ESCUELA DE INGENIERÍAS INDUSTRIALES

Departamento: Ingeniería Civil, de Materiales y Fabricación

Área de Conocimiento: Ingeniería de los Procesos de Fabricación

TRABAJO FIN DE GRADO

APLICACIÓN DE REDES NEURONALES ARTIFICIALES A LA CARACTERIZACIÓN GEOMÉTRICA DE LA VIRUTA EN EL MECANIZADO SOSTENIBLE DE ALEACIONES LIGERAS DE USO AERONÁUTICO

Grado en

Ingeniería en Tecnologías Industriales

Autor: D. Pablo Márquez Postigo

Tutor: D. Francisco Javier Trujillo Vilches

Cotutor: D. Fermín Bañón García

Málaga, Octubre de 2023

Agradecimientos

En esta sección, deseo expresar mi más sincero agradecimiento a todas las personas que han sido un apoyo fundamental durante el proceso de realización del presente trabajo de fin de grado, así como en el proceso de mis estudios de grado. Sin su colaboración, este logro habría sido inalcanzable. Sus contribuciones han sido un faro de luz en mi camino académico y, por ello, merecen mi más profundo reconocimiento.

Quiero expresar mi gratitud de manera especial a mis padres, Rosa y Alonso, así como a mi hermano Fernando, cuyo amor, apoyo incondicional y sacrificio han sido pilares en este viaje académico. Su constante respaldo me ha fortalecido en los momentos más desafiantes y, por ello, merecen mi más profundo reconocimiento.

Sin olvidarme de mi abuela Cristobalina, cuyo amor, sabiduría y apoyo han sido un faro en mi vida; ni de todos mis tíos y tías, quienes siempre me han apoyado de igual manera.

Además, quiero agradecer a todos y cada uno de mis amigos, quienes han sido fuentes inagotables de ánimo y apoyo a lo largo de esta travesía académica. Siempre han estado en los momentos buenos, así como en los malos.

Por último, me gustaría mostrar agradecimientos a mi tutor, Javier, a quien acudí desde un inicio para realizar el presente trabajo. Su orientación experta, dedicación incansable y sus desafiantes propuestas han sido fundamentales en la mejora continua de este trabajo.

Resumen

Las aleaciones ligeras, especialmente las de aluminio, se han utilizado tradicionalmente en la industria aeronáutica, principalmente en la fabricación de componentes estructurales de aeronaves. Entre los diferentes procesos de conformado utilizados en la fabricación de estas piezas, el mecanizado es uno de las más comunes, principalmente las operaciones de torneado, taladrado y fresado. Debido a razones medioambientales, la tendencia actual es la de realizar estos procesos en seco (ausencia total de fluidos de corte). En estas condiciones, la evacuación de viruta se convierte en un factor especialmente relevante. Dentro de este contexto se encuentra trabajando el grupo de investigación TEP933 “Ingeniería de Fabricación” de la Universidad de Málaga, en el marco del proyecto de investigación PID2021-125988OBI00, por título “SISTEMA EXPERTO PARA LA MEJORA DE LA INTEGRIDAD SUPERFICIAL EN EL MECANIZADO SOSTENIBLE DE ALEACIONES LIGERAS”, dentro del programa “Proyectos de Generación de Conocimiento” del “Programa Estatal para Impulsar la Investigación Científico-Técnica y su Transferencia del Plan Estatal de Investigación Científica, Técnica y de Innovación 2021-2023”. Entre los objetivos de este proyecto se encuentra la aplicación de técnicas de Machine Learning, en concreto la utilización de redes neuronales artificiales (ANN, Artificial Neural Networks), para la obtención de modelos predictivos de distintas variables de salida del proceso en función de los parámetros de corte. Entre estas variables se encuentran distintos parámetros geométricos relacionados con la morfología de la viruta.

De este modo, el objetivo principal de este Trabajo Fin de Grado es la utilización de redes neuronales artificiales (ANN) para la caracterización de distintos parámetros geométricos de la geometría de la viruta, en el mecanizado sostenible de aleaciones ligeras de uso aeronáutico.

En primer lugar, se han obtenido imágenes de tres tipos de aleaciones (Ti6Al4V, UNS A92024 y UNS A97075) provenientes de los previos procesos de mecanizado de una pieza, encapsulado de la viruta, ataque con ácido, escalado y fotografiado al microscopio. Una vez se tienen las imágenes de los tres tipos de viruta, entra en juego el primero de los objetivos de este trabajo, que es la clasificación automática con inteligencia artificial. Para ello se ha empleado la librería dedicada a las redes neuronales artificiales de Matlab (Neural Network Toolbox). El proceso consiste en la identificación de patrones, formas y colores de cada viruta por parte del programa, dando como resultado una clasificación automática entre los tres tipos de aleación con los que se trata. Una vez completada esta parte, se procedió con el procesamiento de la imagen, el cual consistió en seguir una serie de pasos tales como binarizado de la imagen, detección de contornos, picos y valles, identificación de elementos y obtención de parámetros de la viruta de forma automática (altura de picos y valles, factor de recalcado, ángulo de deslizamiento etc).

Los resultados obtenidos han sido comparados con los obtenidos mediante procedimientos no automatizados, utilizados previamente por el grupo, procediendo a la validación de la metodología empleada.

La metodología desarrollada en este TFG ha permitido ahorrar una gran cantidad de tiempo en la caracterización geométrica de la viruta obtenida bajo las condiciones estudiadas, en comparación con la metodología empleada por el grupo con anterioridad.

Palabras clave: Mecanizado / Aleaciones ligeras / Morfología de viruta / Machine Learning / Reconocimiento de imágenes / Inteligencia Artificial

Abstract

Light alloys, especially alluminium-based ones, have traditionally been used in the aerospace industry, primarily in the manufacturing of aircraft structural components. Among the various forming processes in the production of these parts, machining is one of the most common, particularly turning, drilling and milling operations. Due to environmental reasons, the current tren is to carry out these processes in a dry state (complete absence of cutting fluids). Under these conditions, chip evacuation becomes a particularly relevant factor. Within this context, the research group TEP933 “Manufacturing Engineering” at the University of Málaga is working within the framework of the research project PID2021-125988OBI00, titled “EXPERT SYSTEM FOR ENHANCING SURFACE INTEGRITY IN SUSTAINABLE MACHINING OF LIGHT ALLOYS”, as part of the “Knowledge Generation Projects” programo f the “State Program to Promote Scientific-Technical Research and its Transfer of the 2021-2023 State Plan for Scientific, Technical and innovation Research”. Among the objetives of this project is the application of Machine Learning techniques, specially the use of artificial neural networks (ANN), to obtain predictive models for various output variables of the process base don cutting parameters. These variables include different geometric parameters related to chip morphology.

Hence, the main goa lof this Bachelor’s Thesis is the utilization of artificial neural networks (ANN) for the characterization of different geometric parameters of chip morphology in the sustainable machining of lightweight alloys used in the aerospace industry.

Firstly, images of three types of alloys (Ti6Al4V, UNS A92024, and UNS A97075) are obtained from previous processes of machining a part, chip encapsulation, acid etching, scaling and photographed under a microscope. Once the images of the three types of chips are acquired, the first objetive of this work comes into play, which is automatic classification using artificial intelligence. For this purpose, the Matlab library dedicated to artificial neural networks (Neural Network Toolbox) has been employed. The process consists of identifying patterns, shapes and colors of each chip by the program, resulting in an automatic classification among the three types of alloys being studied. Once this part is completed, image processing follows, which involves a series of steps such as image binarization, contour detection, peaks and valleys identification, element recognition, and obtaining chip parameters automaticalli (peak and valley heights, shrinkage factor, shear angle, etc.).

The obtained results have been compared with those obtained through non-automated procedures previously used by the group, proceeding with the validation of the methodology employed.

The methodology developed in this Bachelor’s Thesis has allowed for a significant amount of time to be saved in the geometric characterization of the chip obtained under the studied conditions, compared to the methodology previously employed by the group.

Keywords: Machining / Light alloys / Chip Morphology / Machine Learning / Image Recognition / Artificial Intelligence

Índice de contenido

Índice de Figuras

Índice de Tablas

1. Introducción

1.1. Antecedentes

Tradicionalmente, las aleaciones ligeras han sido utilizadas en la industria aeronáutica, sobre todo para la fabricación de componentes estructurales. Son conocidas como aleaciones ligeras por su ratio propiedades mecánicas/densidad. Concretamente, las aleaciones más empleadas son las de aluminio y titanio. Estos componentes son críticos desde el punto de vista de la fiabilidad, por lo cual su diseño y fabricación están sometidos a unos niveles de calidad muy estrictos, siendo los márgenes de tolerancia muy estrechos.

Para la fabricación de este tipo de componentes, es necesario aplicar una serie de procesos de conformado, entre los cuales destacan los procesos de mecanizado, principalmente las operaciones de torneado, taladrado y fresado. Debido a razones medioambientales, se está tendiendo a realizar estos procesos en seco, lo que quiere decir que no se usan en ningún momento fluidos de corte. Esto trae consigo un problema, y es que debido a las altas temperaturas que se alcanzan por la ausencia de fluidos de corte, se producen fenómenos de adherencia de material a la herramienta de corte (fig. 1.1), dificultando así la evacuación de la viruta. Por otro lado, también influye en las desviaciones geométricas de la pieza que se está mecanizando, así como en la continuidad y estabilidad del proceso. Es por ello por lo que ganan importancia los aspectos de control y monitorización del proceso de formación de viruta de cara a la mejora de rendimiento [1,2].

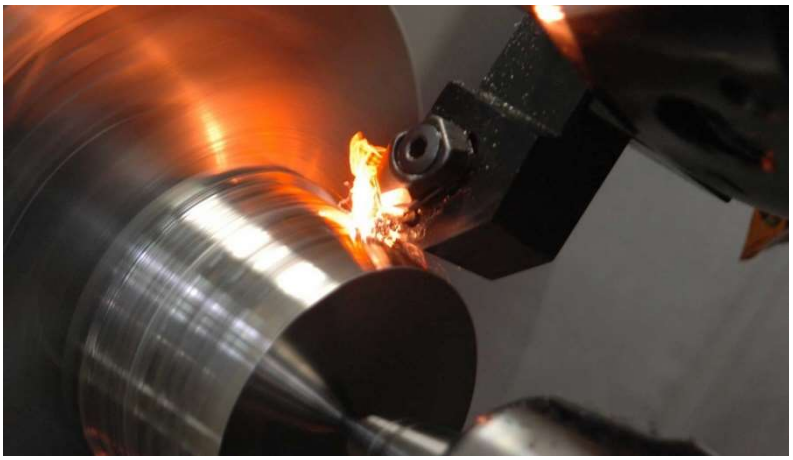


Figura 1.1- Fenómeno de adherencia de material a la herramienta de corte [3].

Dentro de este contexto se encuentra trabajando el grupo de investigación TEP933 “Ingeniería de Fabricación” de la Universidad de Málaga, en el marco del proyecto de investigación PID2021-125988OBI00, por título “SISTEMA EXPERTO PARA LA MEJORA DE LA INTEGRIDAD SUPERFICIAL EN EL MECANIZADO SOSTENIBLE DE ALEACIONES LIGERAS”, dentro del programa “Proyectos de Generación de Conocimiento” del “Programa Estatal para Impulsar la Investigación Científico-Técnica y su Transferencia del Plan Estatal para Impulsar la Investigación Científico-

Técnica y su Transferencia del Plan Estatal de Investigación Científica, Técnica y de Innovación 2021-2023".

Este es el contexto en el cual se sitúa este trabajo de fin de grado, el cual se enfoca en las aleaciones ligeras Ti6Al4V, UNS A92024 y UNS A97075 empleadas en la industria aeronáutica, siendo la primera de las mencionadas de titanio, mientras que las dos restantes pertenecen a la familia del aluminio. Se estudiarán formas de mejorar la eficiencia de los procesos de mecanizado para estas aleaciones de cara a reducir el impacto ambiental, ya que como se ha mencionado anteriormente, se tratará de eliminar los fluidos de corte en el proceso, abriendo la puerta al llamado *Dry Machining* o corte en seco [4].

1.2. Objetivos

El objetivo principal de este Trabajo Fin de Grado es la utilización de redes neuronales artificiales (ANN) para la caracterización de distintos parámetros geométricos de la geometría de la viruta, tales como el factor de recalcado, sección y espesor de la viruta, el grado de segmentación, entre otros. La metodología será de aplicación en el mecanizado en seco de distintas aleaciones ligeras de uso aeronáutico, tales como las aleaciones de aluminio UNS A97075 (Al-Zn) y UNS A92024 (Al-Cu), así como la aleación de Titanio TiA6Al4V. Partiendo de imágenes obtenidas previamente por el Grupo de Investigación, se pretende desarrollar una metodología que permita automatizar el proceso de medida de estos parámetros a partir de dichas imágenes. Para que esto sea posible, se hace uso del software de tratamiento de imágenes IMAGEJ, el cual cabe destacar que es de uso libre. Asimismo, se emplea el software Matlab tanto para la programación de la red neuronal artificial (ANN), como para el procesamiento digital que requieren las imágenes para obtener lograr mediciones automáticas de los parámetros de la viruta [5,6].

Al tratarse de un campo de investigación en el que la componente experimental tiene una gran importancia, se dan algunos objetivos particulares, aparte del objetivo principal ya mencionado. Uno de esos objetivos es que la metodología experimental del presente trabajo sirva como referencia de un modo u otro y que facilite futuras investigaciones sobre temas relacionados. Del mismo modo, se ha tomado como referencia la metodología aplicada por miembros de los grupos de investigación mencionados anteriormente, adaptando la misma al objetivo principal a llevar a cabo en este trabajo [4]. Para ello, los resultados obtenidos aplicando esta nueva metodología se han contrastado con los resultados obtenidos con metodologías previas aplicadas por el grupo de investigación, al objeto de realizar su validación.

1.3. Estructura de la memoria

La presente memoria está organizada en diferentes capítulos, cada uno de los cuales estará enfocado en abordar aspectos específicos relacionados con la

morfología de la viruta de aleaciones ligeras en procesos de mecanizado, así como en la aplicación de redes neuronales artificiales (ANN) para la obtención de parámetros de las propias virutas de manera automatizada. Cada capítulo de la memoria se estructura de forma coherente, proporcionando así un enfoque completo sobre el tema. Esta división en capítulos permite una presentación ordenada, mostrando de manera clara y concisa la investigación realizada.

En este primer capítulo, se presenta el contexto dentro del cual se desarrolla este TFG y sus objetivos. Por otro lado, se presentan los objetivos de este proyecto, que de forma resumida consiste en la aplicación del *Machine Learning* e Inteligencia artificial en el estudio de la viruta de las ya mencionadas aleaciones ligeras.

El segundo capítulo se centra en la caracterización morfológica de la viruta en procesos de mecanizado. El tema principal que se abarca en este capítulo es la importancia de estudiar y analizar la viruta de un material tras su mecanizado. Se tratan los diferentes mecanismos de formación de la viruta, ya que no todas las virutas se forman de igual manera. Se realiza una clasificación entre tipos diferenciables de viruta que se producen en el mecanizado y se tratan los factores que hacen que una viruta se forme de una manera o de otra. Para finalizar, se presentan los aspectos geométricos de la viruta que son de interés, haciendo especial énfasis en las aleaciones ligeras Ti6Al4V, UNS A92024 y UNS A97075.

El tercer capítulo está enfocado en la inteligencia artificial (IA), explicando qué es el *Machine Learning* y como puede ser utilizado para el reconocimiento de imágenes. Se habla también del uso de la inteligencia artificial en el campo de la industria, finalizando con el empleo de esta en reconocimiento de imágenes relacionadas con procesos de mecanizado.

El cuarto capítulo adquiere una gran importancia, ya que trata sobre la metodología experimental de este trabajo en concreto. En este apartado se muestran los datos de partida que se tienen como base, para posteriormente explicar los procedimientos y pasos seguidos para conseguir un reconocimiento automático de imágenes, así como el procesamiento de cada imagen. Para cerrar el capítulo, se muestra cómo se comprimen los algoritmos desarrollados en una aplicación informática de interfaz amigable, facilitando así el uso de este trabajo en líneas futuras de investigación.

El capítulo cinco es también de gran importancia, pues en él se muestran los resultados obtenidos al usar los algoritmos desarrollados en el capítulo anterior. Estos resultados vienen en forma de parámetros de la viruta, contando además con gráficas donde se puede ver la comparativa entre las mediciones realizadas por el algoritmo y las medidas obtenidas mediante el uso de una metodología no automatizada, empleada previamente por el grupo de investigación. Con ello se pretende validar la nueva metodología implementada mediante el uso de inteligencia artificial para el reconocimiento de imágenes

Para acabar, se da un sexto capítulo para las conclusiones, donde se explican tanto las conclusiones generales del trabajo, como las particulares sobre algún tema en

concreto, comentando las ventajas, desventajas y puntos donde se puede mejorar el trabajo realizado.

2. Caracterización morfológica de la viruta en procesos de mecanizado

2.1. Morfología de la viruta durante el mecanizado

En primer lugar, es de importancia explicar el concepto de viruta, el cual se encuentra referenciado como el material que se desprende por consecuencia de las modificaciones experimentadas por un material en un proceso de mecanizado. A primera vista, la viruta de mecanizado puede ser considerada simplemente como un residuo del proceso, sin ninguna utilidad aparente. Sin embargo, esta es una visión bastante equivocada, puesto que el estudio morfológico de la viruta trae consigo bastantes beneficios e información de utilidad de cara a mejorar los procesos de mecanizado, tanto en términos económicos como de rendimiento. De hecho, en todo proceso de arranque de material por mecanizado, la viruta juega un papel fundamental; es por ello por lo que surge el interés por analizarla. Su estudio da la oportunidad de observar a la viruta desde una perspectiva en la que no es un desecho metálico, sino un indicador el cual permite ciertas facetas del proceso de mecanizado, como puede ser la vida útil de la herramienta de corte o la calidad del material entre otras. Este estudio se realiza en torno a los factores geométricos de la viruta.

A lo largo de la historia del mecanizado, varios investigadores han realizado estudios sobre la caracterización de la viruta, tales como *F.W.Taylor*, quien desarrolló un modelo matemático capaz de caracterizar el desgaste de la herramienta a través del estudio de la viruta. También destacaron *Erns y Merchant*, realizando un análisis del proceso de corte observando el estado morfológico de la viruta. Más adelante, *Palmer y Oxel* propusieron una nueva teoría, focalizándose en la zona de cizalladura. Estos son únicamente algunos ejemplos de muchos investigadores destacados sobre el estudio morfológico de la viruta.

Después de este período, surge la era de la digitalización, la cual trae consigo múltiples avances tecnológicos en todas las ramas de la industria. En lo que al mecanizado respecta, la revolución informática permite la incorporación de computadores capaces de realizar operaciones de alta complejidad, mejorando así el acabado superficial de las piezas. Asimismo, surgen programas informáticos de simulación, los cuales son capaces de determinar comportamientos estructurales complejos, destacando métodos como el de elementos finitos.

Hoy en día, siguen existiendo estudios y teorías que relacionan los procesos de mecanizado con la morfología de la viruta. Estas se centran sobre todo tanto en la identificación de las variables que influyen en la formación de viruta como en su relación con cada parámetro de corte [7].

2.2. Mecanismos de formación de la viruta

El proceso de formación de viruta se lleva a cabo mediante un proceso de cizalladura, produciéndose una deformación plástica del material al incidir la herramienta de corte. Esto se produce a partir de una compresión que genera un desprendimiento de material en la denominada zona de corte. Cuando la herramienta incide sobre la pieza, se generan dos caras, siendo la cara de desprendimiento por la que cae la viruta, y la cara de incidencia la que sigue en contacto con la pieza (fig. 2.1).

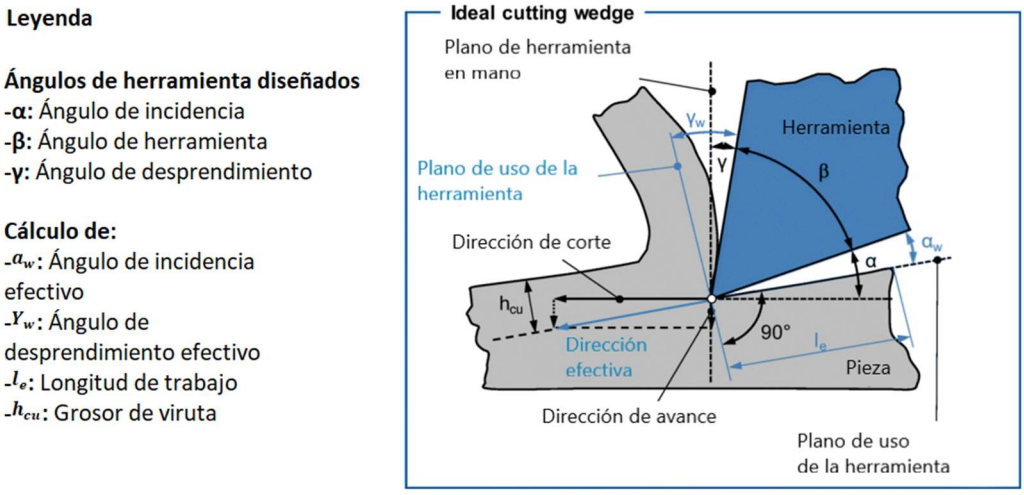


Figura 2.1- Esquema de formación de viruta [8].

En la figura puede apreciarse cómo la herramienta incide en el material a mecanizar. En este proceso se distinguen tres ángulos principales, que son los ángulos de desprendimiento (α), herramienta (β), e incidencia (γ). Al observar la formación de viruta en una imagen ampliada (fig. 2.2), se puede apreciar el mecanismo de formación de viruta en la punta de la herramienta, generando una estructura laminar que aumenta conforme avanza el proceso de mecanizado.

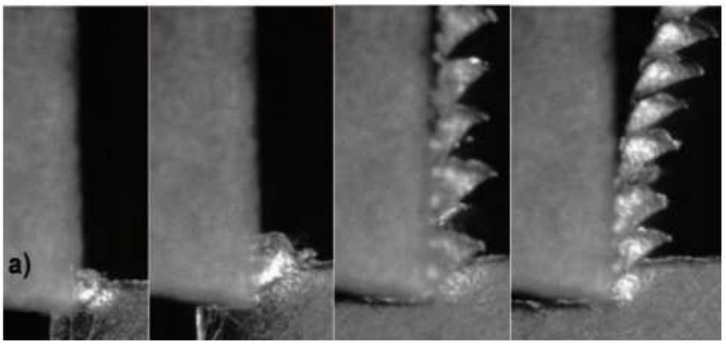


Figura 2.2- Formación de viruta en forma de dientes de sierra [7].

Para poder estudiar el mecanismo de formación de la viruta, es necesario observar cómo varía dicho mecanismo en función a la geometría que presenta tras el mecanizado y en base a como se produce el corte. Esto quiere decir que dos virutas resultantes serán diferentes si sus condiciones de corte también lo son. Gracias a conocer esto, se han ido creando diferentes teorías y modelos de formación de viruta, entre los cuales se destaca el modelo de *Time y Merchant* [9] (fig. 2.3).

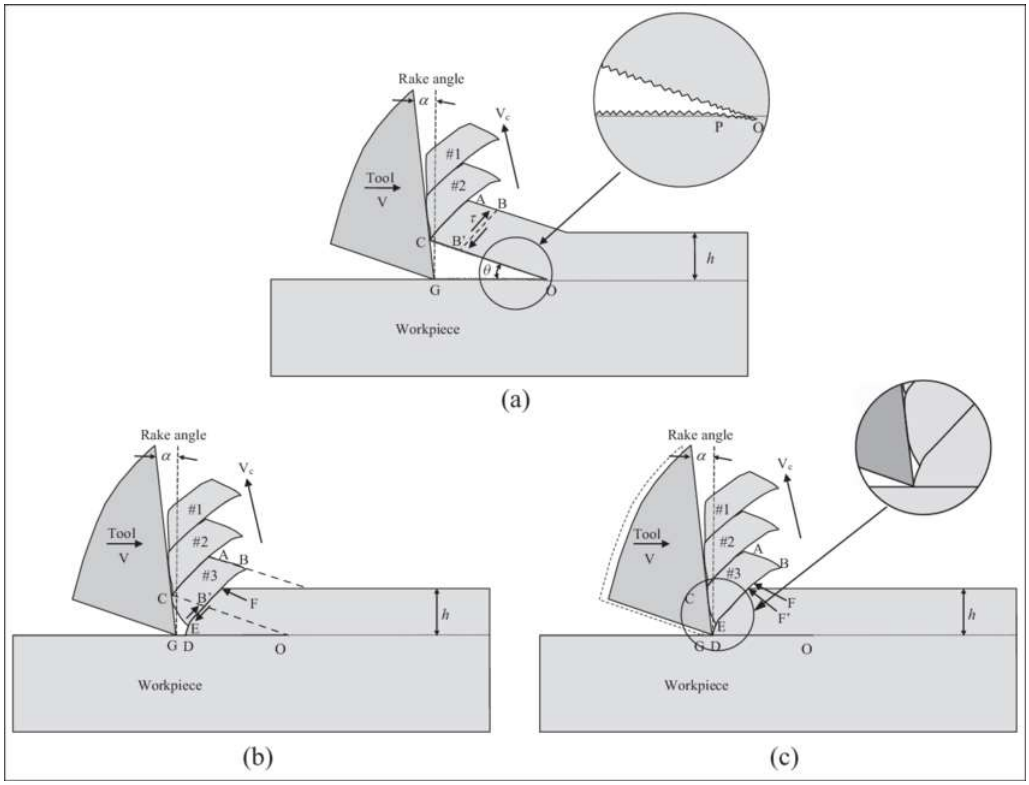


Figura 2.3- Esquema del modelo de Time y Merchant.

Este modelo se desarrolla en tres fases. En la primera de ella es en la que se genera el fenómeno de cizalladura o recalcado, que ocurre cuando el material mecanizado se desplaza por el filo de la herramienta de corte. Esto hace que se produzca una pequeña grieta en la zona de contacto, donde además se produce un aumento de la energía térmica debido a la deformación, dándose un ablandamiento en el material. En la segunda fase, al ser el esfuerzo cortante es bastante mayor que la resistencia del material en la zona de cizalladura, se generan una serie de bandas de deformación, ya que se corta una parte de la viruta. Para acabar, en la tercera y última fase, se da el desprendimiento de la viruta debido a la tensión que se genera en la cara de desprendimiento. La forma en la que se da el desprendimiento dependerá en su mayor medida al material sobre el que se está trabajando [7].

Otros modelos de formación de viruta destacados son los modelos de *Merchant* y *Stabler*, los cuales coinciden en que el proceso de corte se encuentra limitado por el plano de corte y la cara de desprendimiento; y el modelo de *Lee y Shaffer*, el cual se centra en la zona de plastificación de la viruta durante el proceso de mecanizado.

2.3. Tipos de viruta producidas en el mecanizado

Hasta el momento, se ha abarcado la importancia de la caracterización morfológica de la viruta, así como de sus mecanismos de formación. Tal y como se ha explicado, es un aspecto crucial identificar y controlar la forma de la viruta en el proceso de mecanizado; es por ello por lo que se hace necesario conocer de antemano el mecanismo de formación, para posteriormente poder estudiar las variables que influyen en la misma. Existen diferentes criterios para la clasificación de la viruta, siendo uno de los pioneros el planteado por *Ernst y Merchant* [10], el cual abarcaba 3 tipos de viruta en base a su forma, siendo forma continua, discontinua y filo recrecido. Posteriormente surgen nuevos criterios que complementan al anterior, añadiendo factores como la temperatura en la zona de corte, así como la velocidad y ángulo de corte y el espesor de viruta. A raíz de estos factores, es posible realizar una primera clasificación según la forma que adquiere la viruta, agrupándolas en viruta continua, fragmentada, segmentada e irregular [7].

Viruta continua

Es aquella que adquiere forma de cinta. Este tipo de viruta se obtiene cuando el material mecanizado es dúctil, empleando un rango de velocidades medias o altas y un ángulo de ataque agudo. Suele tener un buen acabado superficial, siempre y cuando la herramienta de corte se encuentre en buenas condiciones y tiende a enredarse entre sí, generando nidos (fig. 2.4).

Figura 2.4- Ejemplo de viruta continua [7].

Viruta fragmentada

Este tipo de viruta se da al mecanizar materiales de dureza media, para su obtención se emplea un rango bajo de velocidades de corte con alto avance y profundidad de corte. Tiende a darse vibración, por lo que el acabado superficial de la pieza no es bueno. La geometría del filo de corte se ve alterada tras el proceso (fig. 2.5).

Figura 2.5- Ejemplo de viruta fragmentada [11].

Viruta segmentada

Se produce en materiales de baja conductividad térmica y deformación cortante elevada. Se caracteriza porque es fácil de evacuar y tiene aspecto de diente de sierra en vista transversal. Las fuerzas de corte empleadas son variadas y la profundidad de corte es baja (fig. 2.6).

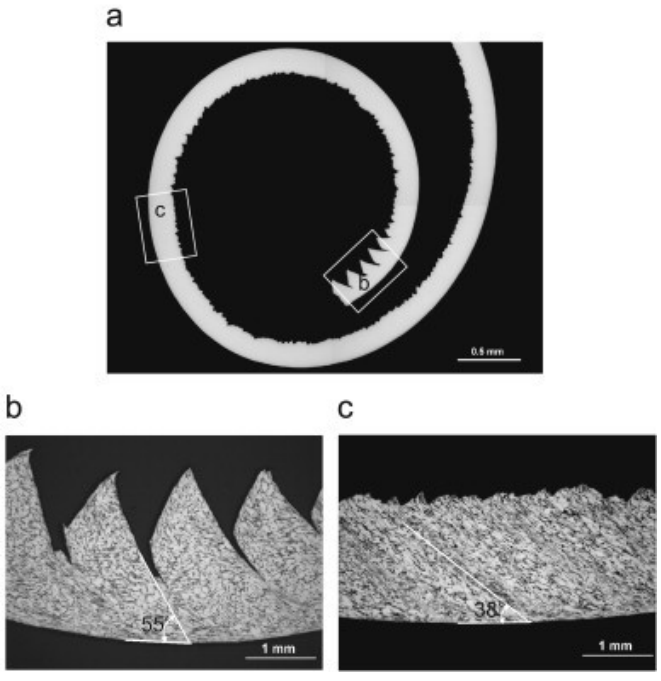


Figura 2.6- Ejemplo de viruta segmentada vista al microscopio [12]

a) Viruta completa, b) Zona de alta segmentación, c) Zona de baja segmentación.

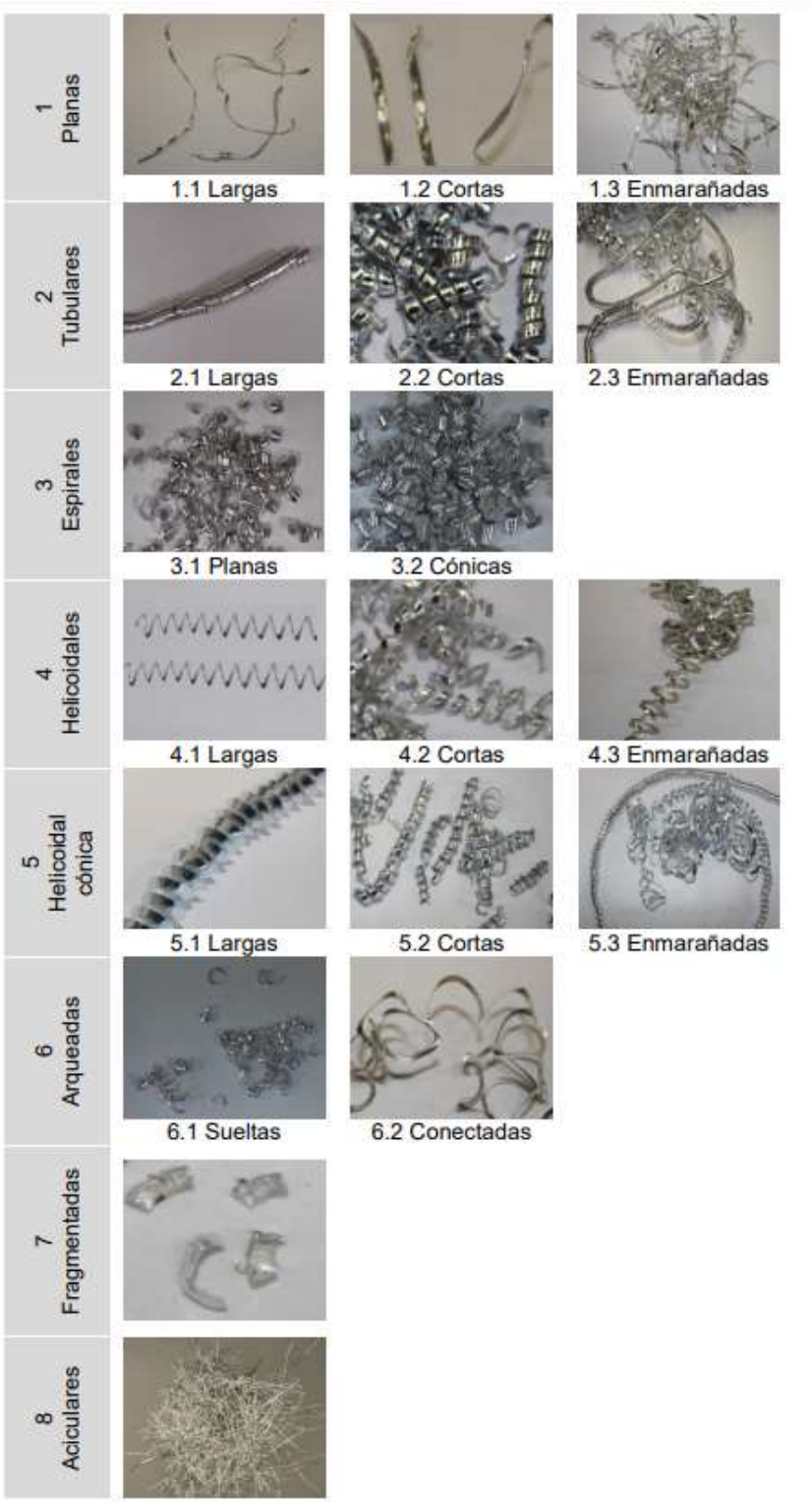
Viruta irregular

Es aquella que resulta de materiales frágiles, presentando una gran cantidad de impurezas duras. Se mecaniza con una baja velocidad de corte y un rango de profundidad de corte elevado, con un bajo ángulo de ataque. Su principal característica es que su mecanizado se realiza en seco (fig. 2.7).

Figura 2.7- Ejemplo de viruta irregular [13].

Sin embargo, hoy en día existe una clasificación estandarizada de tipos de viruta, la cual se recoge en la norma ISO 3685:1993 [14], la cual engloba las pruebas de vida útil de herramientas de mecanizado de un único punto de contacto. De esta norma se extrae la tabla de clasificación de virutas que se muestra en la Tabla 2.1.

Tabla 2.1- Tipos de virutas según la norma ISO 3685:1993 [4].



2.4. Factores de influencia en la morfología de la viruta

Una vez abarcado el concepto de morfología de la viruta, sus mecanismos de formación y cómo se clasifican se llega a la conclusión de que todo ello depende de ciertos factores. La comprensión de estos factores que dan lugar es crucial para poder optimizar los resultados de los procesos de mecanizado, así como mejorar la eficiencia de estos. A continuación, se detallan los 5 factores que desempeñan un papel determinante en la morfología de la viruta [7]. En la (fig. 2.x) se muestra un diagrama de Ishikawa que refleja estos factores.

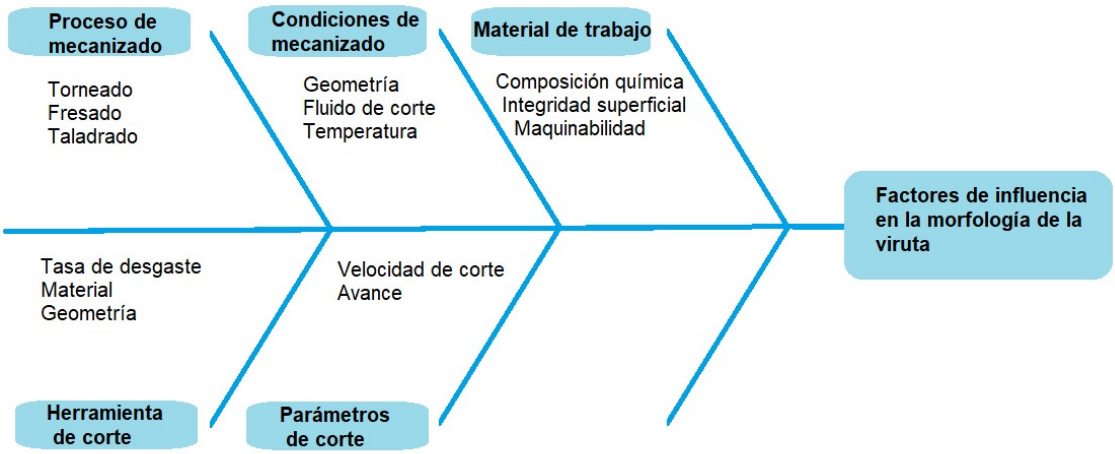


Figura 2.8- Diagrama de Ishikawa en el cual se muestran los factores que influyen en la morfología de la viruta.

