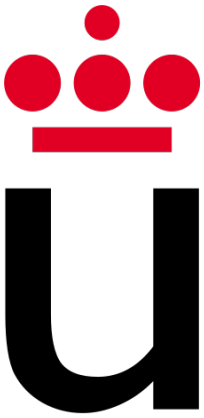

Universidad Rey Juan Carlos

Escuela Técnica Superior de Ingeniería Informática

Grado en Ingeniería de Computadores

Curso académico 2024/2025

Trabajo Fin de Grado

# Algoritmo Open Source para el Conteo Automático de Colonias Bacterianas

Autora: Sarai Méndiz Sal

Tutor: José Ángel Moreno San Segundo

## Agradecimientos

Quiero expresar mi más profundo agradecimiento a todas las personas que me han acompañado y apoyado durante este camino, tanto en la realización de este trabajo de fin de grado como a lo largo de mi carrera universitaria.

A mi familia, que con su apoyo incondicional y paciencia infinita han sido mi pilar fundamental en este proceso. También a quien incluso en los momentos en los que me daba por vencida y me venía abajo, siempre ha estado ahí para apoyarme, brindándome ánimo y comprensión, ayudándome a seguir adelante cuando todo parecía más difícil.

A mis amigos y hermanos, quienes con su compañía y complicidad han hecho más llevaderos los desafíos de estos años. Gracias por los momentos compartidos, por el ánimo en los días más difíciles y por las risas que aligeraron la carga.

De manera especial, quiero reconocer a los profesores de mi instituto, quienes me aconsejaron seguir adelante y no abandonar, incluso en los momentos más difíciles. A pesar de que algunas cosas me costaron más que otras, siempre estuvieron ahí para ayudarme, brindándome apoyo constante. Sus valiosas lecciones sobre la constancia y el esfuerzo me han acompañado a lo largo de este viaje académico, y siempre llevaré esos consejos conmigo.

**Sarai Méndiz Sal**

## Resumen

Este presente trabajo se enfoca en el desarrollo de un algoritmo para el conteo automático de colonias bacterianas en imágenes de placas de *Petri*, una tarea fundamental en microbiología que involucra la identificación y cuantificación de microorganismos cultivados. Este proceso es esencial para aplicaciones en la industria alimentaria, donde se monitorea la contaminación microbiana; en la industria farmacéutica, donde se evalúa la eficacia de antibióticos y otros agentes antimicrobianos; y en la investigación clínica, donde se realizan diagnósticos y estudios epidemiológicos. Tradicionalmente, el conteo de colonias se realiza manualmente, lo cual es un proceso laborioso y propenso a errores humanos debido a la fatiga y la subjetividad del operador. La automatización de este proceso busca no solo reducir el esfuerzo manual y minimizar los errores humanos, sino también mejorar la precisión y eficiencia de los análisis microbiológicos, proporcionando resultados más rápidos y fiables que pueden ser cruciales para la toma de decisiones en contextos críticos.

El algoritmo desarrollado utiliza métodos como la umbralización, la transformada de *Hough* y redes neuronales convolucionales para identificar y contar colonias bacterianas. La implementación se ha realizado en *C++* con *OpenCV*, y se ha complementado con una aplicación web desarrollada en *Python* y *Flask* para proporcionar una interfaz de usuario accesible. Además, se ha planteado que este proyecto sea licenciado bajo la *GNU General Public License v3 (GPLv3),* lo que permitirá su libre distribución y modificación, fomentando la colaboración y mejora continua del software por parte de la comunidad científica y técnica. El código fuente de este proyecto está disponible en el siguiente repositorio de *GitHub*: [https://github.com/SaraiMendiz/Microbial-Colony-Counter].

Las pruebas realizadas con un conjunto de 30 imágenes han mostrado que el algoritmo es preciso y eficiente, aunque la calidad de las imágenes y las variaciones en la iluminación pueden afectar los resultados. Para superar estas limitaciones, se proponen mejoras futuras, como la implementación de técnicas de paralelismo y concurrencia, y el desarrollo de métodos avanzados de corrección de iluminación y eliminación de reflejos.

Este proyecto representa un probable avance significativo en la automatización del conteo de colonias bacterianas, optimizando los flujos de trabajo en laboratorios de microbiología y contribuyendo al progreso de la investigación científica en este campo.

# Índice

## Índice de ilustraciones



## Índice de tablas

## 1. Introducción

El conteo y análisis de colonias bacterianas en placas de *Petri* es una tarea fundamental en microbiología, con aplicaciones cruciales en la industria alimentaria, farmacéutica y en la investigación clínica [1]. Tradicionalmente, este proceso se ha realizado manualmente, lo que implica un esfuerzo considerable y un alto riesgo de errores humanos [2]. La necesidad de métodos más eficientes y precisos ha impulsado el desarrollo de técnicas automatizadas de procesamiento de imágenes y análisis computacional [3].

Las bacterias son organismos unicelulares microscópicos que no pueden ser vistos a simple vista. Por ello, se cultivan sobre un agar o medio de cultivo en placas de *Petri*, donde forman colonias visibles. Estas colonias son agrupaciones de bacterias que se originan de una sola célula o de un pequeño grupo de células, y es a estas colonias a las que se refiere el conteo en microbiología [4]. La identificación y conteo de estas colonias es un paso crítico en el control de calidad y en la investigación científica [5].

En el contexto de la salud pública, la investigación microbiológica es vital para abordar problemas graves como las enfermedades transmitidas por el agua contaminada. Cada año, alrededor de 1,5 millones de personas mueren a causa de diarreas provocadas por el consumo de agua contaminada [6]. La capacidad de detectar y cuantificar rápidamente las bacterias patógenas en el agua puede salvar vidas y prevenir brotes de enfermedades.

Otro problema importante es la resistencia antibiótica, que se ha convertido en una amenaza global. Las bacterias resistentes a los antibióticos causan infecciones que son difíciles de tratar y que pueden llevar a la muerte [7]. La investigación en microbiología es esencial para desarrollar nuevos antibióticos y estrategias de tratamiento, así como para monitorear y controlar la propagación de bacterias resistentes.

En este contexto, el presente trabajo se enfoca en la creación de un algoritmo capaz de contar automáticamente el número de colonias bacterianas presentes en una imagen de una placa de *Petri*. El desarrollo del algoritmo se basa en técnicas avanzadas de procesamiento de imágenes, como la umbralización, la detección de bordes y la segmentación [8]. La implementación se ha realizado utilizando de la librería *OpenCV*, una herramienta muy útil y ampliamente utilizada en el campo de la visión artificial [9]. Este trabajo no solo se enfoca en la precisión del conteo, sino también en la robustez del algoritmo frente a variaciones en la iluminación y la calidad de la imagen [10].

La aplicación práctica de métodos automatizados no solo optimiza los flujos de trabajo en los laboratorios, sino que también contribuye al avance de la investigación científica y al desarrollo de nuevas tecnologías para el control y tratamiento de enfermedades infecciosas. Esto es particularmente relevante en el contexto actual, donde la rápida evolución de las bacterias y la aparición de nuevas cepas patógenas requieren soluciones rápidas y eficientes.

## 2. Estado del Arte

El análisis y conteo de colonias bacterianas en imágenes de placas de *Petri* ha evolucionado significativamente gracias a los avances en el procesamiento de imágenes y el aprendizaje automático. Estos avances han permitido el desarrollo de métodos automatizados que superan las limitaciones de los enfoques manuales, caracterizados por su laboriosidad y susceptibilidad a errores humanos [2][3]. A continuación, se revisan las principales técnicas utilizadas en este campo y sus aplicaciones.

### 2.1. Técnicas de Procesamiento de Imágenes

#### 2.1.1. Umbralización

La umbralización es una técnica fundamental en el procesamiento de imágenes, utilizada para convertir una imagen en escala de grises en una imagen binaria. Esto se logra mediante la selección de un valor de umbral que separa las colonias bacterianas del fondo de la placa de *Petri* [3]. Este proceso es crucial en muchas aplicaciones de análisis de imágenes, ya que permite la segmentación de objetos de interés del fondo, facilitando su análisis y conteo.

El método *Otsu*, por ejemplo, es muy efectivo para seleccionar automáticamente un umbral óptimo que minimiza la varianza intra-clase y maximiza la varianza inter-clase, facilitando así la segmentación precisa en condiciones de iluminación uniforme [11]. Este método evalúa todos los posibles umbrales y selecciona el que minimiza la suma ponderada de varianzas dentro de las clases, proporcionando una segmentación óptima cuando la imagen tiene un histograma bimodal.

Sin embrago, la umbralización simple puede ser insuficiente en condiciones de iluminación no uniforme, lo que ha llevado al desarrollo de métodos adaptativos que ajustan el umbral localmente. La umbralización adaptativa divide la imagen en pequeñas regiones y calcula un umbral para cada una de ellas, adaptándose así a variaciones locales en la iluminación [3]. Este enfoque es especialmente útil en imágenes donde las condiciones de iluminación no son homogéneas, permitiendo una segmentación más precisa en presencia de sombras y reflejos.

Además de los métodos tradicionales de umbralización y los algoritmos adaptativos, también se han desarrollado técnicas avanzadas basadas en el aprendizaje automático. Estos métodos utilizan redes neuronales para aprender características de las imágenes y realizar la segmentación basada en umbral de manera más robusta. Las redes neuronales son capaces de aprender representaciones complejas y pueden ser entrenadas para realizar tareas de segmentación [8].

En aplicaciones prácticas, la elección del método de umbralización depende de varios factores, incluyendo la calidad de la imagen, las condiciones de iluminación, y el nivel de detalle requerido en la segmentación. Los avances de hardware y software han facilitado la implementación de estos métodos, permitiendo su uso en tiempo real en diversas aplicaciones industriales y clínicas [3][12].

#### 2.1.2. Transformada de Hough

La transformada de *Hough* es una técnica eficaz para la detección de formas geométricas en imágenes, como líneas, círculos y elipses. Fue desarrollada por primera vez por *Paul Hough* en 1962 y desde entonces ha sido una herramienta fundamental en el procesamiento de imágenes y visión por computadora [13].

En el contexto del conteo de colonias bacterianas, esta técnica se utiliza para identificar y contar colonias bacterianas esféricas, incluso cuando están parcialmente solapadas. La

transformada de *Hough* es particularmente útil para detectar patrones circulares en las imágenes de placas de *Petri* debido a su robustez frente al ruido y a su capacidad para identificar formas con cierta variación [14]. La técnica opera transformando la imagen original en un espacio de parámetros, donde cada punto en el espacio de parámetros representa una posible forma geométrica en la imagen original. Los picos en el espacio de parámetros indican la presencia de la forma geométrica correspondiente en la imagen.
En la práctica, la transformada de *Hough* para la detección de círculos, conocida como la Transformada de *Hough* Circular, es ampliamente utilizada [15]. Este método implica la detección de bordes en la imagen, generalmente a través de un operador de *Canny*, seguido de la transformación de *Hough* para detectar círculos de diferentes radios. Una ventaja significativa de este enfoque es su capacidad para manejar imágenes con ruido y objetos parcialmente ocultos. Los algoritmos avanzados de transformada de *Hough* pueden detectar múltiples círculos en una sola pasada, optimizando así el proceso de detección [16].
El uso de la transformada de *Hough* en la detección de colonias bacterianas no solo facilita el conteo, sino que también permite analizar características adicionales, como el tamaño y la distribución de las colonias. Esto es crucial en aplicaciones de microbiología donde estas características pueden indicar la presencia de contaminantes o la efectividad de tratamientos antimicrobianos [17]. Además, la transformada de *Hough* se ha integrado con técnicas de preprocesamiento de imágenes, como el filtrado Gaussiano para suavizar la imagen y reducir el ruido, y la umbralización para mejorar la detección de bordes [18].
La robustez de la transformada de *Hough* frente a variaciones en la forma y el tamaño de las colonias bacterianas la convierte en una herramienta invaluable en la automatización del análisis de imágenes microbiológicas. Sin embargo, es importante tener en cuenta que la precisión de la detección puede verse afectada por factores como la calidad de la imagen y la presencia de artefactos [19]. Por lo tanto, la selección adecuada de los parámetros del algoritmo, como los umbrales de detección de bordes y los rangos de radios de los círculos, es crucial para obtener resultados precisos y confiables [20].
En comparación con otras técnicas de detección de formas, la transformada de *Hough* ofrece un equilibrio óptimo entre precisión y eficiencia computacional, especialmente en aplicaciones donde la detección de formas circulares es predominante. Su implementación en hardware moderno y su integración con algoritmos de aprendizaje automático han ampliado aún más su aplicabilidad en diversos campos, incluyendo la microbiología, la medicina y la industria alimentaria [21].

### 2.1.3. Filtros de Suavizado y Detección de Bordes

Los filtros de suavizado, como el filtro *Gaussiano*, y los métodos de detección de bordes, como el operador de *Canny*, son esenciales para el procesamiento de imágenes. Estos métodos ayudan a reducir el ruido y a resaltar las estructuras importantes en las imágenes, mejorando la precisión de la segmentación y facilitando la detección de colonias bacterianas [3][12].
El suavizado *Gaussiano* es particularmente útil para eliminar detalles finos y ruido de la imagen antes de segmentación. Este filtro aplica una convolución de la imagen en una función *gaussiana*, lo que suaviza las variaciones de intensidad. La ventaja del filtro *Gaussiano* es que es isotrópico, es decir, trata todas las direcciones de la misma manera, lo que resulta en una imagen suavizada sin introducir direcciones preferenciales [22]. Este proceso de suavizado es crucial en imágenes de placas de *Petri*, donde las bacterias pueden estar parcialmente oculta por ruido o variaciones de iluminación.
El operador de *Canny* es un método popular para la detección de bordes debido a su precisión y robustez. Fue desarrollado por *John F. Canny* en 1986 y se basa en varios pasos: suavizado de la imagen, cálculo del gradiente de intensidad, supresión no máxima y

trazado de bordes por histéresis. Este enfoque permite detectar bordes finos y continuos en la imagen, lo que es esencial para la segmentación precisa de colonias bacterianas [23]. La detección de bordes mediante el operador de *Canny* es particularmente eficaz para delinear los contornos de las bacterias, facilitando su posterior conteo y análisis [24].
Además de los filtros de suavizado y los métodos de detección de bordes mencionados, existen otros enfoques avanzados que combinan estas técnicas con métodos de aprendizaje automático para mejorar aún más la segmentación de imágenes. Por ejemplo, las técnicas de suavizado adaptativo ajustan el nivel de suavizado en función de las características locales de la imagen, mientras que los algoritmos de detección de bordes basados en redes neuronales pueden aprender a identificar bordes en condiciones de iluminación y ruido variables [25].
La combinación de suavizado *Gaussiano* y detección de bordes con el operador de *Canny* ha demostrado ser efectiva en aplicaciones de procesamiento de imágenes biomédicas. Por ejemplo, en un conteo de células y bacterias en imágenes con alta variabilidad. La capacidad de estos métodos para manejar ruido y variaciones de iluminación es crucial en aplicaciones donde la precisión del conteo es fundamental para el análisis científico y clínico [26].

## 2.2. Aprendizaje Automático y Redes Neuronales

El aprendizaje automático y las redes neuronales convolucionales (CNN) han revolucionado el análisis de imágenes biológicas en los últimos años. Las CNN pueden ser entradas para reconocer patrones complejos y realizar tareas de segmentación y clasificación con alta precisión.

