



**INSTITUTO TECNOLÓGICO DE SONORA**

---

---

---

**DEPARTAMENTO DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA**

**“CONTADOR DE COLONIAS DE MICROORGANISMOS  
POR MEDIO DE VISIÓN ARTIFICIAL”**

**T E S I S**

**QUE PARA OBTENER EL TÍTULO DE:  
INGENIERO EN ELECTRÓNICA**

**PRESENTA:**

**JUAN DIEGO PÉREZ CARRAZCO**

**CD. OBREGÓN, SONORA JUNIO DE 2007**

## **Dedicatoria**

*Dedico este trabajo a todas aquellas personas que hicieron posible el que consiguiera ser un profesionalista.*

*A mis padres:*

*Dr. Federico Pérez y la enfermera Silvia Carrasco*

# Agradecimientos

Para toda mi familia, que no pongo los nombres porque son muchos, pero que todos y cada uno de ellos me han dado de una u otra forma su apoyo.

A mi asesor por haberme ayudado a culminar mi trabajo, al igual que a mis maestros revisores

A todos mis amigos y compañeros de la carrera, que también son bastantes

Al 615 en el cual viví 4 años con mis compañeros, que nunca se derrumbo, por mas mal que lo tratábamos. Y también a los caseros por no corrernos del 615 aunque nos gastáramos el dinero de la renta.

Y no podía faltar a todos aquellos compañeros del Elektroshow y Roboshow que tanto nos divertimos organizando este par de eventos los cuales nos costaron materias y unas buenas desveladas.

Gracias a ti por dedicarle tu tiempo en leer este trabajo ya que fue escrito esperando ser de utilidad para aquellas personas interesadas en seguir trabajando en este tipo proyectos, esperando que lo que yo pude lograr tu puedas mejorarlo

# Índice

<b>Índice de tablas</b> .....	<b>V</b>
<b>Resumen</b> .....	<b>VIII</b>
<b>1 Introducción</b> .....	<b>1</b>
<b>1.1 Antecedentes</b> .....	<b>2</b>
1.1.1 El laboratorio de microbiología.....	<b>2</b>
1.1.2 Conteo de placa.....	<b>4</b>
1.1.2.1 El medio de cultivo.....	<b>4</b>
1.1.2.2 La caja de Petri.....	<b>5</b>
1.1.3 Conteo.....	<b>6</b>
1.1.4 Formación de colonias.....	<b>8</b>
1.1.5 Contadores de colonias.....	<b>9</b>
1.1.5.1 Contadores manuales.....	<b>9</b>
1.1.5.2 Contadores digitales.....	<b>10</b>
<b>1.2 Planteamiento del problema</b> .....	<b>11</b>
<b>1.3 Objetivos</b> .....	<b>12</b>
<b>1.4 Justificación</b> .....	<b>13</b>
<b>1.5 Alcances</b> .....	<b>14</b>
<b>1.6 Bibliografía</b> .....	<b>15</b>

<b>2 Marco teórico</b> .....	<b>16</b>
<b>2.1 ¿Qué es una imagen digital?</b> .....	<b>17</b>
2.1.1 Imagen digital.....	17
2.1.2 Adquisición de una imagen.....	18
2.1.3 Iluminación.....	20
2.1.3.1 Luz trasera difusa ( <i>Backlight</i> ).....	20
2.1.3.2 Luz radial difusa ( <i>Frontlight</i> ).....	21
2.1.3.3 Luz axial difusa.....	21
2.1.3.4 Campo oscuro ( <i>Dark field</i> ).....	21
2.1.3.5 Campo brillante.....	22
2.1.3.6 Día nublado.....	22
2.1.4 Representación de una imagen digital.....	23
2.1.5 Tipos de imágenes.....	26
<b>2.2 Relaciones básicas entre píxeles</b> .....	<b>27</b>
2.2.1 Vecinos de un píxel.....	28
2.2.2 Adyacencia, Conectividad, Regiones y Límites.....	29
<b>2.3 Dominio espacial</b> .....	<b>30</b>
<b>2.4 Operaciones morfológicas</b> .....	<b>31</b>
2.4.1 Teoría de conjuntos.....	32
2.4.2 Operaciones lógicas con imágenes binarias.....	34
2.4.3 Dilatación y Erosión.....	35
2.4.3.1 Dilatación.....	36
2.4.3.2 Erosión.....	40
2.4.4 Rellenado de regiones.....	42
<b>2.5 Segmentación de imágenes</b> .....	<b>43</b>