Proceso de mecanizado

Desempeña un papel esencial en la formación de la viruta, afectando a su morfología de maneras específicas que variarán según la operación de mecanizado a la que se encuentren sometidas, siendo diferente para torneado, fresado, taladrado etc. Asimismo, también influye la maquinaria utilizada según como haya sido configurada para el proceso. Cada tipo de proceso de mecanizado y maquinaria empleada contribuyen a la generación de una viruta de características únicas en cada proceso de mecanizado.

Condiciones de mecanizado

Se tratan de variables fundamentales que se ajustan de acuerdo con varios factores, tales como la geometría de la pieza a mecanizar, la presencia y el tipo de fluido de corte utilizado, la temperatura alcanzada durante el proceso y la posible presencia de vibraciones. Estas condiciones, entrelazadas entre sí de forma compleja, juegan un papel muy importante en la formación de la viruta, ya que afecta a la interacción existente entre la herramienta de corte y la pieza. Cada ajuste en estas variables puede llegar a tener un gran impacto en la morfología resultante de la viruta.

Material de trabajo

El material de trabajo, del cual se obtiene posteriormente la pieza, se presenta como un elemento clave en el proceso. Este depende en su totalidad de su composición química, integridad superficial y maquinabilidad. La composición química del material influye en su comportamiento ante las fuerzas de corte, afectando directamente a la manera en que se deformará y segmentará durante el proceso. Por otro lado, la integridad superficial se refiere a la resistencia del material a las tensiones generadas por el filo de la herramienta de corte, contribuyendo a la forma en la que la viruta se desprenderá. Por último, la maquinabilidad es referida a la capacidad del material a ser trabajado, jugando un papel importante al determinar cómo se formará y separará la viruta del material inicial.

Herramienta de corte

Se trata de otro factor determinante que influye en la formación de la viruta. Cada tipo de herramienta de corte cuenta con características diferentes, tales como su tasa de desgaste, el material del que está compuesta y su geometría. Estas variables generan diferencias en la manera en la que se forma la viruta. La interacción entre la herramienta y el material a mecanizar determina cómo se propagan las fuerzas y tensiones, lo cual a su vez influye en la forma que tendrá la viruta tras el proceso.

Parámetros de corte

Desempeñan un papel bastante relevante, teniendo un impacto determinante en la formación de la viruta. Estos parámetros son las velocidades de corte y avance, así como la fuerza aplicada, los cuales determinan de manera significativa la interacción entre la herramienta de corte y el material. Variar estos parámetros conduce a variaciones importantes en la morfología de la viruta, ya que afectan directamente a la cantidad de energía generada en el proceso y a la distribución de las tensiones resultantes. Dicho en otras palabras, ajustar los parámetros de corte puede llevar a cambios considerables en la configuración de la viruta generada. Por ejemplo, pueden determinar si la viruta es continua o segmentada.

2.5. Aspectos geométricos de la viruta

Los aspectos geométricos son algo de especial importancia en el estudio de la viruta. De hecho, una buena parte de ellos son el resultado que se busca obtener de forma automática en este trabajo. En la (fig. 2.9) se muestran algunos de los aspectos geométricos a estudiar [7].

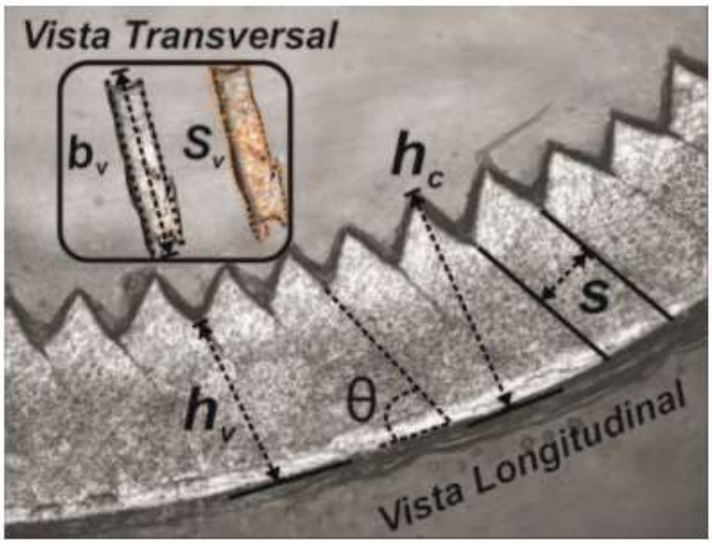


Figura 2.9- Aspectos geométricos de la viruta, tanto en vista transversal como longitudinal [7].

Cada vista es de utilidad para visualizar al microscopio ciertos aspectos en concreto. En la vista longitudinal se aprecian parámetros como la altura de pico (h_p), altura de valle (h_v), espesor de viruta (t) y ángulo de deslizamiento (φ). Por otro lado, en la vista transversal es posible observar valores como el ancho de la viruta (b) y la superficie (S_v).

Altura de pico (h_p)

Representa el espesor de material eliminado tras el mecanizado. Su valor depende de varios factores, que son el ángulo de filo de la herramienta (K_r), la velocidad de avance (f). Por otro lado, es necesario conocer que esta altura tendrá siempre un mayor valor que el espesor inicial del material antes del proceso de corte (h_0), valor que puede ser obtenido mediante la siguiente ecuación:

$$h_0 = f \cdot sin\, K_r \tag{1}$$

El principal beneficio de conocer la altura de pico no es más que identificar la influencia de f. Por lo tanto, se llega a la conclusión de que al aumentar f, la altura de pico se verá aumentada; mientras que al mantener una f baja, esta altura será menor.

Altura de valle (h_v)

La función de conocer este valor es determinar la inestabilidad termoplástica del proceso mediante la diferencia de altura entre pico y valle. Es esta diferencia de alturas la responsable de que se formen capas laminares al mecanizar. Conocer la secuencia de formación de estas capas permite comprender el concepto de segmentación en la formación de la viruta.

Espesor de viruta (t)

Una vez conocidos los valores de altura de valle (h_v) y espesor inicial de la viruta (h_0), es posible determinar el espesor de la viruta mediante la siguiente fórmula:

$$t_v = \frac{(h_0 - h_v)}{2} \tag{2}$$

El espesor de la viruta (t) se refiere al grosor de la porción de material que se desprende durante el proceso de mecanizado. Se trata de un parámetro fundamental en la caracterización de la viruta, influyendo de manera significativa tanto en la eficiencia del proceso de mecanizado en sí como en la calidad de la pieza final.

Ángulo de deslizamiento (φ)

Se encuentra comprendido en torno al plano de cizalladura, es por ello que también recibe el nombre de ángulo de cizalladura. Por debajo de este plano se encuentra la pieza sin deformar; mientras que por la parte superior del plano se encuentra la viruta desprendida. Para su obtención existen varios modelos, según el tipo de mecanizado que se esté realizando, destacando los que a continuación se muestran [15]:

Ernst & Merchant $$\varphi = 45º + \frac{\alpha}{2} + \frac{\beta}{2} + \frac{\gamma}{2} \tag{3}$$

Piispanen & Merchant $$\varphi = 45º - \frac{\beta}{2} + \frac{\alpha}{2} \tag{4}$$

Stabler $$\varphi = 45º - \beta + \frac{\alpha}{2} \tag{5}$$

Lee & Shaffer $$\varphi = 45º - \beta + \alpha \tag{6}$$

Oxley $$\gamma = \varphi + \beta - \alpha \tag{7}$$

Donde α es el ángulo de incidencia de la herramienta, β es el ángulo de herramienta o de corte y γ es el ángulo de desprendimiento de la viruta.

Ancho de viruta (b)

Se aprecia en la vista transversal de la viruta y se refiere a la distancia lateral a lo largo de la dirección de corte en la que se extiende la viruta formada mediante el proceso de mecanizado. El valor de b influye en aspectos como la evacuación de la propia viruta y de calor en el proceso. Su valor se puede obtener mediante la siguiente ecuación:

$$b = \frac{a_p}{\sin K_r} \tag{8}$$

Superficie de viruta (S_v)

Representa la cantidad de material que es arrancado de la pieza en bruto, el cual depende de la profundidad de corte (a_p), ya que a mayor profundidad se tenga, mayor será el área de mecanizado, así como la cantidad de material a eliminar. Por otro lado, la a_p no influye en la forma que tendrá la superficie tras el mecanizado. Hay diferentes métodos para calcularla, como por ejemplo los que a continuación se muestran:

$$S_v = a_p \times f \tag{8}$$

$$S_v = S \times a_p \tag{9}$$

2.6. Características de la viruta producida en el mecanizado de las aleaciones ligeras de uso aeronáutico Ti6Al4V, UNS A92024 y UNS A97075

Las aleaciones ligeras Ti6Al4V, UNS A92024 y UNS A97075 son empleadas en el ámbito de la aeronáutica, una industria en la cual los componentes estructurales requieren de un equilibrio preciso entre resistencia y peso. Es por ello por lo que cada una de estas aleaciones cuenta con propiedades únicas y usos específicos y sus respectivas virutas tendrán características propias que las diferencie entre sí y de virutas de otros tipos de aleaciones. En este contexto, se exploran las características distintivas de la viruta producida en el mecanizado de estas aleaciones, examinando cómo influyen las condiciones de mecanizado en su formación y morfología. Se muestran por un lado las características de la aleación Ti6Al4V, y por otro, las propiedades de las dos aleaciones de aluminio.

Características de la viruta de Ti6Al4V

Es una aleación de titanio compuesta principalmente de titanio (Ti), aluminio (Al) y vanadio (V). Destaca por su alta resistencia a la corrosión, gran relación resistencia-peso y biocompatibilidad, y se utiliza en aplicaciones aeroespaciales y biomédicas.

Esta aleación cuenta con unas propiedades excepcionales, lo que la convierte en una buena elección para muchas aplicaciones industriales, incluyendo el uso aeronáutico. Las propiedades que destacan más en esta aleación son su baja densidad, es decir, una relación resistencia-peso muy alta. Además, cuenta con una excelente resistencia a la corrosión, así como a las altas temperaturas. Es por ello por lo que, dentro de la industria aeronáutica, es utilizada en partes como las aspas de turbinas y las cámaras de combustión entre otras.

Al contar con estas tan buenas propiedades, trae consigo como contraparte problemas a la hora de mecanizar, es por ello por lo que la aleación de Ti6Al4V está incluida en el grupo de materiales difíciles de mecanizar. Uno de estos problemas es debido a la baja conductividad térmica del material, lo cual impide una rápida evacuación del calor generado, produciendo sobrecalentamiento en el filo de la herramienta y afectando así negativamente a su vida útil.

Tal y como era de esperar, todo esto provoca que las características de la viruta producida en el mecanizado de esta aleación sean únicas. Gracias a la investigación llevada a cabo por el grupo de investigación TEP933 “Ingeniería de Fabricación” de la Universidad de Málaga, es posible comprobar cómo varían las características de la viruta en función de la velocidad de corte (v_c) y el avance (f). En este estudio se aplica un rango de v_c que va desde 30 hasta 125 $\mathrm{m/min}$. En cuanto al avance, este se encuentra en un rango que va desde 0.05 hasta 0.30 $\mathrm{mm/r}$. En resumidas cuentas, se obtiene que al incrementar v_c y disminuir f (0.05-0.10 $\mathrm{mm/r}$), la morfología de la viruta es continua y helicoidal. Sin embargo, la viruta tiende a ser tubular y fragmentarse al aumentar el avance (0.20-0.30 $\mathrm{mm/r}$) combinado con una baja velocidad de corte (30-65 $\mathrm{m/min}$). De esta manera se aumenta la fuerza de corte, haciéndose más fácil romper la viruta.

Para concluir, los mejores resultados (fig. 2.10) desde el punto de vista de la maquinabilidad se obtienen mediante los valores más altos tanto de velocidad de corte como de avance por revolución [16].

Figura 2.10- Macro viruta de Ti6Al4V obtenida a $v_c = 30\ \mathbf{m/min}$ y $\mathbf{f} = 0.30\ \mathbf{mm/r}$ [1].

Características de las virutas de UNS A92024 y UNS A97075

Por un lado, UNS A92024 es una aleación de aluminio la cual contiene aluminio como componente principal y cobre como principal aleante. Esta aleación es conocida por su resistencia a la corrosión y su capacidad para mantener la resistencia a altas temperaturas. Se utiliza en aplicaciones donde se requiere alta resistencia, como la industria aeroespacial.

Por otro lado, UNS A97075 se trata de una aleación de aluminio cuyo principal aleante es el zinc, y se caracteriza por su alta resistencia y capacidad de endurecimiento. Es ampliamente utilizado en aplicaciones estructurales y aeroespaciales, donde se requiere una alta relación resistencia-peso.

Estas aleaciones ligeras, al igual que otras aleaciones de aluminio, se han utilizado tradicionalmente en la construcción de elementos críticos en la industria aeronáutica. Sin embargo, en las últimas décadas, se han ido incorporando al sector nuevos materiales, tales como la fibra de carbono reforzada y polímeros, los cuales se emplean de manera conjunta con estas aleaciones de aluminio. A pesar de las evoluciones, las aleaciones de aluminio continúan siendo vitales en la industria aeronáutica. Los componentes aeronáuticos suelen tener una vida útil de entre 15 y 20 años; esto ha permitido que las aleaciones de aluminio mantengan su utilidad en partes críticas tales como los elementos de tensión de las alas, nervios y otras partes del fuselaje entre otras muchas [17].

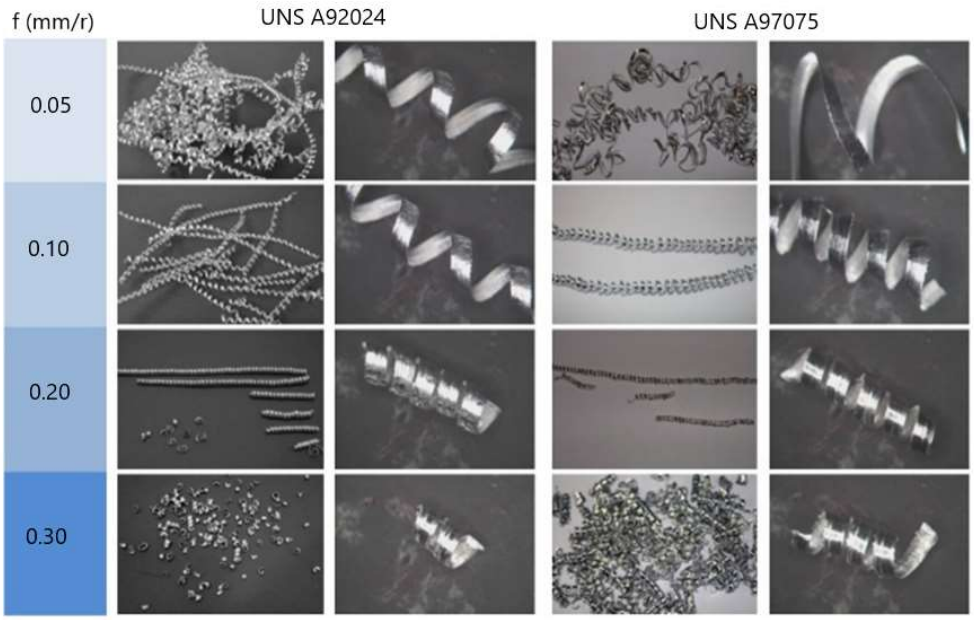
Al igual que se ha explicado en el apartado anterior en el caso de la aleación de titanio, el grupo de investigación TEP933 “Ingeniería de Fabricación” de la Universidad de Málaga ha realizado pruebas, esta vez con las aleaciones UNS A92024 y UNS A97075. En estas pruebas se modifica tanto la velocidad de corte (v_c), como el avance por revolución (f) y la profundidad de corte (a_p) con el fin de caracterizar la viruta procedente de estas aleaciones. Cabe mencionar que el proceso para obtener la morfología de la viruta de estas aleaciones es más complejo

que en la aleación de Ti6Al4V, dado que es necesario realizar un ataque químico sobre ella para así facilitar la observación de parámetros geométricos.

Los resultados obtenidos para la aleación UNS A92024 muestran una viruta continua para rangos bajos de f (0.05-0.10 $\mathrm{mm/r}$); y a medida que este aumenta, la viruta comienza a producirse de forma segmentada, y posteriormente fragmentada (0.20-0.30 $\mathrm{mm/r}$).

En cuanto a la aleación UNS A97075, presenta un comportamiento similar, con una viruta continua para velocidades de avance bajas y conforme esta aumenta, la viruta se empieza a dar de manera segmentada. La diferencia que se aprecia a simple vista es que para avances de 0.30 $\mathrm{mm/r}$, la viruta presenta más continuidad que en el caso de la aleación anterior (Tabla 2.2).

Tabla 2.2- Macro virutas de aleaciones UNS A92024 y UNS A97075 para diferentes velocidades de avance (f) [2].



A modo de conclusión, se muestran de manera resumida las principales diferencias fundamentales entre las aleaciones de titanio y aluminio estudiadas. En el caso de la aleación Ti6Al4V se obtiene un tipo de viruta segmentada, debido a la baja conductividad térmica del titanio. Este tipo de viruta es la menos deseada, ya que da lugar a inestabilidades en el proceso que repercuten en la calidad superficial obtenida y dan lugar a la aparición de fenómenos de vibración. En el caso de las aleaciones UNS A92024 y UNS A97075, el problema fundamental desde el punto de vista del control de la viruta es la obtención de viruta continua, debido a la elevada ductilidad del aluminio. Esto hace que sea difícilmente fragmentable, ya que el proceso no se alcanza la tensión límite a cortante del material, dando lugar a virutas de gran longitud que generan nidos de viruta. Esto supone un problema desde el punto de vista del proceso, dado que frecuentemente se tiene que parar el mismo para eliminar esos nidos generados. El uso

de rompevirutas no es efectivo en estas aleaciones, dado que su bajo punto de fusión hace que el rompevirutas se llene de material adherido, perdiendo su geometría inicial. Las opciones para mejorar la fragmentación pasan por incrementar el avance y la profundidad de corte, lo cual también repercute negativamente a los acabados superficiales. Por tanto, se trata de un tema complejo, requiriendo de estudios profundos para la mejora del rendimiento de estos.

En términos de avance, las virutas son continuas cuando este es bajo y segmentada para valores altos en ambas aleaciones. Sin embargo, la aleación UNS A97075 muestra una mayor continuidad en su morfología de viruta a medida que se aumenta f. Estas diferencias muestran la influencia tanto de la composición de cada aleación como de las condiciones de corte en la formación en forma de características únicas.

3. Inteligencia artificial aplicada al reconocimiento de imágenes en procesos de mecanizado

3.1. Concepto de Machine Learning

Hoy en día, es muy común encontrarse con el término de *Machine Learning*, ya sea leyendo artículos científicos, en televisión o en internet entre otros. Además, siempre está ligado con la inteligencia artificial, y esto se debe a que el *Machine Learning* es una rama de desarrollo de la inteligencia artificial.

Concretamente, el *Machine Learning* es una técnica de modelado que involucra datos. Es decir, emplea un conjunto de datos de entrada, ya sean documentos, audios o imágenes entre otros y logra un modelo, que es el producto final. El proceso que utiliza los datos para obtener el modelo es lo que se conoce como *Machine Learning*. Lo que hace especialmente interesante a esta técnica es que la propia máquina es la que extrae el modelo de los datos, en lugar de que sea un humano quien realiza la tarea. Es por ello por lo que a los datos de entrada se les denomina *Training Data* o datos de entrenamiento. Una vez obtenido el modelo, se emplea para trabajar sobre nuevos datos que esta vez no son de entrenamiento, sino que obtienen una predicción real. En la (fig. 3.1) se muestra un esquema de su funcionamiento.

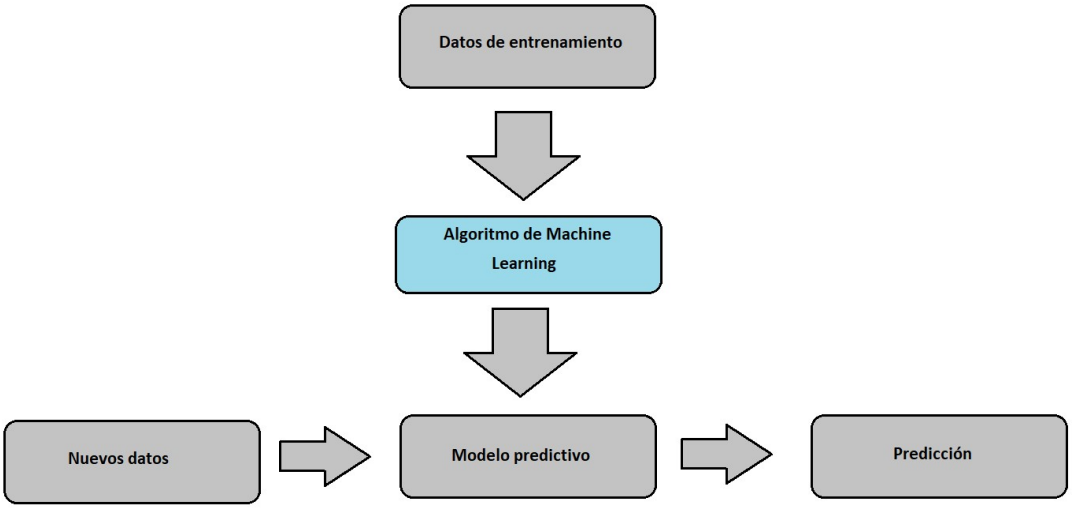


Figura 3.1- Esquema básico de funcionamiento del Machine Learning.

En muchas áreas de conocimiento, como por ejemplo la dinámica, existen modelos preestablecidos que se han ido aplicando a lo largo de los años, en este caso, las leyes de Newton. En el campo de la inteligencia artificial se desarrolla un modelo de resolución de problemas basado en modelo ya existente. Sin embargo, hay algunas áreas en las cuales las leyes y razonamientos lógicos no son demasiado útiles para el desarrollo de un modelo, como es el caso del reconocimiento de imágenes. Véase el siguiente ejemplo mostrado en la (fig. 3.2).

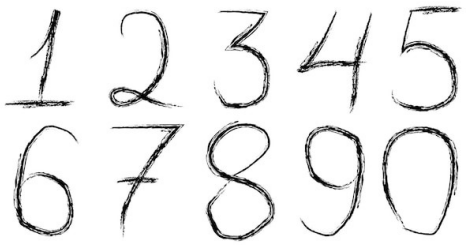


Figura 3.2- ¿Cómo un ordenador puede reconocer números sin un patrón reconocible?

Seguramente ha reconocido los números escritos a mano instantáneamente, tal y como lo haría la mayoría de los seres humanos. El siguiente paso es hacer que el computador realice la misma tarea. Si se quiere realizar mediante un modelado convencional, es necesario encontrar un algoritmo capaz de distinguir los números. No obstante, ¿Por qué no seguir las mismas reglas que el cerebro humano para reconocer los números? Desde pequeños, los seres humanos aprenden lo que es 0 y lo que es 1. ¿Por qué no dejar que los ordenadores hagan lo mismo? Ahí está la respuesta, en eso consiste el *Machine Learning*, el cual se comenzó a desarrollar en sus inicios para conseguir resolver problemas donde los métodos analíticos mediante ecuaciones no son viables [18].

3.2. Aplicación de técnicas de inteligencia artificial al reconocimiento de imágenes

Dentro del campo de la inteligencia artificial, el reconocimiento de imágenes es una de las aplicaciones más novedosas y avanzadas. Su desarrollo permite automatizar tareas tales como detección y clasificación de objetos presentes en una imagen o reconocimiento facial entre otras. Se ha llegado incluso al punto de incorporar el reconocimiento de imágenes a vehículos de conducción autónoma, de manera que se utiliza como visión artificial por computador para que los vehículos sean capaces de interpretar el entorno y tomar decisiones seguras. Para hacer esto posible, es necesario aplicar técnicas de inteligencia artificial y así aplicarlas a las imágenes.

La idea principal es buscar un aprendizaje automático por parte del computador, que es lo que se ha definido como *Machine Learning* en el anterior apartado. Para ello se emplean redes neuronales convolucionales (CNN, Convolutional Neural Networks), que son un tipo de arquitectura de aprendizaje profundo diseñada específicamente para el procesamiento de imágenes. Las CNN utilizan capas convolucionales capaces de detectar características de las imágenes, ya sean texturas, colores, formas o bordes. Después de esto, las capas convolucionales, conectadas entre sí, usan estas características para realizar una clasificación, regresión o la tarea que proceda. Para que todo esto funcione, se debe realizar en primer lugar un preprocesamiento de datos, que consiste en su mayor medida en preparar las imágenes para el entrenamiento. En último lugar se entrena la CNN utilizando conjuntos de imágenes etiquetadas, asociando cada imagen con una

etiqueta que indica su clase. Mientras más datos etiquetados se introduzcan, mejor será el entrenamiento y por consecuencia, tendrá una precisión mayor.

Profundizando en la clasificación de imágenes, esta puede ser llevada a cabo mediante redes neuronales pre entrenadas. En el caso del software *Matlab*, estas pueden ser instaladas como Add-on y hay muchas disponibles en el explorador, tales como *AlexNet, GoogleNet* o *Net50* entre otras. Estas CNN han sido previamente entrenadas con cientos de miles de imágenes y sus modelos cuentan con alrededor de 20 capas convolucionales con 1000 categorías de objetos. Cada capa convolucional tiene una función concreta en la clasificación, desde normalizar la imagen con una desviación estandarizada de 1, hasta el filtrado de parámetros concretos de la imagen. Al realizar un test de clasificación de imagen, el propio software evalúa la probabilidad de 0 a 1 entre todas las categorías presentes en la red pre entrenada para así asignar la imagen de prueba a un grupo, que en este caso será el más cercano a 1 [19].

3.3. Tipos de algoritmos

Dentro de la rama de conocimiento del *Machine Learning*, existen diversos tipos de algoritmos, los cuales pueden englobarse en tres grupos de algoritmos, que son el aprendizaje supervisado, sin supervisión y por refuerzo respectivamente.

El aprendizaje supervisado es aquel en el que se enseña a la máquina con el ejemplo. Es decir, un operador proporciona al algoritmo un conjunto de datos que ya son conocidos, incluyendo las entradas y salidas. El trabajo del algoritmo es encontrar un modo de llegar a dichas entradas y salidas. Cabe destacar que el operador conoce las respuestas correctas al problema, entonces dejará que el algoritmo realice sus predicciones y lo corregirá el número de veces que se considere necesario hasta que el algoritmo alcance el nivel de precisión deseado.

El aprendizaje sin supervisión se caracteriza porque no hay ningún operador humano que corrija al algoritmo, sino que es la propia máquina quien determina las correlaciones necesarias. Al no haber un operario, es necesario proporcionar una gran base de datos sobre la que trabajará el algoritmo. La precisión será mayor mientras más datos evalúe. Este tipo de algoritmos es bastante útil para la identificación de patrones.

El último grupo se trata del aprendizaje por refuerzo, que es aquel que se centra en procesos de aprendizaje reglamentados. En este caso se proporcionan acciones, parámetros y valores finales. Este sistema instruye a la máquina mediante el proceso de prueba y error, aprendiendo de experiencias pasadas y mejorando su precisión en cada ensayo.

Distribuidos entre los tres grupos ya mencionados, existen diversos tipos de algoritmos. A continuación, se explican brevemente algunos de los algoritmos de aprendizaje más comunes. [20].

Algoritmos de regresión

Este tipo de algoritmo es particularmente útil para realizar predicciones y pronósticos. Su función es estimar y comprender las relaciones entre variables en tareas de regresión. Sus datos de partida son una variable dependiente y un conjunto de variables cambiantes.

Algoritmos bayesianos

Tal y como su propio nombre indica, este tipo de algoritmos está basado en el Teorema de Bayes. Su manera de trabajo es clasificar cada valor como independiente del resto, lo cual permite predecir una categoría basándose en un conjunto de características, utilizando la probabilidad. Este tipo de algoritmos ha sido empleado en el presente trabajo, presentando una precisión sorprendentemente alta.

Algoritmos de agrupación

Son empleados en aprendizaje no supervisado para categorizar datos no etiquetados. Su funcionamiento se basa en la búsqueda de grupos dentro de esos datos. Trabaja de forma iterativa, asignando cada punto de datos a uno de los grupos según las características que presente cada objeto.

Algoritmos de árbol de decisión

Presentan una estructura similar a la de un diagrama de flujo, utilizando un método de bifurcación en el que se ilustra cada resultado posible de una decisión. Los nodos representan pruebas para una variable en concreto, mientras que las ramas representan el resultado de dichas pruebas.

Algoritmos de redes neuronales

La estructura de una red neuronal artificial (ANN) está inspirada en sistemas como el cerebro humano, comprendiendo unidades dispuestas en una serie de capas, cada una de las cuales está conectada al resto. Lo que más llama la atención de este tipo de algoritmos es cómo su elevado número de elementos interconectados trabajan al unísono para resolver una tarea específica. En este trabajo se han empleado ANN.

Algoritmos de Deep Learning

Su estructura es similar a la que presentan las redes neuronales artificiales. Lo que las diferencia de estas es que van un paso más allá, funcionando en conjuntos de datos que tienen cientos de características. Este tipo de algoritmo es ideal para trabajar con imágenes, ya que aprende progresivamente más sobre la imagen mientras pasa cada capa. La función de las primeras capas es detectar características simples como bordes y colores, mientras que las campas posteriores combinan la por información obtenida en las capas anteriores en una representación holística. En este trabajo se utiliza este tipo de algoritmos.

3.4. Aplicación en la industria

El *Machine Learning* es un recurso del cual se pueden obtener numerosos beneficios en la industria, de modo que se ha convertido en una herramienta imprescindible para la industria 4.0. Entre esas ventajas se pueden destacar las siguientes [21]:

- Aumento del nivel de planificación, aprovechando que el *Machine Learning* tiene una potente capacidad predictiva. Esto permite a las empresas actuar con antelación ante lo que está por venir y así poder actuar en consecuencia.

- Además de anticiparse a las cambiantes situaciones del mercado, el *Machine Learning* permite a las industrias saber si se encuentran preparadas para estas situaciones.

- Permite la automatización de diversos procesos repetitivos, permitiendo al personal humano dedicar su tiempo a otro tipo de tareas.

- Aumento de la calidad en la fabricación de productos, reduciendo así las pérdidas por devoluciones.

- Ahorro en costes logísticos, debido a una mejor organización.

Al darse este tipo de ventajas, se produce un crecimiento de las aplicaciones que esta novedosa tecnología tiene en las plantas industriales. Entre dichas aplicaciones, se pueden destacar algunas.

Mantenimiento predictivo

Gracias a la alta capacidad predictiva que ofrece el *Machine Learning*, puede ser aplicado en la planificación de operaciones de mantenimiento de la maquinaria. Esto funciona mediante estadística, permitiendo saber con antelación qué máquina tiene más probabilidades de fallar y cuando se dará. Gracias a esto, es posible una anticipación y una actuación a tiempo antes de que el equipo quede inutilizado, generando pérdidas económicas (fig. 3.3).

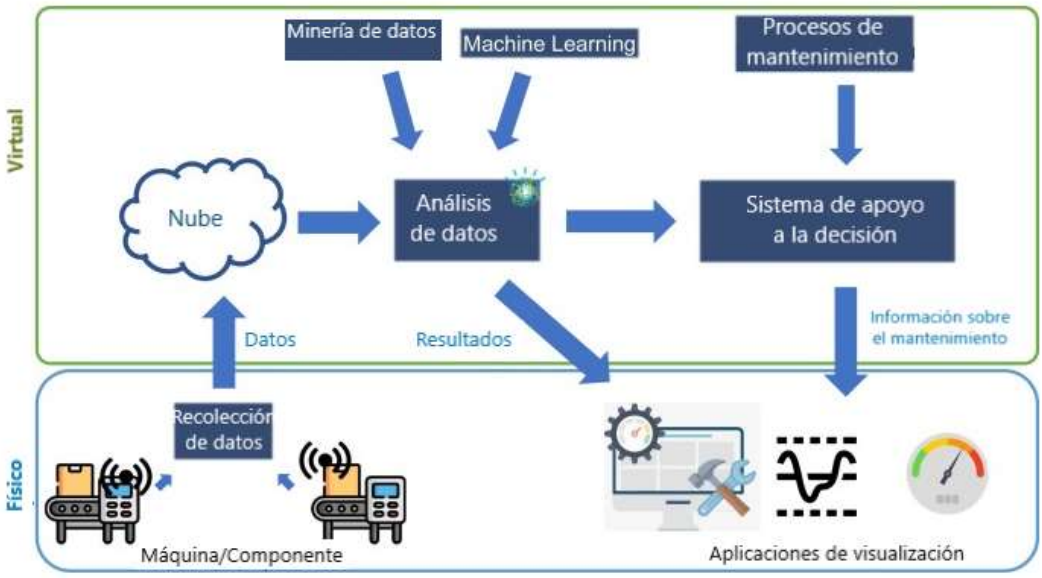


Figura 3.3- Esquema de mantenimiento predictivo empleando machine Learning [22].

Visión artificial

Un caso perfecto para explicar esta aplicación es el del presente trabajo, en el que, como se explica más adelante en el capítulo 4; se emplearán técnicas de visión artificial para extraer datos de imágenes de viruta de tres tipos de aleaciones. Esto permite un procesado automático de las imágenes recogidas en una industria, extrayendo datos de valor de ellas. De este modo, aumenta exponencialmente la velocidad y fiabilidad de las inspecciones visuales, eliminando factores humanos tales como la fatiga y confusiones (fig. 3.4).

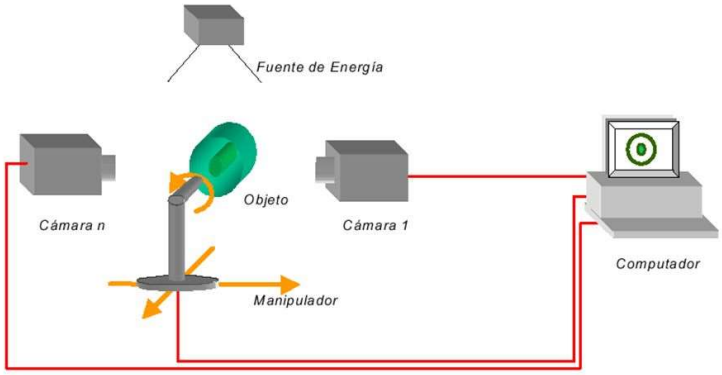


Figura 3.4- Esquema de visión artificial mediante Machine Learning [23].

Control de calidad

En cuanto a control de calidad de materias primas y productos elaborados se trata, también ha crecido la importancia del *Machine Learning*. En primer lugar, vuelve a aparecer la visión artificial, permitiendo identificar defectos con mucha más eficiencia que el ojo humano. Sin embargo, además de la visión artificial, la inteligencia artificial permite recopilar y monitorizar constantemente datos como la temperatura, la presión o la vibración, controlando así automáticamente qué productos pasan el control de calidad y cuáles no (fig. 3.5).