### 2.2.1. Segmentación Semántica

La segmentación semántica que utiliza redes neuronales ha revolucionado el campo del análisis de imágenes biológicas, permitiendo una identificación precisa de colonias bacterianas en imágenes complejas. A diferencia de los métodos tradicionales, que suelen depender de técnicas de procesamiento de imágenes básicas y umbralización, la segmentación semántica emplea modelos avanzados de aprendizaje profundo para diferenciar con mayor precisión las estructuras de interés.
Modelos como *U-Net* se destacan por su eficacia en tareas de segmentación biológica debido a su capacidad para capturar detalles a múltiples escalas. U-Net es una arquitectura de red neuronal convolucional diseñada específicamente para la segmentación de imágenes biomédicas [27]. Su estructura en forma de “U” permite una combinación eficiente de características de diferentes niveles de la imagen, facilitando una segmentación precisa tanto de objetos grandes como pequeños. Esto es particularmente útil en el contexto de conteo de bacterias, donde las colonias pueden estar agrupadas o solapadas [28].
Una de las principales ventajas de *U-Net* es su capacidad para operar en entornos de datos limitados, un escenario común en la biología y la medicina [27]. La red utiliza un proceso de codificación y decodificación, donde la imagen se reduce a una representación de características esenciales y luego se reconstruye con mayor precisión. Este enfoque permite que *U-Net* capture patrones complejos y variaciones en las imágenes, mejorando significativamente la precisión de la segmentación [29].
Estudios han demostrado que *U-Net* puede segmentar células y bacterias en imágenes de microscopía con una precisión notable. Por ejemplo, *Ronneberger et al.* (2015) presentaron la *U-Net* y mostraron su eficacia en la segmentación de imágenes de células en diversas condiciones experimentales [27]. Esta capacidad de segmentar imágenes de alta resolución y complejidad ha reducido considerablemente el tiempo de análisis en

laboratorios de investigación y clínicas, permitiendo un procesamiento de datos más rápido y eficiente.
Además de *U-Net*, otros modelos de segmentación semántica como *Fully Convolutional Networks* (FCN) y *SegNet* también se han utilizado en la segmentación de imágenes biológicas. Estos modelos comparten principios similares en términos de arquitectura, pero *U-Net* ha demostrado ser particularmente robusto en la captura de detalles finos necesarios para la segmentación precisa de bacterias [28][30].
El uso de segmentación semántica con redes neuronales no solo mejora la precisión del conteo de colonias bacterianas, sino que también permite la clasificación de diferentes tipos de bacterias y la detección de características morfológicas específicas. Esto es crucial para aplicaciones clínicas donde la identificación precisa de especies bacterianas puede influir directamente en el diagnóstico y tratamiento de enfermedades [29][30].

### 2.2.2. Clasificación y Reconocimiento de Patrones

Las técnicas de aprendizaje profundo, en particular las redes neuronales convolucionales (CNN), se han convertido en herramientas esenciales para la clasificación y el reconocimiento de patrones en imágenes biológicas [31]. Estas técnicas permiten una identificación precisa y eficiente de bacterias, lo cual es crucial en aplicaciones clínicas y de investigación [32].
Las CNN son especialmente adecuadas para el procesamiento de datos visuales debido a su capacidad para aprender y extraer características jerárquicas de las imágenes [33]. Una CNN típica consiste en varias capas convolucionales seguidas de capas de *pooling* y capas completamente conectadas. Las capas convolucionales aplican filtros para detectar características locales, como bordes y texturas, mientras que las capas más profundas combinan estas características para identificar patrones más complejos [34].
En el ámbito de la microbiología, las CNN se entrenan con grandes conjuntos de datos de imágenes de bacterias, donde cada imagen está etiquetada con la especie bacteriana correspondiente. Durante el entrenamiento, la red aprende a reconocer las características distintivas de cada especie bacteriana, lo que le permite clasificar nuevas imágenes con alta precisión [35]. Esto es especialmente útil en el diagnóstico clínico, donde la identificación rápida y precisa de la especie bacteriana puede influir directamente en el tratamiento y, en última instancia, en los resultados para el paciente.
El aprendizaje profundo también permite la clasificación multiclase, donde un modelo puede identificar y diferenciar entre múltiples especies bacterianas en una sola ejecución [36]. Esta capacidad es crucial en aplicaciones de investigación, donde la diferenciación entre especies puede proporcionar información valiosa sobre las interacciones bacterianas y las respuestas a tratamientos específicos.
Además de la clasificación de especies bacterianas, las técnicas de aprendizaje profundo pueden identificar patrones específicos en las imágenes que pueden no ser evidentes a simple vista [36]. Por ejemplo, las redes neuronales pueden detectar patrones de crecimiento bacteriano, variaciones morfológicas y otros aspectos fenotípicos que son importantes para la investigación y el diagnóstico.
La transferencia de aprendizaje es otra técnica valiosa en este contexto. Utiliza modelos preentrenados en grandes conjuntos de datos, como *ImageNet*, y los adapta a tareas específicas con conjuntos de datos más pequeños. Esta técnica permite aprovechar los patrones y características aprendidos por los modelos en una amplia variedad de imágenes, aplicándolos a la clasificación de imágenes bacterianas [32]. Por ejemplo, un modelo preentrenado puede ajustarse con un conjunto de datos específico de bacterias, reduciendo significativamente el tiempo y los recursos necesarios para entrenar una red desde cero y mejorando la precisión, especialmente cuando se dispone de un conjunto de datos limitado para la tarea específica de clasificación bacteriana [36].

El uso de redes neuronales para la clasificación y el reconocimiento de patrones no solo mejora la precisión diagnóstica, sino que también aumenta la eficiencia del proceso de análisis [31][33]. Esto es especialmente relevante en el contexto de aplicaciones clínicas, donde el tiempo es un factor crítico [35]. Los sistemas automatizados de clasificación pueden procesar grandes volúmenes de datos en poco tiempo, permitiendo a los clínicos tomar decisiones informadas más rápidamente.

## 2.3. Tecnologías en Laboratorios de Microbiología

En los laboratorios de microbiología, se emplean diversas tecnologías para el conteo de colonias bacterianas, cada una con ventajas y desventajas específicas. Estas tecnologías han evolucionado significativamente, mejorando la precisión y eficiencia del análisis microbiológico y permitiendo el procesamiento de grandes volúmenes de muestras. A continuación, se describen las principales tecnologías utilizadas en este campo y sus características distintivas [37][38][39].

### 2.3.1. Contadores automáticos de colonias

Los contadores automáticos de colonias son dispositivos avanzados que emplean cámaras y software de análisis de imágenes para identificar y contar automáticamente las colonias bacterianas en una placa de *Petri*. Estos sistemas han revolucionado el análisis microbiológico, proporcionando una alternativa rápida y precisa al conteo manual, que es laborioso y propenso a errores humanos [37].

El funcionamiento de estos dispositivos se basa en la captura de imágenes de alta resolución de las placas de *Petri* [38]. Una vez capturadas, el software de análisis de imágenes aplica algoritmos avanzados para detectar y contar las colonias bacterianas [39]. Estos algoritmos pueden diferenciar las colonias en función de su tamaño, forma y color, lo que permite una clasificación detallada y precisa. Además, algunos sistemas incorporan técnicas de aprendizaje automático que mejoran continuamente la precisión del conteo a través del entrenamiento con nuevos datos [31].

La alta precisión es una de las principales ventajas de los contadores automáticos de colonias [32]. Estos sistemas son capaces de aplicar criterios consistentes y objetivos en la identificación y conteo de colonias, lo que reduce significativamente la tasa de error en comparación con el conteo manual. Además, la rapidez con la que pueden procesar grandes volúmenes de muestras aumenta la eficiencia del laboratorio y permite una respuesta más rápida en aplicaciones críticas como el diagnóstico clínico y el control de calidad en la industria alimentaria y farmacéutica [33].

Otra ventaja significativa es la reproducibilidad de los resultados [34]. Al eliminar la variabilidad introducida por los diferentes operadores humanos, los contadores automáticos aseguran que los resultados sean reproducibles y consistentes. Esto es crucial para estudios comparativos y seguimiento a largo plazo. Además, estos dispositivos pueden manejar una variedad de tipos de muestras y condiciones de cultivo, desde placas con cultivos de alta densidad hasta aquellas con colonias pequeñas y difíciles de distinguir [35].

A pesar de sus numerosas ventajas, los contadores automáticos de colonias pueden ser costosos, con precios que varían ampliamente según la marca y las características del dispositivo [36]. Los modelos más avanzados, que incluyen capacidades adicionales como la identificación automática de tipos de bacterias y la generación de informes detallados, representan una inversión significativa. Este costo puede ser una barrera para laboratorios con presupuestos limitados. Sin embargo, la inversión en estos dispositivos puede ser justificada por el ahorro de tiempo y la mejora en la precisión y la eficiencia que ofrecen [37].

En términos de aplicaciones, los contadores automáticos de colonias se utilizan ampliamente en diversas industrias y campos de investigación [38]. En la industria alimentaria y farmacéutica, estos sistemas permiten un monitoreo eficiente y preciso de la contaminación microbiológica, asegurando que los productos cumplan con los estándares de seguridad [39]. En el ámbito clínico, la rápida identificación y cuantificación de bacterias en muestras biológicas es esencial para el diagnóstico y tratamiento de infecciones [31]. Los métodos automatizados reducen el tiempo de análisis y mejoran la precisión diagnóstica, lo que es crucial para la gestión efectiva de enfermedades infecciosas. En la investigación científica, las herramientas automatizadas facilitan el análisis de grandes conjuntos de datos, contribuyendo al avance del conocimiento científico en microbiología [32].

### 2.3.2. Microscopios digitales

Los microscopios digitales son herramientas versátiles que permiten la captura de imágenes de alta resolución de muestras bacterianas. A diferencia de los microscopios ópticos tradicionales, los microscopios digitales están equipados con cámaras digitales que facilitan la adquisición de imágenes detalladas, las cuales pueden ser almacenadas y analizadas mediante software especializado. Esta tecnología ha mejorado considerablemente las capacidades de observación y análisis en los laboratorios de microbiología [41][42].

Una de las principales ventajas de los microscopios digitales es su capacidad para capturar y almacenar imágenes electrónicas. Esto no solo facilita la documentación y el seguimiento de las muestras, sino que también permite un análisis detallado y repetible de las mismas. Las imágenes capturadas pueden ser procesadas y analizadas utilizando software avanzado de procesamiento de imágenes, lo que permite la identificación y el conteo preciso de bacterias [43][44]. Estos análisis pueden incluir la medición de tamaños, formas y densidades de las colonias bacterianas, así como la detección de características morfológicas específicas que pueden ser cruciales para la identificación de especies bacterianas [45].

Además, los microscopios digitales permiten la colaboración remota y la telemicroscopía. Las imágenes pueden ser compartidas fácilmente con otros investigadores o profesionales en diferentes ubicaciones, lo que facilita la colaboración en proyectos de investigación y el diagnóstico a distancia [46]. Esta capacidad es especialmente útil en contextos clínicos y educativos, donde la posibilidad de discutir y analizar imágenes en tiempo real puede mejorar la precisión diagnóstica y la calidad de la formación [47].

Aunque los microscopios digitales son más accesibles que los contadores automáticos de colonias, su adquisición y mantenimiento aún representan una inversión significativa. Los costos pueden variar ampliamente dependiendo de las especificaciones y las características del microscopio, como la resolución de la cámara, la capacidad de *zoom*, y las funciones adicionales como la fluorescencia o la capacidad de captura de video [48]. Además, la efectividad del microscopio digital depende en gran medida del software de análisis utilizado. El desarrollo o la adquisición de software específico para el procesamiento y análisis de imágenes es crucial para maximizar el potencial de esta tecnología [49].

El software de procesamiento de imágenes para microscopios digitales puede incluir una variedad de herramientas y algoritmos para la segmentación, el conteo y la clasificación de bacterias. Estos programas a menudo incorporan técnicas avanzadas de aprendizaje automático y redes neuronales para mejorar la precisión del análisis [50]. La integración de estos métodos permite una identificación y cuantificación más rápidas y precisas de las bacterias en las muestras, lo que es esencial para aplicaciones clínicas y de investigación [51].

En términos de aplicaciones, los microscopios digitales son ampliamente utilizados en diversas áreas de la microbiología. En la investigación científica, permiten el estudio detallado de la morfología bacteriana y las interacciones microbianas, facilitando el avance del conocimiento en microbiología y biología celular [52]. En el ámbito clínico, son herramientas valiosas para el diagnóstico de infecciones y la investigación de enfermedades infecciosas [53]. En la industria alimentaria y farmacéutica, se utilizan para monitorear la calidad microbiológica de productos y asegurar el cumplimiento de los estándares de seguridad [54].

### 2.3.3. Sistemas de flujo continuo

Algunos laboratorios utilizan sistemas de citometría de flujo para el análisis y conteo de bacterias en muestras líquidas. Estos sistemas son extremadamente precisos y pueden analizar miles de bacterias por segundo, pero suelen ser muy costosos y requieren un mantenimiento especializado [55].

La citometría de flujo es una técnica avanzada que permite la caracterización y el conteo de partículas, como células y bacterias en suspensión. Funciona haciendo pasar las partículas individuales a través de un haz de luz láser, donde las propiedades ópticas de cada partícula se miden de manera rápida y precisa [56]. Estas propiedades incluyen la dispersión de la luz y la fluorescencia, lo que permite una identificación detallada y la diferenciación de distintos tipos de bacterias en función de su tamaño, forma y composición molecular.

Una de las principales ventajas de los sistemas de flujo continuo es su capacidad para manejar grandes volúmenes de muestras en un período de tiempo muy corto [57]. Esto es particularmente beneficioso en laboratorios que procesan múltiples muestras diariamente, como los laboratorios clínicos, de investigación y de control de calidad en la industria alimentaria y farmacéutica. La alta velocidad y precisión de estos sistemas no solo aumentan la eficiencia operativa, sino que también mejoran la exactitud del análisis, reduciendo el riesgo de errores humanos asociados con los métodos manuales [58].

Además de su velocidad y precisión, los sistemas de flujo continuo también pueden proporcionar datos cuantitativos y cualitativos sobre las bacterias presentes en las muestras. Por ejemplo, pueden diferenciar entre bacterias vivas y muertas, identificar especies bacteriana específicas mediante el uso de tintes fluorescentes y anticuerpos marcados, y evaluar las respuestas bacterianas a diferentes condiciones ambientales o tratamientos antimicrobianos [59].

Sin embargo, la complejidad y el costo de los sistemas de citometría de flujo representan desafíos significativos para su implementación en muchos laboratorios. El equipo en sí es caro y, además requiere personal altamente capacitado para operar y mantener los instrumentos. Los costos de mantenimiento pueden ser sustanciales, ya que los sistemas de flujo continuo necesitan calibración regular y reemplazo de componentes clave para asegurar su funcionamiento óptimo [60].

A pesar de estos desafíos, la inversión en sistemas de citometría de flujo puede ser justificada por los beneficios que aportan en términos de capacidad de análisis y precisión. En la investigación científica, estos sistemas pueden acelerar el descubrimiento de nuevos conocimientos sobre el comportamiento bacteriano y las interacciones microbianas [61].

### 2.3.4. Métodos manuales

A pesar de los avances tecnológicos, muchos laboratorios aún dependen de métodos manuales para el conteo de colonias bacterianas, especialmente en entornos de recursos limitados. Estos métodos son laboriosos y están sujetos a posibles errores humanos, pero no requieren una inversión inicial significativa en equipamiento. Los

métodos manuales generalmente implican el uso de microscopios ópticos tradicionales y técnicas de cultivo en placas de *Petri*, donde los técnicos de laboratorio cuentan las colonias bacterianas a simple vista [62].
El proceso manual puede incluir la preparación de muestras, la inoculación en medios de cultivo, la incubación a temperaturas específicas para permitir el crecimiento bacteriano y, finalmente, el conteo de colonias visibles [63]. Este procedimiento es ampliamente utilizado debido a su simplicidad y a la accesibilidad del equipo requerido. Sin embargo, el tiempo necesario para preparar y analizar las muestras puede ser considerable, lo que limita la cantidad de muestras que se pueden procesar en un período determinado [64].
Los métodos manuales también presentan desafíos relacionados con la precisión y la reproducibilidad. La variabilidad entre operadores puede resultar en discrepancias en los resultados, y la fatiga visual puede afectar la precisión del conteo, especialmente cuando se analizan grandes volúmenes de muestras [65]. Además, el crecimiento de colonias superpuestas o muy pequeñas puede dificultar el conteo exacto, lo que lleva a una posible subestimación o sobreestimación del número real de bacterias presentes [66].
A pesar de estas limitaciones, los métodos manuales continúan siendo una herramienta valiosa en muchos laboratorios, particularmente en aquellos donde los recursos financieros y tecnológicos son limitados. La capacitación adecuada y la estandarización de los procedimientos pueden ayudar a mitigar algunos de los problemas asociados con la variabilidad y los errores humanos [67].
En entornos de investigación, los métodos manuales también permiten un mayor grado de flexibilidad y control sobre las condiciones experimentales. Los investigadores pueden ajustar manualmente los parámetros del experimento, observar directamente los cambios en las colonias bacterianas y realizar anotaciones detalladas que pueden no ser capturadas automáticamente por los sistemas automatizados [68].