2.5.1 Detección de umbral.....	44
2.5.2 Transformada <i>watershed</i> .....	46
2.5.2.1 Construcción de presas.....	47
2.5.2.2 El uso del gradiente.....	49
2.5.2.3 El uso de marcas.....	50
<b>2.6 Bibliografía.....</b>	<b>53</b>
<b>3 Método.....</b>	<b>54</b>
<b>3.1 Hardware.....</b>	<b>55</b>
<b>3.2 Software.....</b>	<b>57</b>
3.2.1 Desarrollo de los algoritmos.....	58
3.2.1.1 Captura de imagen.....	58
3.2.1.2 Conversión a escala de grises.....	60
3.2.1.3 Eliminación del fondo.....	61
3.2.1.4 Filtrado de la imagen.....	66
3.2.1.5 Segmentación de la imagen .....	67
3.2.1.6 Algoritmo de conteo.....	71
<b>4 Análisis de resultados.....</b>	<b>73</b>
<b>4.1 Sensor.....</b>	<b>74</b>
4.1.1 Tipo de iluminación.....	75
4.1.2 Filtro de contraste.....	76
4.1.3 Intensidad de luz.....	77
<b>4.2 algoritmo.....</b>	<b>78</b>

4.2.1 Prueba de contraste.....	78
4.2.2 Intensidad del filtro.....	79
4.2.3 Pruebas de funcionamiento.....	80
4.2.3.1 Error por detección de umbral.....	80
4.2.3.2 Error por erosión.....	81
4.2.3.3 Error por filtrado.....	82
4.2.4 Pruebas con imágenes de alta calidad.....	83
<b>5 Conclusiones.....</b>	<b>85</b>
<b>5.1 Conclusiones obtenidas en la parte teórica.....</b>	<b>86</b>
<b>5.2 Conclusiones obtenidas de la parte práctica.....</b>	<b>87</b>
<b>5.3 Recomendaciones.....</b>	<b>89</b>
<b>Glosario.....</b>	<b>90</b>

# Índice de figuras

Figura 1.1	Cajas de Petri.....	6
Figura 1.2	Contador manual.....	9
Figura 1.3	Contador de colonias digital (aCOLite).....	10
Figura 2.1	Estructura básica de un fotosensor.....	18
Figura 2.2	Proceso de adquisición de una imagen digital.....	19
Figura 2.3	Diversos tipos de iluminación.....	23
Figura 2.4	Convención de coordenadas para representar imágenes digitales.....	24
Figura 2.5	Imagen digital en forma matricial.....	24
Figura 2.6	Imagen con diferente tipo de resolución.....	25
Figura 2.7	Tipos de imágenes.....	27
Figura 2.8	Vecinos de un píxel.....	28
Figura 2.9	Mascara ubicada en el dominio espacial.....	31
Figura 2.10	Teoría de conjuntos.....	33
Figura 2.11	Traslación y reflexión.....	34
Figura 2.12	Operaciones lógicas binarias.....	35
Figura 2.13	Elemento de dilatación.....	36
Figura 2.14	Proceso de dilatación.....	37
Figura 2.15	Imagen de salida.....	38
Figura 2.16	Elemento de erosión.....	40
Figura 2.17	Proceso de erosión.....	41
Figura 2.18	Imagen de salida.....	41
Figura 2.19	Rellenado de regiones.....	43
Figura 2.20	Detección de umbral.....	45
Figura 2.21	Proceso de inundación.....	47
Figura 2.22	Construcción de presas.....	48

Figura 2.23	Uso del gradiente.....	49
Figura 2.24	Uso de marcas.....	51
Figura 3.1	Esquema del adquisición de imágenes.....	57
Figura 3.2	Código de captura de imagen.....	59
Figura 3.3	Imagen a color.....	59
Figura 3.4	Código para separar las matrices.....	60
Figura 3.5	Código para pasar las matrices a coma flotante.....	60
Figura 3.6	Código para promediar las matrices.....	61
Figura 3.7	Imagen en escala de grises.....	61
Figura 3.8	Código para generar una imagen con umbral.....	62
Figura 3.9	Imagen con umbral.....	62
Figura 3.10	Código para rellenado de regiones.....	63
Figura 3.11	Mascara rellenada.....	63
Figura 3.12	Código para generar una matriz circular.....	64
Figura 3.13	Código para erosionar la imagen con umbral.....	64
Figura 3.14	Mascara erosionada.....	65
Figura 3.15	Código para aplicar el molde a la imagen en escala de grises.....	66
Figura 3.16	Imagen enmascarada.....	66
Figura 3.17	Código para suavizar la imagen.....	67
Figura 3.18	Imagen suavizada.....	67
Figura 3.19	Código para generar el gradiente de la imagen.....	68
Figura 3.20	Gradiente de la imagen.....	68
Figura 3.21	Código para la detección de marcas internas.....	69
Figura 3.22	Marcas internas detectadas.....	69
Figura 3.23	Código para detectar marcas externas.....	70
Figura 3.24	Imagen con las marcas internas y externas .....	70
Figura 3.25	Código de segmentación final.....	70
Figura 3.26	Imagen original segmentada.....	71