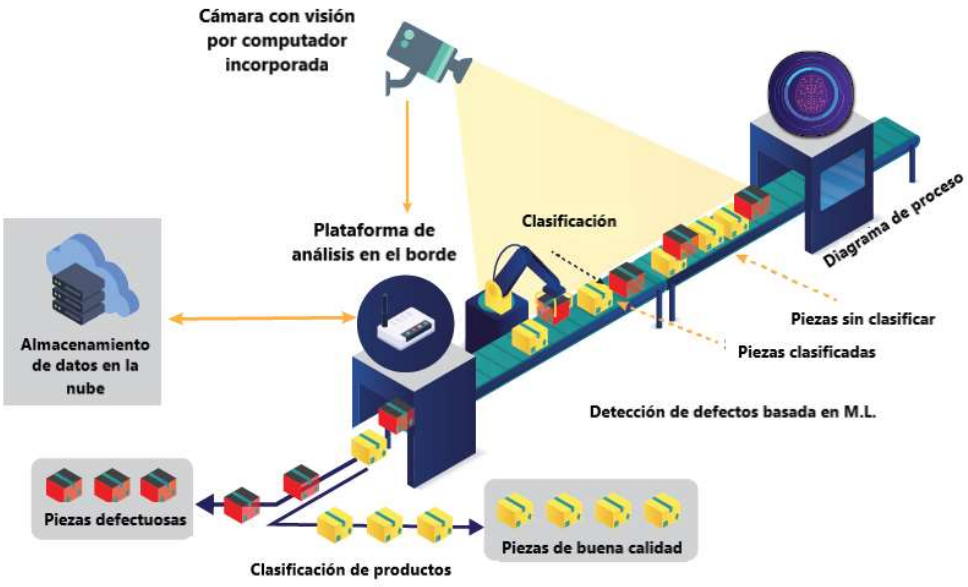


Figura 3.5- Diagrama de proceso de control de calidad mediante Machine Learning [24].

Optimización de recursos

El *Machine Learning* es capaz de trabajar con todo tipo de información, incluyendo a datos sobre recursos, tanto materiales como humanos. Esto permite una óptima organización de las tareas de los empleados, dependiendo de factores como su ubicación, habilidades, rendimiento etc. Asimismo, también se mejora la optimización de los bienes materiales, como los productos en stock.

Clasificación de productos

Volviendo a la visión artificial, al igual que puede ser empleada para controles de calidad, también puede ser usada de cara a la clasificación de objetos, utilizando cámaras y sensores para identificar aspectos en un producto decidiendo así si va a un grupo u otro (fig. 3.6).

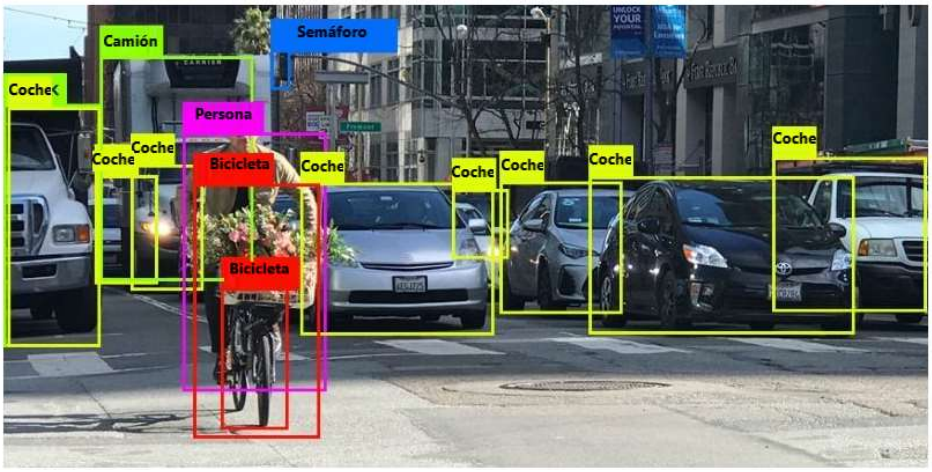


Figura 3.6- Ejemplo de clasificación de objetos mediante Machine Learning [25].

3.5. Aplicación de redes neuronales artificiales en mecanizado

Al igual que en otras áreas de la industria las redes neuronales están teniendo un gran crecimiento, el mecanizado no podía ser menos. De hecho, han ayudado en gran medida a mejorar la eficiencia, calidad y precisión en sus procesos. A continuación, se muestran algunas de las aplicaciones más destacadas dentro de los procesos de mecanizado.

Predicción y optimización de parámetros de corte

Con un buen entrenamiento, las redes neuronales pueden ser de gran ayuda, consiguiendo una predicción de los parámetros de corte óptimos en operaciones de mecanizado, tales como velocidad de corte, velocidad de avance y profundidad de corte. Con esto se consigue una importante reducción del tiempo de prueba y error, mejorando así tanto la productividad como la vida útil de las herramientas de corte.

Detección de defectos en piezas

Tal y como se ha mencionado anteriormente, una red neuronal artificial (ANN) cuenta de lejos con una mayor eficacia y velocidad que el ojo humano a la hora de detectar defectos. En este caso, se emplea una ANN para inspeccionar visualmente una pieza tras su mecanizado y detectar desperfectos e imperfecciones en caso de haberlas. Con esto se logra una clasificación rápida y precisa de las piezas mecanizadas, clasificándolas como aceptables o defectuosas, reduciendo además los costes de control de calidad (fig. 3.7).

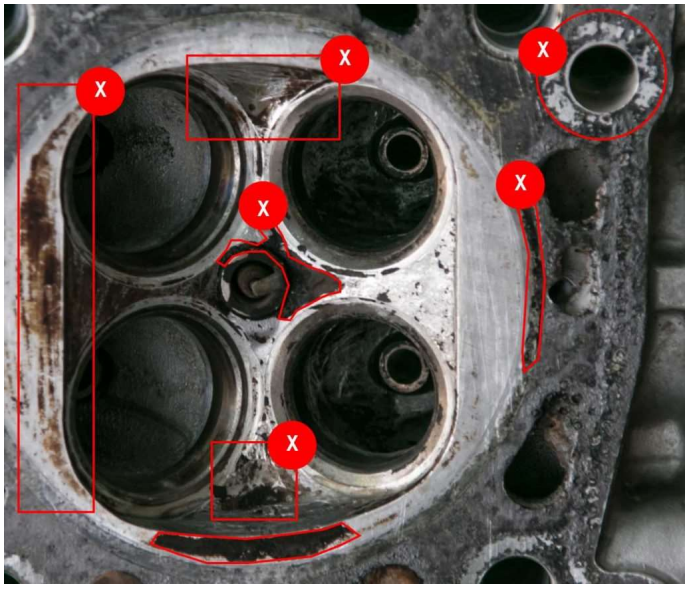


Figura 3.7- Detección de defectos mediante visión artificial [26].

Monitorización de maquinaria

Este es un uso de gran importancia de las ANN, del cual se ha hablado en el apartado anterior, y es que son capaces de analizar una gran cantidad de datos recopilados mediante sensores en las máquinas de mecanizado. Esto permite realizar un mantenimiento predictivo de manera óptima, prediciendo fallos o problemas antes de que ocurran y planificando las fechas de mantenimiento con un margen de tiempo adecuado. De este modo, se evitan tiempos de inactividad de la máquina no planificados.

Clasificación de materiales

Se trata de algo similar a lo que se realiza en este trabajo, y es que las ANN son capaces de clasificar un gran abanico de materiales diferentes en función de sus propiedades físicas y químicas. Esto es bastante útil a la hora de ajustar parámetros de corte en una máquina en relación con el material con el que se vaya a trabajar.

Reducción de vibraciones

Es bien sabido que las vibraciones no deseadas en una máquina de mecanizado pueden ser perjudicial tanto para la propia herramienta de corte como para la pieza mecanizada. Es por ello por lo que una ANN puede ser entrenada para ajustar automáticamente los parámetros de mecanizado en tiempo real, evitando así dichas vibraciones y por consecuencia, conseguir una mejora en la calidad superficial de la pieza (fig. 3.8).

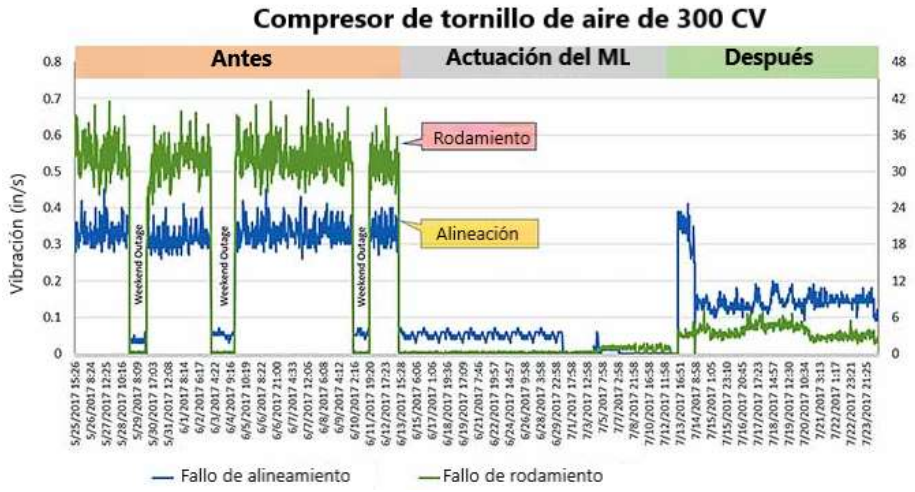


Figura 3.8- Gráfica de reducción de vibraciones mediante Machine Learning [27].

Control adaptativo

Un control adaptativo es aquel que es capaz de modificar su comportamiento en respuesta a cambios en la dinámica del sistema y a sus perturbaciones. En el caso del mecanizado, existen factores cambiantes en el procesado de cada unidad, como pueden ser las irregularidades del material o desgaste de la herramienta. Gracias a las ANN es posible ajustar los parámetros de corte en tiempo real de manera automática, dependiendo de las condiciones de la máquina, la herramienta y el material en cada momento.

3.6. Aplicación del reconocimiento de imágenes mediante redes neuronales a los procesos de mecanizado

El reconocimiento de imágenes por medio de redes neuronales artificiales (ANN) en los procesos de mecanizado tiene bastantes aplicaciones, debido a su versatilidad. La principal utilidad de este reconocimiento automático de imágenes es la de extraer datos de imágenes, de manera precisa y en grandes cantidades, con el objetivo de emplear estos resultados en mejorar procesos. Entre las aplicaciones existentes, las más destacadas se muestran a continuación.

Identificación y obtención de parámetros de virutas

Esta aplicación se coloca en primer lugar, debido a que se trata ni más ni menos que de el objetivo de este trabajo. Tal y como se explicará más adelante en el capítulo 4, el empleo de ANN permite una identificación automática de tipo de viruta por medio de reconocimiento de imágenes de estas. Además, esto abre la puerta a un posterior procesamiento de imagen con el fin de obtener de forma automática varios resultados útiles para conocer el comportamiento tanto de la propia viruta como de la herramienta de corte.

Clasificación de herramientas de corte

Las ANN pueden ser entrenadas para que, mediante reconocimiento de imagen, sean capaces de detectar y clasificar distintos tipos de herramientas de corte. El principal beneficio que esto conlleva es verificar que se encuentra en uso la herramienta correcta con relación al material que se está mecanizando, con el objetivo de evitar daños y errores que se puedan dar por el uso de una herramienta incorrecta (fig. 3.9).

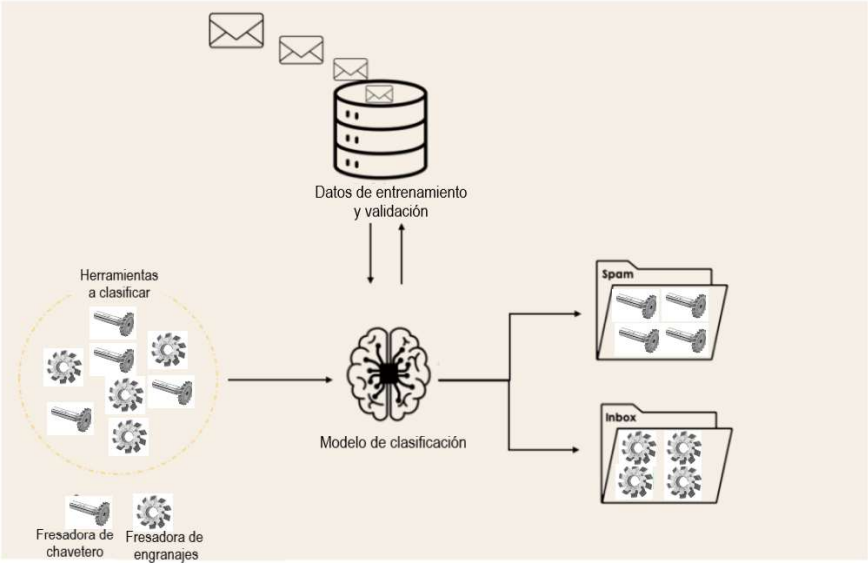


Figura 3.9- Gráfica de reducción de vibraciones mediante Machine Learning [28].

Posicionamiento automático

En muchos de los procesos de mecanizado, se dan múltiples operaciones sobre una única pieza. El reconocimiento de imágenes puede analizar las imágenes de este tipo de piezas en tiempo real, mientras ocurre el proceso. Lo que se consigue con esto es un ajuste automático tanto de la posición como de la alineación de la pieza, logrando un proceso de mecanizado preciso, y por consecuencia logrando una reducción de costes en piezas defectuosas.

Detección de desgaste en herramientas

Analizando las imágenes de una herramienta de corte en tiempo real durante el proceso de mecanizado, las ANN son capaces de detectar los primeros signos de desgaste en el filo de corte. Además, también pueden identificar cuándo estos signos de desgaste llegan al punto en el que reducen la eficacia o inutilizan la herramienta, enviando una señal de alerta sobre la necesidad de reemplazo del filo.

Inspección de geometría

Una vez mecanizada una pieza, mediante el uso de ANN es posible evaluar la geometría de esta, determinando si es apta o no para su distribución. Esta tarea cobra una gran importancia, sobre todo en industrias en las que la precisión dimensional es crítica, como podría ser la aeronáutica o la nanotecnología (fig. 3.10).

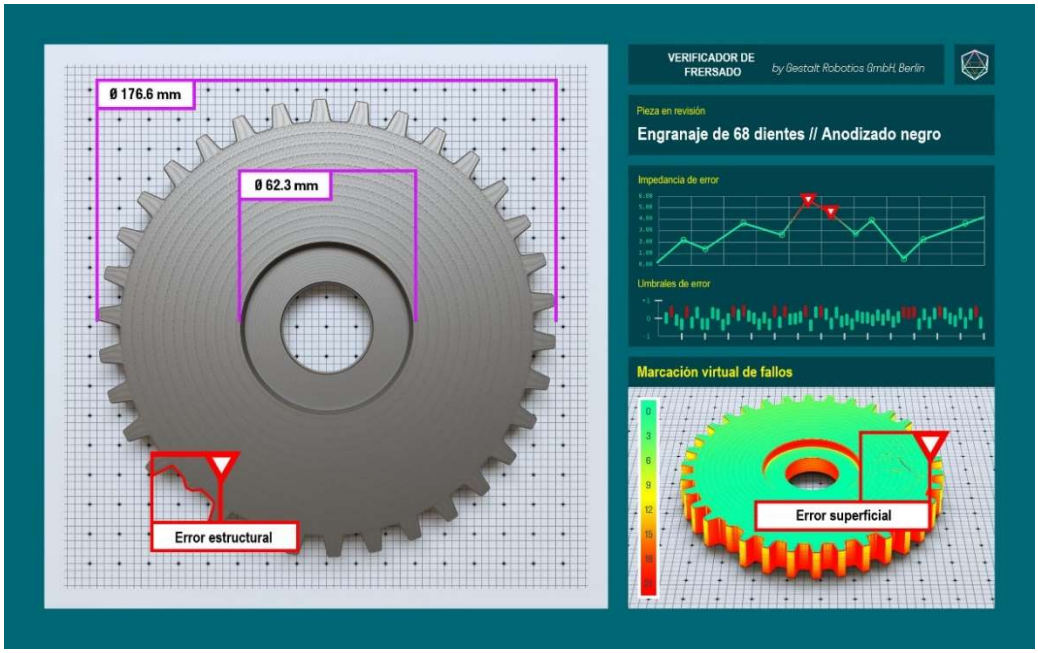


Figura 3.9- Inspección de la geometría de un engranaje mecanizado mediante el uso de ANN [29].

Conclusiones

El uso de redes neuronales en el mecanizado ha demostrado ser una herramienta de vanguardia con aplicaciones prometedoras en diversas áreas. La identificación y obtención de parámetros de virutas, la clasificación de herramientas de corte, el posicionamiento automático, la detección de desgaste en herramientas y la inspección de geometría son solo algunas de las aplicaciones destacadas. Estas tecnologías no solo mejoran la eficiencia y precisión los procesos de mecanizado, sino que también permiten una monitorización en tiempo real y una toma de decisiones más rápida. El uso de redes neuronales en el mecanizado es un campo en auge que promete seguir evolucionando y transformando la industria manufacturera, ofreciendo soluciones innovadoras para desafíos cada vez más complejos.

4. Metodología experimental

4.1. Introducción

El objetivo de este capítulo es explicar detalladamente el procedimiento empleado para la clasificación y procesamiento de imágenes de los tres tipos de aleaciones ligeras con las que se ha tratado en este trabajo (Ti6Al4V, UNS A92024 y UNS A97075), todo ello partiendo desde una serie de datos iniciales.

El primer punto que tratar es la elaboración de una red neuronal artificial (ANN), desarrollada en el entorno del software Matlab y cuya función es clasificar correctamente imágenes de viruta de los tres tipos de aleaciones ligeras con las que se ha trabajado. Esto se hace posible gracias a un entrenamiento de la red neuronal, la cual es capaz de reconocer patrones, formas, tamaño y color entre otros factores para así determinar de qué aleación se trata la viruta sometida a reconocimiento.

Una vez conocido el tipo de aleación, es necesario procesar la imagen seleccionada para la detección de contornos, máximos, mínimos y reconocimiento de dientes de la viruta, para así poder conocer los valores que se necesitan medir (altura de picos, altura de valles, factor de recalcado etc.). Es entonces cuando entra en juego el procesamiento digital de imágenes, el cual consiste en una serie de pasos simples, pero que en su conjunto forman un proceso más complejo; comenzando por el binarizado de la imagen hasta la obtención de todos los valores de interés.

En última instancia, se explica cómo se ha llevado a cabo el desarrollo de una aplicación (APP, application) en el entorno de Matlab, la cual permite realizar lo anteriormente mencionado (reconocimiento y clasificación de imagen y procesamiento digital de la misma) de una forma bastante más cómoda.

A modo de resumen, el procedimiento empleado puede clasificarse en tres bloques; siendo estos la construcción de la ANN, el desarrollo del algoritmo para el procesamiento de imágenes y el desarrollo de la APP.

Los pasos seguidos para la construcción de la ANN son los siguientes: Seleccionar una red neuronal pre-entrenada, elegir y escalar las imágenes empleadas para el entrenamiento, realizar el propio entrenamiento, generar una matriz de confusión y realizar pruebas con imágenes reservadas que no hayan sido empleadas para el entrenamiento.

Para realizar el algoritmo de procesamiento automático de imágenes, se ha realizado lo siguiente sobre la imagen: Escalado, binarizado, obtención de picos y valles, segmentado y obtención de parámetros.

Finalmente, se han tomado tanto los datos del entrenamiento de la ANN como el algoritmo de procesamiento digital de imágenes y se han incorporado en una APP. De este modo, es posible trabajar sobre las imágenes de una manera mucho más cómoda.

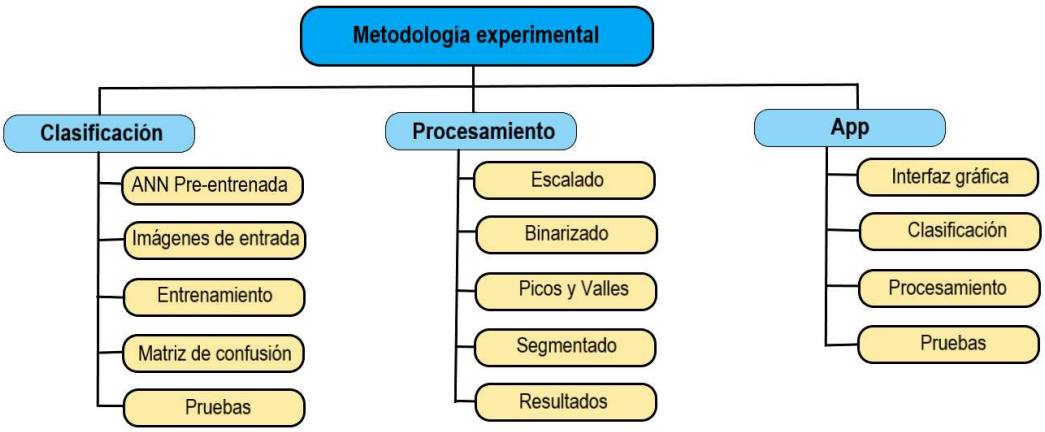


Figura 4.1- Diagrama de flujo donde se muestran los pasos seguidos en la metodología experimental.

4.2. Datos de partida

En el marco de este trabajo de fin de grado, el objetivo principal se trata del desarrollo de un algoritmo con la capacidad de reconocer y clasificar imágenes de virutas de 3 tipos de aleaciones, que son Ti6Al4V, UNS A92024 y UNS A97075; para realizar un posterior procesamiento de imagen, obteniendo ciertos resultados relacionados con la geometría de la viruta. Para hacer esto posible, se ha partido de una sólida base de datos iniciales, entre los cuales hay imágenes, libros, artículos científicos, así como otras investigaciones. Gracias a esto, ha sido posible establecer ciertos fundamentos a partir de los cuales se ha construido el presente trabajo. La calidad de esta base de datos ha resultado ser un recurso de gran valor para obtener resultados sólidos tanto en el reconocimiento como en el procesamiento de imágenes. Entre todos los datos de partida, se podrían destacar los que a continuación se detallan. Cabe resaltar que la mayor parte de los datos de partida se han obtenido a partir de los trabajos realizados por el grupo de investigación TEP933 "Ingeniería de Fabricación" de la Universidad de Málaga, a lo largo de más de 15 años de trabajo en su principal linea de investigación, el mecanizado sostenible de aleaciones ligeras de uso aeronáutico.

Biblioteca de imágenes

Un recurso crucial para hacer posible el desarrollo de este trabajo es una buena base de datos de imágenes, pues sin ella no sería posible realizar un entrenamiento de la red neuronal artificial (ANN) para reconocer imágenes, ni tampoco habría ejemplos para realizar pruebas de procesamiento de imagen sobre ellos. Gracias al trabajo de mecanizado, encapsulado, ataque con ácido y fotografiado al microscopio entre otros procesos por parte del grupo de investigación TEP933 “Ingeniería de Fabricación” de la Universidad de Málaga; se ha contado con una colección de

alrededor de 800 imágenes, las cuales contienen virutas de los tres tipos de aleación ya mencionados, en vista longitudinal y transversal, tomadas a diferentes aumentos y mecanizadas con distintos parámetros de corte. En la (fig. 4.2) aparece un pequeño ejemplo del tipo de imágenes proporcionadas, en este caso de Ti6Al4V.

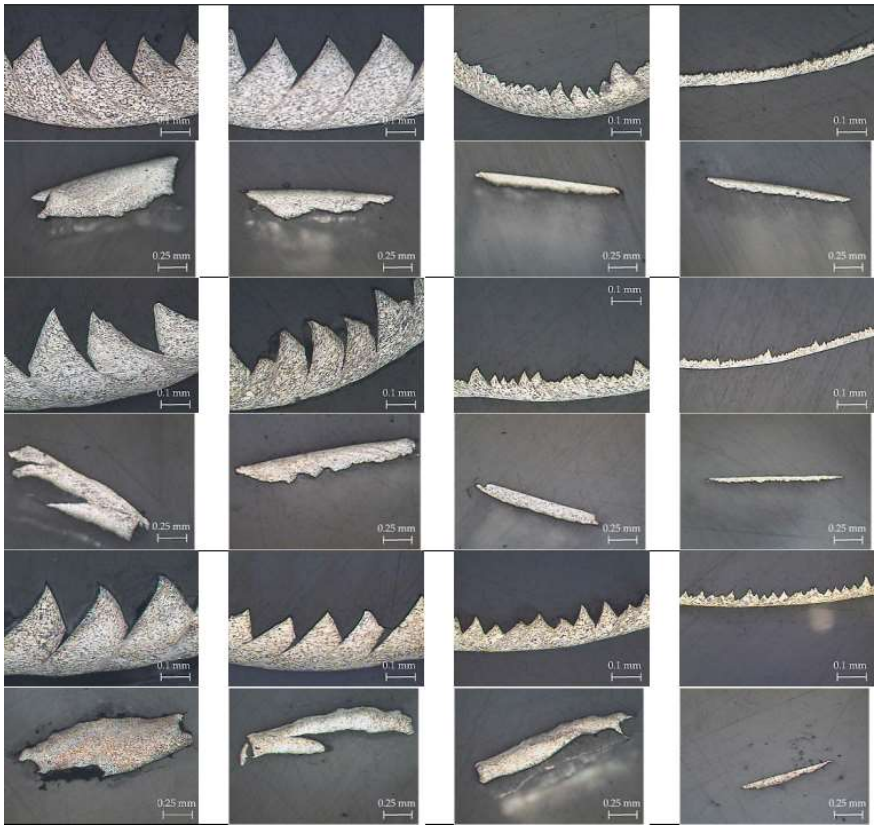


Figura 4.2- Ejemplo de imágenes de viruta de Ti6Al4V en vista longitudinal y transversal.

Bibliografía

También proporcionada por la Universidad de Málaga, se ha tratado de otro pilar fundamental a la hora de desarrollar este trabajo, ya que sin ella no habría una base sobre la que empezar a trabajar. Se encuentran tesis doctorales, artículos científicos, libros y trabajos de fin de grado y Máster. De entre todos estos documentos, se pueden ser destacados algunos que han sido de referencia para los objetivos principales del trabajo, sin quitar importancia al resto, que también ha sido de gran ayuda y se encuentran referenciados.

- Morphological characterization of chip segmentation in Ti-6Al-7Nb machining: A novel method based on digital image processing: Un artículo científico realizado por Sílvia Carvalho, Ana Horovistiz y J.P. Davim en el cual se muestra el procesamiento de imagen de viruta en el que se ha basado este trabajo. Aunque no usando los mismos métodos, sí que se han seguido de forma similar los mismos pasos [30].

- Analysis of the Chip Geometry in Dry Machining of Aeronautical Aluminum Alloys: Artículo científico llevado a cabo por Francisco Javier Trujillo Vilches, Lorenzo Sevilla Hurtado, Francisco Martín Fernández y Carolina Bermudo Gamboa, pertenecientes a la Universidad de Málaga. En él aparece toda la información sobre las dos aleaciones de aluminio con las que se ha trabajado, así como los parámetros geométricos de la viruta que se deseaban obtener entre otros datos de interés [17].

- Experimental Parametric Relationships for Chip Geometry in Dry Machining of the Ti6Al4V Alloy: Realizado por Yezika Sánchez Hernández, Francisco Javier Trujillo Vilches, Carolina Bermudo Gamboa y Lorenzo Sevilla Hurtado, se trata de otro artículo científico similar al anteriormente mencionado, siendo la única diferencia que toda la información que aparece es sobre la aleación de titanio con la que también se ha trabajado en el contexto de este proyecto de fin de grado [1].

- MATLAB Deep Learning: Libro escrito por el autor de Phil Kim. Debido a que la parte técnica de este trabajo ha sido realizada en su mayor parte mediante el software Matlab, el libro ha sido de gran importancia, concretamente para ayudar a comprender los términos de *Machine Learning* y *Red Neuronal Artificial (ANN)*, así como el proceso de entrenamiento del segundo de estos [18].

- Practical MATLAB Deep Learning: Este libro ha sido escrito por los autores Michael Paluszek y Stephanie Thomas. Gracias a él, se ha determinado que la mejor manera para realizar el entrenamiento de una ANN es emplear una red pre-entrenada [19].

Páginas web

Además de toda la información y datos de partida aportados por la Universidad de Málaga, es necesario obtener toda la información de más posible, aumentando así el número de recursos sobre todo a la hora de generar los algoritmos de reconocimiento y procesamiento de imágenes. Entre estas páginas web, podrían destacar:

- Centro de ayuda de MATLAB: Gracias a este recurso se ha conseguido el conocimiento de la existencia de numerosos comandos que son de gran utilidad en la generación del algoritmo, así como de sus funcionalidades [31].

- Matlab Add-On explorer: Para llevar a cabo el algoritmo, es necesario instalar algunos Adds-On sin los cuales no es posible realizar el entrenamiento de la ANN ni procesar las correspondientes imágenes. Se pueden destacar algunos tales como *ResNet-50 Network*, para el entrenamiento de la red neuronal, o *Image Processing ToolBox*, para realizar el procesado de imagen [32].

4.3. Reconocimiento de imágenes

En el contexto de este trabajo se ha llevado a cabo un reconocimiento y clasificación automática de imágenes de virutas de mecanizado, de manera que el algoritmo recibe una imagen y automáticamente la clasifica correctamente según el tipo de aleación como perteneciente a Ti6Al4V, UNS A92024 o UNS A97075 según proceda. Para lograr este objetivo, se ha diseñado y desarrollado un algoritmo de reconocimiento de imágenes empleando el entorno de programación de Matlab. Este algoritmo aprovecha las capacidades de las redes neuronales artificiales (ANN) para analizar minuciosamente las imágenes por capas, identificando patrones, formas, bordes y colores entre otros parámetros.

Gracias a la naturaleza de las redes neuronales, el algoritmo es capaz de aprender a partir de una base de datos proporcionada en la denominada Fase de entrenamiento, en este caso mediante imágenes. Una vez realizado el entrenamiento, el algoritmo es capaz de realizar una clasificación precisa de las virutas de mecanizado en función del tipo de aleación.

Esta automatización tiene el potencial de agilizar el proceso de análisis de virutas, ofreciendo resultados precisos que podrán ser de ayuda en futuras líneas de investigación en el marco de los procesos de mecanizado. A continuación, se muestran los pasos seguidos para la construcción del algoritmo. Es de importancia mencionar que todo el proceso que a continuación se detalla ha sido llevado a cabo mediante el software Matlab.

Red neuronal pre-entrenada

Tal y como se explica en el libro [19], es una buena idea utilizar una red neuronal convolucional (CNN) pre entrenada, como pueden serlo GoogleNet o AlexNet. *Nuruzzaman Faruqui* es un investigador bangladesí en el campo de la de visión artificial y procesamiento de imágenes, graduado en Ingeniería Eléctrica en *North South University* y que posee un Máster en Tecnología de la Información por la *Universidad de Jagangirnagar*. En uno de los videos de su canal en la plataforma *Youtube*, muestra un esquema de algoritmo en Matlab para reconocimiento y clasificación de imágenes [33], el cual se ha seguido en el presente trabajo, eso sí; adaptándolo a las necesidades que en él se han presentado. Para ello, emplea la CNN pre entrenada *ResNet-50*, la cual también se ha usado en este caso.

ResNet-50 es un tipo de arquitectura de CNN que se emplea en tareas de visión artificial, concretamente en el campo del Deep Learning. Su principal característica es su capacidad de entrenar redes extremadamente profundas sin sufrir problemas de rendimiento. Su nombre se divide en dos partes; “ResNet” y “50”. ResNet hace alusión a red residual (Residual Network) y con 50 se refiere al número de capas convolucionales del que dispone.

ResNet-50 cuenta con 50 capas, cada una de las cuales incluye bloques residuales, los cuales son bloques capaces de aprender diferencias en lugar de características directas en el entrenamiento, a través conexiones directas entre capas.

La principal ventaja de usar una CNN pre entrenada es el ahorro de tiempo, ya que, al tener su propia arquitectura ya formada, es posible realizar sobre ella un entrenamiento específico empleando las imágenes deseadas para ello, en este caso imágenes de los tres tipos de viruta con las que se ha trabajado. Entrenar desde cero una red neuronal requiere una enorme cantidad de tiempo, esfuerzo y un banco de datos extremadamente grande.

Para hacer posible el uso de la CNN *ResNet-50*, se ha empleado el Add-On de Matlab denominado "*Deep Learning Toolbox model for ResNet-50 Network*" [32]. En la (fig. 4.3) se muestra la arquitectura de la Red Neuronal, donde se puede apreciar cómo se organiza por capas.

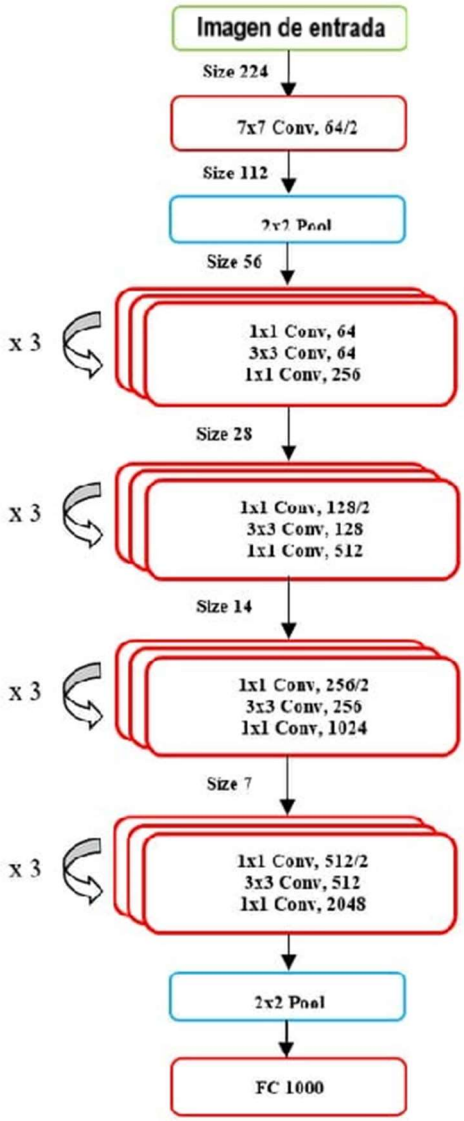


Figura 4.3- Arquitectura de ResNet-50.

Datos para el entrenamiento

Para llevar a cabo el entrenamiento de la CNN, ha sido necesario tomar el máximo número de imágenes posibles, dado que mientras más imágenes se introduzcan en esta fase de desarrollo, mejores pueden ser los resultados obtenidos, los cuales se verán reflejados en la denominada matriz de confusión, la cual se muestra más adelante. Otro factor a tener en cuenta en este punto es la necesidad de dividir en categorías las imágenes. En este caso se han creado 3 carpetas para Ti6Al4V, UNS A92024 y UNS A97075 respectivamente. Tras analizar la biblioteca de imágenes, se ha comprobado que hay más de 600 imágenes correspondientes a virutas de

Ti6Al4V; Unas 100 de UNS A92024; y unas 50 de UNS A97075. Teniendo en cuenta que algunas imágenes de cada categoría deben ser reservadas para la fase de testeo, se tiene que la categoría con menos imágenes para el entrenamiento es UNS A97075, con un total de 38 imágenes. En la (Tabla 4.1) aparece el número de imágenes de cada categoría que se han utilizado en la fase de entrenamiento.

Tabla 4.1- Número de imágenes iniciales de cada aleación empleadas en el entrenamiento.

Imágenes insertadas	
Categoría	**Cantidad**
Ti6Al4V	397
UNS A92024	75
UNS A97075	38

Llama la atención que solo aparezcan 397 datos de la categoría Ti6Al4V, cuando anteriormente se ha mencionado que se cuenta con alrededor de 600 en la biblioteca de imágenes. Esto se debe a que, por razones computacionales, para que el entrenamiento se lleve a cabo de manera adecuada, debe haber exactamente el mismo número de datos de las tres categorías, quedando finalmente el siguiente número de imágenes de cada tipo de aleación (Tabla 4.2).

Tabla 4.2- Número de imágenes definitivas de cada aleación empleadas en el entrenamiento.

Imágenes definitivas	
Categoría	**Cantidad**
Ti6Al4V	38
UNS A92024	38
UNS A97075	38

Para que exista el mismo número de datos para las tres categorías, el propio algoritmo es quien elige como número de muestras por categoría al que presenta la categoría de la que se tienen menos imágenes. Esto se ha realizado de manera aleatoria, tomando en este caso los 38 datos de UNS A97075; y en el caso de las otras dos aleaciones, 38 datos aleatorios de entre los 397 de Ti6Al4V y 38 de entre los 75 de UNS A92024. Por este motivo, no es necesario emplear más imágenes de viruta de la aleación Ti6Al4V, ya que la diferencia de números de muestra entre esta

y las otras dos aleaciones es abismal, por lo cual no se ha notado una mejoría en el resultado del entrenamiento. Para realizar un entrenamiento más eficaz, habría que aumentar el número de muestras de los tres tipos de viruta, intentando mantener un número similar de cada una de ellas. No obstante, al ser *ResNet-50* una CNN con tantas capas convolucionales, con los números que se han obtenido es suficiente para este trabajo.