## 2.4. Aplicaciones en la Industria y la Investigación

Las soluciones automatizadas para el conteo de colonias bacterianas se aplican en diversas industrias, mejorando la precisión y velocidad del análisis microbiológico. En la industria alimentaria y farmacéutica, aseguran la calidad y seguridad de los productos al monitorear la contaminación bacteriana [69]. En el ámbito clínico, facilitan la identificación rápida y precisa de bacterias, crucial para el diagnóstico y tratamiento de infecciones [70]. Además, en la investigación científica, permiten un estudio detallado del comportamiento bacteriano, impulsando el avance del conocimiento en microbiología [71].

### 2.4.1. Industria Alimentaria y Farmacéutica

En estas industrias, el control de calidad es vital. Los sistemas automatizados de conteo de bacterias permiten un monitoreo eficiente y preciso de la contaminación microbiológica, asegurando que los productos cumplan con los estándares de seguridad [1]. Esto no solo reduce los riesgos de contaminación, sino que también optimiza los procesos de producción, aumentando la productividad y la rentabilidad [2].

### 2.4.2. Diagnóstico Clínico

En el ámbito clínico, la rápida identificación y cuantificación de bacterias en muestras biológicas es esencial para el diagnóstico y tratamiento de infecciones [1]. Los métodos automatizados reducen el tiempo de análisis y mejoran la precisión diagnóstica, lo que es crucial para la gestión efectiva de enfermedades infecciosas [2]. Esto permite a los médicos tomar decisiones informadas más rápidamente, mejorando así los resultados para los pacientes [3].

### 2.4.3. Investigación Científica

Los investigadores utilizan técnicas de procesamiento de imágenes para estudiar el comportamiento bacteriano, las interacciones con otros microorganismos y la respuesta a tratamientos [1]. Las herramientas automatizadas facilitan el análisis de grandes conjuntos de datos y contribuyen al avance del conocimiento científico en microbiología [2]. La automatización del conteo de bacterias permite a los científicos enfocarse en el análisis y la interpretación de datos, acelerando el ritmo de los descubrimientos [3].

## 2.5. Diferenciación del trabajo propuesto

El presente trabajo se diferencia de los estudios anteriores en varios aspectos innovadores y prácticos. En primer lugar, se combina el uso de técnicas tradicionales de procesamiento de imágenes, como la umbralización y la transformada de *Hough*, con avanzados algoritmos de aprendizaje profundo para mejorar la precisión y eficiencia en el conteo bacteriano en imágenes de placas de *Petri*. Esta combinación permite aprovechar lo mejor de ambos mundos: la robustez de las técnicas clásicas y la capacidad de aprendizaje y adaptación de las modernas redes neuronales. Además, este proyecto se centra en la segmentación y clasificación simultánea de bacterias, lo cual representa una mejora significativa en términos de reducción del tiempo de análisis. Mientras que muchos estudios se enfocan en una sola tarea, este trabajo aborda múltiples etapas del proceso de identificación y conteo de bacterias de manera integrada. Esta integración no solo optimiza el flujo de trabajo, sino que también mejora la precisión al reducir los errores acumulados de un sistema fragmentado. Otro aspecto diferenciador es el objetivo de proporcionar una solución accesible y de bajo costo. A diferencia de los costosos sistemas automatizados comerciales que requieren inversiones significativas, el enfoque de este trabajo es desarrollar una herramienta que pueda ser implementada con recursos limitados, sin sacrificar la precisión y eficiencia. Esta accesibilidad es crucial para laboratorios pequeños o con presupuestos restringidos, que no pueden permitirse tecnologías avanzadas pero que necesitan mejorar sus capacidades de análisis microbiológico. La facilidad de uso es otro componente esencial del proyecto. Se ha diseñado la solución pensando en usuarios que quizás no tengan una formación técnica avanzada en procesamiento de imágenes o aprendizaje automático. La interfaz intuitiva y los procesos automatizados permiten que los técnicos de laboratorio puedan utilizar la herramienta con un mínimo de entrenamiento, lo que aumenta la adopción y efectividad del sistema en entornos diversos. Finalmente, el enfoque en la adaptabilidad y escalabilidad del sistema garantiza que pueda ser ajustado y mejorado con el tiempo. A medida que se desarrollan nuevos algoritmos y técnicas, estos pueden ser integrados en la plataforma existente, asegurando que la solución se mantenga a la vanguardia de la tecnología en el análisis microbiológico.

# 3. Objetivos

En esta sección hablaremos de los objetivos que se presentan en este presente trabajo.

## 3.1. Objetivo principal

El objetivo principal de este trabajo de fin de grado es crear una aplicación de acceso abierto *(Open Access)* que permita identificar y contar de manera automática las colonias bacterianas en imágenes de placas de *Petri*. Este desarrollo busca mejorar la precisión y eficiencia en los análisis microbiológicos, reduciendo el tiempo y los recursos necesarios en los laboratorios. Para facilitar el acceso y la colaboración en el desarrollo y mejora continua de esta tecnología, el código de la aplicación se colgará en un repositorio público con una licencia *GPLv3*.

## 3.2. Objetivos específicos

A continuación, definiremos los objetivos específicos que se han planteado para cumplir el anterior:

### 3.2.1. Desarrollo de un algoritmo de procesamiento de imágenes

1. El primer paso para identificar las colonias bacterianas es la umbralización de la imagen. Se implementarán técnicas avanzadas, como el método de *Otsu*, para determinar automáticamente el mejor umbral y separar las bacterias del fondo.

2. Una vez que las imágenes están umbralizadas, se procederá a segmentarlas y etiquetar las regiones de interés. Eso permitirá diferenciar las colonias bacterianas entre sí y el fondo.

3. Las imágenes pueden presentar diferentes niveles de iluminación y contrastes. Se desarrollarán y aplicarán técnicas para corregir las variaciones, asegurando una detección uniforme de las bacterias independientemente de las condiciones de luz.

### 3.2.2. Automatización del proceso de conteo

1. El objetivo es desarrollar un sistema que no requiera intervención manual, desde la corrección de la iluminación hasta el conteo final de las colonias. Esto incluye el procesamiento de la imagen, la segmentación, el etiquetado y el conteo de las bacterias.

2. Se validará el algoritmo en un amplio conjunto de imágenes para asegurar su robustez y fiabilidad en diversas situaciones experimentales. Esto implicará pruebas en imágenes con diferentes condiciones de iluminación, contraste y disposición de las colonias bacterianas.

### 3.2.3. Mejora de la precisión en la detección y conteo de colonias

1. Se implementarán funciones en el algoritmo para cálculos estadísticos para poder detectar y diferenciar las colonias bacterianas en función del tamaño de las regiones para una mejor precisión.

2. Se compararán los resultados obtenidos con métodos manuales y otros métodos automáticos ya existentes. Esto permitirá evaluar la mejora en precisión y eficiencia, destacando las ventajas del algoritmo desarrollado.

#### 3.2.4. Desarrollo de una interfaz amigable para el usuario

1. Se desarrollará una interfaz gráfica de usuario (GUI) que permita a los usuarios cargar imágenes de palcas de *Petri* y obtener el conteo de colonias bacterianas de manera rápida y sencilla. Esta interfaz incluirá opciones para ajustar parámetros del algoritmo en caso de ser necesario.

2. La interfaz proporcionará flexibilidad y control al usuario, permitiendo ajustes en tiempo real y visualización de resultados intermedios.

#### 3.2.5. Evaluación y mejora continua

1. Se evaluará el rendimiento del algoritmo en términos de precisión, velocidad y robustez. Esto incluirá métricas de desempeño específicas para la tarea de conteo de colonias bacterianas.

2. Basándose en la retroalimentación y los resultados obtenidos, se implementarán mejoras continuas en el algoritmo. Esto puede incluir ajustes en los parámetros, optimización del código y adición de las nuevas funcionalidades.

### 3.3. Objetivos de aprendizaje

1. **Conocimiento de técnicas existentes**: Adquirir un conocimiento profundo de las técnicas actuales de procesamiento de imágenes y aprendizaje automático aplicadas al conteo de colonias bacterianas.
2. **Problemática del conteo manual**: Comprender las limitaciones y desafíos del conteo manual de colonias, incluyendo la variabilidad entre operadores y la susceptibilidad a errores humanos.

### 3.4. Objetivos adicionales y lecciones aprendidas

1. **Exploración de métodos alternativos**: Intentar implementar y evaluar métodos alternativos para el conteo de colonias, incluso si algunos no cumplen con las expectativas. Documentar estos intentos y los fracasos asociados para orientar futuros trabajos en el área.
2. **Optimización de recursos**: Evaluar y optimizar el uso de recursos computacionales y de tiempo durante el desarrollo del algoritmo, identificando posibles cuellos de botella y soluciones para mitigarlos.

### 3.5. Objetivos no cumplidos

1. **Clusterización y esfericidad**: Se intentó emplear la *clusterización* para el conteo de colonias bacterianas, pero se encontraron problemas significativos en la determinación de la esfericidad de las bacterias, lo que incrementaba considerablemente el tiempo de ejecución del programa. Estos problemas llevaron a la implementación de un nuevo método que se enfocaba en considerar solo las

regiones dentro de la placa de *Petri*, descartando aquellas que pudieran interferir en el conteo. Además, se permitió al usuario calibrar la umbralización para reducir el "ruido" de la imagen y asegurar que solo las bacterias aparecieran en blanco dentro de la placa de *Petri*. Este enfoque simplificó el análisis al enfocarse en las regiones dentro de la placa y permitió dividir correctamente las colonias agrupadas para un conteo más preciso.

# 4. Descripción informática

## 4.1. Hardware, software y material utilizado

En el desarrollo de este presente trabajo, se han empleado diversos recursos tanto de software como de hardware, los cuales fueron fundamentales para la implementación y pruebas del algoritmo de conteo de colonias bacterianas en placas de *Petri*. A continuación, se detallan los componentes y herramientas utilizados.

### 4.1.1. Fotografías de placas de Petri

Las imágenes de las placas de *Petri*, esenciales para el análisis y desarrollo del algoritmo, fueron obtenidas en los laboratorios de la Universidad Rey Juan Carlos. Estas imágenes proporcionan los datos de entrada necesarios para el procesamiento y análisis del algoritmo.

### 4.1.2. Hardware utilizado

El desarrollo y ejecución del código se realizaron en un portátil *HP Gaming Pavilion 5 4600H*. Este equipo cuenta con las siguientes especificaciones clave:

- **Procesador**: *AMD Ryzen 5 4600H*, que ofrece un rendimiento adecuado para manejar tareas intensivas de procesamiento y permite realizar pruebas eficientes del algoritmo.
- **Memoria RAM**: 16 GB, suficiente para ejecutar múltiples procesos simultáneamente y manejar imágenes de alta resolución sin ralentizaciones significativas.
- **Tarjeta Gráfica**: *NVIDIA GeForce GTX 1650*, utilizada para acelerar ciertas operaciones de procesamiento de imágenes, aprovechando su capacidad de procesamiento paralelo para mejorar la velocidad de análisis.

Estas especificaciones aseguran que el hardware pueda manejar eficientemente el procesamiento de imágenes, ejecutar el algoritmo y ofrecer resultados rápidos y precisos.

### 4.1.3. Lenguajes de programación y librerías

Para la implementación del algoritmo y la aplicación web, se emplearon varios lenguajes y librerías:

- **C++**: el núcleo del algoritmo de procesamiento de imágenes se desarrolló en *C++*, aprovechando su eficiencia y control de bajo nivel sobre los recursos del sistema. La elección de *C++* permite un manejo más eficiente de la memoria y el procesamiento en tiempo real, lo cual es crucial para aplicaciones de análisis de imágenes.
- **OpenCV**: se utilizó la última versión de esta biblioteca de visión por computadora para tareas de procesamiento de imágenes como detección de bordes,

umbralización y etiquetado de regiones. *OpenCV* es conocida por su eficiencia y su amplia gama de funciones de procesamiento de imágenes.

- **Python**: utilizado para el desarrollo del *backend* de la aplicación web, empleando el *framework Flask* debido a su simplicidad y eficacia. Python es conocido por su facilidad de uso y su extensa biblioteca estándar, lo que facilita el desarrollo rápido y eficiente.
- **HTML, CSS, JavaScript y AJAX**: estos lenguajes y tecnologías se emplearon para desarrollar el *frontend* de la aplicación web, proporcionando una interfaz de usuario interactiva y funcional. *HTML* estructura la página web, *CSS* se encarga del estilo visual, y *JavaScript*, junto con *AJAX*, permite la interacción dinámica y la actualización de contenido sin necesidad de recargar la página completa.
- **Java**: Empleado para ciertas funcionalidades específicas en el desarrollo de la interfaz, aprovechando su robustez y compatibilidad multiplataforma.

#### 4.1.4. Herramientas de desarrollo

El desarrollo del proyecto se realizó utilizando diversas herramientas y entornos de desarrollo:

- **Visual Studio Code**: el editor de código principal, elegido por su flexibilidad y la amplia gama de extensiones disponibles que facilitan el desarrollo en múltiples lenguajes.
- **Extensiones de Visual Studio Code**: se utilizaron diversas extensiones para facilitar el desarrollo en *C++*, *Python* y tecnologías web, como *Pylance* para *Python* y *C/C++ Extension Pack* para *C++*.
- **Máquina Virtual**: para la implementación y pruebas de la aplicación web, se creó una máquina virtual configurada con *Python*. Esto garantiza un entorno controlado y reproducible para el desarrollo, permitiendo aislar el entorno de desarrollo del sistema operativo principal y reduciendo el riesgo de conflictos de dependencias.

#### 4.1.5. Instalaciones y descargas

- **Python**: la última versión de *Python* se descargó desde la página oficial, asegurando acceso a las últimas funcionalidades y mejoras de seguridad.
- **Bibliotecas y Dependencias**: todas las bibliotecas necesarias para el desarrollo del proyecto, como *Flask* y *OpenCV*, se instalaron a través de gestores de paquetes como *pip* para *Python* y otros métodos específicos para *C++*. Se siguieron las instrucciones oficiales de instalación para garantizar la correcta configuración del entorno de desarrollo.

### 4.2. Descripción general

El sistema desarrollado para el conteo automático de colonias bacterianas en placas de *Petri* se compone de dos componentes principales: un algoritmo de procesamiento de imágenes implementado en *C++* y una aplicación web que permite la interacción del usuario, desarrollada con tecnologías de *Python*, *JavaScript* y *HTML*. La aplicación web ofrece una interfaz intuitiva donde los usuarios pueden cargar imágenes, ajustar parámetros de umbralización y visualizar los resultados del conteo de colonias.

#### 4.2.1. Componentes del sistema

**Frontend**

- **Tecnologías utilizadas:** *HTML*, *CSS*, *JavaScript*.

- **Descripción:** La interfaz de usuario se desarrolla utilizando *HTML* para la estructura, *CSS* para el estilo visual y *JavaScript* para la interacción dinámica. Los usuarios pueden cargar imágenes de placas de *Petri*, ajustar el umbral de las imágenes en tiempo real y visualizar los resultados del conteo de colonias.

**Backend**

- **Tecnologías utilizadas:** *Python (Flask)*.
- **Descripción:** El *backend* está implementado en *Python* utilizando el *framework Flask*. Este componente maneja las solicitudes del usuario, ejecuta el algoritmo de conteo de colonias y devuelve los resultados al *frontend*. *Flask* se elige por su simplicidad y eficacia para crear aplicaciones web ligeras y rápidas.

**Algoritmo de procesamiento de imágenes**

- **Tecnologías utilizadas:** *C++ (OpenCV)*.
- **Descripción:** El núcleo del sistema es el algoritmo de procesamiento de imágenes, implementado en *C++* con la biblioteca *OpenCV*. Este algoritmo realiza varias etapas de procesamiento, que incluyen preprocesamiento, umbralización, segmentación y conteo de colonias bacterianas. La elección de *C++* y *OpenCV* se debe a su alta eficiencia y capacidades avanzadas de procesamiento de imágenes.

#### 4.2.2. Flujo de datos

El flujo de datos en la aplicación web se realiza en varias etapas, asegurando que el usuario pueda cargar, procesar y visualizar las imágenes de placas de *Petri* de manera eficiente y precisa.