Figura 3.27 Imagen lista para aplicar el algoritmo de conteo.....	<b>72</b>
Figura 3.28 Código para contar el numero de colonias.....	<b>72</b>
Figura 4.1 Imágenes con diferentes cámaras.....	<b>75</b>
Figura 4.2 Caja de Petri iluminada.....	<b>76</b>
Figura 4.3 Colores de contraste.....	<b>76</b>
Figura 4.4 Prueba de contraste.....	<b>78</b>
Figura 4.5 Cuatro diferentes grados de intensidad del filtro.....	<b>79</b>
Figura 4.6 Error por detección de umbral.....	<b>80</b>
Figura 4.7 Error por erosión.....	<b>81</b>
Figura 4.8 Error por filtrado.....	<b>83</b>
Figura 4.9 Imágenes de alta calidad.....	<b>84</b>



El presente trabajo trata de cómo se desarrolló un sistema de visión artificial para el conteo en placa de microorganismos, realizado para solventar la necesidad de realizar dichas cuentas de manera rápida y automática.

Básicamente este trabajo se dividió en dos partes, una destinada a la parte hardware en donde se construyó una base que mantenía fijo el sensor (cámara web) y la colocación de la iluminación. En lo referente a la parte hardware el trabajo realizado se puede considerar no muy extenso y tan sólo se trabajó lo mínimo para que éste funcionara correctamente. Se mostrará en la sección referente al marco teórico un poco sobre las formas de cómo iluminar un objeto, pero en general no se profundizó demasiado en el tema de iluminación.

La parte correspondiente al software es la que recibe mejor atención en el presente trabajo, pues se decidió que el peso de los problemas que se presentaron recayera en la programación. Todo lo referente al desarrollo se trabajó en el entorno Matlab usando su *toolbox* de procesamiento de imágenes. Sólo se utilizaron funciones ya programadas, las cuales forman parte del *toolbox*, no se desarrollaron usando algún otro lenguaje de programación que nos permitiera tener un programa autónomo.

El algoritmo se desarrolló exclusivamente para el método de cuenta en placa de Petri y más específicamente para las cajas de Petri de 10 cm de diámetro. Básicamente este algoritmo consiste en procesar la imagen a tal grado, de que sólo

tengamos aquello que nos interesa, es decir las colonias de microorganismos para después aplicar un algoritmo de conteo.

La parte correspondiente al marco teórico trata de los fundamentos del procesamiento de imágenes, siguiendo una marcada tendencia al ir explicando solamente los fundamentos en los cuales se basan los comandos usados para nuestro caso en particular. Mientras que en el método desglosa el código desarrollado acompañado de imágenes, que permiten mostrar la forma en que cada uno de estas funciones afecta a la imagen.

Para finalizar en el apartado de análisis de resultados, se muestra el funcionamiento del algoritmo aplicado a otras imágenes con características diferentes a la usada durante el desarrollo del método, señalando los errores que se produjeron y como se le pudiera dar una posible solución, además de cómo este trabajo pudiera contribuir para posteriores investigaciones en este mismo tema, y en el campo del procesamiento de imágenes en general.



# Introducción

“Cuando algo se puede leer sin esfuerzo, el gran esfuerzo se hizo en la escritura”

-- Enrique Jardiel Poncela --

En el primer capítulo acerca de este trabajo se aborda la necesidad que fomentó la creación de esta investigación, así como los principios de los cuales se tuvo que partir. Se detalló de manera muy general los aspectos referentes a la microbiología que involucra el cultivo de colonias dando hincapié a solo aquello que permite tener las bases referentes al proceso que se requiere para preparar un cultivo y su posterior conteo.

## **1.1 Antecedentes**

### *1.1.1 El laboratorio de microbiología*

La constante generación de necesidades que se presentan en nuestra comunidad, no solo estudiantil, ha sido un foco de importancia al cual se le ha prestado especial interés en el departamento de ingeniería eléctrica y electrónica. Debido a que muchas de estas necesidades se les puede dar una solución de ingeniería con especial énfasis en la electrónica. Con esto se obtiene un beneficio mutuo tanto para el organismo que requiere tal necesidad como para la formación de Ingenieros Electrónicos que requieren el desarrollo de proyectos para alcanzar su madurez como ingenieros y poder presentar un trabajo con validez de titulación.

Tal es el caso del laboratorio de microbiología con sede en la Unidad Obregón, en el cual se desarrollan importantes proyectos, tanto internos como para organismos externos, los cuales pagan por recibir estos servicios. Uno de los factores para otorgar un buen servicio en las investigaciones que se realizan, es contar con los instrumentos necesarios que faciliten sus labores.

La existencia de organismos reguladores, como es el caso de la Secretaría de Salubridad, que hacen énfasis en el nivel de higiene que deben presentar establecimientos o fábricas en donde el manejo de alimentos o sustancias para el consumo humano, hace necesario que las empresas tengan un control de la cantidad de microorganismos presentes en el ambiente, alimentos o bebidas. Usualmente se hace mediante mediciones del número de microorganismos por litro.

Una de las técnicas empleadas para determinar la calidad de higiene de algún alimento, agua potable o envasada, es la cuenta en placa (caja de Petri) que en términos de la NORMA OFICIAL MEXICANA NOM-092-SSA1-1994, BIENES Y SERVICIOS de la secretaria de salud.