Posteriormente, surgió un nuevo problema, y es que las imágenes de las que se ha dispuesto para el entrenamiento cuentan con una resolución de 760x560 píxeles en su totalidad. Sin embargo, el esquema propuesto por *Nuruzzaman Faruqui* trabaja con imágenes de entrenamiento cuadradas con una resolución de 224x224 píxeles, debido a que las capas convolucionales que detectan patrones, bordes y colores entre otros parámetros, son de ese tamaño. Para solucionar esto, se han redimensionado las imágenes propuestas para el entrenamiento al tamaño indicado, incluyendo a aquellas que son descartadas por el algoritmo en el proceso de igualado mencionado anteriormente. En la (fig. 4.4) se observa una imagen de cada tipo de viruta tras ser redimensionada, conservando los parámetros que deben ser reconocidos por la CNN.

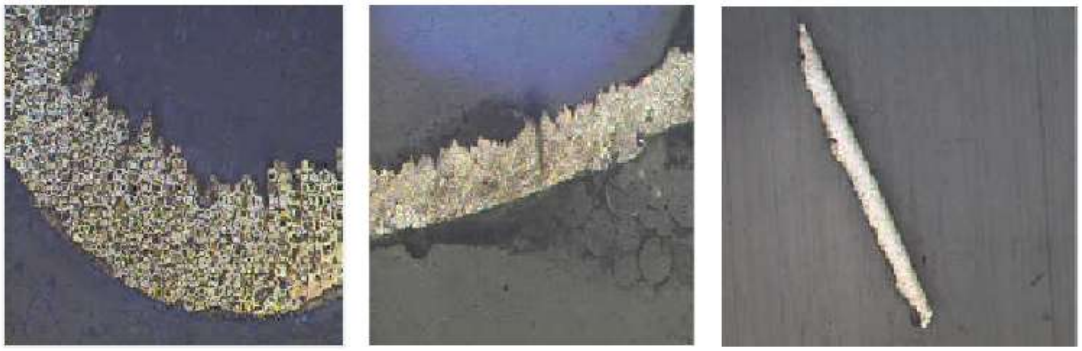


Figura 4.4- Imágenes de virutas de mecanizado tras ser redimensionadas a 224x224 px.

Capas convolucionales

Como se ha mencionado anteriormente, el reconocimiento de imágenes mediante redes neuronales convolucionales (CNN) trabaja con capas convolucionales, las cuales son capaces tanto de detectar patrones y parámetros simples, como realizar conexiones entre ellas creando así un proceso de aprendizaje. En el caso de *ResNet-50*, arquitectura empleada en este trabajo; hay un total de 50 capas, cada una de las cuales tiene una función específica. A continuación, se enumeran estas funciones:

- En primer lugar, actúan las capas de convolución inicial, cuya función es extraer características de bajo nivel de la imagen. Para ello emplea una serie de filtros, cada uno de los cuales se encarga de un parámetro simple, como pueden ser bordes o texturas.

- Acto seguido, actúan las denominadas capas de de normalización por lotes (Batch Normalization), las cuales se encargan de acelerar el entrenamiento. Después de esto se aplica una función de activación como puede ser *Rectified Linear Unit (Relu)* para introducir no linealidad.

- Posteriormente, trabajan las capas de convolución profunda y conexiones residuales. La función de estas es aplicar numerosas capas convolucionales en serie, en este caso 50, estableciendo una jerarquía que baja a niveles en los que la red aprende características complejas y abstractas de la imagen.

- Finalmente, las capas de ajuste de dimensiones se encargan de garantizar que las dimensiones de todas y cada una de las capas coincidan en dimensiones, para así poder superponerse entre sí.

De esta manera, *ResNet-50* es una arquitectura de CNN ideal para tareas de visión por computadora, siendo una de las más utilizadas en este ámbito. En la (fig. 4.5) se muestra cómo se ve la primera capa convolucional en el entrenamiento de la red neuronal para las imágenes de viruta de aleaciones de Ti6Al4V, UNS A92024 y UNS A97075.

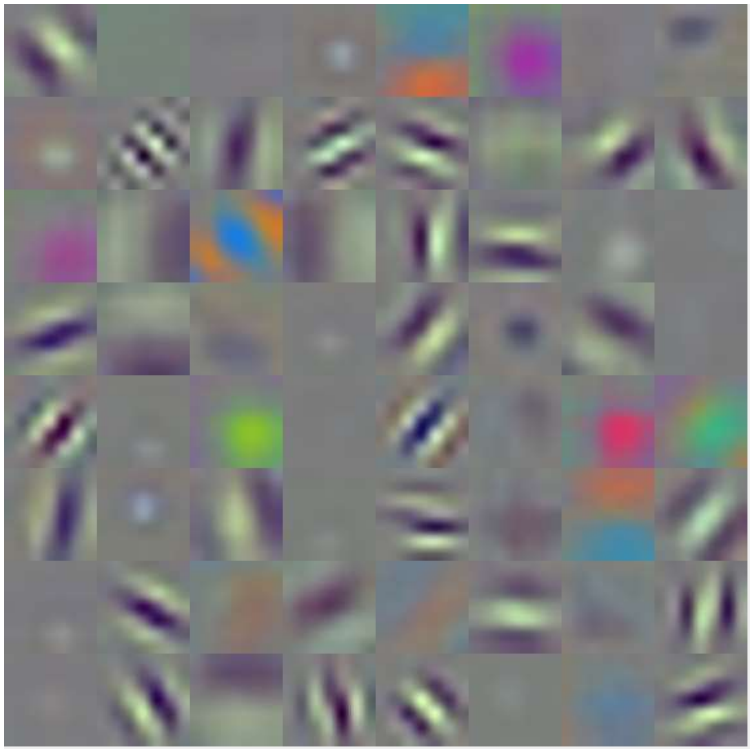


Figura 4.5- Representación visual de la primera capa convolucional en el proceso de entrenamiento.

Matriz de confusión

La matriz de confusión (fig. 4.6) es una herramienta de gran utilidad a la hora de valorar un modelo de clasificación, en este caso de imágenes, basado en aprendizaje automático. Esta contará con un tamaño de N filas x N columnas, siendo N el número de variables a tratar. En el caso del presente trabajo, la matriz de confusión generada ha sido de tamaño 3x3, ya son tres clases de imágenes las que se necesitaban clasificar, correspondiendo a cada respectivo tipo de viruta.

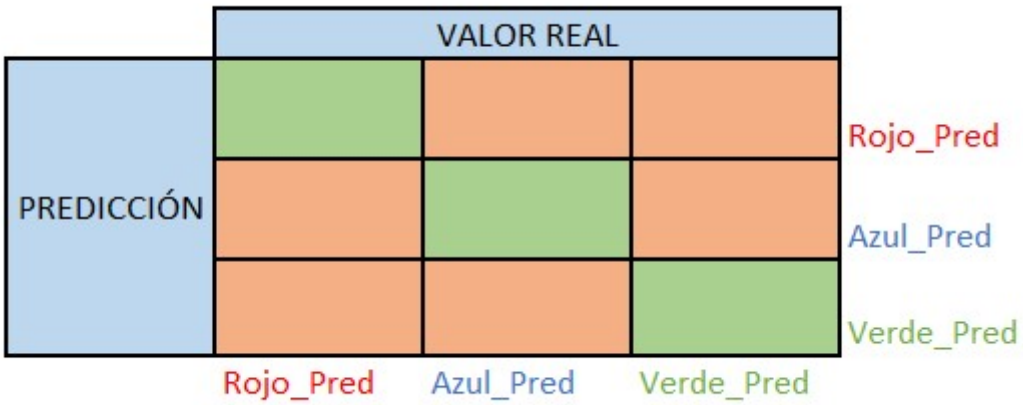


Figura 4.6- Esquema de matriz de confusión para un clasificador de tres clases.

Tras la actuación de todas las capas convolucionales, finalizando así la fase de entrenamiento, es necesario medir el número exacto tanto de aciertos como de desaciertos. Para interpretar los resultados de la matriz de confusión, se ha de tener en cuenta que los aciertos son los valores mostrados en la diagonal principal de la matriz (color verde en la imagen anterior). Por otro lado, los valores que resultan ser erróneos se muestran en los términos de la matriz que no pertenecen a la diagonal principal (color rojo en la imagen anterior) [34]. Esto quiere decir que esta matriz mostrará resultados más eficaces mientras mayores sean los números de la diagonal principal y, por consecuencia, mientras más cercanos a 0 sean los valores situados en los términos no diagonales. En el caso que acontece a este trabajo, se realizaron 20 entrenamientos, obteniendo resultados bastante similares, siendo la matriz de confusión con mayor número de aciertos la que a continuación se muestra (fig. 4.7).

MATRIZ DE CONFUSIÓN		
23	0	4
0	23	4
1	6	20

Figura 4.7- Matriz de confusión obtenida tras el entrenamiento.

A partir de la matriz de confusión obtenida, es posible obtener el porcentaje de acierto por parte de la red mediante un simple cálculo. Este consiste en sumar los términos de la diagonal y posteriormente dividirlo entre la suma de todos los términos de la matriz:

$$\%aciertos = \frac{(23+23+20)}{(23+23+20)+(4+4+1+6)} \times 100 = 81.49\% \quad (10)$$

Pruebas

Una vez finalizado el entrenamiento, se tomaron las imágenes que fueron apartadas del resto anteriormente, las cuales se tenían organizadas de manera que se conocía a qué tipo de viruta pertenecen. De este modo, se han empleado para realizar pruebas y verificar que el algoritmo efectivamente funciona, clasificando cada imagen de prueba de manera correcta de acuerdo con el tipo de aleación que representa.

Finalmente, se guardaron en un archivo de datos los resultados del entrenamiento. De este modo, el algoritmo puede ser usado en el futuro para identificar y clasificar imágenes, y así dar paso al procesamiento de estas para obtener parámetros de la correspondiente viruta.

4.4. Procesamiento digital de imágenes

Para llevar a cabo el procesamiento digital de imagen, se han seguido una serie de pasos, que debieron ser realizados estrictamente en el orden que a continuación se detalla para que el proceso funcione correctamente [30]. Es de importancia recordar que toda la parte del procesamiento se ha llevado a cabo mediante el software Matlab, de modo que, al hacer menciones sobre funciones, todas ellas son referidas a dicho software.

4.4.1. Viruta longitudinal

En primera instancia se aborda el caso de la viruta en vista longitudinal, que es aquella en la cual se pueden apreciar dientes en forma de hoja de sierra. En este caso, se ha tomado como ejemplo una imagen de viruta de la aleación de titanio Ti6Al4V para mostrar los pasos seguidos, aunque para el caso de las otras dos aleaciones el proceso es exactamente el mismo.

Selección de superficie en la imagen

Cuando se está trabajando con una imagen de viruta, hay ocasiones en las que es de necesario para el usuario tratar la imagen completa. Sin embargo, puede ocurrir que sea de interés tratar solo con una sección dentro de la imagen, ya sea porque se quieren observar valores de una zona concreta de la viruta, o porque a partir de cierto límite la viruta se curve. Es por ello por lo que el primer paso que se ha seguido en este trabajo ha sido incorporar la posibilidad para el usuario de seleccionar una zona concreta de la imagen. Para ello, se ha empleado la función *"imcrop"* (fig. 4.8). Es de importancia destacar que para leer y mostrar imágenes por pantalla se han empleado las funciones *"imread"* e *"imshow"* respectivamente.

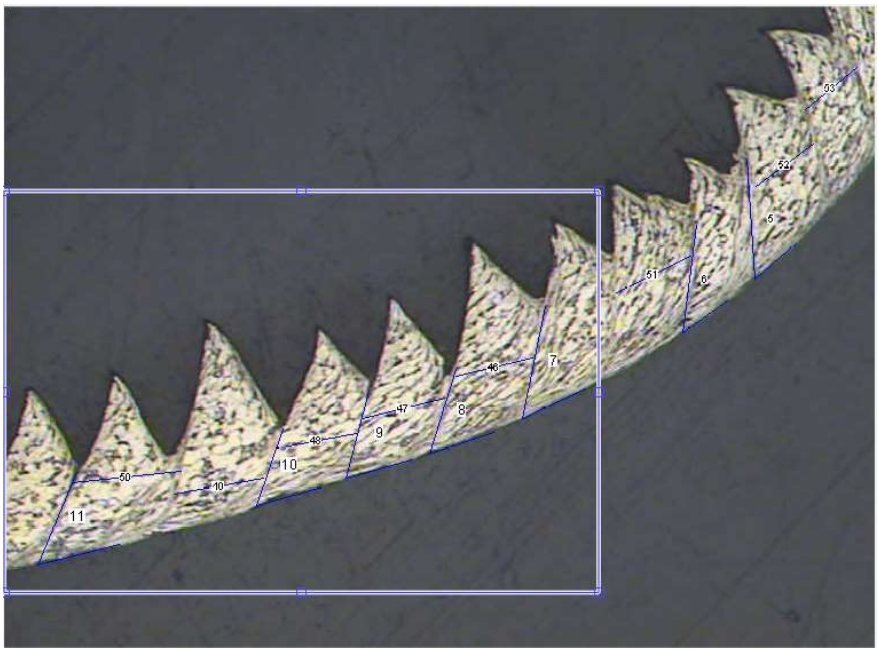


Figura 4.8- Selección de área en la imagen de la viruta.

Binarizado de la imagen

Al trabajar con una imagen a color, que es lo más común en el caso de las virutas de aleaciones ya que así se obtienen en el microscopio; es necesario saber que

dicha imagen está en escala RGB (Red, Green, Blue). En otras palabras, la imagen está compuesta por tres matrices, cada una de las cuales representa al color rojo, verde y azul respectivamente. La superposición de estas tres matrices, con diferentes valores numéricos de cada uno de los tres colores en cada píxel de la imagen, da lugar a todos los colores que se pueden apreciar en la imagen (fig. 4.9).

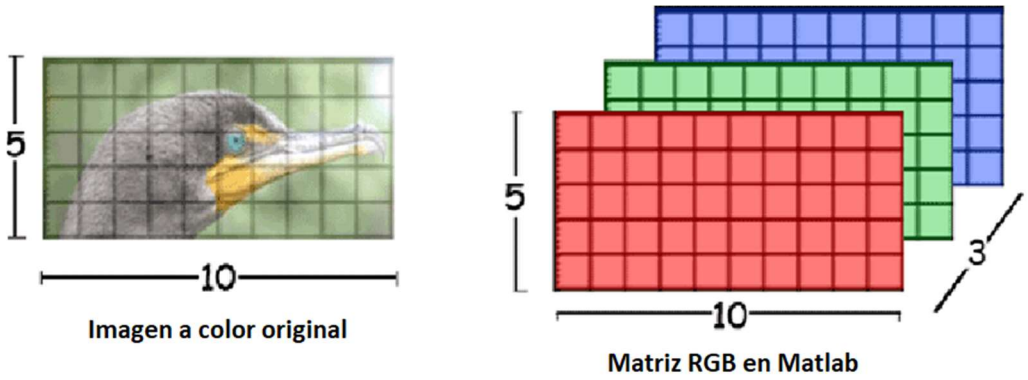


Figura 4.9- Matriz de una imagen en escala RGB.

No obstante, es muy compleja la tarea de trabajar sobre una imagen compuesta por tres matrices. Es entonces cuando se hace necesario convertir la imagen a escala de grises, de modo que solo se puedan apreciar tonos grises, desde el blanco hasta el negro. La ventaja de esto es que, a partir de ahora, la imagen estará compuesta solo por una sola matriz numérica. Para conseguir esto, se ha utilizado la función *"rgb2gray"* (fig. 4.10).

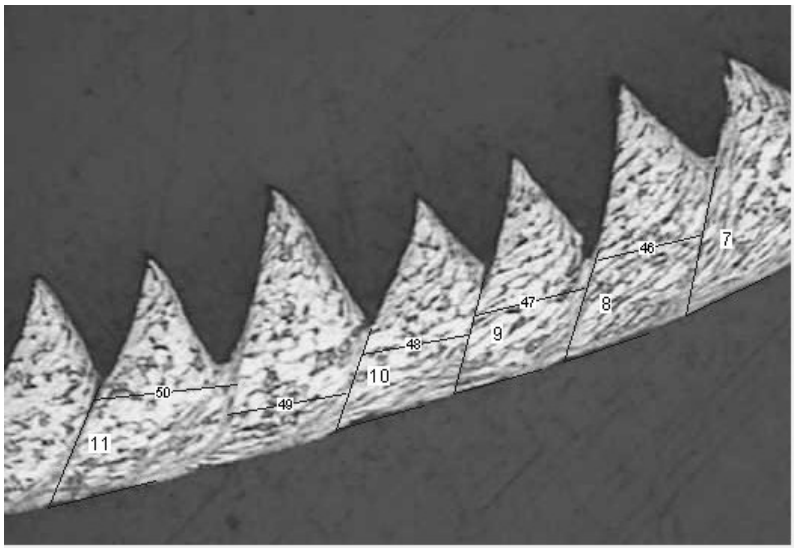


Figura 4.10- Imagen de la viruta en escala de grises.

Hasta el momento, se ha convertido la imagen RGB a escala de grises, lo cual facilita el trabajo posterior, al estar trabajando sobre una única matriz, con diferentes valores en cada píxel. Sin embargo, aún se puede facilitar más el trabajo posterior si la imagen es binarizada. Binarizar una imagen consiste en tomar esa única matriz en escala de grises, y convertir sus valores en unos y ceros. Esto da como resultado una imagen en la que solo existe el color blanco, asociado con el 1, y el color negro, asociado con el 0. Para lograr esto, se ha empleado la función *"imbinarize"*, dando como resultado lo que se muestra en la siguiente figura (fig. 4.11).

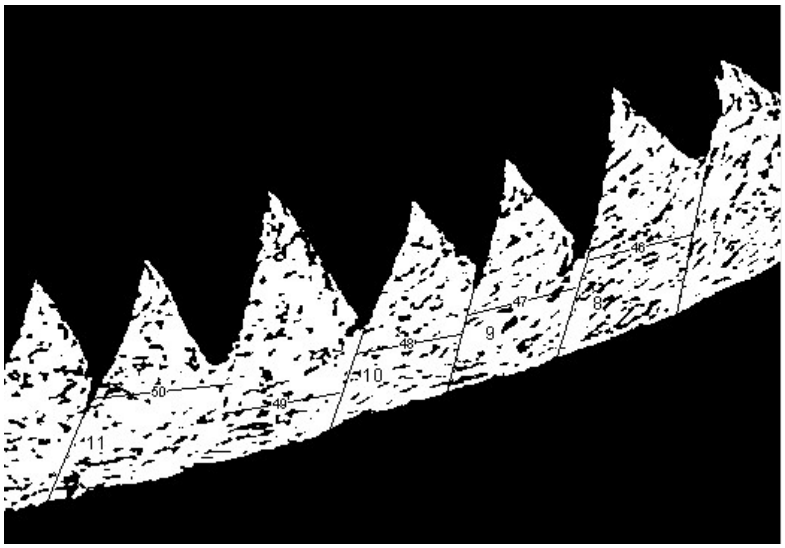


Figura 4.11- Imagen de la viruta en escala binaria.

Detección de bordes

Para medir valores de la viruta, tales como picos y valles; es necesario realizar una previa detección de bordes, de manera que el software tenga localizados los contornos en todo momento. El problema que impide lograr esta detección se puede apreciar en la (fig. 4.10), y es que se trata de una imagen en escala binaria, lo cual es correcto, pero está completamente llena de agujeros e irregularidades. Esto hace imposible a Matlab detectar de forma correcta los bordes, por lo tanto, la imagen debe ser dilatada. Para ello se ha hecho uso de tres funciones en Matlab. La primera de ellas se denomina *"strel"*, el cual permite realizar una dilatación, de forma que convierte las agrupaciones pequeñas de píxeles negros en blancos, siempre y cuando sea el color blanco el que abunda alrededor. A continuación, se ha aplicado la función *"imclose"*, que permite aplicar en la imagen el factor obtenido con el anterior comando. Por último, se ha usado la función *"imfill"*, cuya función es mejorar el resultado obtenido mediante el comando anterior rellenando posibles irregularidades que hayan quedado sin erosionar la imagen (fig. 4.12).

Figura 4.12- Imagen de la viruta binarizada tras limpiar irregularidades.

Una vez realizada la limpieza de la imagen, es posible pasar a la detección de contornos. Para llevar esto a cabo, se han empleado las funciones *"bwboundaries"* y *"visboundaries"* respectivamente. El primero de ellos se ha encargado de realizar la detección de contornos en la matriz de la imagen, mientras que la función del segundo ha sido mostrar los bordes superpuestos en la imagen binaria de la viruta, de forma que se puedan observar las coordenadas cartesianas de cada punto del contorno como si de una gráfica se tratase (fig. 4.13).

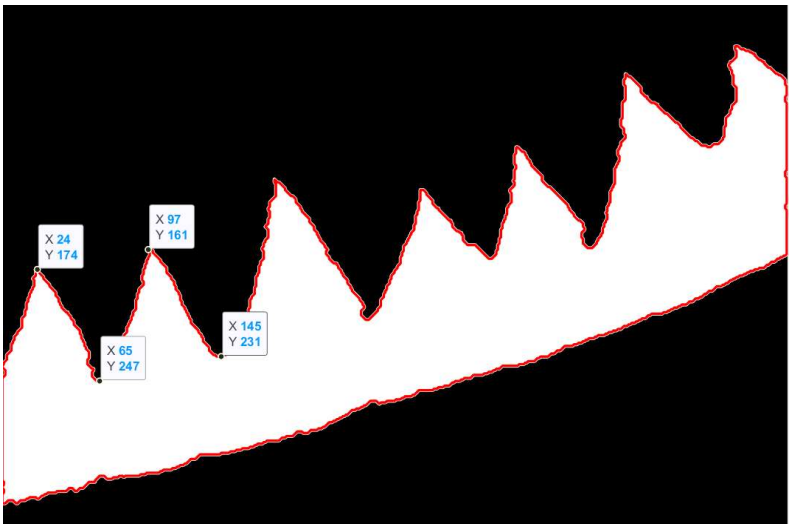


Figura 4.13- Imagen de la viruta con resalto de bordes.

Obtención de picos y valles

Una vez detectado el contorno, se procedió a obtener las coordenadas de los puntos que representan picos y valles. Para ello, se utilizó la función llamada *"findpeaks"*, cuya función es encontrar todos los picos, ya sean máximos o mínimos. El nuevo problema que surge al usar este comando es que localiza todos y cada uno de los picos. Teniendo en cuenta que el borde de la viruta es bastante irregular, van a existir pequeños picos y valles a lo largo de toda la viruta, incluyendo toda la parte baja de la viruta. Estas irregularidades se pueden apreciar al ampliar la imagen (fig. 4.14).

Figura 4.14- Imagen ampliada con línea de irregularidades (rojo).

El método por el que se ha optado para solucionar este problema ha sido desarrollar dos filtros, vertical y horizontal respectivamente, siendo el primero un poco más sencillo de realizar.

En primer lugar, se ha desarrollado el filtro vertical, el cual se ha empleado para filtrar toda la parte baja de la viruta, ya que no es de interés por el momento. Para su realización, se tomó el vector numérico de valores en "y" (eje vertical) tanto de picos como de valles y se compararon sus valores, todo ello dentro de un bucle. De esta manera se fueron recorriendo los valores tanto del vector vertical de picos como el de valles de modo que, al encontrar un valor demasiado bajo, significa que se encuentra en la parte baja de la viruta y, por lo tanto, es eliminado del vector junto a su pareja en el vector de coordenadas horizontales.

Posteriormente, se realizó algo parecido para el filtro horizontal. En este caso se tomó el vector numérico de valores en "x", tanto para picos como para valles. A partir de este punto, se desarrolló un bucle cuya función es recorrer dichos vectores comparando cada valor con el que le precede en el vector. En caso de haber dos picos demasiado juntos, o dos valles demasiado juntos, se elimina el valor del vector

"x" cuya altura en su vector correspondiente "y" sea menor, así como su pareja en el vector vertical.

En la (fig. 4.15 y 4.16) se puede observar cómo se reduce el número tanto de picos como de valles y se puede comprobar que, tras la limpieza, coinciden con el número de picos y valles reales que se muestran en la (fig. 4.12).

Picos Antes		Picos Después	
x	y	x	y
24	181	24	181
97	168	97	168
180	122	180	122
276	129	276	129
339	101	339	101
411	53	411	53
483	35	483	35
247	267		
198	287		
80	316		
68	316		
47	324		
23	330		
8	332		

Figura 4.15- Coordenadas de picos antes y después del filtrado de irregularidades.

Valles Antes		Valles Después	
x		x	y
63	262	63	261
142	246	142	245
240	222	240	221
321	182	321	181
386	175	386	174
465	109	465	108
249	277		
202	296		
81	325		
73	326		
50	334		
25	339		
11	342		
3	342		

Figura 4.16- Coordenadas de picos antes y después del filtrado de irregularidades.

Segmentación de la imagen

Una vez obtenido los valores de picos y valles, es necesario segmentar la imagen, lo cual consiste en separar cada diente de la viruta longitudinal, para así poder calcular individualmente la superficie de cada diente, altura de pico, altura de valle etc.

Para realizar la segmentación, la solución que mejor resultado mostró fue usar la técnica de superposición de imágenes, la cual consiste en crear una máscara, que en realidad es una segunda imagen del mismo tamaño que la que se ha está trabajando, de modo que al superponer ambas, se da una tercera imagen completamente diferente como resultado (fig. 4.17).

Figura 4.17- Creación de una imagen mediante superposición de otras dos.

Para el caso de este trabajo, se creó una máscara que actúa como segunda imagen en la superposición a partir de los picos y valles obtenidos anteriormente. Para ello, se desarrolló una rutina que en primer lugar crea una imagen negra con las mismas dimensiones que la original. Posteriormente se trazaron varias líneas que partían desde cada pico y que finalizan en cada respectivo valle. Después de esto, se volvieron a trazar varias líneas en la imagen, esta vez comenzaban en cada valle, manteniendo la misma pendiente que tenían las primeras líneas trazadas. Esta segunda serie de líneas finalizaba al encontrarse con un borde de la imagen, ya sea horizontal o vertical. Para acabar, se eliminaron todas las líneas del primer tramo, quedando solo su extensión desde cada valle hasta el borde de la imagen (fig. 4.18).

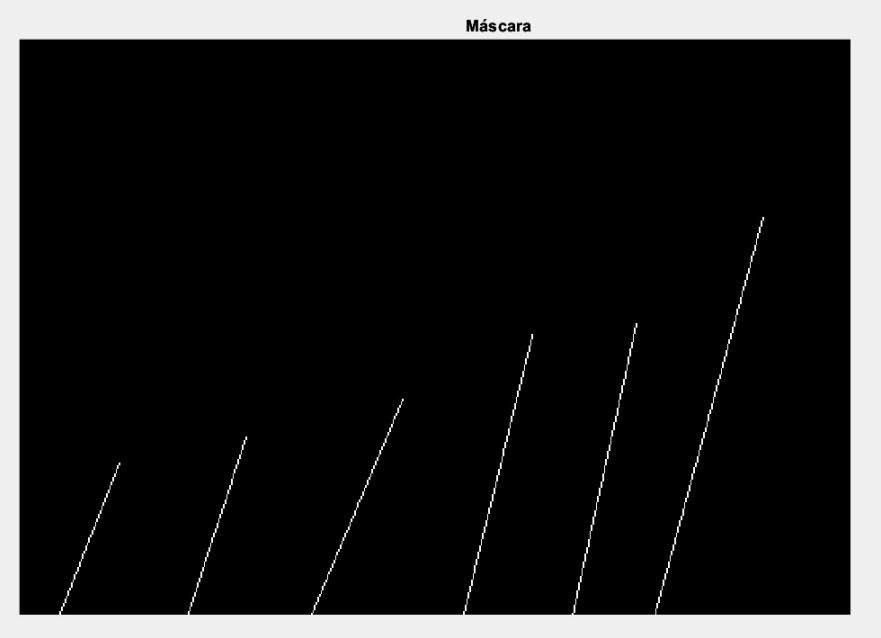


Figura 4.18- Máscara empleada para superposición de imágenes.

Una vez creada la máscara, el último paso para la segmentación fue superponer la imagen de la viruta binarizada (fig. 4.11) con la imagen de la recién creada máscara (fig. 4.18), especificando que las líneas trazadas en esta última imagen fuesen de color negro, usando para ello la función *"imoverlay"*. El resultado de la segmentación puede apreciarse en la (fig. 4.19).

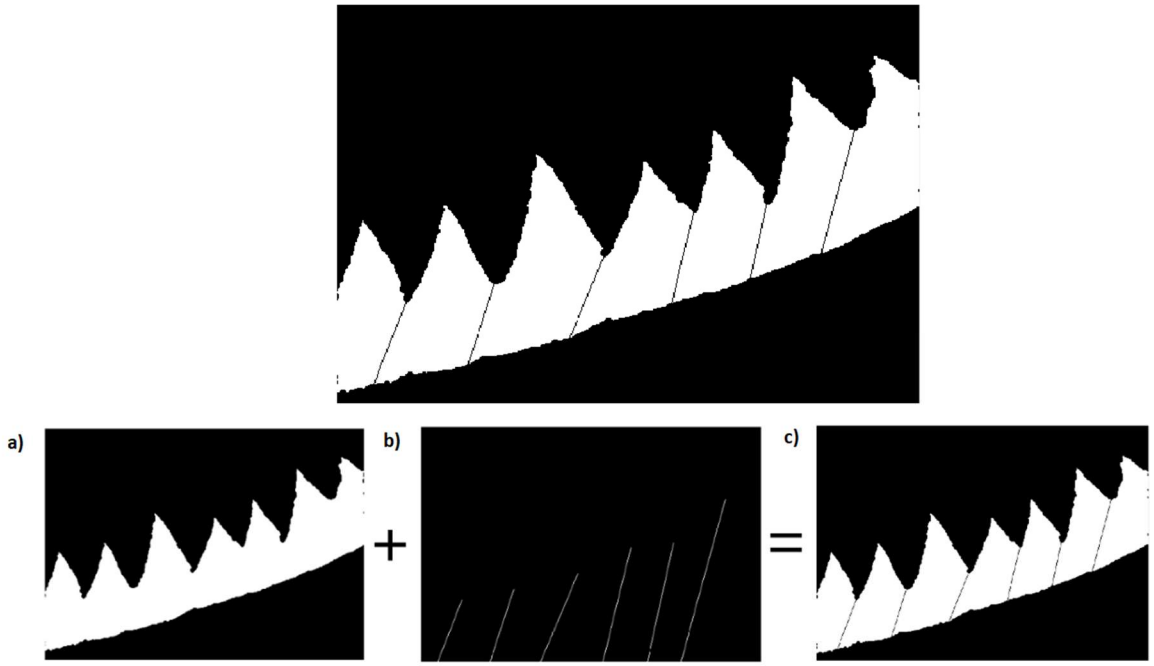


Figura 4.19- Imagen de viruta segmentada y proceso de superposición de imágenes.

a) Imagen binarizada, b) Líneas trazadas, c) Imagen segmentada

Identificación de objetos y obtención de resultados

Una vez realizados todos los pasos anteriores, se finalizó el procesamiento de imagen con una identificación de objetos presentes en la imagen y con la obtención de algunos parámetros básicos a partir de los cuales se pudo obtener el resto.

Se comenzó por una limpieza de bordes, es decir, se eliminaron los dientes que estaban en contacto con el borde de la imagen. Con esto se consiguió que no quedase ningún diente incompleto dentro de la imagen, de modo que al obtener parámetros posteriormente y promediar no afecten de manera negativa al resultado final. Esto se llevó a cabo mediante la función *"imclearborder"* (fig. 4.20).

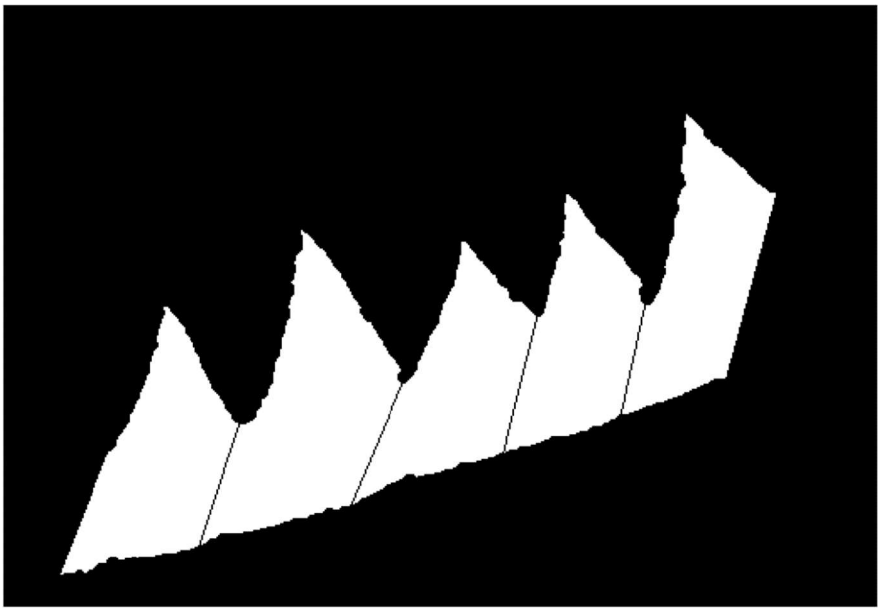


Figura 4.20- Imagen de viruta segmentada tras la limpieza de bordes.

Una vez están limpios los bordes, al haber segmentado cada diente previamente, es posible obtener diferentes resultados numéricos mediante la función *"regionprops"*. Además, el mismo comando permite obtener las denominadas "*Bounding Boxes"*, que no son más que cajas de forma rectangular del tamaño mínimo posible capaces de contener cada diente y además ofrece otros resultados (fig. 4.21). En este punto, la imagen ya está preparada para extraer resultados de ella, los cuales se explican y comparan en el capítulo 5.

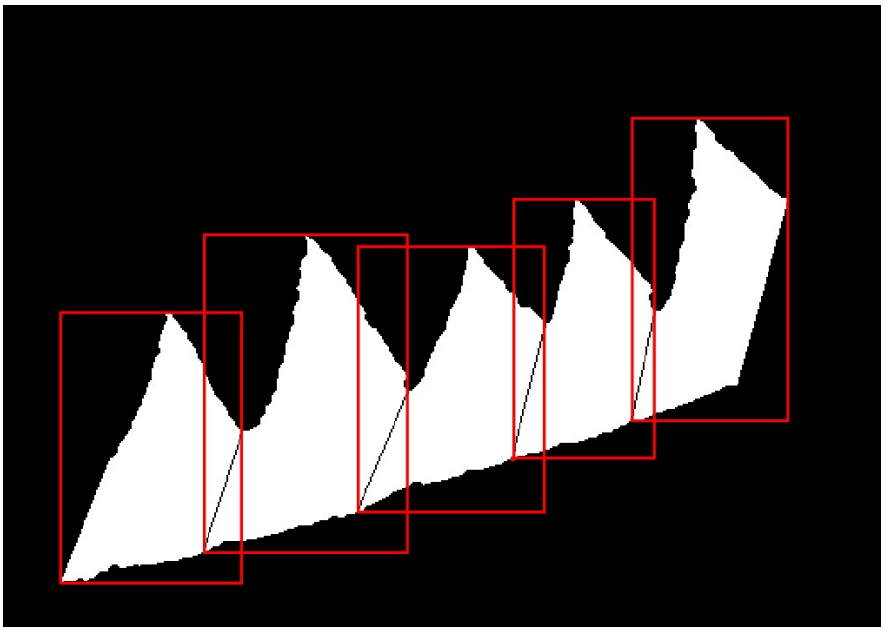


Figura 4.21- Imagen de viruta segmentada con Bounding Boxes.

Para finalizar, se muestra en la (fig. 4.22) un diagrama de flujo a modo de resumen en el cual se indican todos los pasos de mayor interés seguidos en cuanto al procesamiento de imagen; desde la obtención de la imagen hasta que está preparada para obtener resultados numéricos.