- **Carga de Imágenes:** el usuario carga una imagen de una placa de *Petri* a través de la interfaz web.
- **Ajuste del Círculo:** una vez cargada la imagen, el usuario debe ajustar manualmente el tamaño del círculo rojo que se traza alrededor de la placa de *Petri*.
- **Ajuste del Umbral:** el usuario puede ajustar el umbral de la imagen en tiempo real utilizando un control deslizante.
- **Procesamiento de Imágenes:** la imagen ajustada se envía al backend, donde el algoritmo de procesamiento de imágenes en *C++* realiza el conteo de colonias.
- **Visualización de Resultados:** los resultados del conteo se devuelven al frontend y se muestran al usuario, tanto visualmente (en la imagen) como numéricamente.

Estos pasos se detallarán más adelante y se explicará para qué son necesarios, proporcionando una comprensión completa del flujo de datos y la funcionalidad de la aplicación.

#### 4.2.3. Detalle de la implementación

**Frontend:**

- **HTML:** Estructura básica de la página web.
- **CSS:** Estilos para una presentación visual atractiva.
- **JavaScript:** Lógica para manejar la interacción del usuario, como el ajuste del umbral y la actualización de la imagen en tiempo real.

**Backend:**

- **Python/Flask:** Manejo de rutas, procesamiento de solicitudes y ejecución del algoritmo de procesamiento de imágenes.
- **Endpoints:** Rutas específicas para cargar imágenes, ajustar el umbral, y procesar y devolver resultados.

**Algoritmo de Procesamiento de Imágenes:**

- **Preprocesamiento:** Conversión de la imagen a escala de grises y aplicación de filtros de mediana para reducir el ruido.
- **Umbralización:** Utilización del método de *Otsu* para determinar el mejor umbral automáticamente.
- **Segmentación:** Etiquetado de regiones de interés en la imagen umbralizada.
- **Conteo de Colonias:** Implementación de la función *DividirRegionSiEsGrande* para ajustar dinámicamente los parámetros y contar las colonias bacterianas con alta precisión.

Al completar esta sección, se proporciona una visión clara y detallada de la arquitectura del sistema, sus componentes principales y el flujo de datos desde la carga de imágenes hasta la visualización de resultados.

### 4.3. Detalles del algoritmo

El algoritmo desarrollado para identificar y contar colonias bacterianas en imágenes de placas de *Petri*, incluye varios pasos fundamentales, desde la carga de las imágenes hasta el conteo final de las colonias. Es importante tener en cuenta que, para ejecutar el algoritmo de manera eficiente y sin problemas, se requiere un procesador potente y una cantidad adecuada de memoria RAM. Esto asegura que el procesamiento de imágenes, que puede ser intensivo en recursos, se realice de manera rápida y efectiva. Además, es importante saber que se trabajarán con imágenes en formato *Bitmap* (*.bmp*). A continuación, se explican estos pasos en detalle.

**Importancia del formato de imagen Bitmap (.bmp)**
Trabajar con imágenes en formato *Bitmap* (*.bmp*) es recomendable por varias razones técnicas y prácticas, especialmente en el contexto de procesamiento de imágenes para análisis científico y técnico como el conteo de colonias bacterianas. Aquí se detallan algunas de las principales ventajas:

- **Sin Compresión**: El formato *.bmp* no utiliza compresión, lo que significa que no hay pérdida de calidad de imagen. Esto es crucial en aplicaciones científicas donde la precisión de cada píxel es importante para el análisis [72][73].
- **Fidelidad de Color**: Las imágenes *.bmp* pueden almacenar colores de alta fidelidad, lo cual es importante para análisis que dependen de la exactitud y consistencia del color [72].
- **Facilidad de Acceso**: Debido a su estructura simple y la falta de compresión, las imágenes *.bmp* pueden ser procesadas rápidamente. No se necesita decodificación compleja, lo que facilita y acelera el acceso y la manipulación de los datos de la imagen [72][73].
- **Compatibilidad**: El formato *.bmp* es ampliamente compatible con diferentes programas y sistemas operativos. Esto asegura que las imágenes puedan ser abiertas y manipuladas en casi cualquier entorno de desarrollo sin necesidad de conversión [72].
- **Integridad de Datos**: La falta de compresión significa que la imagen se mantiene intacta y no sufre pérdidas de información, lo que es esencial para tareas que requieren un análisis detallado y preciso [73].
- **Simplicidad**: El formato *.bmp* es sencillo y directo en su estructura, lo que facilita su manejo y procesamiento en aplicaciones de bajo nivel, como algoritmos escritos en *C++* con *OpenCV* [72][73].

#### 4.3.1. Carga de imágenes

El algoritmo trabaja a partir de dos imágenes fundamentales, pero antes de ello, desde la interfaz de usuario, se carga una imagen sin modificaciones de la placa de Petri (Figura 1). Posteriormente, el proceso continúa con una imagen de la placa de *Petri* con un círculo marcado (Figura 2) y una imagen umbralizada que ha sido convertida a escala de grises y escalada a 16 bits (Figura 3). La imagen con el círculo, que es ajustada por el usuario, se utiliza para delimitar la región de interés (ROI) dentro de la placa de *Petri*, mientras que la imagen umbralizada se usa para identificar y etiquetar las colonias bacterianas. A pesar de que la placa ya es redonda en sí, reflejos o mala calidad de la imagen pueden dificultar que el algoritmo detecte adecuadamente el círculo de la placa. Por ello, el ajuste manual del círculo rojo asegura que la región de interés esté correctamente delimitada. El valor de umbralización se puede calcular automáticamente mediante un algoritmo de *Otsu*, o puede ser ajustado manualmente por el usuario para adaptarse a las condiciones específicas de la imagen a través de la interfaz gráfica de usuario (GUI). Estas imágenes son esenciales para establecer los límites dentro de los cuales se realizará el análisis de las colonias, garantizando que solo se consideren las áreas relevantes de la placa de *Petri*.

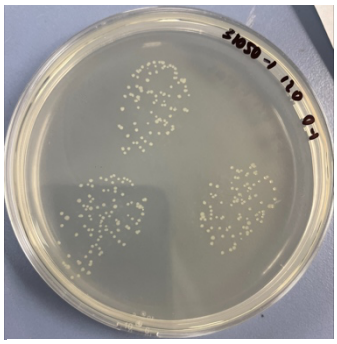

Figura 1

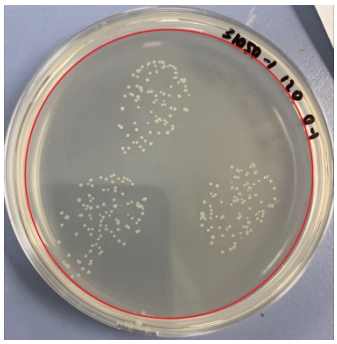

Figura 2

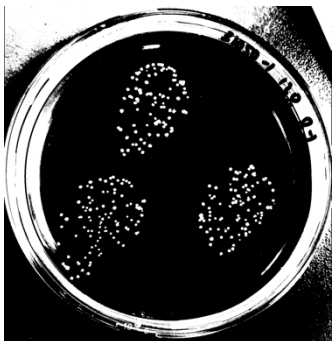

Figura 3

**Figura 1:** imagen original.
**Figura 2:** imagen con círculo trazado.
**Figura 3:** imagen umbralizada.

#### 4.3.2. Detección del círculo más grande

Para detectar el círculo más grande en la imagen de la placa de *Petri*, se utiliza la transformada de *Hough*. Este proceso comienza trabajando con la imagen a escala de grises, lo que simplifica el procesamiento al eliminar la información de color y reducir la imagen a un solo canal de intensidad. A continuación, se aplica un filtro *Gaussiano* para suavizar la imagen y reducir el ruido, mejorando así la detección de bordes y detalles significativos.

La transformada de *Hough* se utiliza para identificar círculos en la imagen, aplicando ciertos parámetros que controlan la sensibilidad y precisión de la detección. Entre estos parámetros, se ajustan los umbrales que determinan qué bordes se consideran para la formación de círculos y la distancia mínima entre los centros de los círculos detectados. El círculo más grande detectado se considera como la placa de *Petri*, y se registran su centro y radio. En caso de el círculo de la placa no pueda distinguirse adecuadamente debido a la iluminación, calidad de la imagen, etc., se detectará el círculo rojo dibujado sobre la imagen anteriormente. Esta información es crucial para los pasos posteriores del

algoritmo, ya que permite filtrar las regiones relevantes dentro de la placa de *Petri*, asegurando que solo se analicen las colonias bacterianas dentro de estos límites. A continuación, se proporciona una representación visual del proceso descrito (Figura 4).

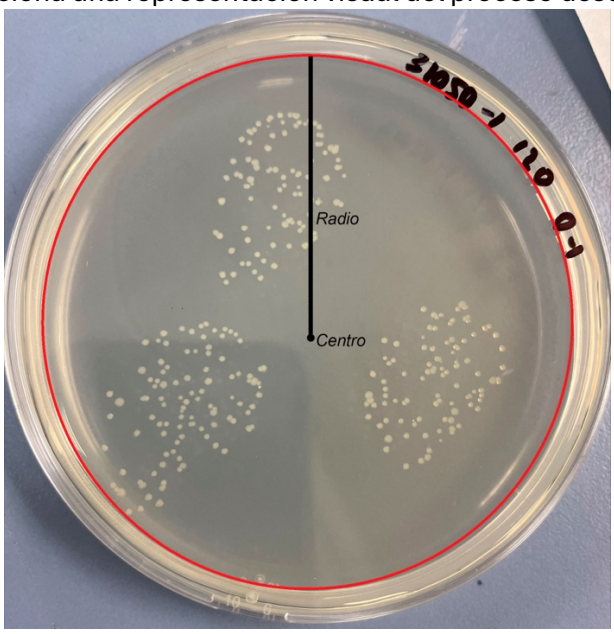


Figura 4

**Figura 4**: transformada de *Hough*.

#### 4.3.3. Etiquetado de regiones

El etiquetado de regiones en la imagen umbralizada es un proceso crucial en el algoritmo de conteo de colonias bacterianas. Este proceso permite identificar y agrupar los píxeles blancos, que representan posibles colonias bacterianas, en regiones conectadas. Estas regiones se etiquetan con números únicos, lo que facilita su diferenciación y conteo individual. A continuación, se detallan los pasos involucrados en este proceso:

El proceso de etiquetado comienza con un recorrido sistemático de la imagen umbralizada. Durante este recorrido, se inspeccionan cada uno de los píxeles de la imagen. Los píxeles blancos (aquellos con un valor umbral superior, que representan las colonias bacterianas) son identificados como posibles partes de colonias.

Una vez que se identifica un píxel blanco, se utiliza un algoritmo de etiquetado de componentes conectados para agrupar los píxeles blancos adyacentes en una misma región. Este algoritmo examina los píxeles vecinos (generalmente en una ventana de 8 vecinos) y agrupa aquellos que están conectados, es decir, que son adyacentes entre sí y también son blancos. Este proceso de agrupación asegura que todos los píxeles conectados que forman parte de una misma colonia bacteriana se agrupen en una región única.

Cada una de estas regiones conectadas recibe un número de etiqueta único. Esta etiqueta es un identificador numérico que distingue una región de otra. Por ejemplo, la primera región detectada podría recibir la etiqueta "1", la segunda "2", y así sucesivamente. Este etiquetado único permite contar y analizar individualmente cada colonia bacteriana detectada en la imagen.

El etiquetado de regiones no solo facilita el conteo de colonias, sino que también es esencial para su posterior análisis. Por ejemplo, una vez que las regiones están

etiquetadas, se pueden calcular diversas propiedades de cada región, como su tamaño (número de píxeles), forma, y posición en la imagen. Esta información es crucial para los pasos posteriores del algoritmo, como el filtrado de regiones y la división de regiones grandes.

Como un caso de ejemplo, en la Figura 5 se presentan las colonias bacterianas que se utilizarán para ilustrar el proceso de etiquetado de regiones. Esta imagen sirve como base para la identificación y separación de las colonias, permitiendo una visualización clara del área de análisis. Luego, en la Figura 6, se muestra una colonia bacteriana etiquetada, donde cada píxel perteneciente a una región identificada ha sido asignado con un número único. Este etiquetado distintivo permite diferenciar y contar individualmente las colonias, facilitando un análisis preciso y automatizado. La imagen ha sido editada específicamente para proporcionar una representación visual clara del proceso de etiquetado, destacando cómo se numeran las regiones de las colonias para su análisis.

Figura 5

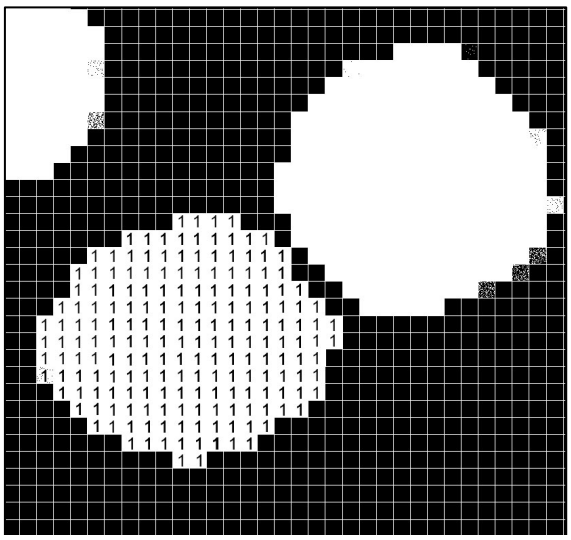

Figura 6

**Figura 5**: colonias bacterianas.
**Figura 6:** colonia bacteriana etiquetada.

**Información guardada en el vector regiones**

El vector regiones es una estructura de datos que almacena la información de las regiones etiquetadas. Cada entrada en este vector es un vector de *cv::Point*, que contiene las coordenadas de los píxeles que pertenecen a una región específica. Esta estructura permite acceder y manipular directamente las regiones identificadas, facilitando operaciones como el cálculo de propiedades de la región, el filtrado y la división de regiones grandes.

**Información guardada en el vector labeled**

El vector *labeled* es una matriz que contiene la imagen umbralizada con las etiquetas de cada región. Cada píxel de la imagen tiene un valor correspondiente a la etiqueta de la región a la que pertenece. Esta matriz es esencial para mantener la referencia espacial y la conectividad de las regiones en la imagen.

**Información guardada en el vector labeledArray**

El *labeledArray* es una estructura que guarda el tamaño de cada región etiquetada en la imagen. Esta información es utilizada para el cálculo de propiedades como el tamaño promedio de las regiones y su desviación estándar, lo que es crucial para el filtrado de regiones y la división de regiones grandes.

### 4.3.4. Descarte de regiones

Para mejorar la precisión del conteo de colonias bacterianas, es fundamental descartar las regiones que no cumplen con ciertos criterios basados en el tamaño promedio y la desviación estándar de las regiones detectadas (Figura 7). Este proceso de filtrado ayuda a eliminar el ruido y las falsas detecciones, asegurando que solo las verdaderas colonias bacterianas sean consideradas en el análisis final.

El proceso comienza con el cálculo del tamaño promedio de las regiones detectadas. Esto se logra sumando el área de todas las regiones y dividiendo el resultado entre el número total de regiones. El tamaño promedio proporciona una referencia sobre qué tan grandes deberían ser, en promedio, las colonias bacterianas en la imagen.

A continuación, se calcula la desviación estándar del tamaño de las regiones. La desviación estándar es una medida de la cantidad de variación o dispersión de los tamaños de las regiones. Si la desviación estándar es pequeña, significa que los tamaños de las regiones son bastante uniformes. Por otro lado, una desviación estándar grande indica que hay una considerable variabilidad en los tamaños de las regiones.

Con el tamaño promedio y la desviación estándar calculados, se definen límites superiores e inferiores para determinar qué regiones deben ser consideradas o descartadas. Estos límites se establecen multiplicando la desviación estándar por una constante (*K*) y añadiendo (o restando) este valor al tamaño promedio. Los límites se definen como:

- **Límite superior:** Tamaño promedio + *K* * Desviación estándar
- **Límite inferior:** Tamaño promedio – *K* * Desviación Estándar

Las regiones cuyo tamaño se encuentra fuera de estos límites son descartadas del análisis. Por ejemplo, las regiones que son significativamente más pequeñas que el límite inferior o mucho más grandes que el límite superior no se consideran verdaderas colonias bacterianas y se eliminan del conteo final. Este paso es esencial para reducir el ruido y aumentar la precisión del análisis, ya que elimina objetos que no son colonias bacterianas, tales como artefactos de la imagen o áreas de fondo mal segmentadas.

