-1)'+anchoBin/2;

% Test de Kolmogorov-Smirnov:
% Obtención de la distribución de probabilidad acumulada experimental (ECDF):
Fexper = cumsum(hAlturas*anchoBin);
```

```
% Obtención de la versión muestreada con el mismo número de elementos que la experimental:
FteorCentrosBins = GaussianCDF(rCentrosBins,muTeor,sigmaTeor);
FteorCentrosBinsEscalada = FteorCentrosBins * areaHist;




% El test de Kolmogorov-Smirnov se basa en la máxima distancia vertical que se observe entre las dos
% funciones de probabilidad acumuladas. Normalizamos con el valor final, que es la integral total:
distKS = max(abs(Fexper - FteorCentrosBinsEscalada)) / max(FteorCentrosBinsEscalada);


close(fig);
end
```

ANEXO B: OTRAS FUNCIONES

Función para incorporar Java a Matlab

copytoimagePlus.m

allows you to open an array I with an instance of ImageJ

within MATLAB with a proper data type and hyperstack dimensions.

```
function imp = copytoImagePlus(I,varargin)


import ij.process.ShortProcessor
import ij.process.ByteProcessor
import ij.process.FloatProcessor


p = inputParser;
p.addRequired('I',@(x) isnumeric(x));
p.addOptional('dimorder','YXCZT',@(x) ischar(x) && isrow(x) ...
    && all(arrayfun(@(y) ismember(y,'XYCZT'),upper(x))) && length(x) >=2 ...
    && all(arrayfun(@(y) ismember(y,'XY'),upper(x(1:2))))...
    );
p.addParameter('NewName','new',@(x) ischar(x) && isrow(x));
p.addParameter('FrameInterval',[],@(x) isreal(x) && x > 0);


p.parse(I,varargin{:});


dimorder = upper(p.Results.dimorder);
newname = p.Results.NewName;
frameinterval = p.Results.FrameInterval;




switch dimorder(1:2)
    case 'XY'
        order1 = [1 2];
    case 'YX'
        order1 = [2 1];
end


switch dimorder(3:ndims(I))
    case 'CZT'
        order2 = 3:5;
    case 'CTZ'
        order2 = [3 5 4];
    case 'ZCT'
        order2 = [4 3 5];
```

```
    case 'ZTC'
        order2 = [4 5 3];
    case 'TCZ'
        order2 = [5 3 4];
    case 'TZC'
        order2 = [5 4 3];
    case 'CZ'
        order2 = [3 4];
    case 'CT'
        order2 = [3 5 4];
    case 'ZC'
        order2 = [4 3];
    case 'ZT'
        order2 = [5 3 4];
    case 'TC'
        order2 = [4 5 3];
    case 'TZ'
        order2 = [5 4 3];
    case 'C'
        order2 = [3 4 5];
    case 'Z'
        order2 = [4 3 5];
    case 'T'
        order2 = [4 5 3];
    otherwise
        order2 = 3:5;
end


I0 = permute(I, [order1, order2]);


nX = int32(size(I0,1));
nY = int32(size(I0,2));
nC = int32(size(I0,3));
nZ = int32(size(I0,4));
nT = int32(size(I0,5));


try
    switch class(I0)
        case 'uint8'
            bitdepth = 8;


        case 'int8'

            bitdepth = 8;

        case 'uint16'

            bitdepth = 16;

        case 'int16'

            bitdepth = 8;
```

```
        case 'uint32'

            bitdepth = 32;

        case 'int32'

            bitdepth = 32;

        case 'uint64'
            error('MATLAB:copytoImg:UnsupportedType', ...
                'uint64 is not supported.');
        case 'int64'
            error('MATLAB:copytoImg:UnsupportedType', ...
                'uint64 is not supported.');
        case 'single'

            bitdepth = 32;

        case 'double'

            bitdepth = 32;

        case 'logical'

            bitdepth = 8;


        otherwise
            error('MATLAB:copytoImg:UnsupportedType', ...
                '%s is not supported.', class(I0));
    end


catch merr
    if strcmp(merr.identifier, 'MATLAB:undefinedVarOrClass')
            error('MATLAB:copytoImg:undefinedVarOrClass', ...
                'Could not find ImgLib2 on the path. Did you forget to run
''Miji(false)'' before calling this function?');
    else
        rethrow(merr);
    end
end


imp = ij.IJ.createHyperStack(newname,nX,nY,nC,nZ,nT,bitdepth);


for t = 1:nT
    imp.setT(t);
    for z = 1:nZ
        imp.setZ(z);
        for c = 1:nC
            imp.setC(c);

            XY = I0(:,:,c,z,t);
```

```
            xy = XY(:)';

            switch bitdepth
                case 16
                    ip = ShortProcessor(nX,nY);
                    ip.setPixels(xy);
                    imp.setProcessor(ip);
                case 8
                    ip = ByteProcessor(nX,nY);
                    ip.setPixels(xy);
                    imp.setProcessor(ip);
                otherwise
                    ip = FloatProcessor(nX,nY);
                    ip.setPixels(single(xy));
                    imp.setProcessor(ip);
            end
        end
    end
end


imp.setT(1);
imp.setZ(1);
imp.setC(1);
imp.setDisplayMode(ij.IJ.COLOR) %NOTE this is required to enable the next line
imp.setDisplayMode(ij.IJ.COMPOSITE)


try
    imp.resetDisplayRanges();
catch mexc
    if strcmpi(mexc.identifier,'MATLAB:UndefinedFunction')
        warning('resetDisplayRanges did not work')
    else
        throw(mexc)
    end
end


if ~isempty(frameinterval)

    fi = imp.getFileInfo();
    fi.frameInterval = frameinterval;
    imp.setFileInfo(fi);
end
end
```

Función para obtener la suma acumulada en la gaussiana

GaussianCDF.m

Cálculo de la Cummulative Distribution Function de una Gaussiana. Es decir, su integral.