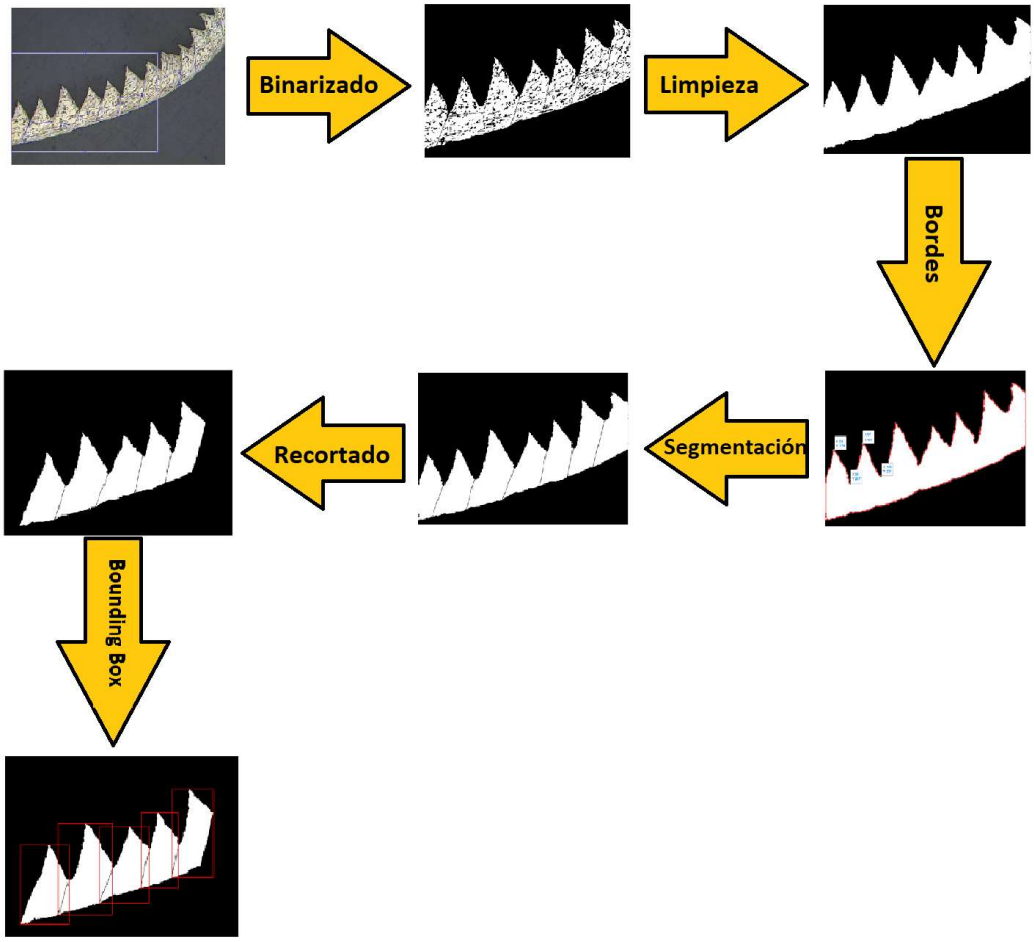


Figura 4.22- Diagrama de flujo de procesado de imagen en vista longitudinal.

4.4.2. Viruta transversal

Además de imágenes de viruta longitudinal, también es necesario procesar imágenes de viruta en vista transversal, que es la que se obtiene al fotografiar la viruta de perfil. En este caso, el proceso es diferente, ya que no se observan picos ni valles, de modo que se buscan otros resultados completamente diferentes. Para mostrar los pasos seguidos, se ha tomado como ejemplo una imagen de viruta transversal de aluminio UNS A97075, aunque el procedimiento es exactamente el mismo para los demás tipos de aleaciones.

Binarizado de imagen y detección de bordes

Tal y como se ha explicado en el caso de viruta en vista longitudinal, es necesario convertir la imagen primero a escala de grises, para después pasarla a escala binaria y realizarle una limpieza de huecos (fig. 4.23). No es necesario una explicación al detalle de esta parte, ya que el proceso es idéntico al realizado anteriormente en la viruta en vista longitudinal. No obstante, cabe destacar que en este caso no se ha habilitado la opción de recortar la imagen para seleccionar una zona de interés, puesto que en este tipo de vista solo habrá una única superficie de interés para el estudio de la viruta, la cual se selecciona automáticamente.

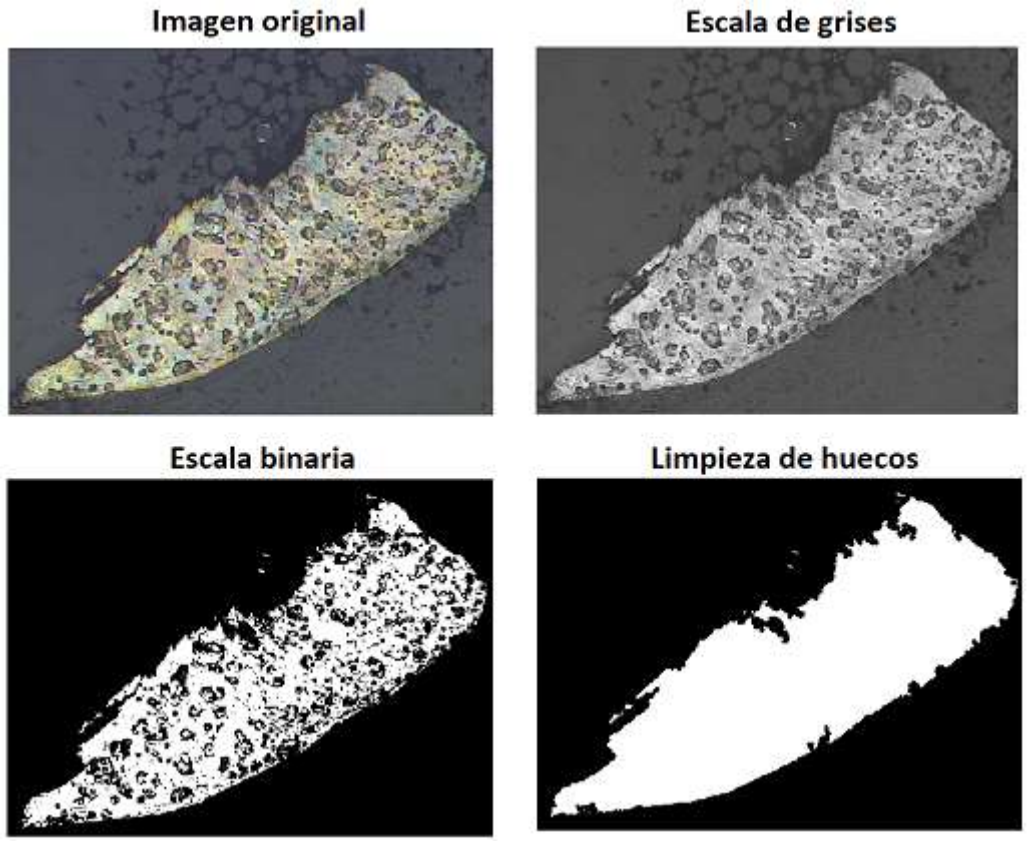


Figura 4.23- Proceso de binarizado y limpieza de la imagen.

En cuanto a la detección de bordes, se volvió a emplear la función *"bwboundaries"* para que el software reconozca los límites entre viruta y fondo.

Identificación de objetos y obtención de resultados

En el caso de la vista transversal, al no haber dientes que separar, no es necesario segmentar la imagen. Sin embargo, se dio un pequeño problema, y es que, alrededor de la superficie que se deseaba estudiar, aparecieron pequeñas superficies que el software toma como viruta. Esto se debía a que, al binarizar, algunas de las irregularidades de los bordes de la viruta podían quedar separadas del elemento

principal, o simplemente a que el software pueda detectar en el fondo de la imagen un reflejo que pueda confundir con un tono de blanco. Para solucionarlo, se desarrolló una rutina que cuenta el número de superficies blancas que existen en la imagen, creando una nueva imagen en la que solo aparecerá la superficie que más píxeles tenga en la primera imagen. Para ello se ha empleado la función *"bwlabel"*. A partir de este momento, se continuó trabajando sobre la segunda imagen, que solo presentaba la mayor superficie, la cual siempre se correspondía con la viruta a estudiar, ya que el resto de los elementos son de un tamaño mucho menor al de la viruta (fig. 4.24).

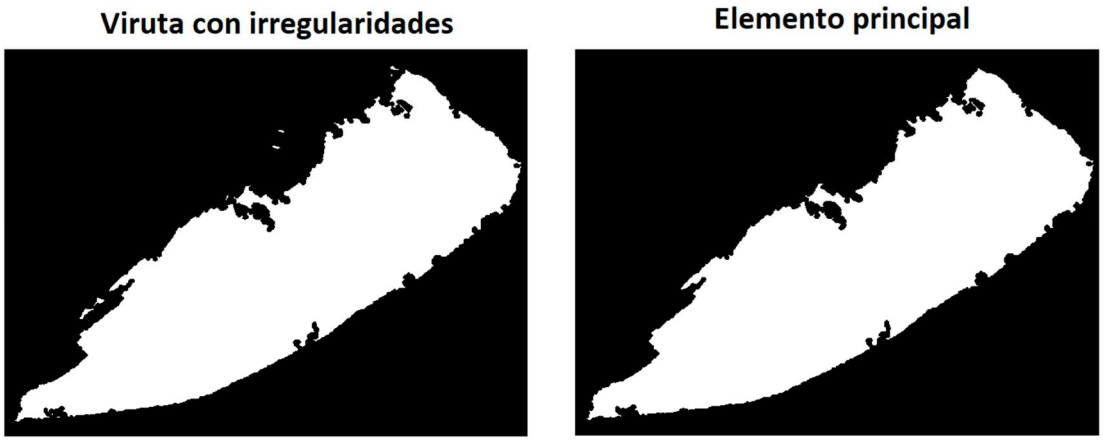


Figura 4.24- Proceso de limpieza de elementos sobrantes.

Una vez realizado todo lo comentado, la imagen queda procesada y se pueden extraer resultados de ella, los cuales se muestran en el capítulo 5. En la (fig. 4.25) se muestran los pasos más importantes seguidos en el procesamiento de imagen en vista transversal, desde la obtención de la imagen hasta que la misma es apta para extraer resultados.

Figura 4.25- Diagrama de flujo de procesado de imagen en vista longitudinal.

4.5. Desarrollo de la App

4.5.1. Introducción

Dentro del marco de este trabajo, hasta el momento se ha logrado desarrollar un algoritmo capaz de detectar y clasificar con bastante precisión el tipo de aleación al que corresponde la viruta, identificando entre las aleaciones Ti6Al4V, UNS A92024 y UNS A97075. Además, se ha llevado a cabo un segundo algoritmo diseñado para procesar digitalmente la imagen y proporcionar resultados numéricos de interés, en este caso altura de picos, valles, espesor y ángulo de deslizamiento para vista longitudinal; así como ancho y superficie de viruta para vista transversal.

No obstante, uno de obstáculos identificados en el presente trabajo es la complejidad para los usuarios a la hora de utilizar estos algoritmos. Con el objetivo de solventar este problema y hacer que la aplicación de estos algoritmos sea más accesible a nivel de usuario, se ha creado una aplicación con interfaz gráfica. Esta aplicación, desarrollada en el entorno de App Designer de Matlab, simplifica significativamente el proceso de detección y clasificación de aleaciones, así como el procesamiento de imágenes.

Una de las ventajas destacadas de esta aplicación es su facilidad de uso. La interfaz de usuario ha sido diseñada pensando en la simplicidad y la intuitividad, lo que permite a los usuarios, incluso aquellos sin experiencia previa en programación, utilizarla de manera efectiva. Esto no solo simplifica la operación de los algoritmos, sino que los unifica en uno, cosa que no era posible anteriormente.

Además de la facilidad de uso, la aplicación proporciona una ventaja significativa en términos de eficiencia y velocidad en la obtención de resultados. Elimina la necesidad de que los usuarios ejecuten manualmente los códigos y realicen tareas de procesamiento de imágenes de manera independiente, lo que agiliza el proceso de análisis y permite una toma de decisiones más rápida y precisa.

Además de todo esto, se ha incorporado a la App una opción la cual permite exportar los resultados obtenidos a Excel, facilitando de esta manera aún más el trabajo a realizar por el usuario. De este modo se pueden lograr resultados en un menor tiempo, aumentando la eficiencia de la aplicación.

En resumen, esta aplicación desarrollada en el entorno de App Designer de Matlab representa un paso importante hacia la simplificación y la accesibilidad en la utilización de algoritmos de detección y clasificación de aleaciones, así como de procesamiento de imágenes. Su utilidad potencial abarca diversos campos de la metalurgia y la investigación científica, proporcionando a los usuarios una herramienta poderosa y fácil de usar para optimizar sus procesos de análisis y toma de decisiones.

4.5.2. Funcionamiento de la App

Tal y como se ha explicado, la App diseñada cuenta con una interfaz gráfica bastante intuitiva y visualmente atractiva, lo cual facilita la interacción del usuario con sus diversas funciones. Tal y como se puede apreciar en la (fig. 4.26), la interfaz presenta una disposición ordenada de elementos, tales como botones, iconos, menús desplegables y deslizaderas, que permiten una navegación fluida y accesible. Todo esto no solo mejora la experiencia del usuario, sino que también contribuye al uso de la aplicación, asegurando que se puedan aprovechar al máximo todas las características que ofrece.

Figura 4.26- Interfaz general de la APP.

Para proporcionar una comprensión detallada del funcionamiento y uso de la aplicación, se ha dividido su interfaz en cuatro zonas distintas (fig. 4.27).

En primer lugar, la zona 1 se ha destinado a la selección y visualización de la imagen con la que se trabaja, permitiendo además la configuración de los aumentos según hayan sido seleccionados previamente en el microscopio.

A continuación, la zona 2 se ha enfocado en el procesamiento de imágenes de viruta en vista longitudinal, ofreciendo herramientas específicas para esta tarea. Aquí, el usuario puede realizar análisis detallados de la viruta en esta orientación, siempre y cuando esta se encuentre segmentada, habiendo una clara diferencia entre picos y valles.

La zona 3 ha sido diseñada para el procesamiento de imágenes de virutas en vista longitudinal en las que no se puede apreciar una diferencia clara entre picos y valles. En este espacio, la aplicación proporciona una herramienta capaz de obtener la longitud del eje menor de dicha viruta.

Por último, la zona 4 se ha dedicado al procesamiento de imágenes de virutas en vista transversal, ofreciendo herramientas especializadas para este tipo de análisis.

Esta división de la interfaz garantiza una experiencia de usuario eficiente y efectiva, adaptada a las necesidades específicas de cada tipo de análisis de viruta.

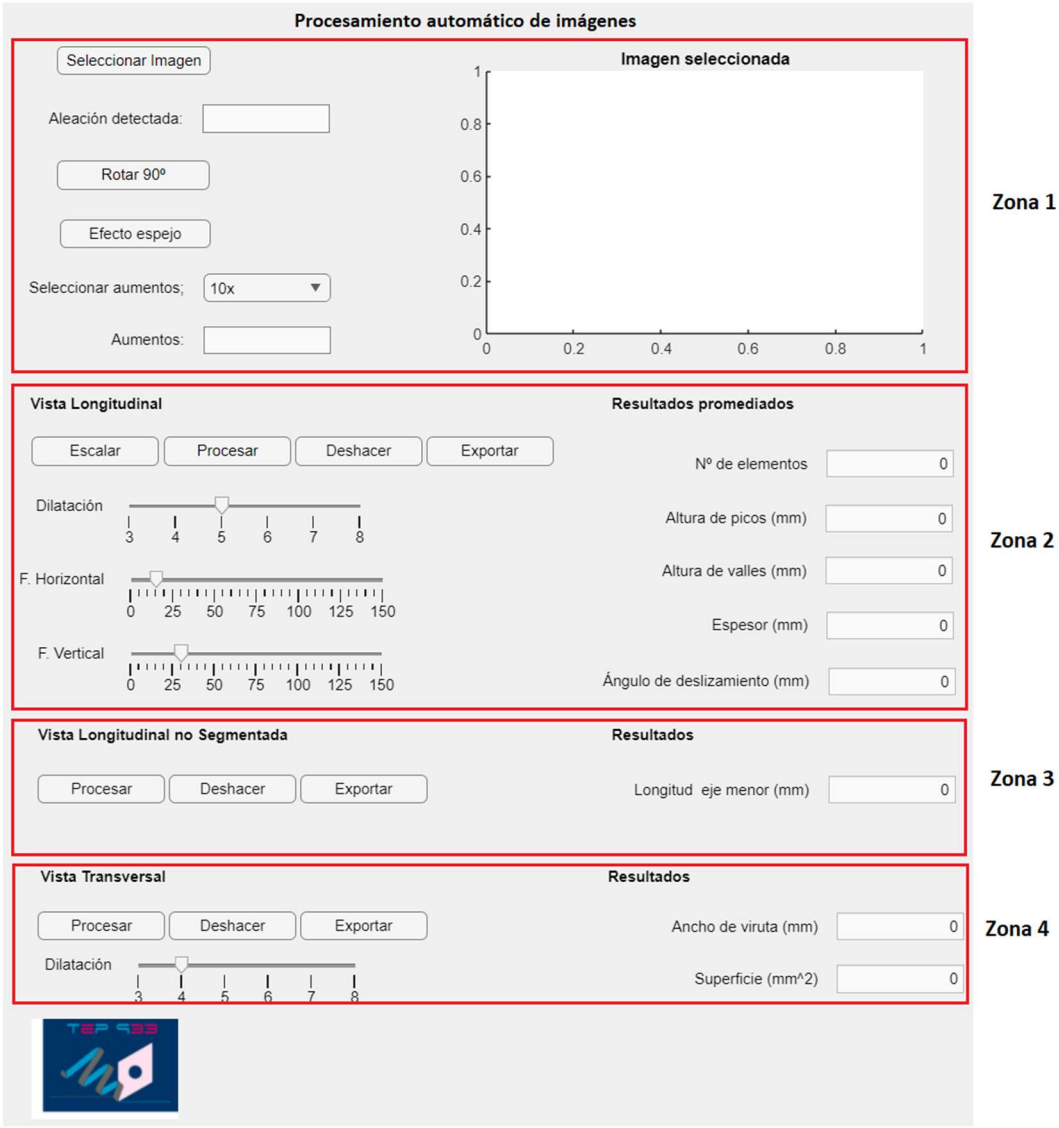


Figura 4.27- Zonas en las que se divide la interfaz.

A continuación, se explica el funcionamiento de todas las herramientas disponibles en cada una de estas zonas, con el objetivo de proporcionar al usuario una comprensión completa de cómo aprovechar al máximo de esta aplicación. A medida que se explore cada zona, se aprecian las funcionalidades específicas las cuales permiten llevar a cabo análisis precisos y eficientes en los distintos contextos de procesamiento de imágenes de viruta.

Zona 1. Selección de imagen

En primer lugar, en la interfaz se encuentra una tecla fundamental denominada “Seleccionar imagen”, la cual permite al usuario importar imágenes desde su dispositivo con facilidad. Al hacer clic en esta tecla, se habilita la selección de una

imagen de interés para su posterior análisis. En la parte derecha de la pantalla hay una ventana de visualización donde se muestra la imagen seleccionada de manera inmediata, tras ser seleccionada. Justo debajo de la tecla “Seleccionar imagen”, se encuentra un cuadro de texto informativo en donde se muestra el nombre de la aleación detectada en la imagen, gracias a las capacidades de análisis de visión artificial del algoritmo (fig. 4.28).

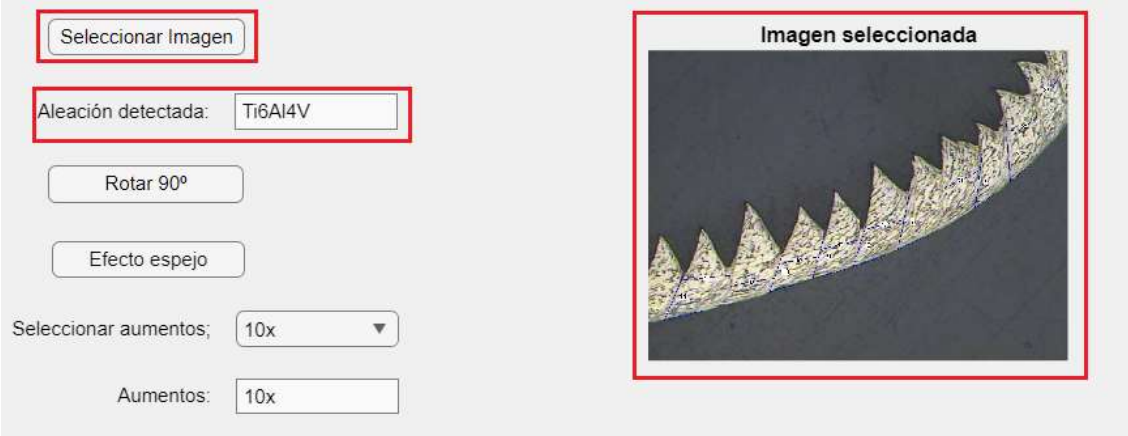


Figura 4.28- Tecla de selección de imagen, detección de aleación y ventana de imagen.

Además de la funcionalidad de importar y visualizar imágenes, la interfaz de nuestra aplicación ofrece dos herramientas adicionales para el ajuste de imágenes según las necesidades del usuario. En primer lugar, se encuentran las teclas “Rotar 90º” y “Efecto espejo”. La tecla “Rotar 90º” permite al usuario girar la imagen seleccionada en incrementos de 90 grados, permitiendo orientar los picos de la viruta en la parte superior, en caso de que la imagen original esté volteada. Por otro lado, la tecla “Efecto espejo” ofrece la posibilidad de crear una versión reflejada de la imagen, lo cual es de gran utilidad, ya que para que el procesamiento de imagen funcione correctamente, es necesario que la orientación de los dientes de la viruta quede hacia la derecha. Adicionalmente, se presenta un menú desplegable que permite seleccionar los aumentos de la imagen según corresponda; acompañado de un cuadro de texto en el que se muestran los aumentos seleccionados (fig. 4.29).

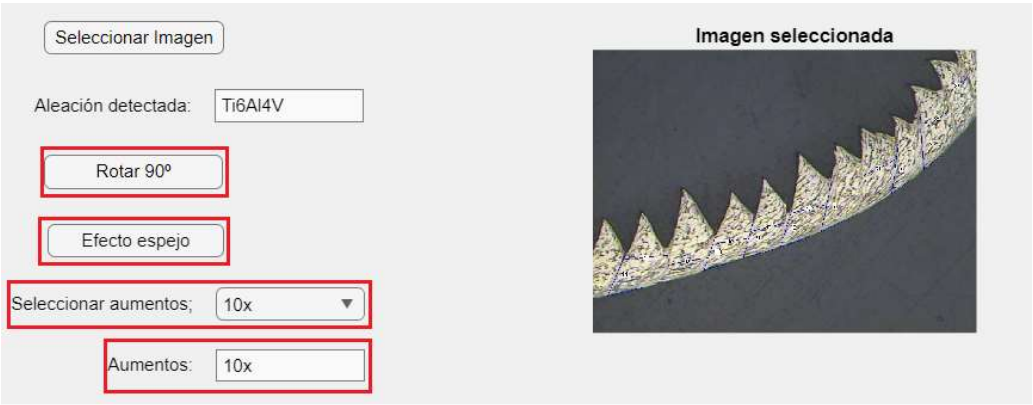


Figura 4.29- Teclas para rotar y voltear la imagen; menú desplegable de aumentos.

Zona 2. Viruta en vista longitudinal segmentada

En primer lugar, se encuentra una tecla esencial denominada “Escalar”, cuya función es realizar un escalado previo de la imagen tal y como se explica en profundidad en el apartado 4.4. correspondiente al procesamiento de imagen. Es de importancia destacar que el escalado se trata de una acción obligatoria en el caso de las imágenes de viruta en vista longitudinal segmentada. Dicho escalado es un paso fundamental para el procesamiento adecuado de estas imágenes y garantiza resultados precisos y coherentes. La tecla “Escalar” se convierte, por lo tanto, en un componente muy importante para un procesamiento eficiente, asegurando que cada imagen quede recortada mostrando única y exclusivamente el área de interés a procesar (fig. 4.30). Una vez recortada, la imagen se muestra en la ventana anteriormente mencionada.

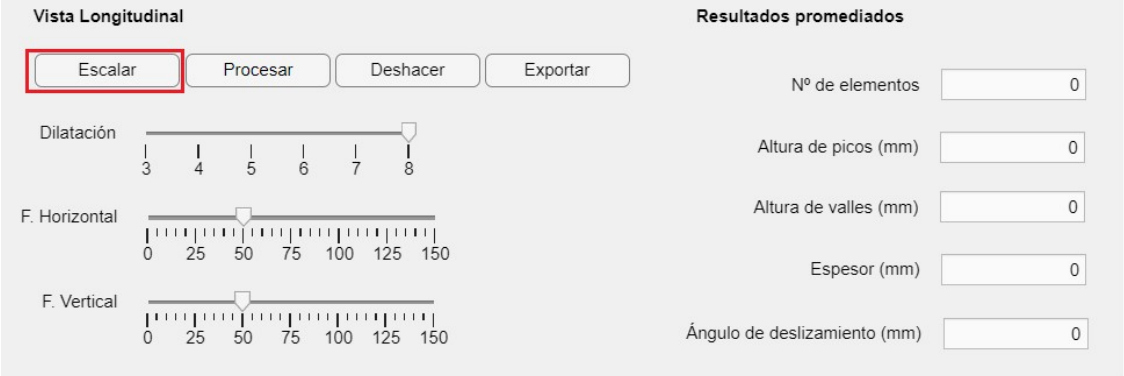


Figura 4.30- Botón para escalar la imagen.

A continuación, se presentan tres deslizaderas que son clave en la interfaz, cada una con un propósito específico en el procesamiento de imágenes. La primera de ellas es la deslizadera denominada “Dilatación”, cuyo rango de ajuste va desde 3 hasta 8. A medida que se incrementa el valor en esta deslizadera, la imagen procesada experimentará una dilatación mayor, lo cual empeora muy ligeramente los resultados finales, pero que por otro lado es extremadamente útil en imágenes en las cuales la viruta presenta muchas irregularidades, por lo que permite su procesamiento. Por otro lado, la deslizadera denominada “Factor horizontal” permite ajustar el algoritmo según la separación entre dientes y varía en un intervalo de 0 a 150, brindando flexibilidad para adaptar el análisis a diferentes tipos de virutas. Por último, la deslizadera llamada “Factor vertical” regula el procesamiento en función a la distancia que se da entre los valles y la parte inferior de la viruta, también en un rango que va de 0 a 150. Estas tres herramientas proporcionan a los usuarios un control preciso sobre el procesamiento de imágenes, permitiéndoles ajustar los parámetros según las necesidades específicas de la inspección, y garantizando resultados de alta calidad (fig. 4.31).

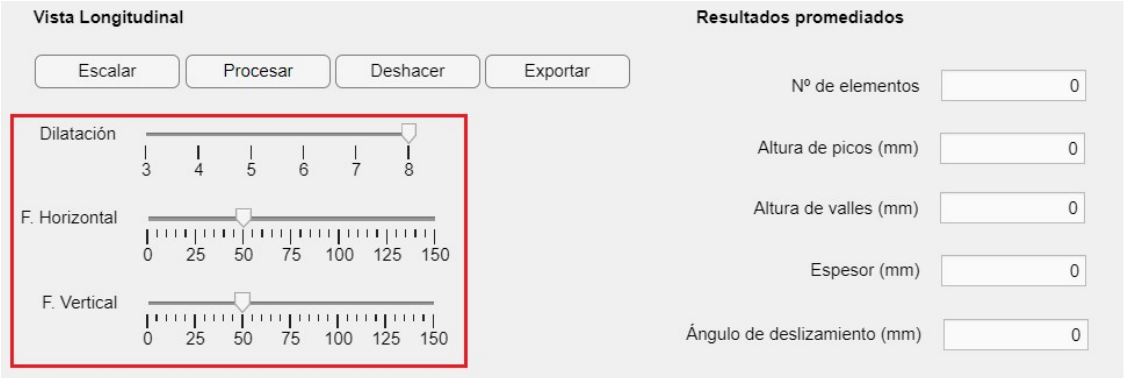


Figura 4.31-. Deslizaderas para modificar los factores horizontal, vertical y dilatación.

Para lograr resultados precisos y adaptados a cada viruta, es fundamental comprender cómo ajustar adecuadamente las deslizaderas disponibles en la APP. En primer lugar, al abrir una imagen, se recomienda observar los valores por defecto en las deslizaderas, que son “Dilatación” igual a 8, “Factor horizontal” igual a 30 y “Factor vertical” igual a 30. A continuación, al pulsar la tecla “Procesar”, se obtiene una primera vista del procesamiento y pueden darse dos casos diferentes.

En el primero de los casos la imagen resulta bien procesada (fig. 4.31), entonces, en caso de querer mejorar la calidad de la imagen procesada, el siguiente paso es disminuir gradualmente el valor de “Dilatación” y volver a procesar la imagen hasta que el algoritmo lo permita, lo que proporcionará una representación más detallada de la viruta.

En el segundo caso (el más común), la imagen no se procesa adecuadamente, entonces será necesario ajustar tanto el “Factor horizontal” como el ”Factor vertical”. Para ello, el usuario debe aumentar el valor del “Factor horizontal” mientras mayor sea la separación entre los picos de la viruta observando la imagen a simple vista y viceversa. Respecto al “Factor vertical”, debe ser incrementado a medida que se dé una mayor separación entre los valles y la parte inferior de la viruta, o disminuido, si la separación es menor. Se deben realizar estos ajustes hasta lograr una segmentación satisfactoria (fig. 4.32). Una vez ajustados los factores horizontal y vertical según sea necesario, se debe disminuir gradualmente el valor de “Dilatación” y reprocesar la imagen de manera progresiva hasta alcanzar el resultado deseado al igual que en el primer caso. Este enfoque iterativo permite al usuario adaptar eficazmente el procesamiento de imágenes a las características únicas de cada viruta, asegurando un análisis preciso y detallado.

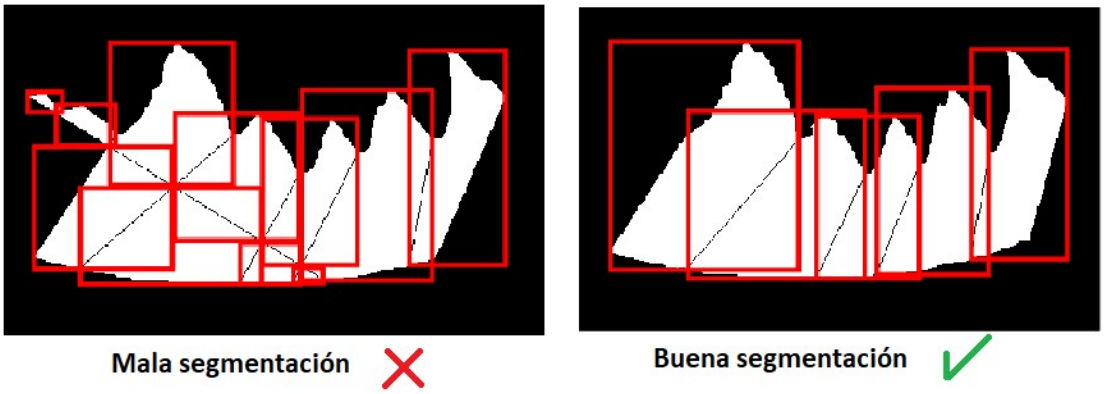


Figura 4.32- Ejemplo de viruta mal segmentada y bien segmentada.

Tal como se ha mencionado previamente, una vez ajustadas las deslizaderas según las necesidades específicas, el siguiente paso es utilizar el botón “Procesar”. Esta función aplicará a la imagen seleccionada todo el proceso detallado en el apartado 4.4, desde el binarizado de la imagen hasta la obtención de resultados finales.

Además, en la interfaz, se incorpora un botón denominado “Deshacer”, cuya función principal es revertir los cambios realizados sobre la imagen después de haberla procesado, devolviéndola a su estado base, manteniendo únicamente la escala ajustada. Si bien este botón puede ser de utilidad en casos donde el procesamiento inicial no sea satisfactorio, es importante destacar que no es necesario su uso, ya que el usuario puede simplemente volver a pulsar el botón “Procesar” para obtener el mismo resultado. No obstante, el botón “Deshacer” permanece disponible en la interfaz para aquellos momentos en que el usuario desee regresar a la imagen original a modo de referencia o comparación en cualquier punto del proceso (fig. 4.33).

Cabe recalcar que, tras procesar o deshacer cambios sobre una imagen, se podrán observar todos los cambios gráficamente, ya que la ventana en la que se muestra la imagen se va actualizando automáticamente con cada cambio realizado.

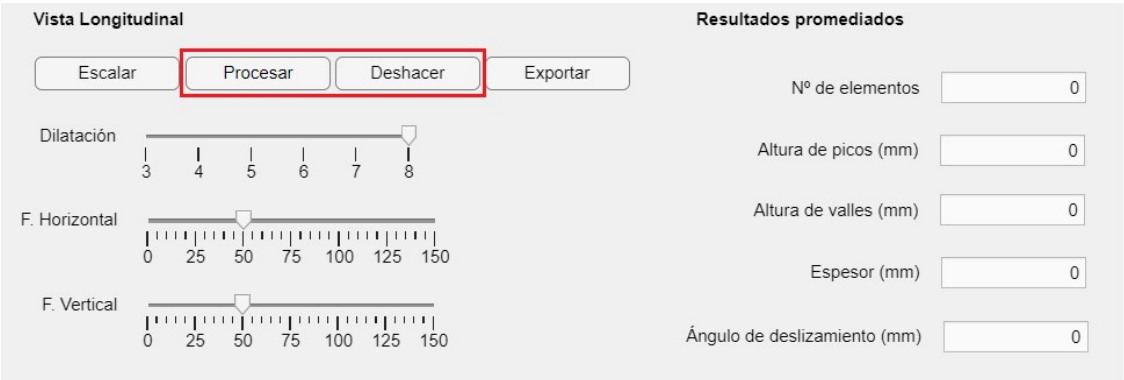


Figura 4.33- Botones para procesar la imagen y deshacer el procesado respectivamente.

Una vez que la imagen ha sido procesada, en la parte derecha de la interfaz se presentan los resultados obtenidos para cada uno de los parámetros en una serie de cuadros de texto (fig. 4.34). Estos cuadros proporcionan una representación numérica de las mediciones y datos relevantes extraídos del análisis de la viruta, ofreciendo al usuario una información detallada sobre las características de la muestra. Esta disposición facilita la revisión de los resultados para su posterior análisis.

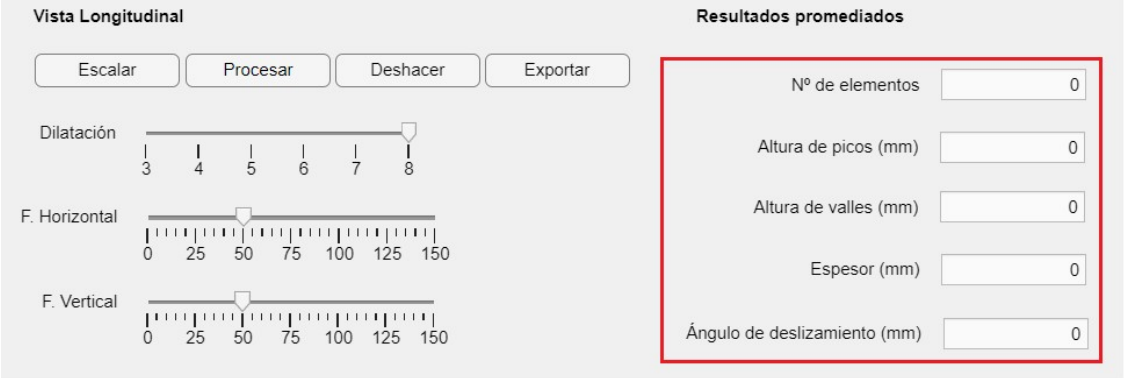


Figura 4.34- Resultados numéricos del procesamiento de imagen.

Finalmente, se encuentra la tecla denominada “Exportar” (fig. 4.35). La función principal de esta tecla es generar un documento en formato xlsx (Excel) o crear una nueva hoja si ya existe dicho documento, y exportar los resultados obtenidos previamente durante el procesado de la imagen. Esto facilita la documentación y el almacenamiento de los datos procesados, permitiendo al usuario conservar un registro organizado y accesible de los resultados de cada imagen. La opción de exportar es esencial para compartir resultados, realizar un seguimiento de las mediciones y realizar un posterior análisis o comparación, lo que contribuye a un flujo de trabajo eficiente y completo.