El uso de la constante k permite ajustar la sensibilidad del filtrado. Un valor más alto de *K* resultará en límites más amplios, permitiendo que más regiones sean consideradas. Un valor más bajo de k hará que los límites sean más estrictos, descartando más regiones.

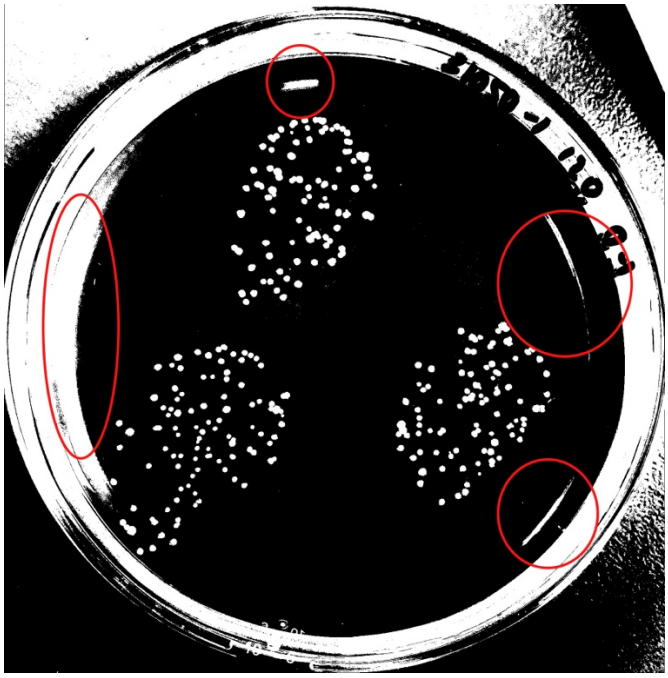

Figura 7

**Figura 7:** regiones que se descartarían.

#### 4.3.5. Filtrado de regiones

Una vez que se han etiquetado y filtrado las regiones basadas en su tamaño y desviación estándar, se lleva a cabo un filtrado adicional para considerar únicamente aquellas regiones que están dentro del círculo de la placa de *Petri*. Este paso es crucial para asegurar que el análisis se centre exclusivamente en las colonias bacterianas que se encuentran dentro de la zona de interés, eliminando cualquier región que se encuentre fuera de la placa de *Petri* y que podría introducir ruido o datos irrelevantes en el conteo final (Figura 8).

Para realizar este filtrado, se verifica si los puntos de cada región etiquetada se encuentran dentro del radio del círculo detectado previamente en el paso de detección del círculo más grande. Se calcula la distancia de cada punto al centro del círculo y se compara con el radio del círculo. Si todos los puntos de una región están dentro de este radio, la región se considera válida y se mantiene para el análisis posterior. Por el contrario, si algún punto de la región se encuentra fuera del radio del círculo, esa región se descarta.

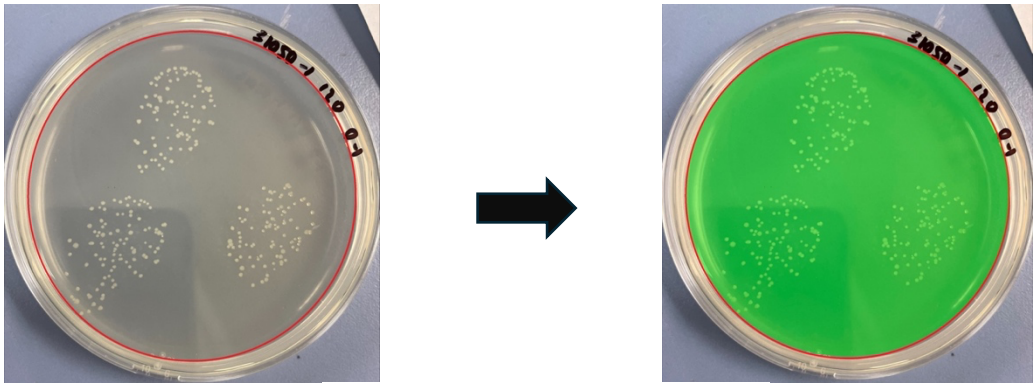

Figura 8

**Figura 8:** zona de interés de la placa de *Petri*.

#### 4.3.6. División de regiones grandes

Para manejar colonias bacterianas que pueden estar agrupadas o superpuestas, se implementa una técnica de división de regiones. Este paso es crucial para asegurar un conteo preciso de las colonias, especialmente en casos donde varias colonias están tan cerca unas de otras que forman una única región grande en la imagen umbralizada (Figura 9).

El proceso comienza identificando aquellas regiones que son significativamente más grandes que el tamaño promedio de las colonias detectadas. Se calcula el tamaño promedio de las regiones y se determina un umbral de tamaño que, si es superado, indica la necesidad de dividir esa región en subregiones más pequeñas (Figura 10). Este umbral se ajusta dinámicamente en función del tamaño promedio y puede ser modificado iterativamente para mejorar la precisión del conteo.

La división de regiones se realiza mediante un algoritmo que segmenta la región grande en varias subregiones, basándose en la distribución de los píxeles dentro de la región. Una técnica común es la de dividir la región a lo largo de líneas de menor densidad de píxeles, lo que típicamente indica la presencia de límites entre colonias agrupadas. Este proceso se repite varias veces, cada vez con un valor ligeramente diferente del parámetro de tamaño, permitiendo obtener una media de los conteos de colonias en las distintas iteraciones.

Figura 9

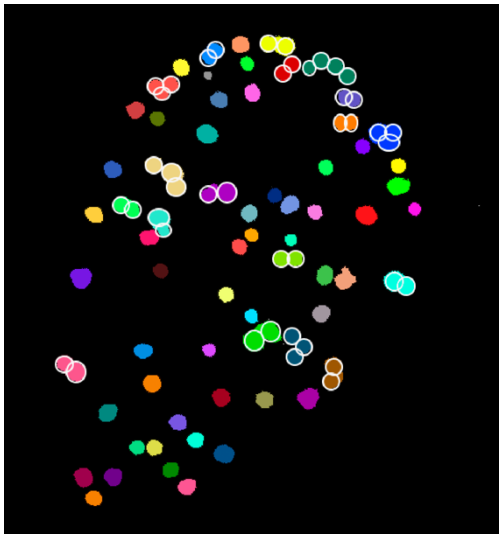

Figura 10

**Figura 9:** regiones de colonias bacterianas distinguidas.
**Figura 10:** regiones de colonias bacterianas divididas en subregiones.

#### 4.3.7. Cálculo del conteo medio de colonias

Para obtener una estimación precisa del número de colonias bacterianas, se realiza un proceso de conteo múltiple, variando ligeramente los parámetros de división de las regiones grandes. Este enfoque se basa en la técnica iterativa que se detalló en el punto anterior (4.3.6), donde se ajusta dinámicamente el tamaño de las regiones a dividir. La razón de este enfoque es compensar las variaciones y posibles errores que podrían ocurrir al utilizar un único conjunto de parámetros fijos, asegurando así una mayor precisión en el conteo final.

El proceso comienza ejecutando el algoritmo de división de regiones varias veces, cada vez con un factor de tamaño ligeramente diferente. Estos factores se incrementan en pequeños pasos, permitiendo al algoritmo explorar diversas configuraciones para la segmentación de las regiones grandes. Durante cada iteración, se realiza un conteo del número de colonias detectadas y se registra este número.

Al finalizar todas las iteraciones, se dispone de una serie de conteos de colonias, correspondientes a cada una de las configuraciones de parámetros probadas. La media de estos conteos se calcula como la estimación final del número de colonias bacterianas presentes en la imagen. Este método de promediado ayuda a mitigar cualquier anomalía que pudiera surgir debido a una configuración específica de los parámetros, proporcionando una estimación más robusta y precisa.

#### 4.3.8. Control del tiempo en el procesamiento de imágenes

El algoritmo implementado no solo se encarga de identificar y contar las colonias bacterianas en las imágenes de las placas de *Petri*, sino que también mide el tiempo necesario para procesar cada imagen. Esta medición del tiempo de procesamiento es fundamental para el control del rendimiento del servidor donde se ejecutará la aplicación en el futuro. Al disponer de estos datos de procesamiento, se podrá monitorear constantemente el rendimiento del sistema. Si se observa que ciertas imágenes requieren más tiempo de procesamiento en comparación con el promedio, esto indicará que el servidor podría estar experimentando una sobrecarga. Al monitorear constantemente estos tiempos, se podrán tomar medidas preventivas para asegurar la eficiencia del sistema y evitar interrupciones en el servicio, manteniendo así un rendimiento óptimo y una experiencia de usuario fluida. Esta funcionalidad permitirá prevenir posibles saturaciones del servidor y garantizar la operatividad continua de la aplicación.

### 4.4. Desarrollo de la Interfaz Gráfica de Usuario (GUI)

El desarrollo de la interfaz gráfica de usuario (GUI) es una parte fundamental de este proyecto, ya que permite a los usuarios interactuar de manera eficiente y efectiva con el algoritmo de conteo de colonias bacterianas. La GUI se ha desarrollado como una aplicación web utilizando tecnologías como *Python* (*Flask*), *JavaScript* y *HTML*, facilitando el acceso y uso del software desde cualquier dispositivo con acceso a internet.

#### 4.4.1. Arquitectura de la aplicación web

La arquitectura de la aplicación web está diseñada para ser modular y escalable, separando claramente la lógica de negocio del algoritmo de procesamiento de imágenes y la presentación e interacción con el usuario.

- **Backend:** el *backend* está implementado en *Python* utilizando el *framework Flask*, que es ligero y flexible. *Flask* se encarga de manejar las solicitudes *HTTP*, gestionar la lógica de la aplicación y comunicar con el algoritmo de procesamiento de imágenes, así como la ejecución de procesos en el servidor. Este diseño modular permite escalar la aplicación y añadir nuevas funcionalidades sin necesidad de reescribir la lógica central.
- **Frontend:** el *frontend* de la aplicación está construido con *HTML*, *CSS* y *JavaScript*, proporcionando una interfaz interactiva y fácil de usar. La biblioteca *jQuery* se utiliza para simplificar las operaciones *DOM* (*Document Object Model*) y las iteracciones *AJAX*, permitiendo actualizar partes de la página sin necesidad de

recargarla por completo. Esto mejora la experiencia del usuario, haciendo que la aplicación sea más rápida y reactiva.

#### 4.4.2. Carga de imágenes y selección de parámetros

La primera interacción del usuario con la aplicación es la carga de la imagen de la placa de *Petri* la cual debe estar en formato *.bmp* (Figura 11). Este requisito garantiza que las imágenes sean de la más alta calidad y sin compresión, lo que es crucial para un análisis preciso. Sin embargo, en un futuro muy cercano, se planea implementar un sistema de conversión de formatos. Este sistema permitirá a los usuarios cargar imágenes en formatos comunes como *png*, *jpg*, y otros, y convertirlas automáticamente a *.bmp* dentro de la aplicación. Esto eliminará la necesidad de que los usuarios utilicen herramientas externas para cambiar el formato de sus imágenes, facilitando y agilizando el proceso de carga y análisis. Una vez que la imagen es cargada, el usuario es redirigido de una página donde puede ajustar los parámetros necesarios para el análisis.

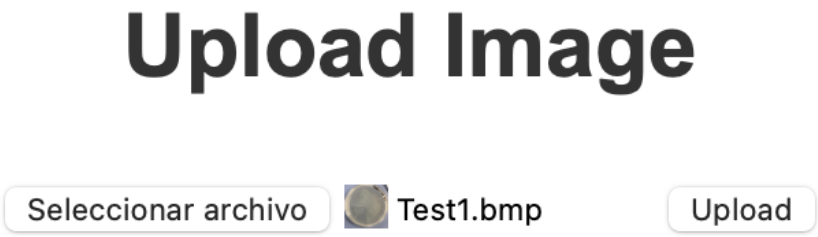


Figura 11

**Figura 11:** vista de la aplicación web para cargar la imagen.

- **Selección de círculo:** el algoritmo de procesamiento de imágenes requiere que el usuario ajuste manualmente el tamaño del círculo rojo que se traza alrededor de la placa de *Petri* (Figura 12). Este ajuste es crucial porque, aunque la placa es inherentemente redonda, factores como reflejos, sombras o la mala calidad de la imagen pueden dificultar la detección precisa del círculo de la placa por parte del algoritmo. Por lo tanto, al permitir que el usuario ajuste el círculo manualmente, se asegura que la región de interés esté correctamente delimitada (Figura 13), evitando así errores en el análisis posterior. El ajuste del círculo por parte del usuario sobre la imagen se logra gracias a *JavaScript*, que proporciona una interfaz interactiva en la aplicación web.

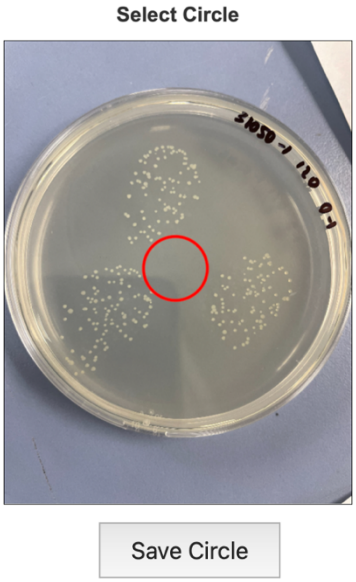


Figura 12

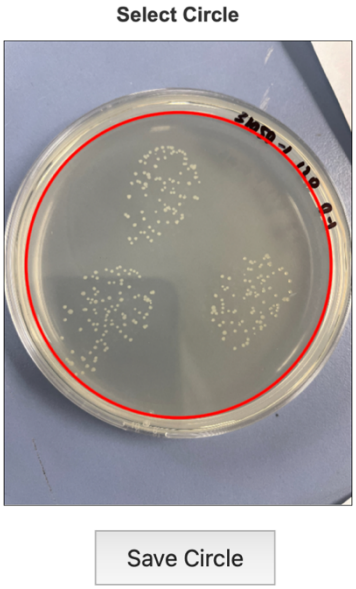


Figura 13

**Figura 12:** imagen cargada junto al círculo ajustable.
**Figura 13:** imagen con el círculo ajustado.

- **Ajuste de umbral:** una de las funcionalidades clave es la posibilidad de ajustar el umbral de la imagen. Esto se realiza mediante un control deslizante que permite al usuario modificar el nivel de umbral en tiempo real (Figura 14). Un *script* en *JavaScript* se encarga de actualizar la imagen umbralizada en el *canvas*, proporcionando una vista previa instantánea de cómo se aplicará el umbral seleccionado. Esta función es esencial para adaptar el análisis a diferentes condiciones de iluminación y contraste en las imágenes.

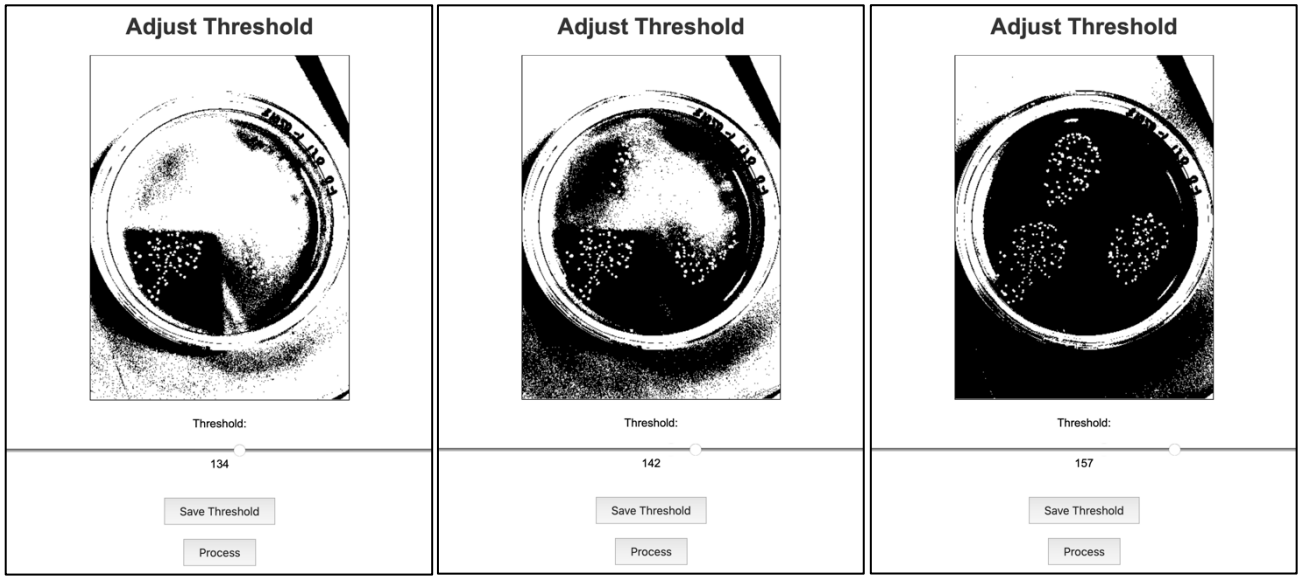


Figura 14

**Figura 14:** interfaz de la aplicación web para el ajuste del umbral.

#### 4.4.3. Procesamiento de visualización de resultados

Una vez que el usuario ha ajustado los parámetros y está satisfecho con la configuración, puede procesar la imagen.