*“Esta técnica no pretende poner en evidencia todos los microorganismos presentes. La variedad de especies y tipos diferenciables por sus necesidades nutricionales, temperatura requerida para su crecimiento, oxígeno disponible, etc., hacen que el número de colonias contadas constituyan una estimación de la cifra realmente presente y la misma refleja si el manejo sanitario del producto ha sido el adecuado.[1]*

*Por otra parte el recuento de termofílicos, psicofílicos y psicotróficos es importante para predecir la estabilidad del producto bajo diferentes condiciones de almacenamiento.[1]*

*Para obtener resultados reproducibles y por lo tanto significativos, es de suma importancia seguir fielmente y controlar cuidadosamente las condiciones.[1]*

*Esta técnica puede aplicarse para la estimación de microorganismos viables en una amplia variedad de alimentos”[1].*

Esta Norma Oficial Mexicana es de observancia obligatoria en el Territorio Nacional para las personas físicas o morales que requieran aplicar este método en productos alimenticios nacionales y de importación, para fines oficiales.

### *1.1.2 Conteo de placa*

La imposibilidad de contar microorganismos debido a su reducido tamaño (en el orden de los micrómetros), hacen necesario recurrir a técnicas alternas que nos permitan tener una visualización de los mismos para poder proceder a una correcta medición. Al tomar una muestra del lugar a estudiar, es necesario almacenarla en un lugar especial que nos permita estudiar los microorganismos, para ello se utilizan los medios de cultivos.

Para que los microorganismos crezcan adecuadamente en un medio de cultivo artificial, éste debe reunir una serie de condiciones como son: temperatura, grado de humedad y presión de oxígeno adecuadas, así como un grado correcto de acidez o alcalinidad, contener los nutrientes y factores de crecimiento necesarios y debe estar exento de todo microorganismo contaminante, de aquí que surge la importancia de realizar el cultivo en un medio controlado como es el laboratorio de microbiología.

#### 1.1.2.1 El medio de cultivo

*“Cualquier preparación líquida o sólida hecha específicamente para cultivo, almacenamiento o transporte de microorganismos u otros tipos de células. La variedad de los medios que existen permiten el cultivo de microorganismos y tipos de*

*células específicos, como medios diferenciales, medios selectivos, medios de tes. y medios definidos. Los medios sólidos están constituidos por medios líquidos que han sido solidificados con un agente como el AGAR o la GELATINA. [2]”*

*“El AGAR es un elemento solidificante muy empleado para la preparación de medios de cultivo. Se licúa completamente a la temperatura del agua hirviendo y se solidifica al enfriarse a 40 grados centígrados. Con mínimas excepciones no tiene efecto sobre el crecimiento de las bacterias y no es atacado por aquellas que crecen en él. La Gelatina es otro agente solidificante pero se emplea mucho menos ya que bastantes bacterias provocan su licuación”.[2]*

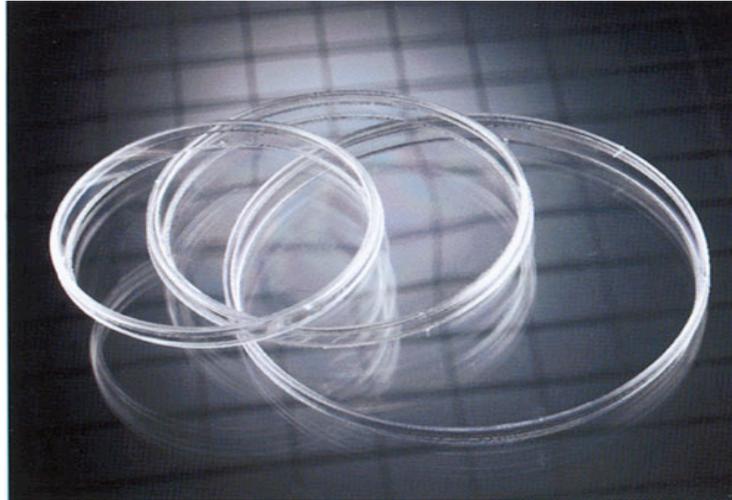
*“En los diferentes medios de cultivo se encuentran numerosos materiales de enriquecimiento como hidratos de carbono, suero, sangre completa, bilis, etc. Los hidratos de Carbono se adicionan por dos motivos fundamentales: para incrementar el valor nutritivo del medio y para detectar reacciones de fermentación de los microorganismos que ayuden a identificarlos. El suero y la sangre completa se añaden para promover el crecimiento de los microorganismos menos resistentes”.[2]*

*“También se añaden colorantes que actúan como indicadores para detectar, por ejemplo, la formación de ácido o como inhibidores del crecimiento de unas bacterias y no de otras (el Rojo Fenol se usa como indicador ya que es rojo en pH básico y amarillo en pH ácido. La Violeta de Genciana se usa como inhibidor ya que impide el crecimiento de la mayoría de las bacterias Gram-positivas)”.[2]*