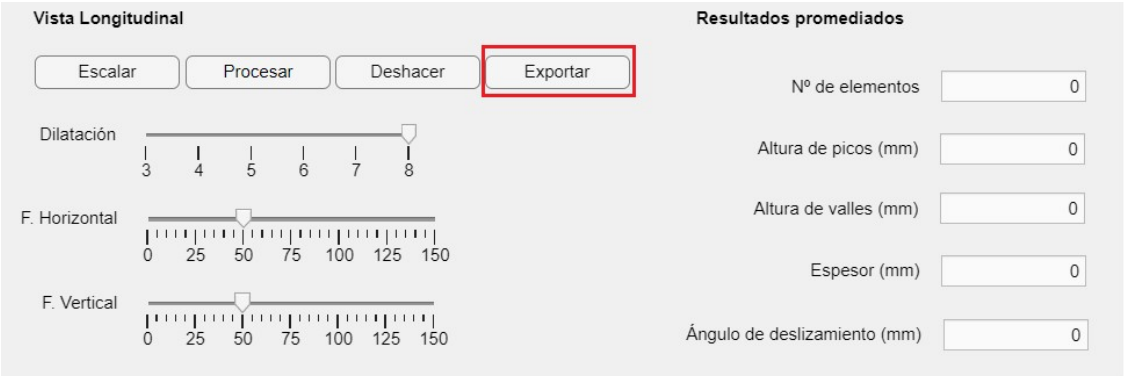


Figura 4.35- Botón para exportar los resultados numéricos a Excel.

Los resultados exportados se presentan de manera ordenada y estructurada en el archivo Excel (fig. 4.36). En primer lugar, se muestra el nombre del archivo de la imagen importada, lo cual permite referenciar de forma inmediata la muestra analizada. A continuación, se indica el tipo de aleación detectada mediante visión artificial.

Posteriormente, se especifica si la viruta se encuentra en vista longitudinal o transversal, lo cual ayuda a contextualizar los resultados en función de la orientación de la muestra.

Finalmente, en las columnas subsiguientes, se presentan de manera detallada los resultados numéricos correspondientes a los parámetros analizados, proporcionando una visión completa y precisa de las mediciones realizadas durante el procesamiento. Esta estructura facilita la interpretación y el uso posterior de los datos exportados, lo que es esencial para un análisis efectivo y una documentación eficiente.

Nombre del Archivo	125_01_20x_4a_s_a.jpg
Aleación detectada	Ti6Al4V
Vista	Longitudinal
Aumentos	20x
Altura de Picos (mm)	0,11789
Altura de Valles (mm)	0,068063
Espesor (mm)	0,11336
Angulo de Deslizamiento (°)	70,8615

Figura 4.35- Resultados numéricos una vez exportados a Excel.

Zona 3. Viruta en vista longitudinal no segmentada

En algunos casos, se enfrenta la dificultad de no poder apreciar claramente la diferencia entre picos y valles en la viruta en vista longitudinal, o bien el ajuste realizado previamente no proporciona resultados satisfactorios. Para abordar este problema, se ha incorporado una característica especial en la interfaz de la aplicación. Esta función tiene la capacidad de medir la longitud del eje menor de la viruta, que representa un promedio entre picos y valles. Esta medida proporciona una solución eficaz para casos en los que la diferenciación entre picos y valles es complicada, permitiendo obtener algunos resultados de la viruta incluso en situaciones desafiantes de inspección.

En esta variante del proceso, las teclas “Procesar” y “Deshacer” cumplen la misma función que en el caso anterior, aunque con una diferencia importante. En este contexto, no es necesario realizar el escalado de la imagen ni ajustar ninguna de las deslizaderas, ya que la aplicación trabaja directamente con la imagen tal como se presenta. Las teclas “Procesar” y “Deshacer” permiten aplicar o revertir el procesamiento de imagen sin la necesidad de ajustes adicionales, simplificando así

la tarea de procesamiento, su función es exactamente la misma que en el contexto anterior (fig. 4.37).

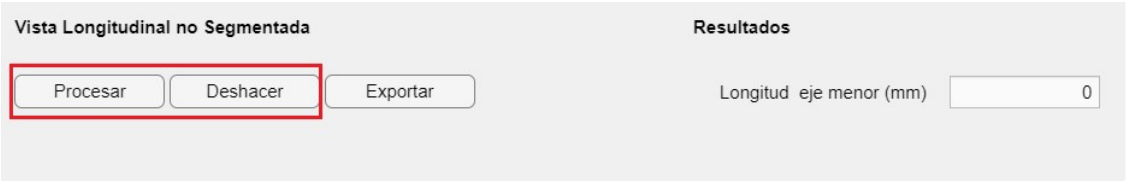


Figura 4.37- Botones para procesar la imagen y deshacer el procesado respectivamente.

Posteriormente, después de procesar la imagen, se presenta nuevamente un cuadro donde se muestran los resultados obtenidos, permitiendo al usuario revisar y documentar las mediciones. Asimismo, se incluye la tecla “Exportar”, cuya función es exactamente la misma que en el contexto anterior. Esta tecla facilita la exportación de los resultados en un archivo Excel, manteniendo el mismo formato estructurado que permite un fácil análisis de los datos obtenidos (figs. 4.38 y 4.39).

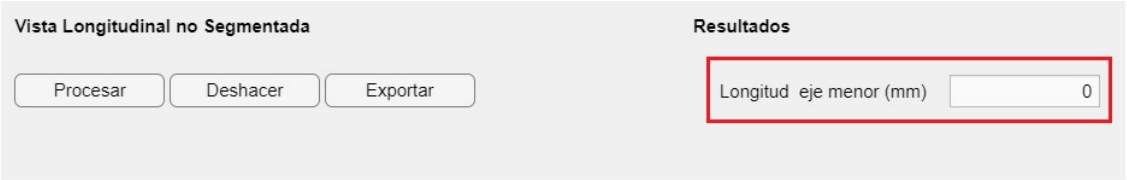


Figura 4.38- Resultados numéricos del procesamiento de imagen.

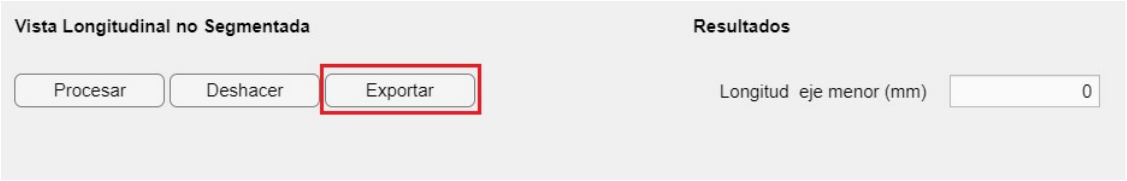


Figura 4.39- Botón para exportar los resultados numéricos a Excel.

Zona 4. Viruta en vista transversal

En ciertos casos, las imágenes de viruta se presentan en vista transversal, lo cual implica la necesidad de extraer diferentes datos. En este contexto, la interfaz de la aplicación incluye una deslizadera denominada “Dilatación”, cuya función es idéntica a la previamente explicada. Sin embargo, en este caso, se recomienda mantener la deslizadera en su valor por defecto, que es igual a 4, a menos que se produzca algún error inusual, lo cual es poco probable. En el caso de que se presente un error

en el procesamiento, se recomienda aumentar progresivamente el valor de la deslizadera y volver a procesar la imagen para lograr una segmentación adecuada.

Por otro lado, las teclas “Procesar” y “Deshacer” vuelven a estar disponibles en esta variante, desempeñando las mismas funciones explicadas anteriormente. Estas teclas permiten aplicar o revertir el procesamiento de imagen según sea necesario (figs. 4.40 y 4.41).

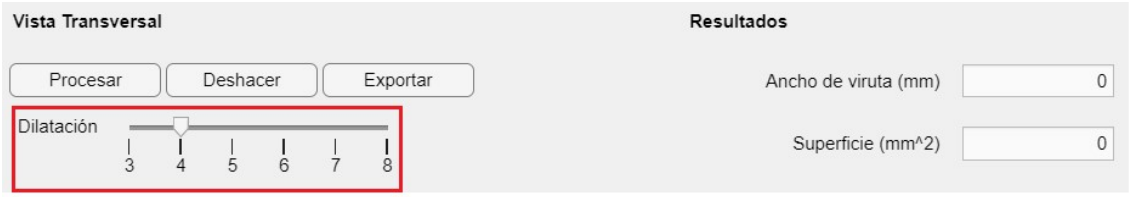


Figura 4.40- Deslizadera para seleccionar el factor de dilatación de la imagen.

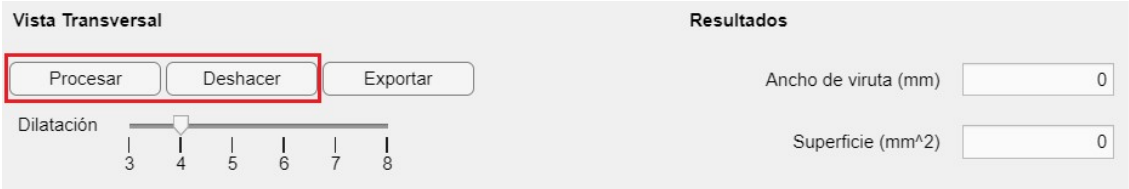


Figura 4.41- Botones para procesar la imagen y deshacer el procesado respectivamente.

Finalmente, en la parte final de la interfaz, vuelven a aparecer cuadros de texto en los que se muestran los resultados obtenidos tras el procesamiento de la imagen. Estos cuadros proporcionan una representación numérica de las mediciones extraídas del análisis de la viruta, permitiendo al usuario acceder a una información detallada sobre las características de la muestra. Además, nuevamente se encuentra disponible una tecla "Exportar" que facilita la opción de exportar los resultados a Excel para su posterior análisis (figs. 4.42 y 4.43).

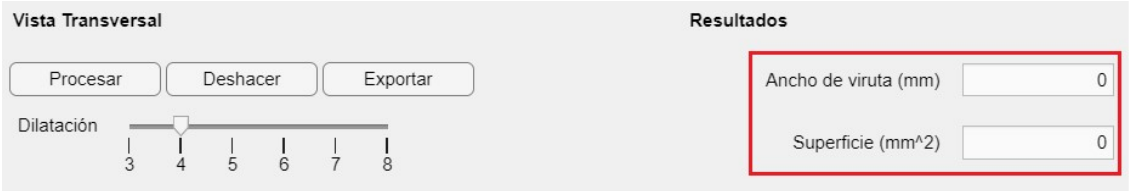


Figura 4.42- Resultados numéricos del procesamiento de imagen.

Figura 4.43- Botón para exportar los resultados numéricos a Excel.

En conclusión, la interfaz de la aplicación permite al usuario importar imágenes y ajustarlas según sus necesidades con herramientas como rotación y espejo. Además, tres deslizaderas permiten un control preciso. Tras los ajustes, se aplica fácilmente el procesamiento de imágenes, y los resultados se muestran en cuadros de texto. También hay una opción para exportar los resultados. En resumen, la interfaz brinda un conjunto completo de herramientas para el procesamiento y análisis de imágenes de virutas.

4.5.3. Guía de instalación sin necesidad de tener Matlab instalado

Hasta el momento, se ha explicado detenidamente el funcionamiento de la aplicación diseñada en el entorno de Matlab para agilizar el proceso de selección de imágenes, su procesamiento, la obtención de resultados y la exportación de estos. Sin embargo, ¿qué tal si se la comodidad del usuario un paso más allá? Para lograrlo, se propone la conversión de esta aplicación de Matlab en una aplicación de escritorio. De esta manera, se elimina la necesidad de que el usuario tenga que contar con Matlab instalado, junto con sus diversas extensiones y Add-ons necesarios para que el algoritmo funcione correctamente. A continuación, se detallan los pasos necesarios para llevar a cabo la instalación de esta aplicación de escritorio.

Paso 1

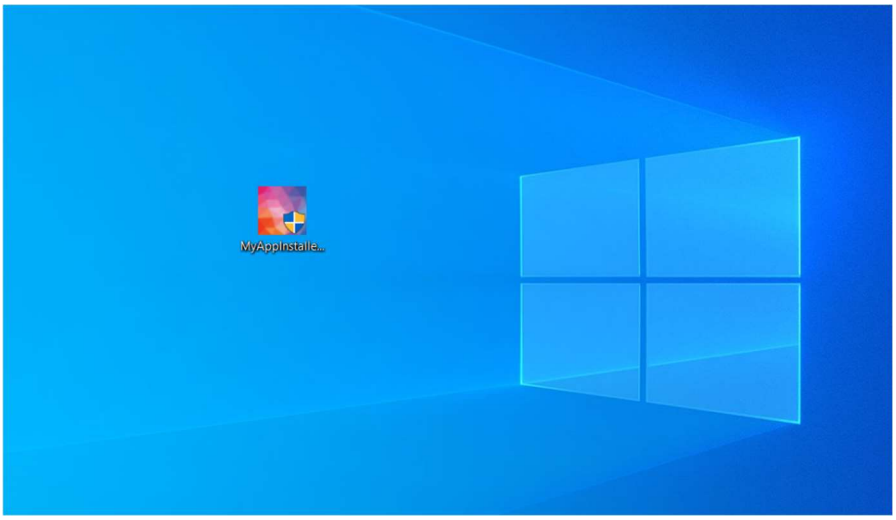


Figura 4.44- Instalación de la APP, paso 1.

Para comenzar el proceso de instalación de la aplicación de escritorio, el primer paso consiste en ejecutar el archivo ".exe" denominado "MyAppInstaller_mcr". Este archivo será el punto de partida para configurar y habilitar la aplicación en el equipo. Asegúrate de seguir atentamente las indicaciones que se presentarán durante la instalación para garantizar un proceso sin problemas.

Paso 2

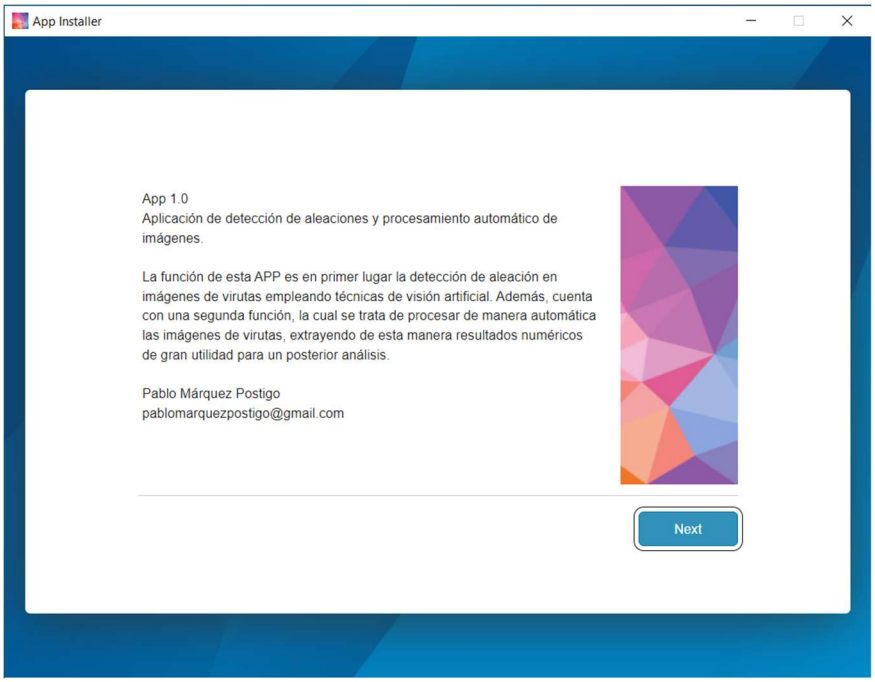


Figura 4.45- Instalación de la APP, paso 2.

Una vez ejecutado el archivo, se abrirá una ventana que mostrará las especificaciones de la aplicación. En este punto, se debe presionar la tecla "Next" para continuar con el proceso de instalación.

Paso 3

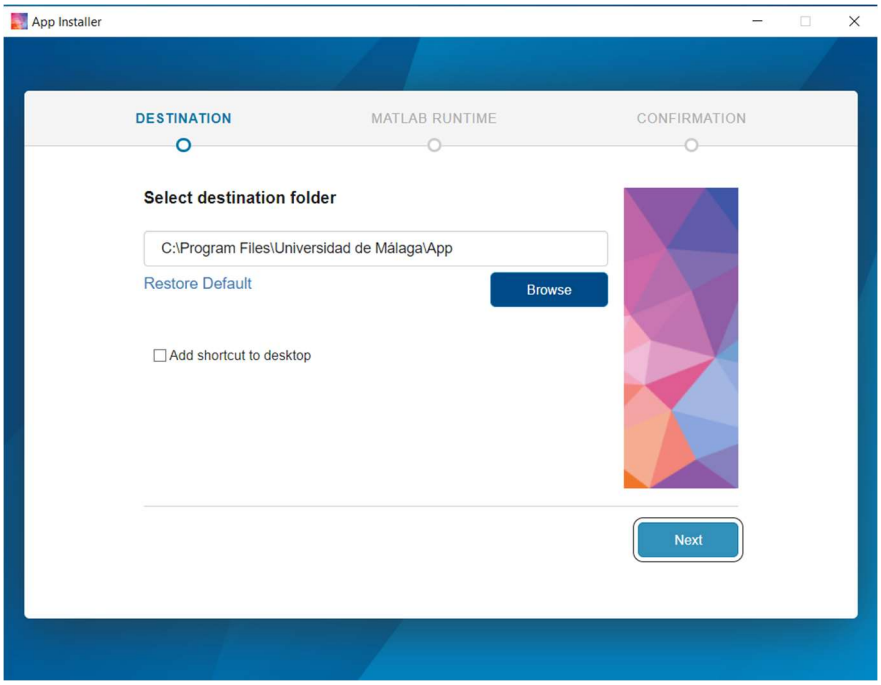


Figura 4.46- Instalación de la APP, paso 3.

A continuación, se abrirá una ventana que permite seleccionar el directorio de instalación para la aplicación. Aquí, se da la opción de elegir la ubicación en la que se instalará la aplicación. Además, hay una casilla que permite añadir un acceso directo en el escritorio, lo cual puede ser conveniente para un acceso rápido.

Una vez seleccionado el directorio de instalación y, si se desea, marcado la casilla para agregar un acceso directo en el escritorio, se debe pulsar la tecla "Next" para avanzar en el proceso de instalación. Esto garantizará que la aplicación se instale en la ubicación y con las opciones elegidas.

Paso 4

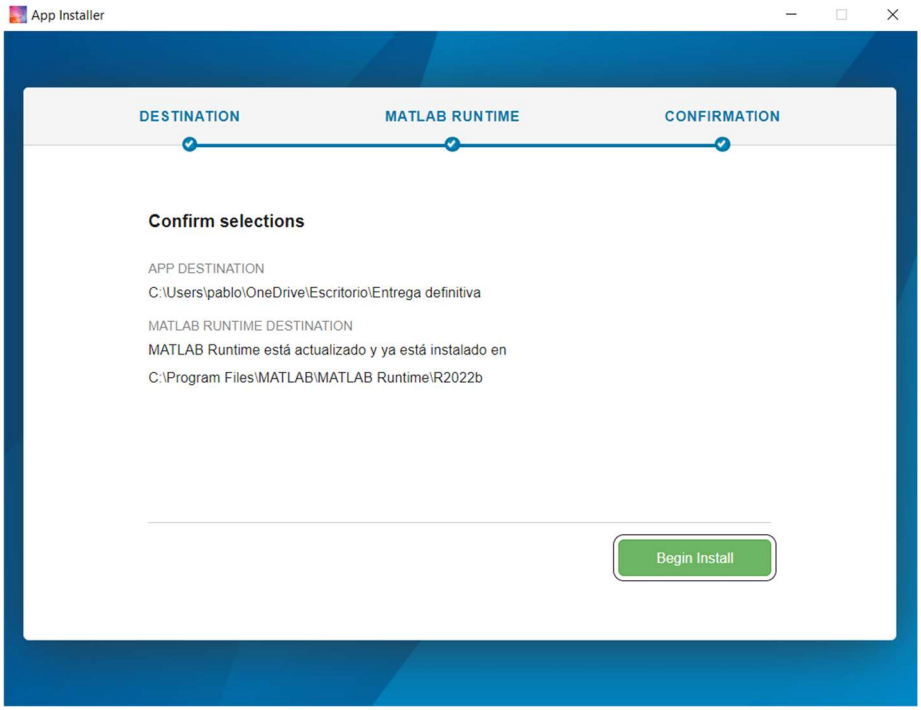


Figura 4.47- Instalación de la APP, paso 4.

En la siguiente pantalla, se te presentará una vista previa del directorio seleccionado para la instalación de la aplicación. Aquí se puede revisar que todo esté configurado de acuerdo con las preferencias. Si el directorio es el correcto, se puede pulsar la tecla de instalación. Esto iniciará el proceso de instalación de la aplicación de escritorio en el directorio elegido.

Paso 5

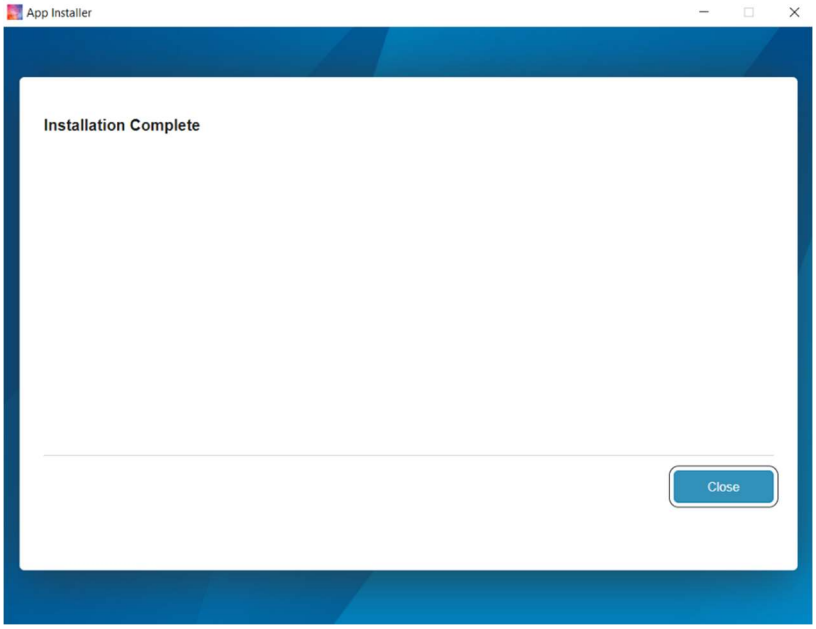


Figura 4.48- Instalación de la APP, paso 5.

Una vez que la instalación se haya completado con éxito, aparecerá una ventana que indica que el proceso ha finalizado. En este punto, simplemente se debe pulsar la tecla "Cerrar" para finalizar la instalación de la aplicación de escritorio. A partir de este punto, ya se podría utilizar la aplicación de manera conveniente, sin la necesidad de tener instalado Matlab ni sus extensiones, mejorando así la experiencia del usuario.

En la (fig. 4.49), se presenta una visualización de la aplicación de escritorio instalada con éxito. Esta interfaz proporciona una forma más cómoda y accesible para el usuario, eliminando la necesidad de contar con Matlab y sus extensiones. Ahora, el usuario puede disfrutar de las funcionalidades de la aplicación de una manera más eficiente.

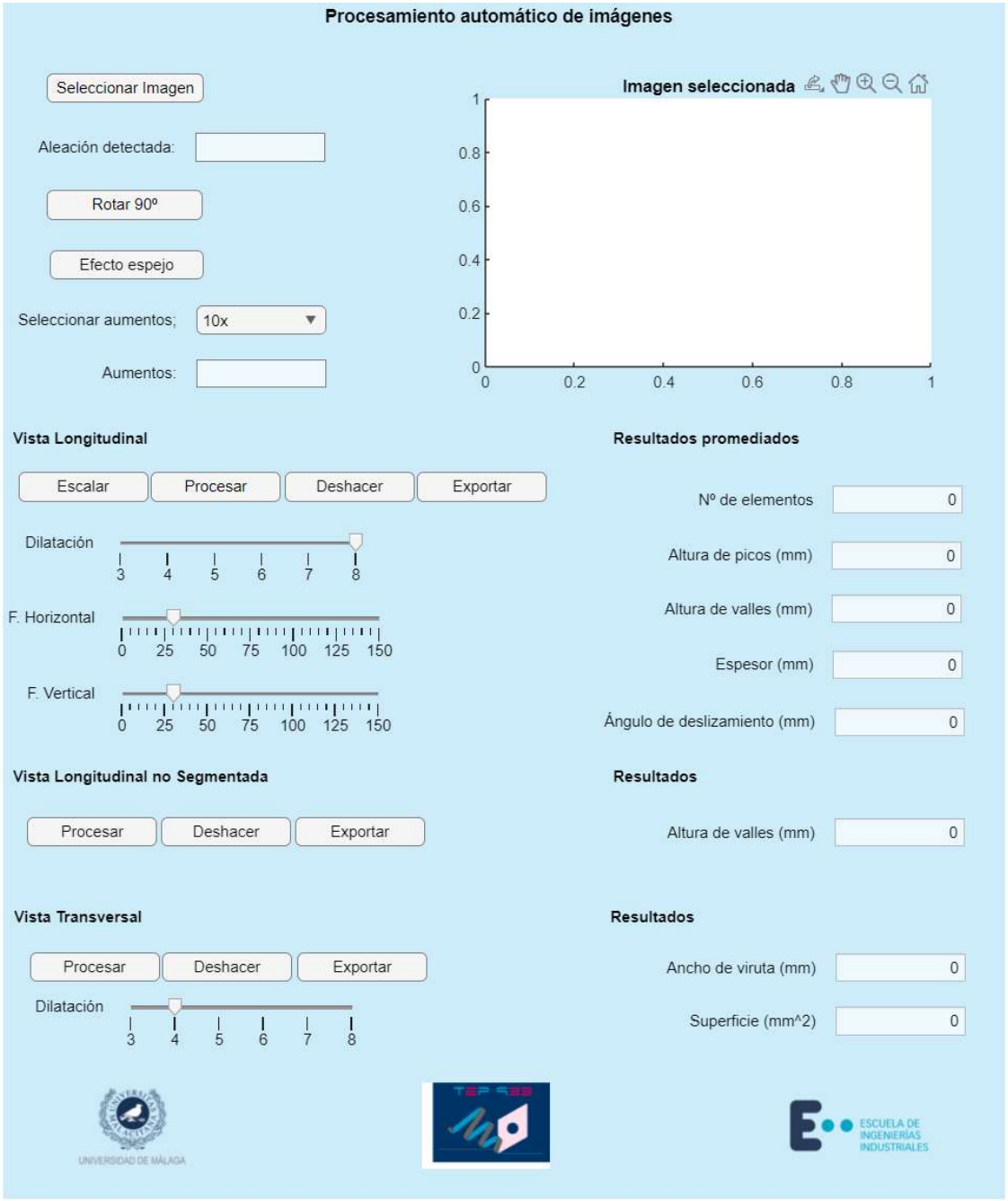


Figura 4.49- Interfaz de la APP de escritorio una vez instalada.

5. Resultados

5.1. Introducción

El objetivo de este capítulo es la visualización de los resultados obtenidos a partir del procesamiento digital de imagen detallado en el capítulo 4. Se explica cómo se obtienen dichos resultados automáticamente mediante el software Matlab y se ha realizado una comparación de los valores con los obtenidos mediante una metodología no automatizada utilizada por el grupo de investigación en sus trabajos previos. Es de importancia mencionar que en Matlab, los resultados obtenidos se dan en píxeles, por lo cual es necesario convertirlos a unidades métricas antes de realizar la comparación. Para realizar esta conversión, se ha utilizado ImageJ sobre una imagen que cuenta con una escala métrica y cuyo número de aumentos en el microscopio es conocido (fig. 5.1).

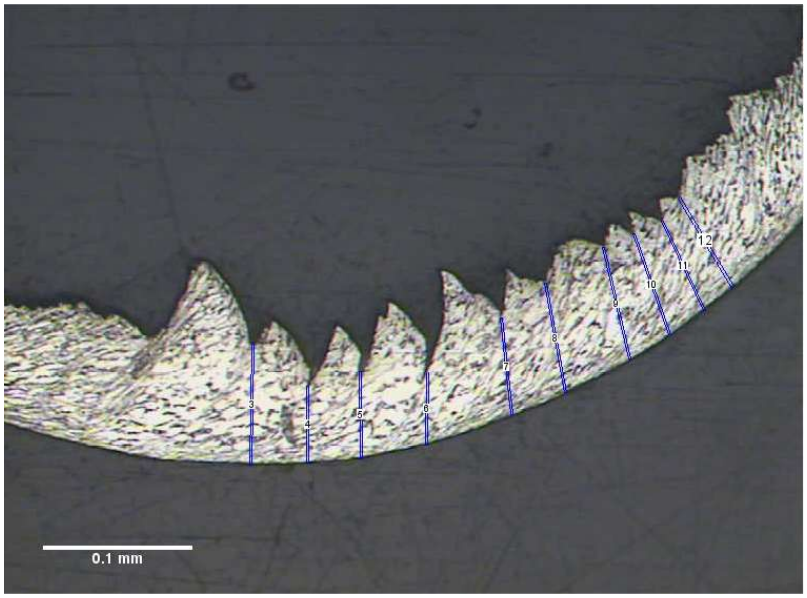


Figura 5.1- Imagen de viruta con escala métrica.

A partir de esta imagen, de la cual se conoce que ha sido tomada a 20 aumentos, es posible obtener un factor de conversión de píxeles a milímetros para dicho número de aumentos, así como para 10, 40 y 100 aumentos, cubriendo así todos los aumentos diferentes que aparecen en el banco de imágenes. Se han obtenido un total de 4 factores de conversión, uno para cada número de aumentos. De este modo, son implementados en Matlab y es el propio usuario quien selecciona el número de aumentos con el que está trabajando para que los resultados sean convertidos correctamente de píxeles a mm. Esos factores de conversión quedan reflejados en la (Tabla 5.1).

Tabla 5.1- Factores de conversión de píxeles a mm.

Aumentos	Distancia (mm)	Distancia (px)	Relación (mm/px)
10x	0,1	71	0,1/ 71
20x	0,1	142	0,1/ 142
40x	0,1	284	0,1/ 284
100x	0,1	710	0,1/ 710

5.2. Viruta longitudinal

En primera instancia se ha mostrado la obtención de resultados de una viruta longitudinal que, como se ha mencionado anteriormente, es aquella vista en la que se aprecian los diferentes dientes en forma de hoja de sierra. En este caso, se ha utilizado como ejemplo la viruta longitudinal de aleación de titanio Ti6Al4V mostrada en el capítulo 4 (fig. 4.7). No obstante, más adelante se muestra una comparación de resultados de distintas virutas con diferentes aumentos. Cabe destacar que cada resultado se muestra como un promedio. Por ejemplo, si en la imagen con la que se trabaja aparecen tres dientes, se obtienen sus tres respectivas alturas de pico, y estas son promediadas para ofrecer un único valor. Los resultados a obtener son las alturas de picos y valles, espesor de viruta y ángulo de deslizamiento [2,16].

5.2.1. Número de segmentos

Aunque no se trata de un resultado en sí, es de suma importancia que el programa sea capaz de detectar cuántos segmentos de viruta hay en la imagen. Como anteriormente se ha comentado, los resultados se han promediado entre todos los segmentos, por lo cual se hace necesario conocer el número exacto de ellos. En el caso que se muestra (fig. 5.2), se aprecia a simple vista que hay un total de cinco segmentos de viruta. Si el programa detectase más de cinco elementos, sería un indicativo de que algo va mal. En concreto, estaría detectando como diente alguna superficie pequeña separada de la viruta, lo cual provoca que afecte de manera negativa a los resultados, reduciendo el valor promedio. Para asegurar que la detección de segmentos funcione correctamente, se ha desarrollado una rutina cuya función es eliminar pequeñas aglomeraciones de píxeles, empleando para ello la función *“bwlabel”*.

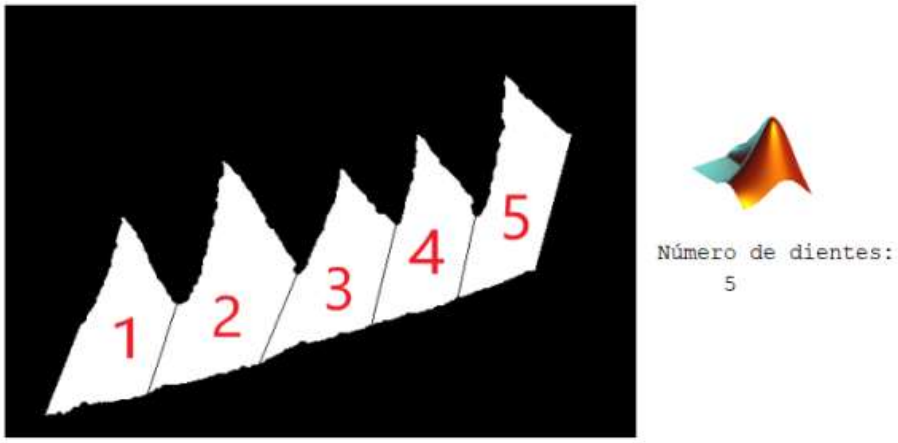


Figura 5.2- Número de segmentos obtenido de forma automática en Matlab.

5.2.2. Altura de picos

La altura de pico (h_p) puede ser definida como la distancia que va desde la parte más alta de un diente hasta la base de este siguiendo una línea paralela al eje vertical, tal y como se muestra en la (fig. 5.3).

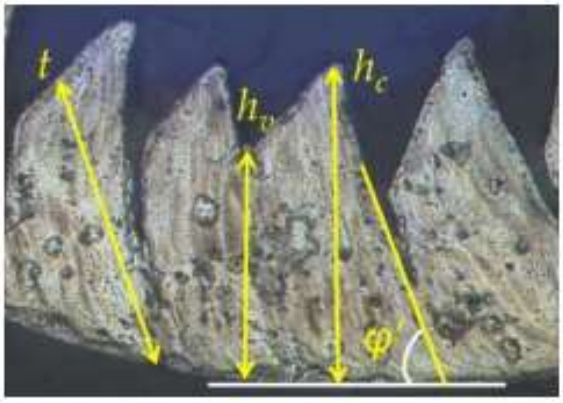


Figura 5.3- Representación de altura de picos, altura de valles, ángulo de deslizamiento y espesor de la viruta.

Si bien es cierto que en el capítulo 4 se obtuvieron las coordenadas de picos y valles, en una primera instancia se podría llegar a pensar que tomando sus coordenadas verticales se pueden conseguir directamente las respectivas alturas de picos y de valles. Sin embargo, no es así de sencillo, ya que dichas coordenadas verticales están tomadas desde la parte inferior de la imagen y no de la base de cada diente. No obstante, sí que se ha guardado la diferencia promedio entre picos y valles, ya que fue de utilidad más adelante. La manera más aproximada de conseguir las alturas de pico para posteriormente promediar es mediante las ya mencionadas *"Bounding Boxes"*, ya que al ser rectangulares y estar limitadas verticalmente por la base de diente como límite inferior, y por la parte más alta del diente como límite

superior; se convierten en una herramienta que otorgan la altura de pico de una manera casi exacta simplemente tomando su altura, siguiendo la definición dada sobre altura de pico (fig. 5.4).

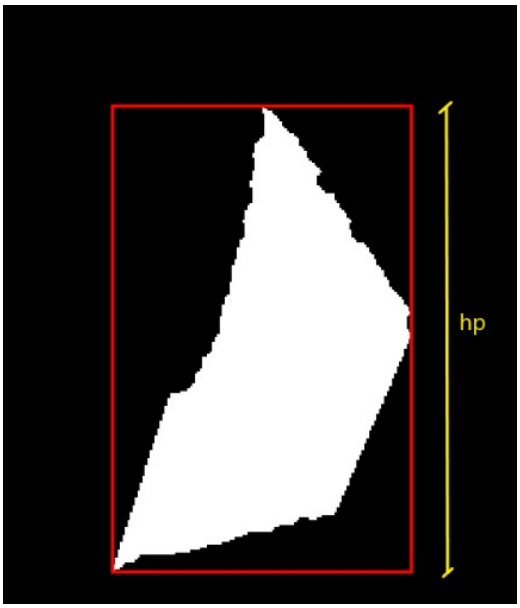


Figura 5.4- Obtención de la altura de pico de un diente a partir de su *Bounding Box*.

5.2.3. Altura de valles

Se entiende por altura de valles (h_v) a la distancia que va desde el valor mínimo en la ranura que se genera entre dos picos al mecanizar, hasta la base del diente, siguiendo una trayectoria perpendicular al plano horizontal (fig. 5.3).