- **Visualización de resultados:** después de procesar la imagen, los resultados se visualizan en una página de resultados. Aquí, el usuario puede ver el número de colonias bacterianas detectadas.

#### 4.4.4. Funcionalidades adicionales

La aplicación web también incluye o incluirá varias funcionalidades adicionales que mejoran la experiencia del usuario y la utilidad del software.

- **Descarga de resultados:** los usuarios podrán descargar la imagen procesada y un informa detallado de los resultados en varios formatos. Esto facilitará el almacenamiento y el análisis posterior de los datos. La capacidad de generar informes automatizados ahorrará tiempo y reduce la posibilidad de errores manuales en la documentación.
- **Almacenamiento de datos:** para laboratorios que utilicen el software de manera regular, la aplicación permitirá almacenar y gestionar las imágenes y resultados obtenidos. Esto creará un historial de análisis que podrá ser útil para estudios longitudinales y comparativos. La gestión de datos incluirá funcionalidades de búsqueda y filtrado, lo que permitirá a los usuarios encontrar y revisar fácilmente análisis anteriores.
- **Interfaz amigable y accesible:** la interfaz está diseñada para ser intuitiva, permitiendo a usuarios con distintos niveles de experiencia técnica utilizar el software sin dificultades. La posibilidad de ajustar parámetros en tiempo real y ver los resultados de manera inmediata mejora significativamente la usabilidad del software. La interfaz también está diseñada para adaptarse a diferentes tamaños de pantalla y dispositivos, lo que asegura una experiencia de usuario consistente en diferentes plataformas.

### 4.5. Despliegue y mantenimiento

El despliegue y mantenimiento de la aplicación web es un proceso crítico para asegurar que el sistema funcione correctamente en un entorno de producción y pueda ser utilizado de manera eficiente por los usuarios finales. Este proceso implica varias etapas, desde la configuración del entorno hasta la implementación de medidas de mantenimiento continuo y seguridad para garantizar la estabilidad y escalabilidad del sistema.

#### 4.5.1. Configuración del entorno de despliegue

La configuración del entorno de despliegue es la primera etapa en el proceso de implementación. Este paso implica preparar la infraestructura y asegurarse de que todos los servicios necesarios estén desplegados y configuradas adecuadamente.

- **Elección del proveedor:** para el despliegue, se ha elegido servicios en nubes públicas o privadas, como *AWS*, *Google Cloud* o *Microsoft Azure* o despliegue de infraestructura On-premise que proporcionarán escalabilidad y flexibilidad. Independientemente del proveedor de servicios, el entorno puede adaptarse a las necesidades específicas del del entorno productivo.
- **Configuración del servidor de la aplicación:** el servidor está basado en un sistema Linux, y se instalan todos los paquetes necesarios, incluidos *Python*, *Flask*, *OpenCV* y cualquier otra biblioteca requerida. También se configura un

servidor web, como *Nginx* o *Apache*, para manejar las solicitudes *HTTP* y dirigirlas a la aplicación *Flask*.

- **Bases de datos:** si la aplicación requiere almacenamiento de datos persistente, se configura una base de datos, como *PostgreSQL* o *MySQL*. Se Asegura que la base de datos esté correctamente instalada y configurada para interactuar con la aplicación.

#### 4.5.2. Despliegue de la aplicación

Una vez que el entorno está configurado, la siguiente etapa es desplegar la aplicación web en el servidor.

- **Control de versiones:** se utiliza un sistema de control de versiones, como *Git*, para gestionar el código de la aplicación. Esto facilita el despliegue de nuevas versiones y el seguimiento de cambios en el código.
- **Integración continua y despliegue continuo (CI/CD):** se implementa un *pipeline* de CI/CD utilizando herramientas como *Jenkins, GitHub Actions* o *GitLab CI/CD*. Esta *pipeline* automatiza el proceso de pruebas y despliegue, asegurando que cada cambio en el código sea probado antes de ser desplegados en el entorno de producción.
- **Despliegue:** el código de la aplicación se despliega en el servidor de producción. Este proceso puede incluir la compilación de archivos, la configuración de variables de entorno y la puesta en marcha de servicios necesarios.

#### 4.5.3. Mantenimiento continuo

El mantenimiento continuo de la aplicación es esencial para asegurar su funcionamiento estable y eficiente a lo largo del tiempo. Esto incluye la monitorización del rendimiento, la gestión de actualizaciones y la resolución de problemas.

- **Monitorización y alertas:** se implementarán herramientas de monitorización como *Prometheus*, *Grafana* o *New Relic* para supervisar el rendimiento de la aplicación y el servidor. Estas herramientas proporcionarán métricas en tiempo real sobre el uso de CPU, memoria, tiempo de respuesta y otros parámetros críticos. Además, se configurarán alertas para notificar al equipo de desarrollo sobre cualquier problema que requiera atención inmediata.
- **Gestión de actualizaciones:** la aplicación y sus dependencias requerirán de actualizaciones periódicas para mejorar la funcionalidad y la seguridad. Se establecerá un calendario de mantenimiento regular para aplicar actualizaciones y parches. Además, se probarán las actualizaciones en un entorno de desarrollo antes de aplicarlas en producción para evitar interrupciones.
- **Gestión de usuarios y seguridad:** la seguridad de la aplicación es una prioridad. Se implementarán medidas como autenticación y autorización de usuarios, cifrado de datos y protección contra ataques comunes como inyecciones *SQL* y ataques de denegación de servicios (DDoS). Además, se revisa y actualiza continuamente la seguridad de la aplicación para proteger contra nuevas amenazas.
- **Copia de seguridad y recuperación:** se implementarán políticas de copias de seguridad regulares para asegurar que los datos estén protegidos contra pérdida accidental o fallos del sistema. Las copias de seguridad se almacenarán de forma segura y se probarán periódicamente para garantizar que se puedan restaurar en caso de necesidad.

#### 4.5.4. Escalabilidad y optimización

A medida que la aplicación crece y atrae a más usuarios, es fundamental asegurar que pueda escalar adecuadamente para manejar el aumento de la carga.

- **Escalabilidad horizontal y vertical:** la infraestructura se diseñará para soportar tanto la escalabilidad horizontal (añadir más servidores) como la vertical (mejorar las capacidades del servidor). Esto asegurará que la aplicación pueda manejar un mayor número de solicitudes y usuarios sin degradación del rendimiento.
- **Optimización del rendimiento:** se realizarán optimizaciones continuas del código y la infraestructura para mejorar el rendimiento. Esto incluirá la optimización de consultas a la base de datos, la reducción del tamaño de las imágenes y otros recursos, y la mejora de la eficiencia del código de procesamiento de imágenes.
- **Balanceo de carga:** se implementarán técnicas de balanceo de carga para distribuir equitativamente las solicitudes entre múltiples servidores, asegurando una respuesta rápida y evitando la sobrecarga de cualquier servidor individual.

### 4.6. Limitaciones

Uno de los principales desafíos al utilizar el algoritmo de procesamiento de imágenes para el conteo de colonias bacterianas es la calidad de las imágenes de entrada. La precisión y efectividad del análisis pueden verse significativamente afectadas por varios factores relacionados con la calidad de la imagen:

1. **Fotografías de mala calidad:** las imágenes borrosas pueden dificultar la identificación precisa de los bordes de las colonias bacterianas. Esto puede llevar a errores en la segmentación y conteo, ya que el algoritmo podría no distinguir claramente entre colonias individuales o confundir áreas de fondo con colonias.
2. **Iluminación incorrecta**: la iluminación desigual o insuficiente es otro problema común. Si una parte de la imagen está más iluminada que otra, el contraste entre las colonias y el fondo puede variar significativamente. Esto afecta la umbralización, ya que el algoritmo puede establecer un umbral incorrecto que no se adapta bien a todas las áreas de la imagen, resultando en regiones mal segmentadas.
3. **Sombras y reflejos**: las sombras pueden crear áreas oscuras en la imagen que el algoritmo podría interpretar incorrectamente como parte del fondo o como colonias adicionales. Los reflejos, por otro lado, pueden crear puntos de alta intensidad que se confunden con colonias bacterianas, alterando el conteo final.
4. **Ruido de la imagen**: el ruido visual, causado por la calidad de la cámara o el entorno de captura, puede introducir falsos positivos en el análisis. Píxeles aleatorios de alta intensidad pueden ser interpretados como colonias bacterianas, mientras que el ruido de baja intensidad puede dificultar la detección de colonias reales.
5. **Resolución de la imagen**: las imágenes de baja resolución pueden no proporcionar suficiente detalle para una segmentación precisa. En contraste, imágenes de muy alta resolución pueden aumentar el tiempo de procesamiento sin necesariamente mejorar la precisión del análisis, a menos que se optimicen adecuadamente los algoritmos de procesamiento de imágenes.

# 5. Resultados experimentales

## 5.1. Metodología de pruebas

En esta sección se describe el procedimiento seguido para evaluar la eficacia del algoritmo desarrollado para el conteo automático de colonias bacterianas en placas de *Petri*. Se utilizó un conjunto de 30 imágenes de placas de *Petri* para realizar las pruebas. A continuación, se detalla el proceso seguido para cada imagen.

### 5.1.1. Preparación de las imágenes

- **Captura de imágenes:** las imágenes de las placas de *Petri* fueron capturadas en los laboratorios de la Universidad Rey Juan Carlos. Cada imagen se guardó en formato *.bmp* para asegurar la máxima calidad sin compresión.
- **Carga y ajuste inicial:** cada imagen ha sido cargada en la interfaz gráfica de la aplicación web. A través de esta interfaz, el usuario ajusta manualmente el círculo rojo alrededor de la placa de *Petri*. Este ajuste es crucial para definir la región de interés (ROI) y asegurar que el análisis se realice únicamente dentro de los límites de la placa.
- **Umbralización:** la umbralización de la imagen se ha realizado automáticamente utilizando un algoritmo de *Otsu*. Sin embargo, en casos donde la calidad de la imagen o la iluminación no han permitido un ajuste óptimo, el usuario ha ajustado el umbral manualmente a través de un control deslizante en la interfaz gráfica.

### 5.1.2. Parámetros del algoritmo

- **Ajuste de *K* en la función *DividirRegionSiEsGrande:*** para el proceso de división de regiones, se utiliza un valor inicial de $K = 1.25$, con un incremento de 0.1 en un bucle que se repite 20 veces. Estos valores han sido seleccionados después de realizar múltiples pruebas con diversas imágenes y determinar que estos parámetros proporcionaban resultados óptimos en la mayoría de los casos.
- **Iteraciones:** el bucle de 20 iteraciones permite ajustar dinámicamente los parámetros de tamaño de las regiones, asegurando una mayor precisión en el conteo de colonias bacterianas.

### 5.1.3. Procedimiento de conteo

- **Conteo automático:** una vez ajustado el círculo y el umbral, el algoritmo procesa la imagen y realiza el conteo automático de las colonias bacterianas. Este conteo se repite para cada iteración del bucle, ajustando los parámetros de tamaño de las regiones.
- **Conteo manual:** paralelamente, se realiza un conteo manual de las colonias en cada imagen para comparar y validar la precisión del algoritmo automático.

### 5.1.4. Análisis de resultados

- **Comparación de resultados:** los resultados obtenidos del conteo automático se comparan con los resultados del conteo manual. Se calcula el margen de error para cada imagen, lo que permite evaluar la precisión y fiabilidad del algoritmo.
- **Cálculo del margen de error:** el margen de error se determina como la diferencia porcentual entre el número de colonias contadas manualmente y el número contado automáticamente por el algoritmo. Este análisis proporciona una medida cuantitativa de la precisión del sistema.

- **Promedio de conteo:** se calcula el promedio de los conteos automáticos obtenidos en las 20 iteraciones para cada imagen. Este promedio ayuda a reducir la variabilidad y a obtener una estimación más robusta del número de colonias bacterianas.

## 5.2. Resultados cuantitativos y cualitativos

### 5.2.1. Resultados cuantitativos

La tabla a continuación resume estos resultados, proporcionando una visión clara de la eficiencia y precisión del algoritmo bajo diferentes condiciones. La precisión se calcula como el porcentaje de colonias detectadas automáticamente que coinciden con las contadas manualmente.

**Tablas de resultados**

| Imagen | Tiempo de procesamiento (ms) | Colonias detectadas | Colonias manuales | Precisión (%) |
|---|---|---|---|---|
| Imagen_1.bmp | 1750 | 205 | 207 | 99,03 |
| Imagen_2.bmp | 1832 | 215 | 218 | 98.62 |
| Imagen_3.bmp | 1615 | 296 | 297 | 99.66 |
| Imagen_4.bmp | 2034 | 301 | 303 | 99.33 |
| Imagen_5.bmp | 1678 | 231 | 234 | 98.72 |
| Imagen_6.bmp | 1827 | 402 | 402 | 100.00 |
| Imagen_7.bmp | 1939 | 157 | 158 | 99.37 |
| Imagen_8.bmp | 2007 | 113 | 116 | 97.41 |
| Imagen_9.bmp | 2004 | 171 | 173 | 98.84 |
| Imagen_10.bmp | 1996 | 158 | 161 | 98.14 |
| Imagen_11.bmp | 1999 | 207 | 207 | 100.00 |
| Imagen_12.bmp | 2021 | 417 | 421 | 99.05 |
| Imagen_13.bmp | 1973 | 331 | 334 | 99.10 |
| Imagen_14.bmp | 1914 | 198 | 198 | 100.00 |
| Imagen_15.bmp | 1854 | 274 | 276 | 99.23 |
| Imagen_16.bmp | 2010 | 309 | 311 | 99.34 |
| Imagen_17.bmp | 1976 | 212 | 212 | 100.00 |
| Imagen_18.bmp | 1728 | 120 | 121 | 99.17 |
| Imagen_19.bmp | 1834 | 187 | 189 | 98.94 |
| Imagen_20.bmp | 2012 | 406 | 409 | 99.27 |
| Imagen_21.bmp | 1978 | 263 | 267 | 98.50 |
| Imagen_22.bmp | 1810 | 204 | 209 | 97.61 |
| Imagen_23.bmp | 2008 | 142 | 144 | 98.61 |
| Imagen_24.bmp | 1711 | 197 | 198 | 99.49 |
| Imagen_25.bmp | 1658 | 96 | 98 | 97.98 |
| Imagen_26.bmp | 1923 | 342 | 345 | 99.13 |
| Imagen_27.bmp | 2124 | 254 | 254 | 100.00 |
| Imagen_28.bmp | 1814 | 111 | 114 | 97.37 |
| Imagen_29.bmp | 1988 | 102 | 103 | 99.03 |
| Imagen_30.bmp | 1723 | 286 | 289 | 98.96 |

Tabla 1: tabla de los resultados obtenidos de cada una de las imágenes.

**Leyenda**

Para facilitar la interpretación de los datos presentados en la tabla, se proporciona la siguiente leyenda:

- **Tiempo de procesamiento (ms):** tiempo en milisegundos que tomó el algoritmo para procesar la imagen
- **Colonias detectadas:** número de colonias bacterianas detectadas automáticamente por el algoritmo.
- **Colonias manuales:** número de colonias contadas manualmente.
- **Precisión (%):** precisión del algoritmo calculada como *(Colonias detectadas / Colonias manuales) * 100*.