#### 1.1.2.2 La caja de Petri

*“Es un recipiente redondo, de cristal o plástico, de color traslúcido de diferentes diámetros (siendo más comunes los de diámetros alrededor de 10 cm), de fondo bajo, con una cubierta de la misma forma que la placa, pero algo más grande de diámetro,*

*para que se pueda colocar encima y cerrar el recipiente (figura 1.1), fue inventada por el bacteriólogo alemán J.R. Petri (1852-1921)”[3].*



**Figura 1.1 Cajas de Petri.**

En el fondo de la caja de Petri (previa esterilización) se coloca el medio de cultivo sólido junto a los microorganismos a cultivar, de esta forma los nutrientes actúan y permiten el crecimiento de las mismas, hasta alcanzar un número en el que éstas empiezan a ser visibles en formas de manchas, es en este punto (se establecen lapsos de 24 horas) cuando se puede realizar un conteo y estimar un aproximado del grado de contaminación que existe en el lugar estudiado.

### **1.1.3 Conteo**

Previamente al cultivo en el momento de preparar las muestras se realizan diluciones en escalas de 10 mL con agua destilada, se colocan en la incubadora y pasadas 24 horas se realiza un conteo, este lapso ya está establecido debido a que de seguir incrementándose el tiempo, el crecimiento también se vería incrementado.

Como se realizaron diferentes cultivos de una misma muestra se deben de seguir las siguientes consideraciones para escoger la muestra que nos dará la información correcta acerca del grado de contaminación que buscamos:

- *“El número de colonias debe ir en los márgenes de 25 a 250”.[1]*
- *“Números superiores a estos márgenes, se consideran como “cultivo masivo” (y no se puede estimar un número aproximado)”[1].*
- *“Se debe tomar el cultivo menos diluido, siempre y cuando el número de colonias sea menor el margen superior anteriormente mencionado”[1].*

Una vez seleccionado el cultivo se procede a realizar el conteo el cual se realiza en su totalidad de manera manual, ayudados únicamente por contadores manuales, los cuales consisten en una base para colocar la caja de Petri, con iluminación del tipo *Backlight* y una lupa para tener una mejor visualización de las colonias más pequeñas.

Con ayuda de un plumón se van marcando aquellas colonias que se van contando, de esta manera se evitan repetir colonias ya contadas con anterioridad, un contador se va incrementado al ir marcando las colonias ya sea por el usuario o en ocasiones, dependiendo del contador éste lo realiza de manera automática al ir presionando la caja de Petri con el plumón.

El número final que se le debe entregar al cliente es el número de colonias contadas multiplicadas por el inverso de la dilución de la muestra, expresadas como UFC/mL, “unidades formadoras de colonias por mililitro”, pues toda colonia que se generó en la placa de Petri lo hizo a partir de una célula principal, es por eso que es válido usar el número de las colonias formadas para determinar un grado de contaminación en un ambiente, alimento o bebida determinada.

#### 1.1.4 *Formación de colonias*

La forma que puede presentar una colonia de microorganismos es muy variada, generalmente son en forma circular con una pequeña corona a su alrededor, en cuanto a su color van desde un tono blanco hasta un color rojo intenso.

La existencia de variaciones en la forma de las colonias provoca casos particulares los cuales son llamados formación de colonias extendidas, que pueden presentar las siguientes formas:

1. *“Cadenas de colonias no separadas claramente entre sí, que parecen ser causadas por la desintegración de un cúmulo de bacterias”.[1]*
2. *“Colonias que se desarrollan en película entre el AGAR y el fondo de la caja”.[1]*
3. *“Colonias que se desarrollan en película en la orilla de la caja sobre la superficie del AGAR”.[1]*
4. *“Colonias de crecimiento extendido y en algunas ocasiones acompañadas de inhibición del crecimiento, que en conjunto exceden el 50% de la caja o represión del crecimiento que por sí mismo excede el 25% de la superficie de la caja”.[1]*

*“Para este tipo de casos se deben tomar las siguientes consideraciones en el momento de realizar el conteo de placa: En el primer caso si la caja contiene una sola cadena, contar como una sola colonia, si la caja contiene varias cadenas que parecen originarse de fuentes separadas, contar cada cadena como colonia individual. No contar cada colonia de la cadena individualmente. Las colonias citadas en los casos 2 y 3 generalmente se observan como crecimiento diferenciable de*

*otras colonias y se cuentan como tales. Los crecimientos tipo 4, reportarlos como crecimiento extendido”.[1]*

### 1.1.5 Contadores de colonias

#### 1.1.5.1 Contadores manuales

Por mucho tiempo los laboratorios contaron con instrumentos manuales para contar colonias de cultivo sólido, siendo la base de estos contadores el uso de la vista por parte del usuario, la única ayuda recibida por parte del instrumento era la facilitación que éste le proporcionaba en cuanto a visibilidad (mediante iluminación y una lupa de aumento) y al manejo de la cantidad de las colonias mediante un registrador mecánico o electrónico con el cual se llevaba la cuenta. Empresas como: SCHÜTT LABORTECHNIK GMBH, o “REICHERT INC” [4]. (figura 1.2) siguen comercializando este tipo de contadores, su bajo costo hace que sea la principal razón por la que la mayoría de los laboratorios los siguen usando.