```
function F = GaussianCDF (r, mu, sigma)


if length(r) == 1
   tabr = r * ones(1,1);
else
   tabr = r;
end


tabF = zeros(size(tabr));


for i=1:length(tabr)

   r = tabr(i);

   x = (r-mu)/sigma;


   % Término en serie infinita:
   nSumandos = 70;
   terminoSerieInfinita = 0;
   for n = 0:nSumandos-1
      terminoSerieInfinita = terminoSerieInfinita + x^(2*n+1)/prod(2*n+1:-2:1);
% Con la función 'prod()' implementamos de forma muy cómoda la función doble
factorial.
   end


   tabF(i) = 1/2 + 1/sqrt(2*pi) * exp(-x^2/2) * terminoSerieInfinita;
end


if length(tabF) == 1
   F = tabF(1);
else
   F = tabF;
end
```

REFERENCIAS

[1] «Tachos al vacío,» [En línea]. Available: https://www.ecured.cu/Tachos_al_vacio.

[2] Acor, «Remolacha y Azúcar,» [En línea]. Available: http://www.cooperativaacor.com/es/extraccion/art/189/.

[3] Mathworks, «Image Processing Toolbox,» [En línea]. Available: https://es.mathworks.com/products/image.html.

[4] ImageJ, «Image Processing and Analysis in Java,» [En línea]. Available: https://imagej.nih.gov/ij/.

[5] Acor, «Cooperativa Acor,» [En línea]. Available: http://www.cooperativaacor.com/es/acor/sec/68/.

[6] «Azucarera,» Conócenos, [En línea]. Available: https://www.azucarera.es/conocenos/.

[7] Iteca, «Iteca Socadei,» [En línea]. Available: https://www.iteca.fr/.

[8] Acor, «Cooperativa Acor. La extracción.,» [En línea]. Available: http://www.cooperativaacor.com/es/extraccion/art/189/.

[9] «Ministerio de Agricultura, Pesca y Alimentación,» [En línea]. Available: https://www.mapa.gob.es/es/agricultura/temas/producciones-agricolas/cultivos-herbaceos/remolacha-azucarera/.

[10] «Tubérculos,» [En línea]. Available: https://www.tuberculos.org/remolacha/azucarera/.

[11] SUCDEN, «Gránulos de pulpa de remolacha y melaza,» [En línea]. Available: https://www.sucden.com/es/products-and-services/sugar/beet-pulp-pellets-and-molasses/.

[12] C. Acor, «Youtube. ACOR: FÁBRICA DE AZÚCAR,» [En línea]. Available: https://youtu.be/UwiBTkRPtdk.

[13] C. d. M. S. COMEL, «La melaza en los piensos,» [En línea]. Available: http://www.ciademelazas.com/usumelazaanimal2.asp.

[14] S. Cerevisiae, «Evaluación de la producción de etanol,» *Biotecnología en el Sector Agropecuario y Agroindustrial,* vol. 13, nº 2, pp. 40 - 48, 2015.

[15] N. Faria, F. Rocha, J. Teixeira y A. Ferreira, «Using an Online Image Analysis Technique to Characterize Sucrose,» *I&EC Research. Industrial and Engineering Chemeistry Research.,* nº 50, p. 6990–7002, 2011.

[16] EcuRed, «Brix,» [En línea]. Available: https://www.ecured.cu/Brix.

[17] P. M. Martins, F. A. Rocha y P. Rein, «Modeling Sucrose Evaporative Crystallization. Part 1. Vacuum Pan,» *Ind. Eng. Chem. Res.,* vol. 44, pp. 8858-8864, 2005.

[18] Z. K. Nagy, G. Fevotte, H. Kramer y L. L. Simon, «Recent advances in the monitoring, modelling and control ofcrystallization systems,» *ELSEVIER,* nº 91, pp. 1903-1922, 2013.

[19] C. Borchert y K. Sundmacher, «Crystal Aggregation in a Flow Tube:,» *Chemical Engineering Technology,* nº 4, pp. 545- 556, 2011.

[20] C. Borchert, E. Temmel, H. Eisenschmidt, H. Lorenz, A. Seidel-Morgenstern y K. Sundmacher, «Image-Based in Situ Identification of Face Specific Crystal Growth,» *Crystal Growth and Design,* nº 14, pp. 952-971, 2014.

[21] N. Faria, M. Pons y F. Rocha, «Quantification of the morphology of sucrose crystals by image analysis,» *Powder Technology,* nº 133, pp. 54-67, 2003.

[22] W. Gonález y R. Woods, Digital Image Processing, Pearson, 1977.

[23] I. J, «Image J. Image Processing and Anatysis in Java.,» [En línea]. Available: https://imagej.nih.gov/ij/upgrade/.

[24] G. Hub, «Git Hub.,» [En línea]. Available: https://github.com/kouichi-c-nakamura/copytoImagePlus.

[25] «VEGA. Tacho de vacío para azúcar.,» [En línea]. Available: https://www.vega.com/es-es/home_es/sectores/alimentos/otras-aplicaciones/tacho-de-vac%C3%ADo-para-az%C3%BAcar.

[26] «Silver Weibull. Continuous Centrifugal.,» [En línea]. Available: http://www.silver-weibull.se/continous-centrifugal/.