### 5.2.2. Resultados cualitativos

Para proporcionar una evaluación detallada del rendimiento del algoritmo, se han calculado la media y la desviación estándar tanto del tiempo de procesamiento como de la precisión. Estos cálculos son fundamentales para entender la consistencia y fiabilidad del algoritmo.

- **Tiempo de procesamiento**
  1. **Media del tiempo de procesamiento:** la media del tiempo de procesamiento es una medida estadística que representa el promedio de los tiempos que el algoritmo tomó para procesar cada imagen.

$$\overline{Xtp} = \frac{1}{30}\sum_{i=1}^{30} Xtp_i = \frac{1}{30}\,(1750 + 1832 + 1615 + \cdots + 1988 + 1723) = 1877.37\ ms$$

  2. **Desviación estándar del tiempo de procesamiento:** la desviación estándar es una medida que indica cuánta variación o dispersión existe respecto a la media. En el contexto del tiempo de procesamiento, una desviación estándar baja indica que los tiempos de procesamiento están muy cerca de la media, mientras que una desviación estándar alta indica una mayor variabilidad.

$$\sigma_{\boldsymbol{tp}} = \sqrt{\frac{1}{30}\,\pi\sum_{i=1}^{30}\,(Xtp_i - 1877.37)^2} = 156.45\ ms$$

- **Precisión**
  1. **Media de la precisión del algoritmo:** la media de la precisión se calcula de manera similar a la media del tiempo de procesamiento, pero en este caso se utilizan los valores de precisión para cada imagen. Esta media nos da una idea del rendimiento general del algoritmo en términos de cuántas colonias detectadas automáticamente coinciden con las contadas manualmente.

$$\overline{Xp} = \frac{1}{30}\sum_{i=1}^{30} Xp_i = \frac{1}{30}\,(99.03 + 98.62 + 99.66 + \cdots + 99.03 + 98.96) = 98.91\%$$

  2. **Desviación estándar de la precisión del algoritmo:** al igual que para el tiempo de procesamiento, la desviación estándar de la precisión indica cuánto varían las precisiones individuales respecto a la media de precisión.

$$\sigma_{\boldsymbol{p}} = \sqrt{\frac{1}{30}\,\pi\sum_{i=1}^{30}\,(Xp_i - 98.91)^2} = 0.67\%$$

### 5.3. Análisis de resultados

En esta sección, se lleva a cabo un análisis exhaustivo de los resultados obtenidos en las pruebas del algoritmo de conteo de colonias bacterianas. La evaluación de los datos recolectados permite determinar la precisión y eficiencia del algoritmo en comparación con los métodos manuales tradicionales.

El análisis se centrará en identificar las fortalezas y debilidades del algoritmo, examinando factores como la consistencia en la detección de colonias, el impacto de las condiciones de las imágenes (como la calidad y la iluminación), y la variabilidad en el tiempo de procesamiento. Además, se discutirá la precisión del algoritmo frente al conteo manual y se destacarán los casos donde el algoritmo ha mostrado un rendimiento notable o donde ha enfrentado dificultades.

Este análisis no solo nos proporcionará una comprensión clara del desempeño actual del algoritmo, sino que también nos permitirá identificar áreas de mejora y ajustes necesarios para optimizar su funcionamiento en futuras implementaciones. Con base en los hallazgos de esta sección, se podrán proponer recomendaciones para mejorar la precisión y eficiencia del algoritmo en el conteo de colonias bacterianas.

#### 5.3.1. Evaluación del rendimiento del algoritmo

Los resultados obtenidos en el análisis del algoritmo de conteo de colonias bacterianas muestran una media de tiempo de procesamiento de 1887.37 milisegundos, lo que indica un rendimiento eficiente en el algoritmo en la mayoría de los casos. La precisión media alcanzada fue de 98.91%, lo que refleja la alta fiabilidad del algoritmo. Además, la desviación estándar del tiempo de procesamiento fue de 156.45 ms, lo que sugiere una variabilidad aceptable en el rendimiento del algoritmo. La precisión, con una desviación estándar de 0.67%, reafirma la consistencia del algoritmo en diferentes pruebas. Estos resultados demuestran que el algoritmo no solo es rápido sino también altamente preciso.

Se puede observar que el algoritmo mantiene una alta consistencia en la detección de colonias bacterianas, incluso bajo diferentes condiciones de calidad e iluminación de las imágenes. Sin embargo, es importante mencionar que las imágenes de baja calidad o con iluminación desigual pueden afectar la precisión del umbral automático, requiriendo ajustes manuales adicionales por parte del usuario. También se debe tomar en cuenta que aquellas imágenes donde hay zonas más iluminadas que otras o reflejos (Figura 15) pueden suponer un gran problema dado que la umbralización no se ajustaría adecuadamente (Figura 16), aun haciéndolo manual, podría llegar a eliminar colonias bacterianas (Figura 17) bajando considerablemente la precisión del algoritmo.

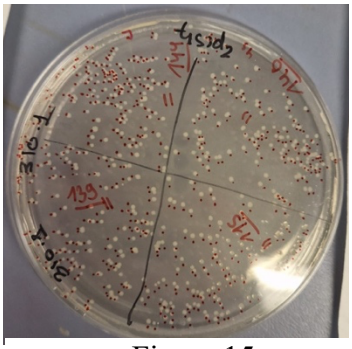

Figura 15

Figura 16

Figura 17

**Figura 15:** imagen con iluminación no equilibrada.
**Figura 16:** imagen con umbralización no uniforme.
**Figura 17:** imagen umbralizada con colonias bacterianas no visibles

En términos de tiempo de procesamiento, la variabilidad detectada se considera aceptable y está influenciada por factores como la resolución de la imagen y la complejidad de las colonias presentes. La eficiencia del algoritmo permite un procesamiento rápido, lo cual es crucial para aplicaciones que requieren resultados en tiempo real.

Finalmente, al comparar los resultados del algoritmo con el conteo manual, se destaca la alta precisión del sistema automatizado. Aunque en algunos casos específicos se observaron ligeras discrepancias, la diferencia promedio entre el conteo manual y el automático fue mínima, lo que valida la efectividad del algoritmo en la mayoría de los escenarios. Sin embargo, es importante mencionar que el algoritmo enfrenta dificultades cuando las bacterias están extremadamente juntas (Figura 18), lo que puede impedir una correcta división de las colonias, y puede afectar la precisión en estos casos.

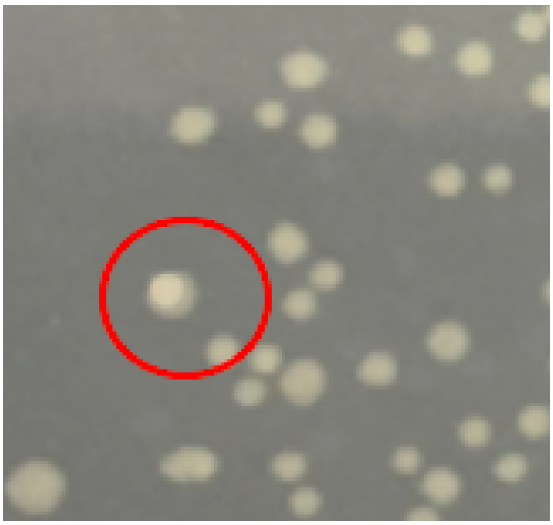

Figura 18

**Figura 18:** dos colonias bacterianas superpuestas.

### 5.3.2. Evaluación del algoritmo

**Fortalezas**

- **Precisión y fiabilidad:** la alta precisión del algoritmo garantiza resultados confiables en la detección de colonias bacterianas, lo que es crucial para aplicaciones en microbiología.
- **Consistencia de resultados:** la baja variabilidad en los resultados demuestra que el algoritmo ofrece un desempeño estable bajo diversas condiciones, minimizando la necesidad de ajustes constantes.
- **Automatización eficiente:** la automatización del conteo de colonias reduce significativamente el tiempo y esfuerzo manual, optimizando el flujo de trabajo en laboratorios y otras aplicaciones.

**Debilidades**

- **Sensibilidad a la calidad de imagen:** imágenes con baja calidad o iluminación desigual pueden afectar negativamente el rendimiento del algoritmo, haciendo necesario un preprocesamiento adicional.
- **Desafíos con colonias superpuestas:** la dificultad del algoritmo para distinguir entre colonias demasiado juntas puede llevar a errores en el conteo, disminuyendo su precisión en situaciones complejas.
- **Problemas con reflejos e iluminación:** reflejos y variaciones en la iluminación de las imágenes pueden causar una umbralización incorrecta, comprometiendo la visibilidad y detección de colonias.

**Oportunidades de mejora**

- **Preprocesamiento de imágenes:** implementar técnicas avanzadas de mejora de imagen para aumentar la calidad antes de la detección de colonias, reduciendo la dependencia de condiciones de iluminación ideales.
- **Optimización del algoritmo:** ajustar y mejorar el algoritmo de detección para manejar mejor las colonias superpuestas y adaptarse a diferentes calidades de imagen.
- **Desarrollo de interfaz de usuario:** crear una interfaz que facilite ajustes manuales mínimos y ofrezca recomendaciones basadas en la calidad de la imagen, mejorando la usabilidad y eficiencia del sistema.

## 6. Conclusión

En este proyecto se ha desarrollado un algoritmo para el conteo automático de colonias bacterianas en placas de Petri licenciado bajo la *GNU General Public License v3 (GPLv3*), utilizando técnicas avanzadas de procesamiento de imágenes y aprendizaje automática. El código fuente de este proyecto está disponible en el siguiente repositorio de *GitHub*: [https://github.com/SaraiMendiz/Microbial-Colony-Counter]. Este trabajo ha demostrado ser un paso significativo hacia la automatización de tareas críticas en microbiología con el fin de lograr avances notables en varios aspectos fundamentales.

En el presente trabajo se ha realizado una revisión exhaustiva de las técnicas actuales de procesamiento de imágenes y aprendizaje automático aplicadas al conteo de colonias bacterianas. Se exploraron métodos como la umbralización, la transformada de *Hough*, filtros de suavizado y detección de bordes, y el uso de redes neuronales convolucionales. Este análisis proporcionó una base sólida para el desarrollo del algoritmo, permitiendo identificar las técnicas más efectivas y adecuadas para nuestro objetivo.

Los objetivos del proyecto han sido detallados, tanto generales como específicos. El objetivo principal fue desarrollar un algoritmo capaz de identificar. Contar automáticamente las colonias bacterianas en imágenes de placas de *Petri*. Los objetivos específicos incluyen el desarrollo de técnicas de procesamiento de imágenes, la automatización del proceso de conteo, la mejora de la precisión en la detección de colonias, la creación de una interfaz amigable para el usuario y la evaluación continua del rendimiento del algoritmo.

Se han puntualizado los recursos de hardware y software utilizados para el desarrollo del proyecto. Esto incluye la descripción del equipo empleado como el portátil *HP Gaming Pavilion 5 4600H* para el desarrollo y pruebas, y las herramientas de software como *C++, OpenCV*, *Python*, *Flask*, *HTML*, *CSS* y *JavaScript*. También se describieron las herramientas de desarrollo empleadas, como *Visual Studio Code* y las extensiones necesarias para facilitar el desarrollo en múltiples lenguajes.

Se presentó una visión general del sistema desarrollado, destacando sus componentes principales: el *frontend*, el *backend* y el algoritmo de procesamiento de imágenes. Se ha explicado el flujo de datos desde la carga de imágenes hasta la visualización de resultados, proporcionando una comprensión clara de la arquitectura del sistema y su funcionamiento.

Los pasos del algoritmo desarrollado han sido descritos desde la carga de imágenes y la detección del círculo más grande, hasta el etiquetado de regiones, el filtrado de estas y su división en caso de que sean más grandes que el promedio. Se ha destacado la importancia del formato de la imagen *Bitmap* (*.bmp*) y se explicó cómo se manejan las imágenes umbralizadas y las regiones etiquetadas.

Se ha detallado el desarrollo de la GUI, que permite a los usuarios interactuar con el sistema de manera eficiente. Se ha explicado la arquitectura de la aplicación web, la carga de imágenes y la selección de parámetros, el ajuste del círculo y del umbral, y la visualización de resultados. También se mencionaron las funcionalidades adicionales y futuras, como la descarga de resultados y el almacenamiento de datos.

Se han abordado los aspectos relacionados con el despliegue y mantenimiento de la aplicación web, incluyendo la configuración del entorno de despliegue, la implementación

de técnicas de integración y despliegue continuos (CI/CD), y las medidas de mantenimiento continuo para asegurar la estabilidad y escalabilidad del sistema.
Finalmente, se ha presentado la metodología de pruebas y los resultados experimentales obtenidos al evaluar la eficacia del algoritmo. Se han analizado los resultados cuantitativos y cualitativos, comparando los conteos automáticos con los manuales y evaluando la precisión y eficiencia del sistema bajo diferentes condiciones.

## 6.1. Trabajo Futuro

### 6.1.1. Pruebas adicionales y validación

Una de las limitaciones significativas ha sido la validación y prueba extensiva del algoritmo con datos reales y bajo diversas condiciones de laboratorio. Aunque se han realizado pruebas con un conjunto limitado de 30 imágenes, es esencial expandir este número y validar el algoritmo con una mayor variedad de muestras y condiciones experimentales. La validación extensiva permitiría ajustar y optimizar el algoritmo para asegurar su robustez y fiabilidad en situaciones reales, como diferentes tipos de agar, variaciones en el tamaño de las colonias, y diferentes especies bacterianas.

La inclusión de escenarios de prueba que simulen condiciones adversas, como imágenes con bajo contraste, ruido elevado y variaciones extremas en la iluminación, proporcionaría una evaluación más completa del desempeño del algoritmo y ayudaría a identificar áreas específicas para mejoras adicionales.

### 6.1.2. Clusterización y esfericidad

Se intentó emplear la clusterización para el conteo de colonias bacterianas, pero se encontraron problemas significativos en la determinación de la esfericidad de las bacterias, lo que incrementaba considerablemente el tiempo de ejecución del programa. Estos problemas llevaron a la implementación de un nuevo método que se enfocaba en considerar solo las regiones dentro de la placa de *Petri*, descartando aquellas que pudieran interferir en el conteo. Además, se ha permitido al usuario calibrar la umbralización para reducir el "ruido" de la imagen y asegurar que solo las bacterias aparecieran en blanco dentro de la placa de *Petri*. Este enfoque ha simplificado el análisis al enfocarse en las regiones dentro de la placa y permitió dividir correctamente las colonias agrupadas para un conteo más preciso.

### 6.1.3. Implementación de paralelismo y concurrencia

En la búsqueda continua de mejorar la eficiencia y el rendimiento del algoritmo de conteo de colonias bacterianas, una de las posibles direcciones futuras es la implementación de técnicas de paralelismo y concurrencia. Estas técnicas permitirán optimizar el procesamiento de imágenes, especialmente cuando se manejen grandes volúmenes de datos o se necesite un tiempo de respuesta más rápido.

El paralelismo implica dividir una tarea grande en varias subtareas que pueden ejecutarse simultáneamente en múltiples núcleos de procesador. Para el algoritmo de conteo de colonias, esto podría significar dividir la imagen en secciona más pequeñas y procesar cada sección en paralelo. De esta forma, el tiempo total de procesamiento se reduce significativamente, ya que las operaciones de detección, etiquetado y filtrado de regiones se realizarían simultáneamente.

Por otro lado, la concurrencia permite que varias tareas se ejecuten de manera intercalada, mejorando la eficiencia del uso del procesador y reduciendo el tiempo de espera. En el contexto del servidor de la aplicación web, implementar concurrencia podría permitir que múltiples solicitudes de procesamiento de imágenes sean atendidas de manera más eficiente, gestionando mejor los recursos y reduciendo los tiempos de respuesta

La adopción de estas técnicas requerirá una revisión y reestructuración del código actual para identificar las partes del algoritmo que pueden ser paralelizadas de manera efectiva. Además, se necesitará la implementación de mecanismo de sincronización para asegurar que los datos compartidos entre diferentes hilos de ejecución sean manejados correctamente, evitando problemas de inconsistencia.