**Figura 1.2 Contador manual.**

### 1.1.5.2 Contadores digitales

Este tipo de contadores hacen uso de sistemas de visión artificial para sintetizar el proceso de conteo. Constituidos por cámaras de video de alta calidad, con la cual se le toma una foto a la caja de Petri, para después por medio de un *software* realizar un procesamiento a dicha imagen, cuyo resultado nos dirá automáticamente el número de colonias.

El procesamiento aplicado a la imagen no es un método rígidamente establecido, ya que el fabricante determina el proceso que su programa deba seguir, o que tan automáticamente nos realizará el conteo, habiendo que configurarlo antes de su funcionamiento en algunos casos. El tipo de procesamiento aplicado a la imagen puede ir desde procesos morfológicos hasta la aplicación de técnicas como lógica difusa o redes neuronales. Empresas como Symbiosis, Perceptive o Lemnatec son algunos de los fabricantes de este tipo de contadores. En la figura podemos ver un ejemplo de uno de ellos, además de cómo nos muestra en pantalla el resultado.



**Figura 1.3 Contador de colonias digital (äCOLyte).**

El uso obligado de una computadora por parte de este tipo de contadores encarecen más su implementación, aparte de ya ser bastante costosos, pero a su favor tenemos que en este tipo de contadores es mínimo el tiempo empleado en su implementación, además de su capacidad de detectar colonias de hasta 0.3 mm de diámetro e incluso en algunos modelos es posible colocar más de una caja de Petri por vez.

## **1.2 Planteamiento del problema**

El proceso de contar las colonias de manera totalmente manual por una persona, constituye un factor de error, ya que las personas son subjetivas al momento de usar el sentido de la vista, además de estar expuestos a cambios en el estado físico y anímico, los cuales afectan la manera en que se desempeñan al realizar un trabajo, dando variaciones en el resultado final que se le entrega a un cliente. Un mal conteo refleja un mal trabajo y la consecuente pérdida de confianza en trabajos posteriores.

El trabajo que se necesita para realizar el conteo es muy simple, si bien los requerimientos son mínimos en cuanto a conocimientos de microbiología, la persona encargada de esta tarea es el mismo Ingeniero que realiza todo el proceso del estudio entero que se requiere para llegar al resultado final. El simple trabajo de contar colonias, al ser un proceso tardado, le quita mucho tiempo, el cual podría aprovecharse para estar realizando otras labores que requieran más de sus habilidades y conocimientos.

Un sistema de conteo automático de colonias de microorganismos existente en el mercado es un gasto fuerte para cualquier laboratorio de microbiología dentro de nuestro país, muy en especial los que se encuentran en instituciones de educación

superior, y aun cuando se consiguiera un equipo como éste, se tendría que hacer fuera del país, ya que al momento no existe un solo fabricante dentro de la republica, lo cual nos motivo a realizar una investigación dentro del campo de la visión artificial y presentar una posible solución al problema de contar colonias de manera rápida y con la mayor exactitud de manera automática.

### **1.3 Objetivos**

- Desarrollar un algoritmo en MATLAB (versión 6.5) que sea capaz de procesar una imagen (extraída con una cámara web) de un cultivo de microorganismo para obtener un número aproximado de colonias cultivadas.
- Construir un prototipo en el cual se colocarán las cajas de Petri a las cuales se les quiere tomar una fotografía, (con la cámara web antes mencionada) con una correcta iluminación.

## 1.4 Justificación

El desarrollo de tecnologías dentro de nuestra institución permite desarrollar nuestra capacidad creativa como profesionista, a la par que damos solución a una necesidad dentro de nuestra comunidad. Aunque un contador de colonias por medio de procesamiento de imágenes, no es para nada nuevo, pues estos aparatos ya existen en el mercado y cuentan con el respaldo de compañías especializadas. Todas ellas son de origen extranjero y además estos dispositivos son muy costosos (entre \$3 250 USD y \$7 995 USD ), lo que nos pone en un lugar muy conveniente para desarrollar nuestra propia tecnología y poder llegar así crear algo en lo que podemos sacar provecho, tanto en lo económico como en el desarrollo de tecnología.

No solo nosotros estaremos sacando provecho en el desarrollo de este proyecto, si no que también estaremos ayudando en el proceso realizado dentro del laboratorio de microbiología. Un contador de colonias automático le daría agilidad a los estudios que se realizan en este laboratorio además de tener los resultados que el cliente exige en un menor tiempo y con una mayor precisión, garantizando calidad en el servicio que se prestó.