Tal y como se ha explicado anteriormente, las coordenadas de picos y valles obtenidas en el capítulo 4 no son válidas para obtener alturas de picos y valles. No obstante, las diferencias entre picos y valles sí que son útiles, concretamente, el promedio de ellas. Ahora que se tiene un promedio de alturas de pico, simplemente habría que restar a este el promedio de diferencias entre picos y valles; de este modo se consiguió directamente la altura promedio de valles (fig. 5.5).

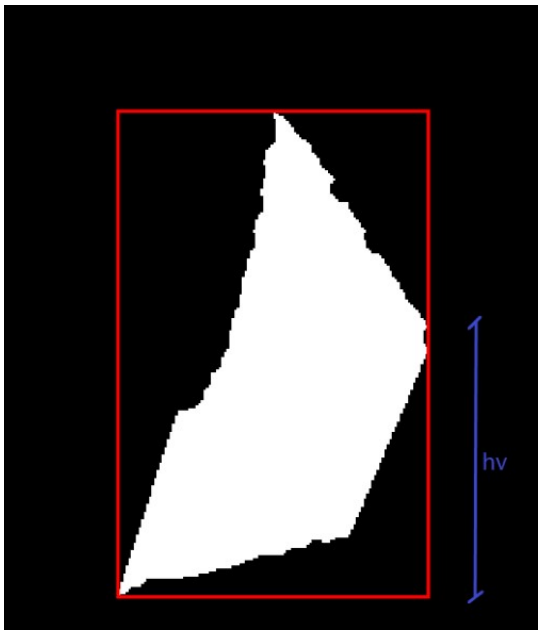


Figura 5.5- Altura de valle de un diente.

5.2.4. Espesor de viruta

Se conoce como espesor de viruta (t) de un diente a la longitud de su eje mayor (fig. 5.3). Para la obtención del promedio de anchos de viruta en la imagen seleccionada, se ha empleado la función *"regionprops.MajorAxisLength"*, consiguiendo así la longitud del eje mayor (fig. 5.6) de cada uno de los segmentos de la viruta. Una vez calculados, se ha realizado el correspondiente promedio, dependiendo del número de segmentos que se estén evaluando.

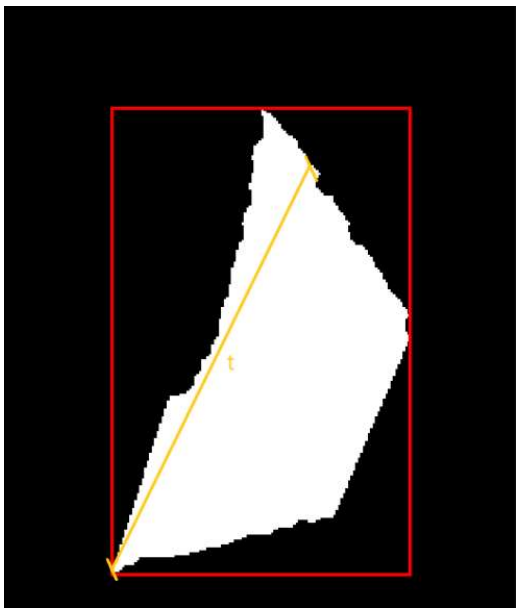


Figura 5.6- Espesor de viruta de un diente.

5.2.5. Ángulo de deslizamiento

El ángulo de deslizamiento o ángulo de corte (φ) se refiere al ángulo entre la dirección de la velocidad de corte y la dirección de la línea central de la viruta al formarse durante un proceso de corte o mecanizado. A efectos prácticos, el ángulo puede ser obtenido como un promedio de la orientación de cada diente de la viruta, el cual se consiguió gracias a la función *"regionprops.Orientation"* (fig. 5.7).

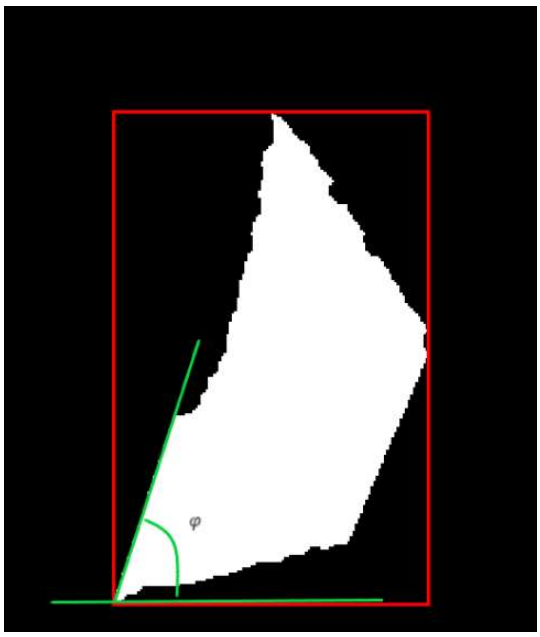


Figura 5.7- Ángulo de deslizamiento de un diente.

5.3. Viruta transversal

Para el caso de la viruta en vista transversal, de la cual ya se explicó anteriormente que es aquella en la que solo se puede observar un diente, ya que se ve de perfil; también se pueden obtener algunos resultados y por ello se creó un programa específico para procesar imágenes de este tipo de viruta en el capítulo 4. Concretamente, los resultados que se buscan extraer de este tipo de imágenes son superficie (S) y ancho (b) de la viruta (fig. 5.8).

Figura 5.8- Representación de superficie y ancho de viruta en vista transversal.

5.3.1. Superficie de viruta

En este caso, con superficie se refiere al área que ocupa la viruta en la imagen en vista de perfil. Para obtener el valor de la misma, sencillamente se utilizó la función *"regionprops.Area"*, obteniéndose su valor directamente.

5.3.2. Ancho de la viruta

Para obtener el valor de ancho de viruta, se volvió a utilizar la función *"regionprops.MajorAxisLength"* tal y como se ha realizado anteriormente para obtener el espesor en vista longitudinal. Esto se debe a que el ancho de viruta coincide con la longitud del eje mayor de la superficie, lo cual facilita bastante su obtención.

5.4. Comparación

Este apartado es de los más importantes de todo el trabajo, si no el que más, ya que en él se realiza una comparativa de los resultados obtenidos automáticamente mediante la generación de un código en *Matlab* con los mismos resultados obtenidos mediante una metodología no automatizada utilizada por el grupo de investigación en sus trabajos previos. Esto quiere decir que se verifica si los resultados obtenidos hasta el momento se asemejan a los valores reales con los que idealmente deberían coincidir.

El proceso de comparación de resultados se ha llevado a cabo siguiendo la siguiente estructura. En primer lugar, se ha realizado una evaluación comparativa de los resultados correspondientes a la aleación Ti6Al4V, abarcando tanto su vista longitudinal como transversal. Este análisis proporcionará una visión completa de las propiedades y características de la aleación en ambas orientaciones. A continuación, se procedió a mostrar una comparativa de los resultados obtenidos para las aleaciones UNS A92024 y UNS A97075, siguiendo la misma estructura, la cual incluye tanto la vista longitudinal como la transversal. Este enfoque permite una análisis preciso y ordenado de las propiedades de estas aleaciones.

5.4.1. Ti6Al4V, vista longitudinal

Altura de picos

En primer lugar, se ha llevado a cabo una comparación de las alturas de picos obtenidas por el equipo en contraste con las medidas realizadas a través de la aplicación para varias velocidades de avance, que abarcaban un rango desde 0.05 hasta 0.3 (mm/rev). Los resultados de este estudio se encuentran detallados en la (Tabla 5.2), la cual presenta de manera clara y concisa los datos recopilados, permitiendo una evaluación de la precisión de ambas metodologías. Cabe destacar

que los resultados mostrados a continuación con respecto a la vista longitudinal para la aleación Ti6Al4V, se corresponden con una velocidad de corte de 125 m/min. No obstante, el método automático proporcionado por la APP es totalmente válido para cualquier otra velocidad de corte.

Tabla 5.1- Datos comparativos de alturas de picos.

Altura de Picos		
Avance	Equipo	APP
0,05	0,08	0,04912
0,1	0,18	0,10329333
0,2	0,3	0,17305
0,3	0,41	0,29396667

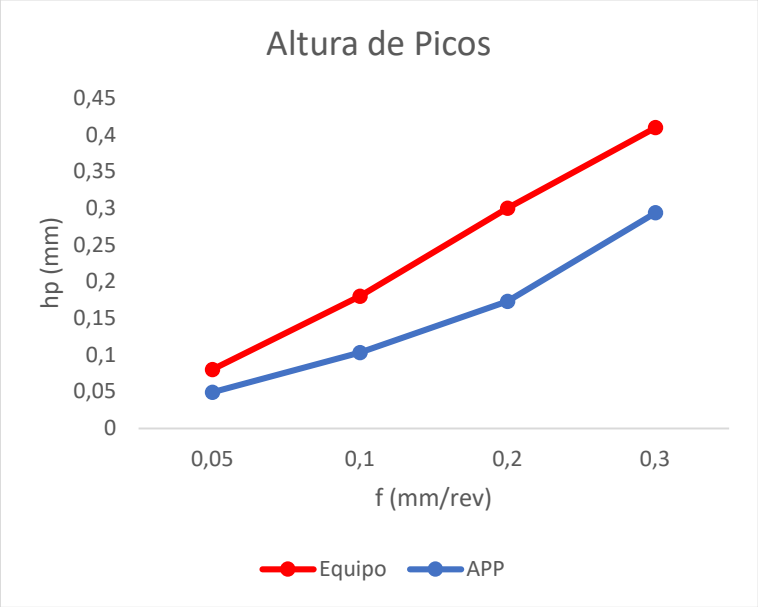


Figura 5.9- Gráfica comparativa de alturas de picos.

En la gráfica (fig. 5.9), se presenta una representación visual que compara la evolución de la altura de los picos a medida que aumenta la velocidad de avance, siendo la curva de color rojo la correspondiente a los datos tomados por el equipo; y la curva azul la que representa los datos tomados por la APP (se han usado los mismos colores para el resto de casos). Es evidente que ambas curvas muestran valores similares, aunque no llegan a converger completamente. Sin embargo, la tendencia general en ambas gráficas es claramente ascendente a medida que se ha incrementado la velocidad de avance. El motivo por el cual no convergen ambas

gráficas puede atribuirse a la deformación que experimenta la viruta en condiciones de altos avances, lo cual dificulta la medición precisa de los picos y los valles mediante el método experimental.

El método propuesto infravalora los resultados obtenidos mediante el procedimiento manual, para todos los avances analizados. Las mayores diferencias se dan a elevados avances. Es decir, las diferencias aumentan al aumentar el avance. Esto es debido a que, a mayor avance, la viruta obtenida presenta mayores picos y valles, es decir, una geometría en diente de sierra más pronunciada. Es en estos casos donde el procedimiento no automatizado puede generar más error, en función de los picos y valles analizados y la forma de medirlos con respecto a una línea tangente a la curvatura de la viruta. En los casos en los que los avances son pequeños, la superficie exterior de la viruta es más uniforme (0.05 y 0.1 mm/rev) por lo que la diferencia con el reconocimiento de imágenes es menor. En la (fig. 5.x) se puede ver una comparativa de la viruta longitudinal mecanizada a bajo y alto avance.

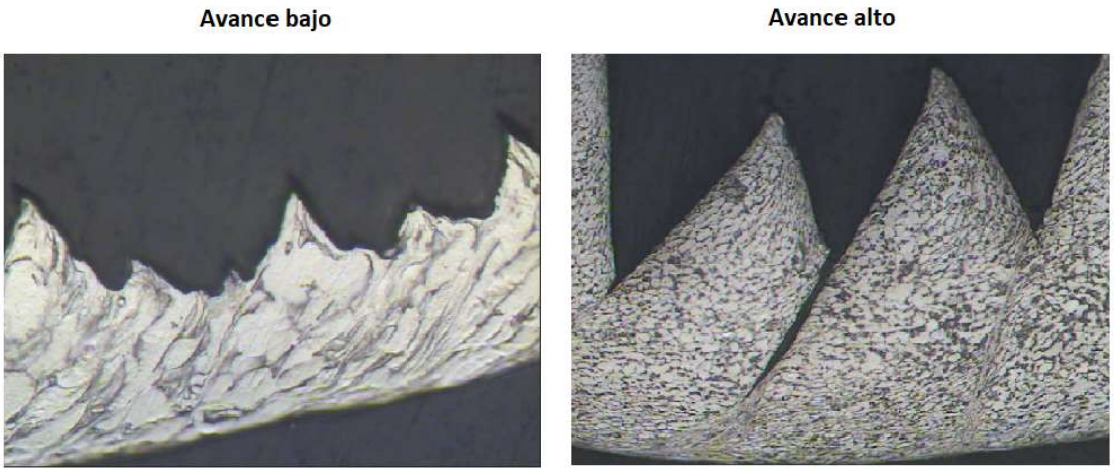


Figura 5.10- Comparación de viruta longitudinal de Ti6Al4V mecanizada a bajo y alto avance.

Altura de valles

Tabla 5.3- Datos comparativos de alturas de valles.

Altura de Valles		
Avance	Equipo	APP
0,05	0,05	0,04137
0,1	0,1	0,06938
0,2	0,2	0,114265
0,3	0,3	0,17729667

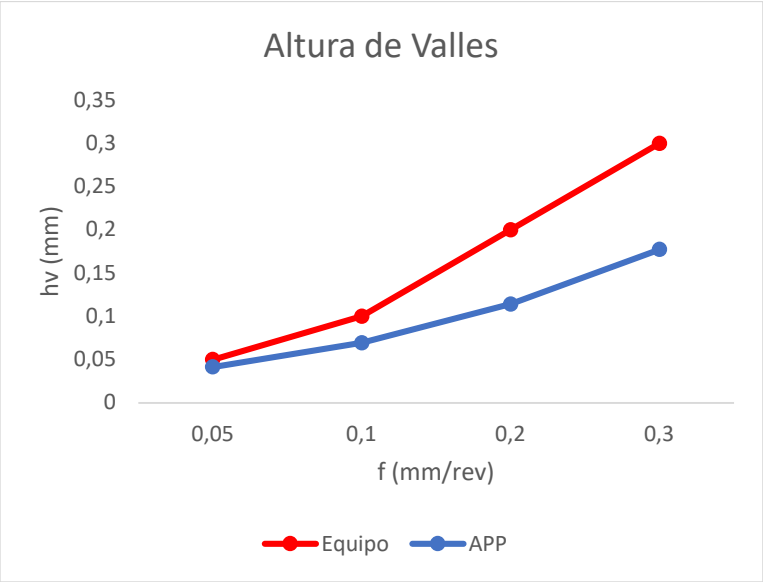


Figura 5.11- Gráfica comparativa de alturas de picos.

En relación con la altura de valles, se ha observado una diferencia notable entre la gráfica que representa los resultados obtenidos a través de la APP y la correspondiente a las mediciones realizadas con el equipo de laboratorio a altos avances. Conforme aumenta la velocidad de avance, los valores en la gráfica de la APP muestran un incremento más gradual en comparación con las mediciones del equipo, lo que resulta en una brecha cada vez más amplia entre ambas curvas. Esta discrepancia puede deberse a la característica de la viruta mencionada anteriormente, relacionada con la deformación de la viruta, la cual se intensifica a mayores velocidades de avance, dificultando así la precisión de las mediciones por el método experimental. No obstante, es importante destacar que la gráfica obtenida a través de la aplicación móvil sigue manteniendo la tendencia general de aumento en la altura de los valles a medida que aumenta el avance, al igual que ocurre con la gráfica correspondiente al método experimental. En la (fig. 5.10) se aprecia una comparación de la viruta longitudinal mecanizada a bajo y alto avance respectivamente.

Espesor

Tabla 5.4- Datos comparativos de espesor de viruta.

Espesor		
Avance	Equipo	APP
0,05	0,07	0,05828
0,1	0,14	0,11723333
0,2	0,25	0,2261
0,3	0,35	0,32733333

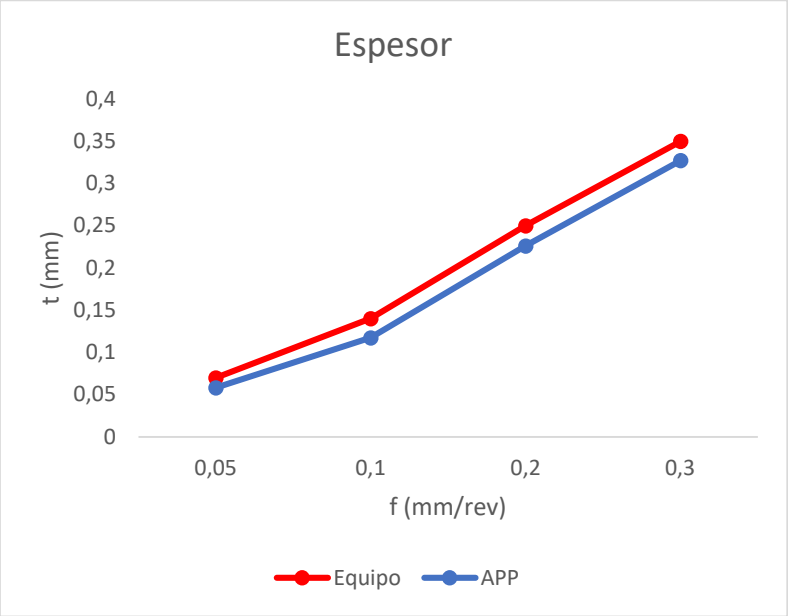


Figura 5.12- Gráfica comparativa de espesor de viruta.

En lo que respecta al espesor, es notable que ambas gráficas son muy parecidas, mostrando valores similares y una tendencia prácticamente igual en ambos casos. Este resultado da a entender que la medición del espesor, tanto mediante el equipo de laboratorio como a través de la aplicación, es consistente y precisa a lo largo de las diferentes velocidades de avance analizadas. La uniformidad en las tendencias y los valores refuerzan la confianza correspondiente a las mediciones de espesor, independientemente de la metodología utilizada, lo cual es un aspecto fundamental en el contexto de esta investigación. En la (fig. 5.10) se puede ver cómo varía el espesor de viruta según la velocidad de avance sea alta o baja

Ángulo de deslizamiento

Tabla 5.5- Datos comparativos correspondientes al ángulo de deslizamiento.

Ángulo de deslizamiento		
Avance	Equipo	APP
0,05	42	54,91
0,1	39	53,9033333
0,2	40	39,085
0,3	39	46,6733333

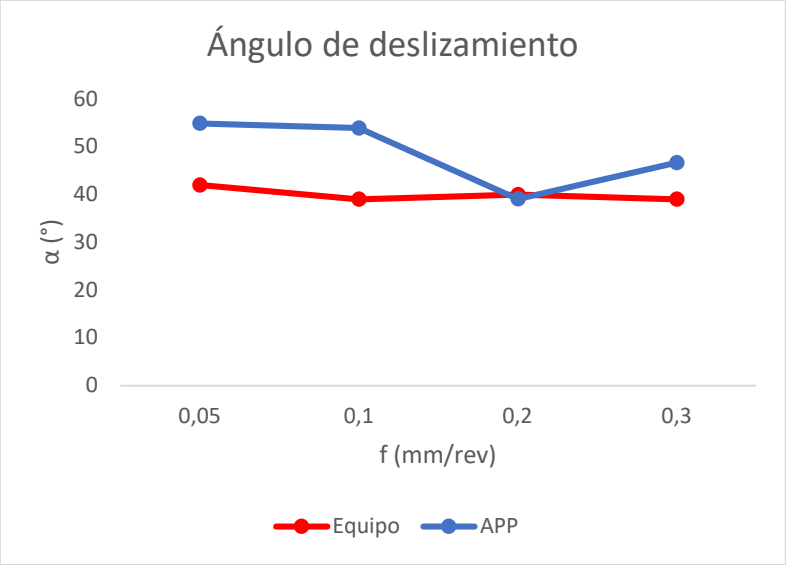


Figura 5.13- Gráfica comparativa de espesor de viruta.

Cuando se ha abordado la medición del ángulo de deslizamiento, se ha dado una complejidad para realizar esta tarea, ya que resulta desafiante obtener mediciones precisas tanto por el método experimental como por el método automático mediante la APP. En este contexto, se puede decir que ambas gráficas muestran resultados similares, aunque no idénticos. La similitud en los resultados obtenidos por ambas metodologías indica un nivel de coherencia en las mediciones, por lo cual se ha considerado un resultado satisfactorio dada la dificultad de esta tarea. Esto respalda la utilidad de ambas aproximaciones en la obtención de datos fiables sobre el ángulo de deslizamiento de la viruta.

5.4.2. Ti6Al4V, vista transversal

A continuación, se presenta una comparativa de los resultados correspondientes a la vista transversal de viruta de aleación Ti6Al4V, que en este caso son el ancho y superficie de viruta. Siguiendo la misma dinámica que se ha empleado para la vista longitudinal, estos resultados han sido obtenidos con una velocidad de corte de 125 m/min, aunque es importante destacar que el método automático es aplicable y válido para cualquier velocidad de corte, lo cual amplía su versatilidad a diversas condiciones de mecanizado.

Ancho de viruta

Tabla 5.6- Datos comparativos correspondientes al ancho de viruta, Ti6Al4V.

Ancho de Viruta		
Avance	Equipo	APP
0,05	1,05	0,42237125
0,1	0,96	0,6471185
0,2	0,99	0,92869251
0,3	0,96	0,89803713

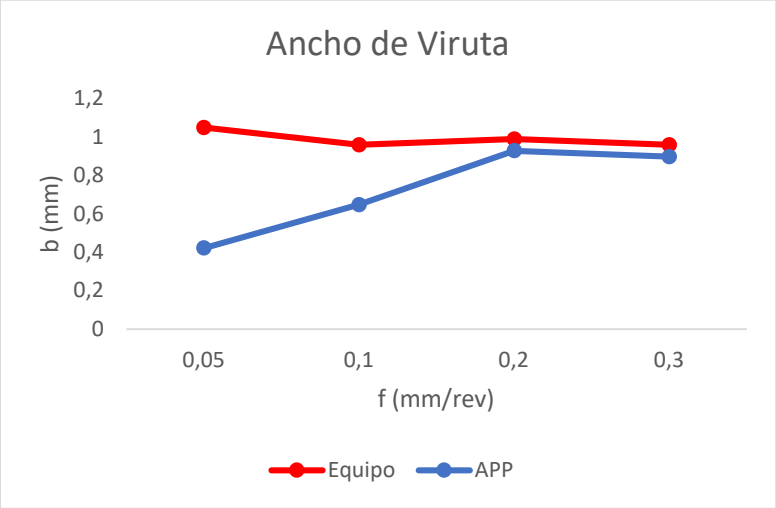


Figura 5.14- Gráfica comparativa de ancho de viruta, Ti6Al4V.

En la (fig. 5.13), se observa claramente que, a velocidades de avance más bajas, las mediciones realizadas a través de métodos no automatizados por el equipo presentan valores notablemente superiores en comparación con los obtenidos automáticamente mediante la aplicación. No obstante, a medida que se incrementa la velocidad de avance, ambas gráficas muestran una convergencia en sus resultados. Esta

discrepancia inicial se debe principalmente a las grandes dificultades que presenta la medición manual del ancho de la viruta, la cual tiende a ser menos precisa.

A avances bajos, las irregularidades en el ancho son mayores, lo cual hace más difícil la medición a estos avances, donde la viruta es bastante más estrecha y tiende a ser muy irregular. Por ello, en este caso, las mayores diferencias se producen a avances bajos, mientras que a avances altos los resultados son razonablemente parecidos. En la (fig. 5.15) se puede apreciar cómo varía la viruta de Ti6Al4V en vista transversal dependiendo de si ha sido mecanizada a una baja o alta velocidad de avance.

Figura 5.15- Comparación de viruta transversal de Ti6Al4V mecanizada a bajo y alto avance.

Superficie de viruta

Tabla 5.7- Datos comparativos correspondientes a la superficie de viruta, Ti6Al4V.

Superficie de Viruta		
Avance	Equipo	APP
0,05	0,08	0,01026251
0,1	0,13	0,0490974
0,2	0,17	0,09685578
0,3	0,42	0,16844012

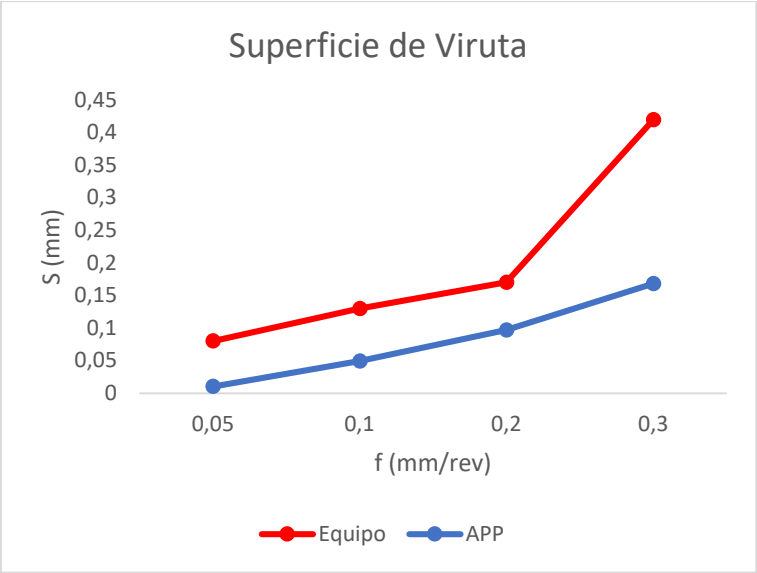


Figura 5.16- Gráfica comparativa de superficie de viruta, Ti6Al4V.

Tal y como se aprecia en la (fig. 5.16), ambas gráficas muestran resultados ligeramente divergentes, aunque la diferencia no es demasiado notable. No obstante, la diferencia sí es notable en el avance muy alto. Esto es debido al elevado grado de deformación que presenta la viruta en este caso, en dirección longitudinal, que la aleja de la forma ideal de un rectángulo. En estos casos es donde el recuento de píxeles es mucho más preciso que la aproximación a un rectángulo.

Por otro lado, la gráfica que representa los datos tomados por el equipo muestra en general valores ligeramente superiores en comparación con los obtenidos mediante la APP Sin embargo, ambas muestran una tendencia ascendente a medida que se incrementa la velocidad de avance, lo cual es coherente, dado que un aumento en el avance debería corresponderse con una mayor superficie en la viruta deformada. Es fundamental destacar que, al igual que en el caso del ancho de la viruta, medir con precisión la superficie de la viruta se ha presentado como una tarea sumamente compleja, más aún que la medición del ancho.

5.4.3. UNS A92024 y UNS A97075, vista longitudinal

A continuación, se procede a efectuar una comparación de los resultados obtenidos en la vista longitudinal para las aleaciones UNS A92024 y UNS A97075. Dado que en la gran mayoría de las imágenes de estas aleaciones la viruta no se presenta segmentada, lo cual dificulta la distinción entre picos y valles, la metodología empleada previamente para las aleaciones de Ti6Al4V no es aplicable en este caso, lo que impide obtener resultados análogos. No obstante, en su defecto, gracias al uso de las herramientas disponibles en MATLAB, se logra obtener una aproximación bastante precisa de las alturas de valles. Para evitar redundancia, a continuación se presentan únicamente los resultados correspondientes a la aleación UNS A97075,

ya que el proceso seguido es idéntico para la aleación UNS A92024 y arroja resultados igualmente precisos.

Tabla 5.8- Datos comparativos correspondientes a la altura de valles, UNS A97075.

Altura de Valles		
Avance	Equipo	APP
0,05	0,09	0,08872667
0,1	0,15	0,14676667
0,2	0,25	0,25936667
0,3	0,23	0,29436667

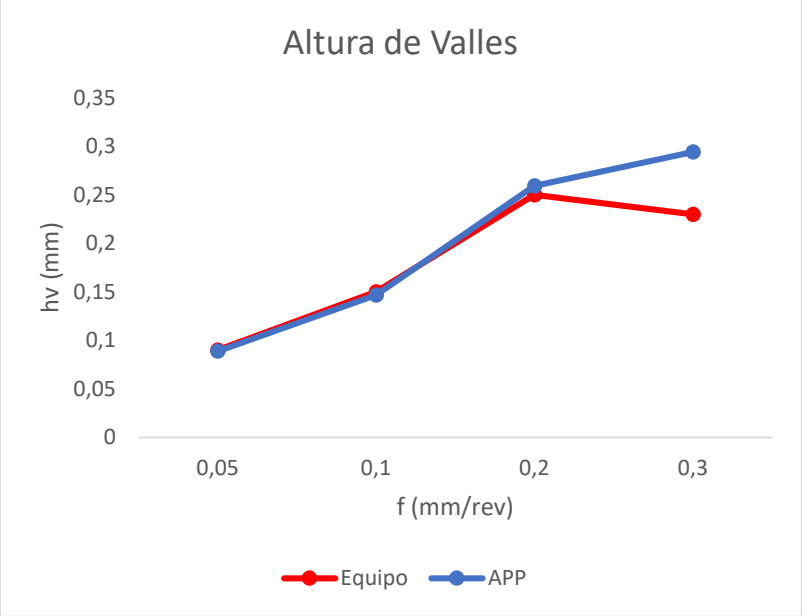


Figura 5.17- Gráfica comparativa de altura de valles, UNS A97075.

Como se puede observar en la (fig. 5.17), ambas gráficas muestran una notable similitud a medida que aumenta la velocidad de avance, manteniendo una tendencia creciente coherente con lo esperado según la teoría. Aquí los resultados son más parecidos, dado que a avances bajos el recuento de picos y valles no es necesario, midiendo el proceso un ancho medio. Es a avances altos, donde la viruta se vuelve segmentada, donde hay más diferencia, por las razones explicadas previamente. En términos generales, los resultados obtenidos de forma automática a través de MATLAB son altamente fieles a los obtenidos por el equipo mediante métodos no automatizados, lo cual respalda su fiabilidad.

Es relevante destacar que, en el caso de disponer de una imagen de viruta segmentada en la cual los picos y valles estén claramente distinguidos, es posible aplicar la misma metodología utilizada previamente para analizar virutas de la aleación Ti6Al4V. Esto abre la posibilidad de obtener un mayor abanico de resultados, lo que resalta la importancia de contar con imágenes de virutas adecuadamente segmentadas para realizar mediciones detalladas en el análisis de virutas de aleaciones UNS A92024 y UNS A97075. En la (fig. 5.18) se puede apreciar como varía la viruta mecanizada de UNS A97075 según sea mecanizada mediante un bajo o alto avance.

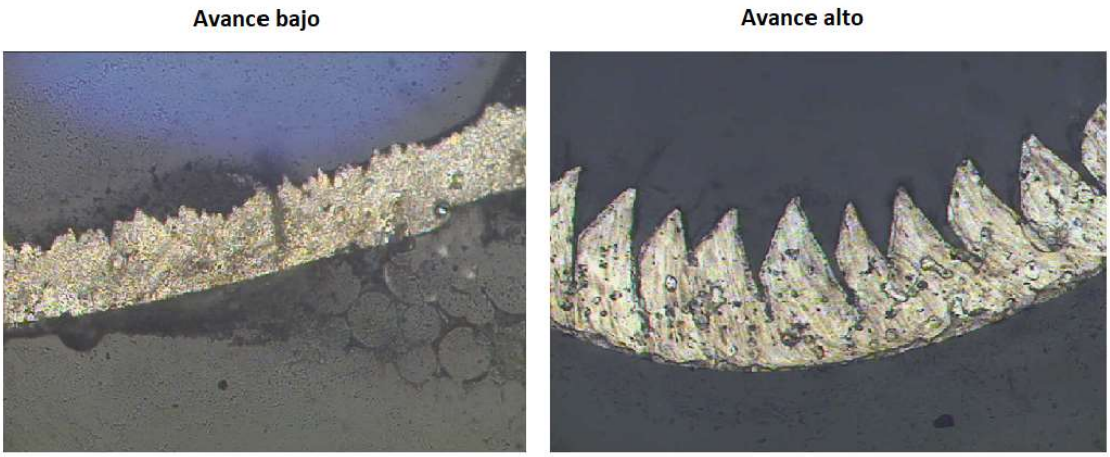


Figura 5.18- Comparación de viruta longitudinal de UNS A97075 mecanizada a bajo y alto avance.

5.4.4. UNS A92024 y UNS A97075, vista transversal

En última instancia, se procede a comparar los resultados obtenidos para la vista transversal de las aleaciones UNS A92024 y UNS A97075. Se ha seguido el mismo procedimiento que el utilizado para la obtención de resultados en las virutas de la aleación Ti6Al4V. En este caso, se presentan los resultados correspondientes a la aleación UNS A92024, aunque el proceso es idéntico para la aleación UNS A97075, generando resultados igualmente precisos y comparables.

Ancho de viruta

Tabla 5.9- Datos comparativos correspondientes al ancho de viruta, UNS A92024.

Ancho de Viruta		
Avance	Equipo	APP
0,05	1,19	1,0078
0,1	1,23	0,98583
0,2	1,2	1,0143
0,3	1,25	1,093

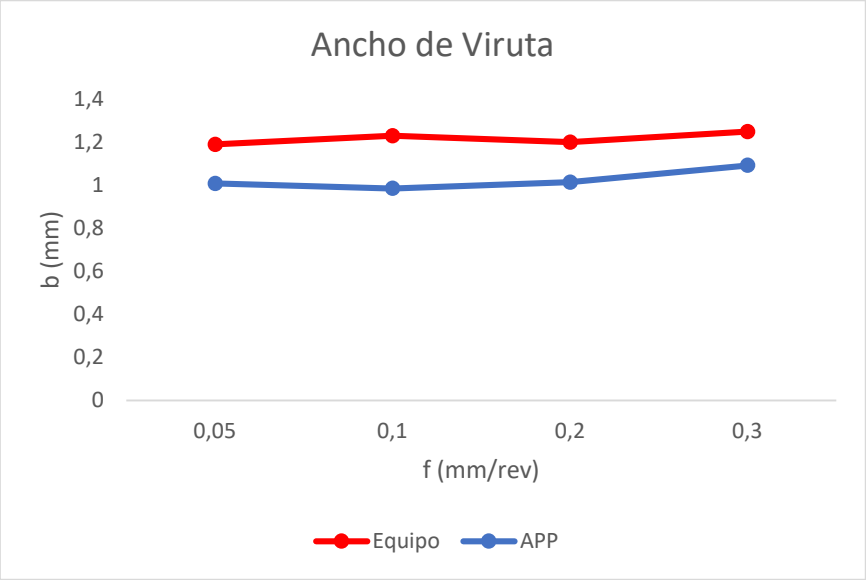


Figura 5.19- Gráfica comparativa de ancho de viruta, UNS A92024.

Como se puede apreciar en la (fig. 5.19), en ambas gráficas se observan valores bastante similares, siendo las mediciones del equipo ligeramente superiores a las obtenidas mediante la aplicación. A pesar de esta diferencia, es destacable que ambas curvas siguen una misma tendencia general, lo que indica una coherencia en los resultados a medida que aumenta el avance. Estos resultados refuerzan la utilidad de ambas metodologías y su capacidad para proporcionar datos consistentes en el contexto de estudio. No obstante, aunque no se aprecia la diferencia a bajos avances, cabe volver a destacar que, para estos avances, las irregularidades en el ancho son mayores, lo que hace más difícil la medición, ya que la viruta es bastante más estrecha y tiende a ser muy irregular. En la (fig. 5.20) se aprecia una comparación de viruta transversal mecanizada a bajo y alto avance respectivamente.

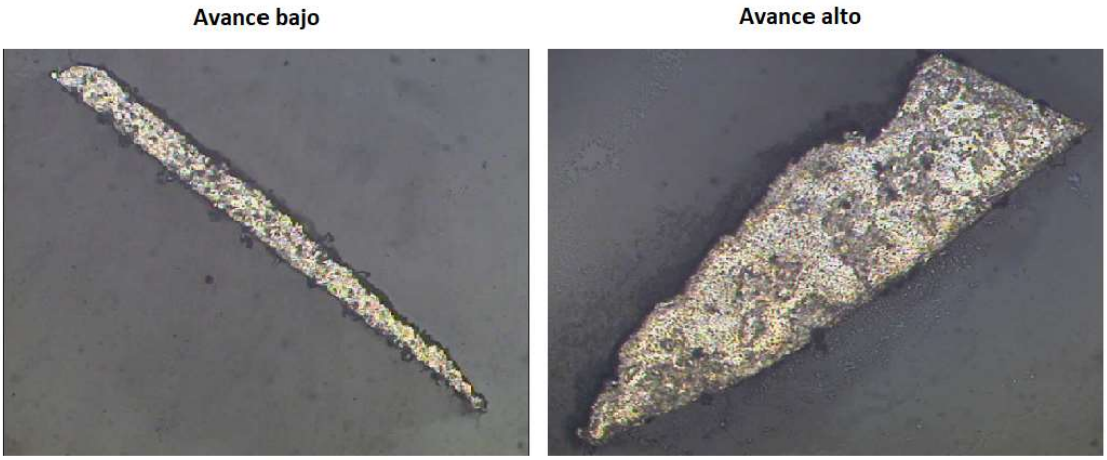


Figura 5.20- Comparación de viruta transversal de Al mecanizada a bajo y alto avance.