A medida que el sistema escala y la demanda de procesamiento aumenta, la implementación de paralelismo y concurrencia se convertirá en una herramienta esencial para mantener y mejorar el rendimiento del sistema. Esto no solo mejorará la eficiencia, haciendo que la aplicación sea más robusta y capaz de manejar mayores cargas de trabajo.

### 6.1.4. Corrección avanzada de iluminación y reflejos

La calidad de las imágenes es crucial para el correcto funcionamiento del algoritmo de conteo de colonias bacterianas. Una de las principales mejoras futuras sería el desarrollo de técnicas avanzadas de corrección de iluminación y eliminación de reflejos. Estas mejoras permitirían que el algoritmo funcione de manera más efectiva bajo diversas condiciones de iluminación y minimicen los errores causados por reflejos en las placas de *Petri*.

La corrección avanzada de iluminación podría incluir la implementación de algoritmos que normalicen la iluminación en la imagen, eliminando sombras y ajustando las variaciones de luz para crear un fondo más uniforme. Esto ayudaría a mejorar la precisión de la umbralización y la segmentación de colonias bacterianas.

La eliminación de reflejos podría lograrse mediante técnicas de filtrado y procesamiento de imágenes que identifiquen y mitiguen los reflejos sin afectar las características importantes de las colonias bacterianas. Estas mejoras son esenciales para asegurar la robustez y fiabilidad del algoritmo en diversas condiciones de captura de imágenes.

### 6.1.5. Redes neuronales

Aunque el algoritmo actual ofrece una alta precisión en el conteo de colonias bacterianas, existe una limitación significativa en casos donde las colonias están muy juntas o solapadas. En estos casos, la determinación del tamaño de la región puede no ser suficiente para un conteo preciso. Habría sido deseable emplear redes neuronales para mejorar la identificación y separación de colonias en estas situaciones complejas.

Las redes neuronales convolucionales (CNN) son especialmente adecuadas para el procesamiento de imágenes debido a su capacidad para aprender características jerárquicas y complejas de las imágenes. En el contexto de la identificación y conteo de colonias bacterianas, las CNN podrían ser entrenadas para identificar colonias individuales, incluso cuando están agrupadas o solapadas.

El proceso de integración de redes neuronales en el algoritmo podría incluir los siguientes pasos:

1. **Recolección y anotación de datos:** primero, sería necesario recolectar un conjunto grande y variado de imágenes de placas de *Petri* con colonias bacterianas, incluyendo casos de colonias agrupadas y solapadas. Estas imágenes deberían ser anotadas manualmente para crear un conjunto de datos de entrenamiento que incluya las ubicaciones exactas de cada colonia bacteriana.
2. **Procesamiento de imágenes:** las imágenes recolectadas serían procesadas para mejorar la calidad de los datos de entrada. Esto podría incluir la normalización de la iluminación, el ajuste de contraste, y la aplicación de filtros para reducir el ruido.
3. **Diseño y entrenamiento de la red neuronal:** se diseñaría una CNN adecuada para la tarea de segmentación y clasificación de colonias bacterianas. Arquitecturas populares como *U-Net* o *Mask R-CNN* podrían ser empleadas debido a su éxito en tareas de segmentación de imágenes biomédicas [27]. La red sería entrenada utilizando el conjunto de datos anotado, optimizando los parámetros para maximizar la precisión de la detección y separación de colonias.
4. **Validación y evaluación:** una vez entrenada, la red sería validad y evaluada utilizando un conjunto de datos independiente para asegurarse de que el modelo generaliza bien las nuevas imágenes. Las métricas de evaluación incluirían precisión, sensibilidad, especificidad y el *F1-score*.
5. **Integración con el algoritmo existente:** la red neuronal entrenada se integraría en el flujo del algoritmo existente. Durante la etapa de identificación de colonias, en lugar de confiar únicamente en métodos de procesamiento de imágenes tradicionales, la CNN se utilizaría para identificar y segmentar las colonias. Esto permitiría una separación más precisa de colonias solapadas y agrupadas.
6. **Ajustes y mejoras continuas:** basándose en los resultados y la retroalimentación de usuarios, se realizarían ajustes y mejoras continuas en el modelo de la red neuronal. Esto podría incluir la recolección de más datos de entrenamiento, la experimentación con diferentes arquitecturas de redes neuronales, y la optimización de los hiperparámetros del modelo.

#### 6.1.6. Interfaz de usuario mejorada

Una mejora significativa para el sistema sería el desarrollo de una interfaz de usuario más accesible e intuitiva. La creación de una interfaz amigable que facilite la interacción del usuario con el sistema, proporcionando herramientas de ajuste y visualización claras y comprensibles, mejoraría significativamente la experiencia del usuario.

El desarrollo de tutoriales interactivos y guías de uso integradas en la interfaz también sería beneficioso, permitiendo a los usuarios comprender y utilizar el sistema de manera eficiente sin necesidad de formación extensa. Además, la capacidad de personalizar y guardar configuraciones de usuario podría mejorar la eficiencia del flujo de trabajo y la adaptación del sistema a necesidades específicas de los usuarios.

## 7. Bibliografía


[1] **Emerging Issues and Approaches in Microbial Food Safety.** (2021). *MDPI*.

[2] **Manual vs. Automated Methods in Clinical Microbiology.** *Clinical Microbiology Reviews*.

[3] **Advancements in Image Processing for Bacterial Counting.** (2021). *Computational Biology and Chemistry*, 90, 107394.

[4] **The Selective Value of Bacterial Shape.** (2006). *Microbiology and Molecular Biology Reviews*, 70(3), 660-703.

[5] **Accuracy of Plate Counts.** (2011). *Journal of Validation Technology,* 17(3), 42-46.

[6] **Diarrhoeal disease**. (2017). *World Health Organization (WHO)*.

[7] **Antibiotic resistance threats in the United States.** (2019). *Center for Disease Control and Prevention (CDC)*.

[8] **Deep Learning in Label-Free Cell Classification.** (2016). *Nature Communications,* 7, 12485.

[9] **The OpenCV Library.** (2000). *Dr. Dobb's Journal of Software Tools*.

[10] **Laboratory Automation: A View of the Clinical Laboratory.** (2018). *Clinical Laboratory Medicine,* 38(1), 15-24.

[11] Otsu, N (1979). **A threshold selection method from gray-level histograms.** *IEEE Trans. Sys., Man., Cyber.*, 9(1), 62-66.

[12] **Microfluidics and Point-of-Care Testing.** (2008) *Lab on a Chip,* 8(12), 1982-1983.

[13] Hough, P.V.C. (1962). **Method and means for recognizing complex patterns.** (1962). *US Patent 3,069,654*.

[14] Ballard, D.H. (1981). **Generalizing the Hough transform to detect arbitrary shapes**. *Pattern Recognition,* 13(2), 111-122.

[15] Yuen, H.K., Princen, J., Illingworth, J., & Kittler, J. (1990). **Comparative study of Hough transform methods for circle finding.** *Image and Vision Computing,* 8(1), 71-77.

[16] Levers, V. (1993). **The application of circular Hough transform to the detection of circular features in engineering drawings.** *Computer-Aided Design,* 25(2), 1-8.

[17] Kimme, C., Ballard, D.H., & Sklansky, J. (1975). **Finding circles by an array of accumulators.** *Communications of the ACM,* 18(2), 120-122.

[18] Duda, R.O., & Hart, P.E. (1972). **Use of the Hough transformation to detect lines and curves in pictures.** *Communications of the ACM,* 15(1), 11-15.

[19] Xu, L., Oja, E., & Kultanen, P. (1990). **A new curve detection method: Randomized Hough Transform (RHT).** *Pattern Recognition Letters,* 11(5), 331-338.

[20] Illingworth, J., & Kittler, J. (1988). **A survey of the Hough transform.** *Computer Vision, Graphics, and Image Processing,* 44(1), 87-116.

[21] Davies, E.R. (2005). **Machine vision: Theory, algorithms, practicalities.** *Elsevier*.

[22] Gonzalez, R.C., & Woods, R.E. (2002). **Digital Image Processing**. *Prentice Hall*.

[23] Canny, J. (1986). **A Computational Approach to Edge Detection**. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 8(6), 679-698.

[24] Marr, D., & Hildreth, E. (1980). **Theory of edge detection**. *Proceedings of the Royal Society of London. Series B. Biological Sciences,* 207(1167), 187-217.

[25] Liu, L., & Wong, A. (2011). **A survey of adaptive smoothing techniques for image noise removal.** *Pattern Recognition*, 44(5), 1182-1197.

[26] Kass, M., Witkin, A., & Terzopoulos, D. (1988). **Snakes: Active contour models.** *International Journal of Computer Vision,* 1(4), 321-331.

[27] Ronneberger, O., Fischer, P., & Brox, T. (2015). **U-Net: Convolutional Networks for Biomedical Image Segmentation.** *MICCAI*.

[28] Diame, Z. E., Brahmbhatt, P., & Ma, Y. (2020). **Multimodal Biomedical Image Segmentation using Multi-Dimensional U-Convolutional Neural Network**. *BMC Medical Imaging*.

[29] Ma, Y., Nawaz, M., Vimala, B. B., & Saravanan, S. (2023). **Medical Image Segmentation Review: The Success of U-Net.** *arXiv*.

[30] Vimala, B. B., Ma, Y., Nawaz, M., & Saravanan, S. (2023). **Advances in Convolutional Neural Networks for Biological Image Analysis.** *Journal of Biomedical Imaging*.

[31] Switzky, B., Deoras, K.S., Cohn, W.M., Weiss, D.S., & Gifford, S.M. (2020). **Automated classification of bacterial cell sub-populations with convolutional neural networks.** *PLOS ONE*, 15(10), e0242423.

[32] Sarvamangala, D.R., & Kulkarni, R.V. (2021). **An Automated Deep Learning Approach for Bacterial Image Classification.** *arXiv*, 2107.04628.

[33] Sudhakar, K., & Vinoth Kumar, T. (2022). **Deep learning approach to bacterial colony classification.** *PLOS ONE*, 17(4), e0266317.

[34] Liu, Y., Wu, Y., Yu, C., & Wang, J. (2021). **A hybrid CNN-Random Forest algorithm for bacterial spore segmentation and classification in TEM images.** *bioRxiv*.

[35] Ramachandran, D., & Mahesh, V. (2023). **Novel neural network application for bacterial colony classification.** *Theoretical Biology and Medical Modelling*, 20(3), 187-202.

[36] Zhang, H., Wang, L., & Liu, Y. (2023). **Improved Bacterial Classification Using Transfer Learning with CNNs.** *BMC Medical Imaging*, 23(1), 14-25.

[37] **Colony Counter- Types, Principle, Parts, Uses, Examples.** (2020). *Microbe Notes*.

[38] **What are the common methods for bacteria counting?.** (2020). *AAT Bioquest*.

[39] Khandpur, R. (2019). **Colony Counter, Automated**. *Compendium Of Biomedical Instrumentation,* 491-494.

[40] Bolte, S., & Cordelières, F.P. (2006). **A guided tour into subcellular colocalization analysis in light microscopy.** *Journal of Microscopy*, 224(3), 213-232.

[41] Murphy, D.B. (2001). **Fundamentals of Light Microscopy and Electronic Imaging**. *Wiley-Liss*.

[42] Yu, F., & Koltun, V. (2015). **Multi-Scale Context Aggregation by Dilated Convolutions.** *arXiv*, 1511.07122.

[43] Paddock, S.W. (2002). **Principles and Practices of Laser Scanning Confocal Microscopy.** *Molecular Biotechnology*, 16(2), 127-149.

[44] Brakenhoff, G.J., Blom, P., & Barends, P. (1979). **Confocal scanning light microscopy with high aperture immersion lenses.** *Journal of Microscopy*, 117(2), 219-232.

[45] Danuser, G., & Waterman-Storer, C.M. (2006). **Quantitative Fluorescent Speckle Microscopy of Cytoskeleton Dynamics.** *Annual Review of Biophysics and Biomolecular Structure,* 35, 361-387.

[46] Inoue, S. (2006). **Foundations of Confocal Scanned Imaging in Light Microscopy.** *Methods in Cell Biology*, 70, 87-127.

[47] Lichtman, J.W., & Conchello, J.A. (2005). **Fluorescence microscopy.** *Nature Methods*, 2(12), 910-919.

[48] Pawley, J.B. (2006). **Handbook of Biological Confocal Microscopy.** *Springer*.

[49] Kessel, R.G. (1992). **Electron Microscopy: A Laboratory Manual and Handbook.** *Academic Press*.

[50] Heintzmann, R., & Ficz, G. (2013). **Breaking the resolution limit in light microscopy.** *Briefings in Functional Genomics*, 12(5), 489-503.

[51] Stelzer, E.H.K. (1998). **The Intermediate Optical System of Laser Scanning Confocal Microscopes.** *In: Pawley, J.B. (ed.) Handbook of Biological Confocal Microscopy. Springer,* pp. 207-220.

[52] Waters, J.C. (2009). **Accuracy and precision in quantitative fluorescence microscopy.** *Journal of Cell Biology*, 185(7), 1135-1148.

[53] Lichtman, J.W., & Conchello, J.A. (2005). **Fluorescence microscopy.** *Nature Methods,* 2(12), 910-919.

[54] Sanderson, M.J., Smith, I., Parker, I., & Bootman, M.D. (2014). **Fluorescence Microscopy.** *Cold Spring Harbor Protocols*, pdb.top071795.

[55] Hoffman, R. A., & Wilkinson, P. (2001). **Flow Cytometry: First Principles.** *Wiley-Liss*.

[56] Shapiro, H. M. (2005). **Practical Flow Cytometry.** *Wiley-Liss*.

[57] Brown, M., & Wittwer, C. (2000). **Flow Cytometry: Principles and Clinical Applications in Hematology.** *Clinical Chemistry,* 46(8), 1221-1229.

[58] Perfetto, S. P., Chattopadhyay, P. K., & Roederer, M. (2004). **Seventeen-colour flow cytometry: unravelling the immune system.** *Nature Reviews Immunology,* 4(8), 648-655.

[59] Ormerod, M. G. (2000). **Flow Cytometry: A Practical Approach.** *Oxford University Press*.

[60] Givan, A. L. (2011). **Flow Cytometry: An Introduction.** *Methods in Cell Biology,* 102, 1-29.

[61] Robinson, J. P., & Darzynkiewicz, Z. (1995). **Current Protocols in Cytometry.** *John Wiley & Sons*.

[62] Delalande, G., Desvaux, M., Hébraud, M., & Chafsey, I. (2016). **Bacterial counting using manual methods: a comparative study.** *Journal of Microbiological Methods,* 129, 133-139.

[63] Sutton, S. (2011). **Accuracy of plate counts.** *Journal of Validation Technology,* 17(3), 42-46.

[64] Brown, A.E. (2005). **Benson's Microbiological Applications: Laboratory Manual in General Microbiology.** *McGraw-Hill*.

[65] Tomasiewicz, D.M., Hotchkiss, D.K., Reinbold, G.W., Read, R.B., & Hartman, P.A. (1980). **Plate count, most probable number, and membranefiltration methods for coliform enumeration.** *Applied and Environmental Microbiology,* 39(1), 55-61.

[66] Buckalew, D.W., Hartman, P.A., & Santiago, M. (1978). **Comparison of methods for the enumeration of coliforms from foods.** *Journal of Food Protection,* 41(4), 323-326.

[67] Ray, B., & Bhunia, A. (2013). **Fundamental Food Microbiology.** *CRC Press*.

[68] Quinn, P.J., Carter, M.E., Markey, B., & Carter, G.R. (1994). **Clinical Veterinary Microbiology.** *Mosby-Year Book Europe Limited*.

[69] Sofos, J. N. (2008). **Improving the safety of fresh meat.** *Woodhead Publishing*.

[70] Barer, M. R., & Harwood, C. R. (1999). **Bacterial viability and culturability.** *Advances in Microbial Physiology*, 41, 93-137.

[71] Madigan, M. T., Bender, K. S., Buckley, D. H., Sattley, W. M., & Stahl, D. A. (2018). **Brock Biology of Microorganisms.** *Pearson*.

[72] **Advantages and Disadvantages of BMP File Format.** (2024). *Konsyse*.

[73] **Image file formats: png, jpg, bmp.** *Advantages & disadvantages*. *IONOS*.