## 1.5 Alcances

- El proyecto se realizó dentro de los laboratorios del Instituto Tecnológico de Sonora, Unidad Nainari, así como en los laboratorios de microbiología de la Unidad Centro.
- Se desarrolló una plataforma prototipo con iluminación controlada para una correcta adquisición de la imagen con la cámara a emplear, además de ser de fácil uso.
- El estudio realizado para el procesamiento de imágenes se desarrolló de tal forma que logremos crear un algoritmo lo suficientemente eficiente para realizar pruebas en el programa MATLAB.
- Se desarrolló el algoritmo tomando como base una única fotografía.

## 1.6 Bibliografía

[1] Secretaría de Salud, Norma oficial mexicana NOM-092-SSA1-1994, <http://www.salud.gob.mx/unidades/cdi/nom/092ssa14.html>, 1995.

[2] Universidad de Buenos Aires, Los medios de cultivo en microbiología, <http://www.qb.fcen.uba.ar/microinmuno/SeminarioMedios.htm>, 2007.

[3] Wikipedia la enciclopedia libre, Placa de Petri, [http://es.wikipedia.org/wiki/Placa\\_de\\_Petri](http://es.wikipedia.org/wiki/Placa_de_Petri), 2007.

[4] Reichert, Analytical instruments, <http://www.reichertai.com/products.html?productID=1>, 2007



## Marco teórico

“Una imagen vale más que mil palabras”

-- Anónimo --

En este capítulo se irán detallando los conceptos en los cuales se basa el método desarrollado para la elaboración de este trabajo. Se sigue una línea de conceptos básicos que permiten explicar a su vez conceptos más elaborados, dejando de lado temas, que si bien están relacionados, no es la intención de este documento abarcar de manera general el campo del procesamiento de imágenes. Los temas correspondientes a este capítulo fueron tomados de [1], excepto en donde se indique

## 2.1 ¿Qué es una imagen digital?

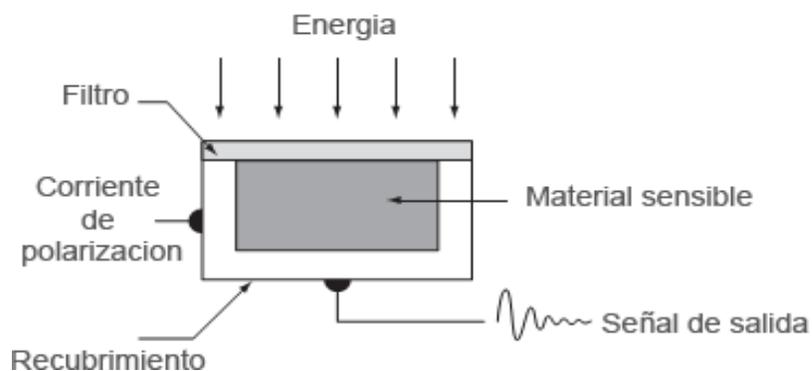
### 2.1.1 *Imagen digital*

Una imagen puede ser definida como una función de dos dimensiones representada como  $f(x,y)$ , donde  $x$  y  $y$  son coordenadas de un plano y la amplitud de  $f$  en cualquier par de coordenadas  $(x,y)$  es llamada intensidad o nivel de gris de la imagen en ese punto. Cuando  $(x,y)$ , y los valores de amplitud de  $f$  son todas cantidades finitas y discretas, nosotros lo llamamos “imagen digital”. El campo del procesamiento digital de imágenes se refiere a procesar una imagen digital mediante el uso de una computadora. A los elementos finitos que componen la imagen y con una localidad y valor particular nos referiremos como elementos de la imagen, pels, o píxeles. Píxel es el término que más comúnmente usamos.

### 2.1.2 Adquisición de una imagen

Los tipos de imagen en las cuales estamos interesados, son generados mediante la combinación de una fuente de iluminación y la reflexión o absorción de la energía de la fuente por los elementos de la escena. Englobamos estos dos conceptos en el echo de que estamos más familiarizados con las situaciones en las que la luz visible ilumina las escenas cotidianas que estamos acostumbrados a observar diariamente, pero en algunas ocasiones especiales la fuente de iluminación puede ser menos tradicional como en el caso de los ultrasonidos o los rayos-x, las cuales no son visibles al ojo humano.