Superficie de viruta

Tabla 5.10- Datos comparativos correspondientes a la superficie de viruta, UNS A92024.

Superficie de Viruta		
Avance	Equipo	APP
0,05	0,09	0,037425
0,1	0,13	0,07949
0,2	0,38	0,1427
0,3	0,47	0,2695

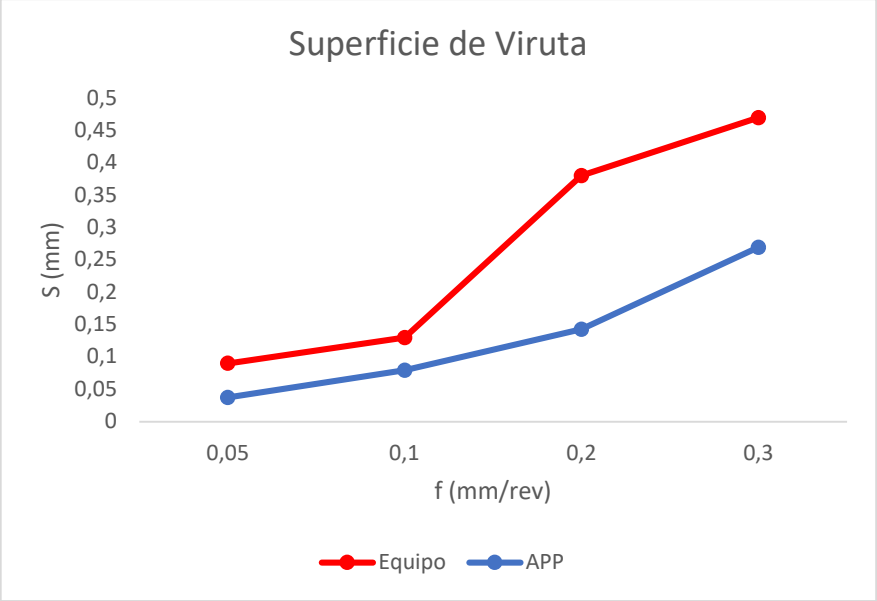


Figura 5.21- Gráfica comparativa de superficie de viruta, UNS A92024.

En la (fig. 5.21), se puede apreciar cómo ambas curvas muestran resultados similares para velocidades de avance bajas. Sin embargo, a medida que las velocidades de avance se incrementan, los resultados obtenidos por el equipo se vuelven superiores a los proporcionados por la APP. Esta disparidad se debe en gran medida a la complejidad que se presenta al medir manualmente la superficie de la viruta, así como a la deformación que experimenta la viruta a velocidades de avance elevadas. A pesar de esta diferencia, es importante destacar que ambas gráficas siguen una tendencia creciente, lo cual es coherente. Esto verifica de nuevo la validez de ambos métodos y su capacidad para proporcionar resultados consistentes, a pesar de los desafíos asociados con las mediciones en condiciones de alto avance.

6. Conclusiones

6.1. Conclusiones generales

Al comienzo de este trabajo de fin de grado, fue propuesta la elaboración de un algoritmo capaz de automatizar algunas tareas en el ámbito de la mejora de la eficiencia en el estudio de virutas obtenidas en procesos de mecanizado. Concretamente, los objetivos propuestos en un inicio fueron los que a continuación se mencionan. Por un lado, el algoritmo desarrollado debe ser capaz de identificar y clasificar imágenes dependiendo de la aleación de la que se trate, siendo estas Ti6Al4V, UNS A92024 y UNS A97075 respectivamente. Por otro lado, una vez clasificada la imagen, el algoritmo debe identificar algunos parámetros característicos de la viruta, siendo estos la altura de valles (h_v) y picos (h_v), espesor (t), ángulo de deslizamiento (φ), superficie (S) y ancho de viruta (b).

En cuanto a estos objetivos propuestos, se puede destacar que se han cumplido en su totalidad, y se han constrastado con otros procedimientos utilizados de forma previa. No obstante, el algoritmo no es perfecto, sino que cuenta con una serie de limitaciones y futuras mejoras, las cuales se detallan más adelante, en el siguiente epígrafe.

Por otro lado, es de importancia mencionar que el objetivo real del desarrollo del presente trabajo de fin de grado es aportar una contribución al Proyecto de Investigación “Sistema experto para la mejora de la integridad superficial en el mecanizado sostenible de aleaciones ligeras (SPAREMETAL)”, llevado a cabo por grupo de investigación TEP933 “Ingeniería de Fabricación” de la Universidad de Málaga en colaboración con otras entidades. Se espera que el algoritmo diseñado sea de utilidad a la hora de facilitar clasificaciones y cálculos, otorgando así un ahorro de tiempo y trabajo.

Por último, hay que mencionar que, durante la elaboración del trabajo, se han adquirido una serie de conocimientos y aprendizajes muy enriquecedores, que van desde la obtención de habilidades técnicas como puede ser el manejo del software Matlab, hasta el aprendizaje en marcos teóricos como la inteligencia artificial, la cual está ganando una importancia exponencial en los últimos años. Sin embargo, la habilidad más importante que se potencia en la elaboración de este tipo de trabajos se trata sin duda de la capacidad para resolver problemas.

6.2. Conclusiones particulares

Para concluir el trabajo, se presentan en forma de ventajas e inconvenientes algunas conclusiones específicas, relacionadas en su mayoría con los resultados obtenidos en la parte técnica.

Particularidades

Para la aleación Ti6Al4V se ha encontrado que la medida de picos y valles se tienen mayores diferencias al aumentar el avance. Esto es debido a que, a mayor avance, la viruta obtenida presenta una geometría en diente de sierra más pronunciada, así como a que el método propuesto infravalora los resultados obtenidos mediante el procedimiento manual, dándose las diferencias más elevadas a altos avances. En cuanto al espesor y al ángulo de deslizamiento, los resultados obtenidos son bastante similares a los de referencia.

Por otro lado, también para la aleación Ti6Al4V se ha observado que a bajos avances difieren bastante las medidas correspondientes al ancho de viruta. Esto se debe a que las irregularidades en el ancho son mayores a bajos avances, lo cual hace más difícil la medición del ancho. En cambio, en lo que a la superficie de viruta respecta, las mayores diferencias se encuentran en la viruta obtenida a altos avances. Esto es debido al elevado grado de deformación que presenta la viruta en este caso, en dirección longitudinal, alejándola de la forma ideal de un rectángulo.

En el caso de las aleaciones de aluminio se han observado resultados bastante similares a los obtenidos por métodos experimentales en su vista longitudinal. Esto se debe a que a avances bajos el recuento de picos y valles no es necesario, midiendo el proceso un ancho medio. Sin embargo, a altos avances la viruta se vuelve segmentada produciéndose una gran diferencia entre picos y valles.

En cuanto a la vista transversal de las aleaciones de aluminio, se vuelven a dar resultados bastante parecidos a los obtenidos para la aleación de titanio, tanto para el ancho de viruta, como para la superficie. La única diferencia radica en que, para el ancho de viruta, los resultados obtenidos por ambos métodos son bastante uniformes, presentando valores similares tanto para bajos como altos avances. En cuanto a la superficie de viruta, se vuelven a apreciar mayores diferencias a altos avances, por el mismo motivo que el mencionado en el caso de la aleación Ti6Al4V.

Ventajas

En cuanto a la parte de reconocimiento y clasificación automática de imágenes, cabe destacar que el algoritmo cuenta con una gran probabilidad de acierto, siendo esta del 81.49%.

Por otro lado, en la parte de procesamiento de imagen, se han obtenido unos resultados bastante similares a los obtenidos mediante métodos no automatizados, lo cual es otro punto favorable.

Los resultados correspondientes a parámetros de la viruta son obtenidos con una gran velocidad, tardando solo algunos segundos. Teniendo en cuenta que hay que promediar, llevaría varios minutos realizar estos cálculos manualmente. En caso de estar realizando un proyecto en el que haya que calcular parámetros de varias imágenes, es bastante notable el ahorro de tiempo que conllevaría usar el algoritmo.

Otro hecho que se ha considerado una ventaja es que, el algoritmo realizado en este trabajo está construido desde cero. Esto quiere decir que tiene un gran margen de mejora y desarrollo para futuras líneas de investigación.

Existe otro punto a favor con respecto al algoritmo realizado, que es su versatilidad. Esto se debe a que es apto para trabajar con imágenes de virutas tanto longitudinales como transversales, tanto en la parte de reconocimiento y clasificación, como en la parte de procesado y obtención de resultados.

Limitaciones

Si bien es cierto que el algoritmo clasificador de imágenes cuenta con una gran precisión, también es de importancia mencionar que tiene una probabilidad de error del 18.51%. Esta probabilidad de error afecta en casi su totalidad en la diferenciación de las dos aleaciones de aluminio, UNS A92024 y UNS A97075, sin apenas afectar a la aleación Ti6Al4V. Esto se debe a que las dos aleaciones de aluminio presentan patrones y formas muy similares entre sí, dificultando así al algoritmo la tarea de identificarlas correctamente. Sin embargo, las imágenes de viruta de Ti6Al4V son diferenciadas casi sin ningún problema, ya que es mucha la diferencia existente con respecto a las demás. La manera más eficaz para solucionar este problema es aumentar el número de imágenes, sobre todo las correspondientes a UNS A92024 y UNS A97075, en la fase de entrenamiento de la CNN.

Otra limitación que se dio, esta vez en la parte de procesado de imagen, es la necesidad de ajustar algunos parámetros antes del inicio del proceso. Esto es debido a que cada viruta es diferente, no existiendo nunca dos virutas iguales. En consecuencia, habrá virutas que presenten los dientes más juntos entre sí con respecto a otras, o valles más o menos pronunciados entre otros factores morfológicos, además de que cada imagen cuenta con un número de aumentos en el microscopio que puede variar con respecto a otras. Todo esto provocó que el algoritmo tuviese dificultades a la hora de detectar picos y valles, lo cual se solucionó con tres parámetros que deben ser ajustados según la imagen a tratar. El primero de estos parámetros está relacionado con la separación horizontal entre dientes; el segundo con la distancia vertical entre valles y base de la viruta; y el tercero con las irregularidades que la viruta tiene en los bordes.

La última limitación encontrada se trata de que, hay algunas imágenes de virutas en las cuales el algoritmo de procesamiento de imagen no es capaz de obtener resultados. Esto ocurre en aquellas virutas en las que no se diferencian claramente los picos y los valles, tal y como se muestra en la (fig. 6.1). Para estos casos, se ha propuesto una alternativa, que no es más que una modificación del código ajustada a este tipo de virutas. Esta modificación permite obtener una aproximación bastante precisa de la altura de valles de la viruta.

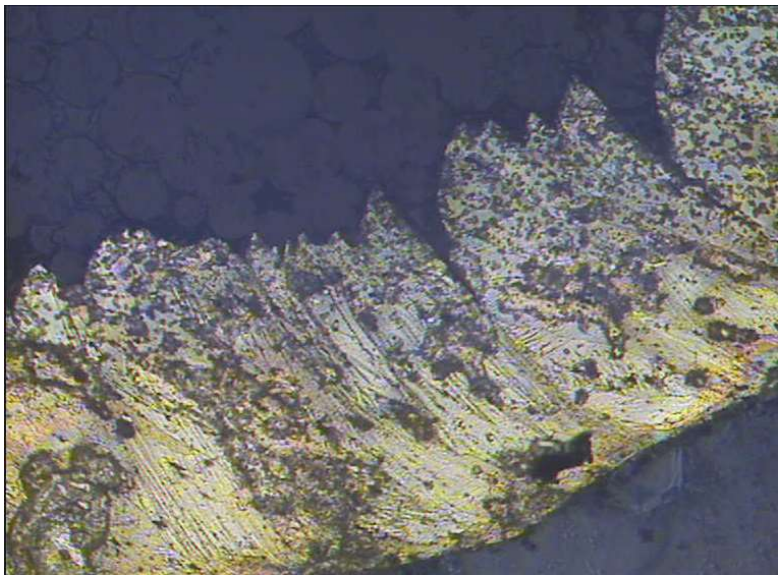


Figura 6.1- Viruta en la cual es difícil detectar una sucesión de picos y valles.

7. Referencias bibliográficas

[1] Y. Sánchez, F. J. Trujillo, C. Bermudo, and L. Sevilla, “Experimental parametric relationships for chip geometry in dry machining of the Ti6Al4V alloy,” *Materials*, vol. 10, no. 7, Jul. 2018, doi: 10.3390/ma11071260.

[2] F. J. Trujillo, L. Sevilla, F. Martín, and C. Bermudo, “Analysis of the Chip Geometry in Dry Machining of Aeronautical Aluminum Alloys,” *Applied Sciences*, vol. 7, no. 2, p. 132, Jan. 2017, doi: 10.3390/app7020132.

[3] D. Heidecker, “El mecanizado en seco y el MQL, cada vez más demandados,” *Metalmecánica*, Apr. 2019.

[4] F. J. Trujillo, “ANÁLISIS PARAMÉTRICO DEL MECANIZADO EN SECO DE LA ALEACIÓN UNS A97075,” Universidad de Málaga , Málaga, 2013.

[5] T. Mikołajczyk, et al. “Predicting tool life in turning operations using neural networks and image processing.,” *Mech Syst Signal Process*, vol. 104, pp. 503–513, 2018.

[6] Y. M. and Ö. T. Arisoy, “Machine Learning Based Predictive Modelling of Machining Induced Microhardness and Grain Size in Ti–6Al–4V Alloy,” *Materials and Manufacturing Processes*, vol. 30, pp. 425–433, 2015.

[7] Y. Sánchez, “Metodología para la Caracterización del Mecanizado de Aleaciones Ligeras de uso Aeronáutico a través del Análisis de la Viruta,” Universidad de Málaga , Málaga, 2017.

[8] C. Löpenhaus, J. Brimmers, T. Bergs, F.Klocke. F. Kühn, *Analysis of the influence of the effective angles on the tool wear in gear hobbing*. 2020.

[9] V. H. Jacobo, R. Schouwenaars, A. Ortiz, J. A. Ortiz, “Revisión de algunos modelos analíticos empleados en el análisis de los procesos de arranque de viruta,” *Unidad de Investigación y Asistencia Técnica en Materiales UDIATEM, Facultad de Ingeniería, Universidad Nacional Autónoma de México*, pp. 1–2, Sep. 2017.

[10] J. Gardner, C. Reich-Weiser, S. Tripathi, A. Vijayaraghavan, D. Dornfeld. M. Ávila, “Strategies for Burr Minimization and Cleanability in Aerospace and Automotive Manufacturing. Laboratory for Manufacturing and Sustainability. ,” *University of California, Berkeley*, Nov. 2006.

[11] Cadem, “CNC turning chips – shape and size, and ROI,” *Cadem*, May 2016.

[12] S. Sun, M. Brandt, and M. S. Dargusch, “Characteristics of cutting forces and chip formation in machining of titanium alloys,” *Int J Mach Tools Manuf*,

vol. 49, no. 7–8, pp. 561–568, Jun. 2009, doi: 10.1016/J.IJMACHTOOLS.2009.02.008.

[13] Apicha, “Steel scrap materials recycling. Aluminum chip waste after machining metal parts on a cnc lathe. Closeup twisted spiral steel shavings. Small roughness sharpness, possible granularity, blurred focus,” Adobe Stock.

[14] S. Dimla., “Sensor signals for tool-wear monitoring in metal cutting operations—a review of methods.,” *Int J Mach Tools Manuf*, pp. 1073–1078, 2000.

[15] O. Kolednik, P. A. F. Martins, A. G. Atkins. P. A. R. Rosa, “The transient beginning to machining and the transition to steady-state cutting,” *International Journal of Machine Tools and Manufacture*, pp. 1904–1915, 2007.

[16] Y. Sánchez, F. J. Trujillo, C. Bermudo, and L. Sevilla, “Experimental Parametric Relationships for Chip Geometry in Dry Machining of the Ti6Al4V Alloy,” *Materials*, vol. 11, no. 7, p. 1260, Jul. 2018, doi: 10.3390/ma11071260.

[17] F. J. Trujillo, L. Sevilla, F. Martín, and C. Bermudo, “Analysis of the chip geometry in dry machining of aeronautical aluminum alloys,” *Applied Sciences (Switzerland)*, vol. 7, no. 2, 2017, doi: 10.3390/app7020132.

[18] P. Kim, *MATLAB Deep Learning*, 1st ed. Apress, 2017.

[19] M. Paluszek, *Practical MATLAB Deep Learning*, 1st ed. Apress, 2020.

[20] Redacción APD, “¿Cuáles son los tipos de algoritmos del Machine Learning?,” pp. 1–1, Apr. 2019.

[21] J. Serrano, “5 aplicaciones de machine learning en la industria,” *Light up your business*, pp. 1–1, Feb. 2021.

[22] J. Dalzochio *et al.*, “Machine learning and reasoning for predictive maintenance in Industry 4.0: Current status and challenges,” *Comput Ind*, vol. 123, p. 103298, Dec. 2020, doi: 10.1016/J.COMPIND.2020.103298.

[23] P. Chaparro, “Introducción al Procesamiento de Imagenes,” *Slide Player*, 2014.

[24] “Improve Production Line Quality Using Machine Learning At The Edge,” *Volansys*, Apr. 2022.

[25] “Image Detection, Recognition, And Classification With Machine Learning,” *Azati Team*, Dec. 2022.

[26] R. Kashyapa, “Machine Vision in Defect Detection Activities,” *Qualitas Tecnologies*, May 2020.

[27] C. Truempi and Dan Yarmoluk, “Vibration Analysis Using Human & Artificial Intelligence,” *Pumps and Systems*, Jun. 2019.

[28] “Classification in Machine Learning: An Introduction,” *Data Camp*, Sep. 2022.

[29] Gestalt Robotics GmbH, “AI-BASED COMPUTER VISION,” Gestalt Robotics GmbH.

[30] S. Carvalho, A. Horovistiz, and J. P. Davim, “Morphological characterization of chip segmentation in Ti-6Al-7Nb machining: A novel method based on digital image processing,” *Measurement (Lond)*, vol. 206, Jan. 2023, doi: 10.1016/j.measurement.2022.112330.

[31] MathWorks, “Matlab, Centro de ayuda.”, https://es.mathworks.com/help/matlab/learn_matlab/help.html, Last Online Access: Sep 2023.

[32] MathWorks, “Matlab Add-On explorer.”, https://es.mathworks.com/products/matlab/add-on-explorer.html, Last Online Access: Sep 2023.

[33] N. Faruqui, “Convolutional Neural Network (CNN) Image Classification in Matlab.” Apr. 03, 2019. https://www.youtube.com/watch?v=j8taOvhHcoU, Last Online Access: Sep 2023.

[34] P. Recuerdo de los Santos, “Cómo interpretar la matriz de confusión: ejemplo práctico,” *Telefonía Tech*, Dec. 2021.

8. Anexos

Anexo I. Código correspondiente al entrenamiento de la Red Neuronal.

```
%% Training
outputFolder = fullfile('General');
rootFolder = fullfile(outputFolder, 'Clases')

categories = {'UNS_A97075','UNS_A92024','Ti6Al4V'};

imds = imageDatastore(fullfile(rootFolder,categories),'LabelSource','foldernames');

tbl = countEachLabel(imds);
minSetCount = min(tbl{:,2});

imds = splitEachLabel(imds, minSetCount, 'randomized');
countEachLabel(imds)

UNS_A97075 = find(imds.Labels == 'UNS_A97075',1);
UNS_A92024 = find(imds.Labels == 'UNS_A92024',1);
Titanio = find(imds.Labels == 'Ti6Al4V',1);

figure
subplot(2,2,1);
imshow(readimage(imds,UNS_A97075));
subplot(2,2,2);
imshow(readimage(imds,UNS_A92024));
subplot(2,2,3);
imshow(readimage(imds,Titanio));

net = resnet50();
figure
plot(net)
title('Architectura de ResNet-50')
set(gca, 'YLim', [150 170]);

net.Layers(1)
```

```
net.Layers(end)

numel(net.Layers(end).ClassNames)
[trainingSet, testSet] = splitEachLabel(imds, 0.3, 'randomize');

imageSize = net.Layers(1).InputSize;

augmentedTrainingSet = augmentedImageDatastore(imageSize, ...
    trainingSet, 'ColorPreprocessing','gray2rgb');

augmentedTestSet = augmentedImageDatastore(imageSize, ...
    testSet, 'ColorPreprocessing','gray2rgb');

w1 = net.Layers(2).Weights;
w1 = mat2gray(w1);

figure
montage(w1)
title('Primera capa convolucional')

featureLayer = 'fc1000';
trainingFeatures = activations(net, ...
    augmentedTrainingSet, featureLayer, 'MiniBatchSize', 32, 'OutputAs','columns');

trainingLabels = trainingSet.Labels;
classifier = fitcecoc(trainingFeatures, trainingLabels, ...
    'Learners', 'linear', 'Coding', 'onevsall', 'ObservationsIn', 'columns');

testFeatures = activations(net, ...
    testSet, featureLayer, 'MiniBatchSize', 32, 'OutputAs','columns');

predictLabels = predict(classifier, testFeatures, 'ObservationsIn', 'columns');

testLabels = testSet.Labels;
confMat = confusionmat(testLabels, predictLabels)
bsxfun(@rdivide, confMat, sum(confMat,2));

mean(diag(confMat));

%% Testeo
newImage = imread(fullfile('Test8.jpg'));

ds = augmentedImageDatastore(imageSize,...
    newImage, 'ColorPreprocessing', 'gray2rgb');

imageFeatures = activations(net, ...
    ds, featureLayer, 'MiniBatchSize', 32, 'OutputAs','columns');

label = predict(classifier, imageFeatures, 'ObservationsIn', 'columns');

sprintf('La imagen cargada pertenece a la clase %s', label)
```

Anexo II. Código correspondiente al procesamiento de imágenes de viruta segmentada en vista longitudinal.

```
%% Binarizado
factor_dilatacion = app.dil1.Value
factor_dilatacion = round(factor_dilatacion);
factor_vertical = app.vert.Value;
factor_horizontal = app.horiz.Value;

im = rgb2gray(im_esc);
im = imbinarize(im);

se = strel('disk',factor_dilatacion);
im = imclose(im,se);
im = imfill(im,'holes');
imagen1 = im;
imshow(imagen1,'Parent',app.UIAxes);
```

```
%% 2) Bordes
%imshow(im)
%hold on
B = bwboundaries(im);
%visboundaries(B)
%hold off
```

```
%% Picos
B = B{1};
Y = B(:,1);
Y = Y*(-1);
X = B(:,2);
[Picos,locs] = findpeaks(Y);
Picos = [abs(Picos),locs];
Picos = abs(B(locs,:));
Picos_y = Picos(:,1);
Picos_x = Picos(:,2);
Picos = [Picos_x Picos_y];
```

```
%% Valles
Y = Y*(-1);
[Valles,locs2] = findpeaks(Y);
Valles = [abs(Valles),locs2];
Valles = abs(B(locs2,:));
Valles_y = Valles(:,1);
Valles_x = Valles(:,2);
Valles = [Valles_x Valles_y];
Y = Y*(-1);

[tam_Picos,residue] = size(Picos_x);
[tam_Valles,residue] = size(Valles_x);
```

```
%% Limpieza vertical de picos
y_max = max(Picos_y);
for i=tam_Picos:-1:1
    if Picos_y(i)>y_max-factor_vertical
    Picos_y(i) = [];
    Picos_x(i) = [];
    end
end

Picos = [Picos_x Picos_y];
[tam_Picos,residue] = size(Picos_x);
```

```
%% Limpieza vertical de valles
y_max = max(Valles_y);
for i=tam_Valles:-1:1
    if Valles_y(i)>y_max-factor_vertical
    Valles_y(i) = [];
    Valles_x(i) = [];
    end
end
Valles = [Valles_x Valles_y];
[tam_Valles,residue] = size(Valles_x);
```

```
%% Limpieza horizontal de picos
% Vectores de entrada
x = Picos_x;
y = Picos_y;
```

```
% Bucle principal
cambio_realizado = true;
while cambio_realizado
    cambio_realizado = false;

    i = 1;
    while i < length(x)
        % Verificar si la diferencia entre elementos consecutivos es menor
        % que el valor elegido
        if abs(x(i+1) - x(i)) < factor_horizontal
            % Comparar los valores de y correspondientes
            if y(i) > y(i+1)
            % Eliminar el primer elemento
                x(i) = [];
                y(i) = [];
            else
                % Eliminar el segundo elemento
                x(i+1) = [];
                y(i+1) = [];
            end
            cambio_realizado = true;
        else
            % Incrementar el índice solo si no se elimina ningún elemento
            i = i + 1;
        end
    end
end

% Mostrar los vectores resultantes
Picos_x = x;
Picos_y = y;
Picos = [Picos_x Picos_y];
```

```
%% Limpieza horizontal de valles
x = Valles_x;  % Vector de coordenadas en x
y = Valles_y;  % Vector de coordenadas en y

% Bucle principal
cambio_realizado = true;
while cambio_realizado
    cambio_realizado = false;

    i = 1;
    while i < length(x)
            % Verificar si la diferencia entre elementos consecutivos es menor que 30
        if abs(x(i+1) - x(i)) < factor_horizontal
            % Comparar los valores de y correspondientes
            if y(i) > y(i+1)
                % Eliminar el segundo elemento
                x(i+1) = [];
                y(i+1) = [];
            else
                % Eliminar el primer elemento
                x(i) = [];
                y(i) = [];
            end
            cambio_realizado = true;
        else
            % Incrementar el índice solo si no se elimina ningún elemento
            i = i + 1;
        end
    end
end

Valles_x = x;
Valles_y = y;
Valles = [Valles_x Valles_y];
```

```
%% Máscara
[tam_Picos,residue] = size(Picos_x);
[tam_Valles,residue] = size(Valles_x);

Size = size(im);
row = Size(1);
col = Size(2);
mask = zeros(row,col);
```

```
%% Líneas

    if Valles_x(end)>Picos_x(end)
        Valles_x(end)=[];
        Valles_y(end)=[];
    end
    Valles = [Valles_x Valles_y];

    [tam_picos,res] = size(Picos_x);
    [tam_valles,res] = size(Valles_x);

    if tam_picos<=tam_valles
        tam = tam_picos;
    else
        tam = tam_valles;
    end



% Crea una imagen vacía
width = col;   % Ancho de la imagen
height = row;  % Alto de la imagen
img = zeros(height, width);
```

```
img = zeros(height, width);

% Crea una imagen vacía
width = col;   % Ancho de la imagen
height = row;  % Alto de la imagen
img = zeros(height, width);

% Número de líneas a trazar
num_lines = tam;

% Coordenadas de los puntos de inicio y fin de las líneas
    x1 = flipud(Picos_x);
    y1 = flipud(Picos_y);
    x2 = flipud(Valles_x);
    y2 = flipud(Valles_y);


% Traza las líneas
for i = 1:num_lines
    % Especifica las coordenadas del punto de inicio y fin
    start_x = x1(i);
    start_y = y1(i);
    end_x = x2(i);
    end_y = y2(i);

    % Calcula la pendiente y la longitud de la línea
    dx = end_x - start_x;
    dy = end_y - start_y;
    line_length = max(abs(dx), abs(dy));

    % Calcula los incrementos en x e y
    x_increment = dx / line_length;
    y_increment = dy / line_length;
```

```
    % Prolonga la línea hasta el borde de la imagen
    x = end_x;
    y = end_y;
    while x >= 1 && x <= width && y >= 1 && y <= height
        img(round(y), round(x)) = 1;  % Establece el pixel en blanco
        x = x + x_increment;
        y = y + y_increment;
    end
end
```

```
%% Overlay
im = imoverlay(im,img,'black');
im = rgb2gray(im);
im = imbinarize(im);
```

```
%% Limpieza de bordes
im = imclearborder(im,4);
```

```
%% Eliminación de elementos pequeños
% Carga la imagen binarizada etiquetada (asegúrate de que la imagen esté cargada en la variable "imagen_etiquetada")
% imagen_etiquetada = imread('ruta_de_la_imagen_etiquetada.png');
image_label = bwlabel(im,4);
imagen_etiquetada = image_label;
% Valor mínimo de píxeles para consideraruna superficie
min_pixeles = 50;

% Etiqueta 0 es el fondo, así que comienza desde 1
num_etiquetas = max(imagen_etiquetada(:));

% Crea una nueva imagen con las mismas dimensiones que la imagen etiquetada
imagen_nueva = zeros(size(imagen_etiquetada));
```

```
% Recorre cada etiqueta y copia las áreas válidas a la nueva imagen
for etiqueta = 1:num_etiquetas
    area_etiqueta = sum(imagen_etiquetada(:) == etiqueta);
    if area_etiqueta >= min_pixeles
        imagen_nueva(imagen_etiquetada == etiqueta) = etiqueta;
    end
end
im=imagen_nueva;
imagen1 = im;
imshow(imagen1,'Parent',app.UIAxes);
```

```
%% Identificación y etiquetado de objetos presentes en la imagen
image_label = bwlabel(imagen1,4); %Identifica los objetos presentes
num_ele = max(max(image_label));
stats = regionprops(image_label,'all');
```

```
%% Relación píxel-mm
if aum=='10x'
    Rel=0.1/71;
elseif aum=='20x'
    Rel=0.1/142;
elseif aum=='40x'
    Rel=0.1/284;
elseif aum=='100x'
    Rel=0.1/710;
else
    Rel=1;
end
%facto = 14/19;
%Rel2 = Rel*facto;
```

```
%% Bounding box y altura de picos
altura = 0;
for i=1:num_ele
    caja = stats(i).BoundingBox;
    rectangle('Position',[caja(1),caja(2),caja(3),caja(4)], ...
        'EdgeColor','r','LineWidth',2,'Parent',app.UIAxes)
    altura = altura + caja(4);
end
altura = altura/num_ele;

altura2 = altura*Rel;
app.picos.Value = altura2;
app.numel.Value = num_ele;
```

```
%% Ángulo de corte
shear=0;
for i=1:num_ele
    shear = shear + stats(i).Orientation;
end
shear = shear/num_ele;
if shear<0
    shear=shear+180;
end

if shear>90
    facto=shear-90;
    shear=90-facto;
end

app.shear.Value = shear;
```

```
%% Altura de valles

% imshow(im)
% hold on
B = bwboundaries(im);
% visboundaries(B)
% hold off

B = B{1};
Y = B(:,1);
Y = Y*(-1);
X = B(:,2);
[Picos,locs] = findpeaks(Y);
Picos = [abs(Picos),locs];
Picos = abs(B(locs,:));
Picos_y = Picos(:,1);
Picos_x = Picos(:,2);
Picos = [Picos_x Picos_y];

Y = Y*(-1);
[Valles,locs2] = findpeaks(Y);
Valles = [abs(Valles),locs2];
Valles = abs(B(locs2,:));
Valles_y = Valles(:,1);
Valles_x = Valles(:,2);
Valles = [Valles_x Valles_y];

[tama_p,res]=size(Picos_y);
[tama_v,res]=size(Valles_y);

Picos_y=Picos_y(1:num_ele);
Valles_y=Valles_y(1:num_ele-1);
```

```
[tam_p,res]=size(Picos_y);
[tam_v,res]=size(Valles_y);

if tam_p<tam_v
    tama=tam_p;
else
    tama=tam_v;
end

dif=0;
for i=1:tama
    dif=dif+(Valles_y(i)-Picos_y(i));
end
dif=dif/tama;

h_valles=altura-dif;


h_valles = h_valles*Rel;
app.valles.Value = h_valles;
```

```
%% Longitud eje mayor
long = 0;
for i=1:num_ele
    long=long+stats(i).MajorAxisLength;
end
long = long/num_ele;

long = long*Rel;
app.espesor.Value = long;
```

Anexo III. Código correspondiente al procesamiento de imágenes de viruta no segmentada en vista longitudinal.

```
%% Testeo
newImage = imagen1;

%% Datos de entrada
factor_dilatacion = app.DilatacinSlider.Value;
factor_dilatacion = round(factor_dilatacion);

%% 1) Binarizado
I = newImage;
deshacer = I;
%[im,rect] = imcrop(I);
im = rgb2gray(I);
im = imbinarize(im);

se = strel('disk',factor_dilatacion);
im = imclose(im,se);
im = imfill(im,'holes');
imshow(im,'Parent',app.UIAxes);

%% 2) Bordes
B = bwboundaries(im);
```

```
%% Identificación de elementos
imagen_binarizada = im;
% Etiquetar los objetos en la imagen
L = bwlabel(imagen_binarizada);
% Obtener las propiedades de cada región etiquetada
stats = regionprops(L,'all');
% Encontrar el área máxima y su correspondiente etiqueta
[area_max, idx_max] = max([stats.Area]);
% Crear una imagen binaria con solo el elemento de área máxima
imagen_max = zeros(size(imagen_binarizada));
imagen_max(L == idx_max) = 1;
stats = regionprops(imagen_max,'all');
% Mostrar la imagen binaria del elemento de área máxima
imshow(imagen_max,'Parent',app.UIAxes);

if aum=='10x'
    Rel=0.1/71;
elseif aum=='20x'
    Rel=0.1/142;
elseif aum=='40x'
    Rel=0.1/284;
elseif aum=='100x'
    Rel=0.1/710;
else
    Rel=1;
end

%facto = 14/19;
%Rel2 = Rel*facto;
```

```
%% Bounding box y altura media
    caja = stats(1).BoundingBox;
    rectangle('Position',[caja(1),caja(2),caja(3),caja(4)], ...
        'EdgeColor','r','LineWidth',2,'Parent',app.UIAxes)


%% Longitudes de eje mayor y eje menor
ancho = [stats.MinorAxisLength]*Rel;
app.shear_2.Value = ancho;
```

Anexo IV. Código correspondiente al procesamiento de imágenes de viruta en vista transversal.

```
%% Testeo
newImage = imagen1;
```

```
%% Datos de entrada
factor_dilatacion = app.DilatacinSlider.Value;
factor_dilatacion = round(factor_dilatacion);
```

```
%% 1) Binarizado
I = newImage;
deshacer = I;
%[im,rect] = imcrop(I);
im = rgb2gray(I);
im = imbinarize(im);

se = strel('disk',factor_dilatacion);
im = imclose(im,se);
im = imfill(im,'holes');
imshow(im,'Parent',app.UIAxes);
```

```
%% 2) Bordes
B = bwboundaries(im);
```

```
%% Identificación de elementos
imagen_binarizada = im;
% Etiquetar los objetos en la imagen
L = bwlabel(imagen_binarizada);
% Obtener las propiedades de cada región etiquetada
stats = regionprops(L,'all');
% Encontrar el área máxima y su correspondiente etiqueta
[area_max, idx_max] = max([stats.Area]);
% Crear una imagen binaria con solo el elemento de área máxima
imagen_max = zeros(size(imagen_binarizada));
imagen_max(L == idx_max) = 1;
stats = regionprops(imagen_max,'all');
% Mostrar la imagen binaria del elemento de área máxima
imshow(imagen_max,'Parent',app.UIAxes);

if aum=='10x'
    Rel=0.1/71;
elseif aum=='20x'
    Rel=0.1/142;
elseif aum=='40x'
    Rel=0.1/284;
elseif aum=='100x'
    Rel=0.1/710;
else
    Rel=1;
end

%facto = 14/19;
%Rel2 = Rel*facto;
```

```
%% Bounding box y altura media
    caja = stats(1).BoundingBox;
    rectangle('Position',[caja(1),caja(2),caja(3),caja(4)], ...
        'EdgeColor','r','LineWidth',2,'Parent',app.UIAxes)
```

```
%% Longitudes de eje mayor y eje menor
area = [stats.Area]*Rel*Rel;
stats.Area
app.sup.Value = area;
ancho = [stats.MajorAxisLength]*Rel;
app.ancho.Value = ancho;
```