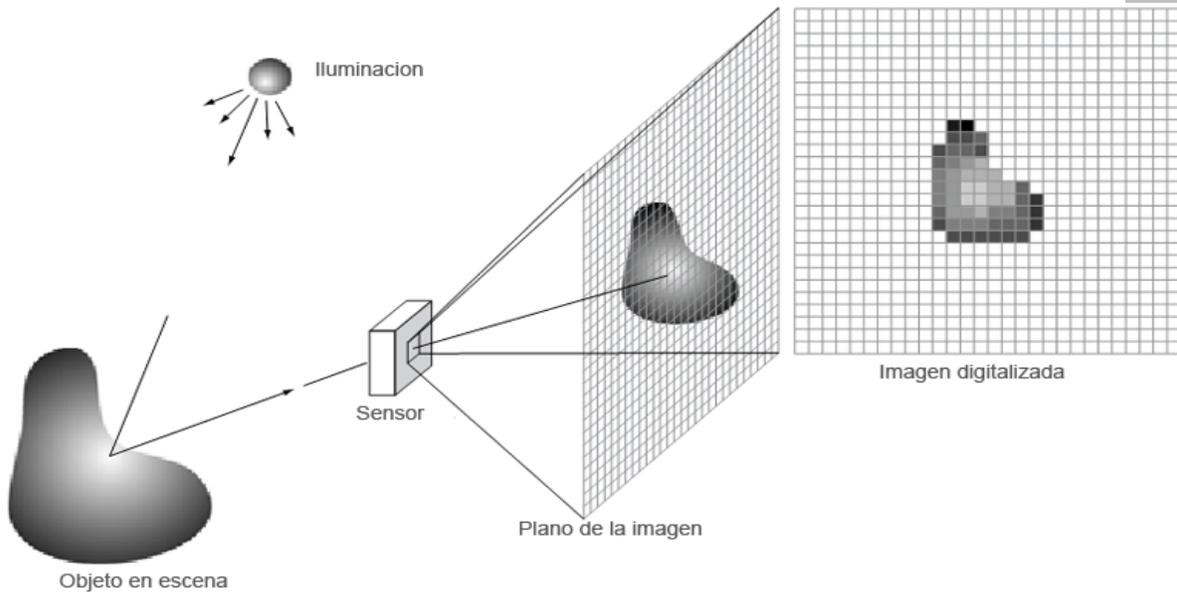
La idea de transformar la energía luminosa en una imagen digital es simple. La energía que llega al sensor es transformada en un voltaje mediante la combinación de un flujo de corriente y el material del sensor, el cual responde a un tipo en particular de energía. La señal de salida es la respuesta obtenida del sensor, de esta manera obtenemos imágenes digitalizadas (Figura 2.1).



**Figura 2.1 Estructura básica de un fotosensor.**

Tal vez el tipo de sensor más conocido sea el fotodiodo, el cual está construido de silicio y cuya señal de salida es proporcional a la cantidad de luz incidente.

Una de las formas en las que se encuentran los sensores es en arreglos en forma de matriz (Figura 2.2), más comúnmente usados por las cámaras digitales. Un sensor típico es el CCD el cual puede estar construido con arreglos de 4000 x 4000 elementos o más, la ventaja de este arreglo es que se puede obtener una imagen completa enfocada en la superficie de la cámara.



**Figura 2.2 Proceso de adquisición de una imagen digital.**

Existen arreglos en los cuales sólo se incluye un solo sensor, usados normalmente en los escáner de alta definición, pero tiene la desventaja de tener que estar moviendo el sensor en las coordenadas de x y y para poder adquirir una imagen completa, mientras que arreglos de una sola fila de sensores son usados para la adquisición de tomografías.

### 2.1.3 Iluminación

En la figura 2.2 podemos observar que parte fundamental en cualquier sistema de adquisición de imágenes es la fuente de iluminación. Existe una gran variedad de formas de iluminar, cada una de ellas facilita el procesado de determinados aspectos del objeto a inspeccionar.

Aunque inicialmente un dispositivo esté especialmente pensado para originar una iluminación determinada, no es raro que se pueda emplear en la generación de otra forma de iluminación, dependiendo de su utilización final. Por lo tanto, un mismo sistema puede ser usado de múltiples formas para iluminar, dependiendo de su colocación final, aunque esté diseñado especialmente para una determinada.

A continuación se mostrarán una lista de diversas formas de iluminación haciendo hincapié en el caso de este estudio en particular. No se detallará cada una de ellas pues este no es un trabajo referente a iluminación en específico.

#### 2.1.3.1 Luz trasera difusa (*Backlight*)

*“La cámara se dirige directamente a la fuente de luz, observando una superficie blanca totalmente uniforme. Cualquier objeto interpuesto entre la cámara y la fuente de luz produce una sombra que es detectada por la cámara como una forma en negro sobre el fondo blanco producido por el iluminador, obteniendo el máximo contraste posible (figura 2.3(a)). Aplicado en la medición precisa de los perfiles de las piezas. Se emplea también en la medición de transmitancia y de impurezas en los objetos transparentes o translúcidos”.[2]*

### 2.1.3.2 Luz radial difusa (*Frontlight*)

*“La iluminación se ubica entre la cámara y el objeto, al provenir de todo el perímetro del eje de la cámara, reduce las sombras, suaviza las texturas y minimiza la influencia de las rayas, polvo e imperfecciones que pueda tener el objeto inspeccionado en la toma de la imagen (figura 2.3(b)). Indicado para la detección de marcas de diferentes colores, caracteres y detección de todo lo que suponga un cambio de color tanto en superficies lisas con brillo como en superficies rugosas”.[2]*

### 2.1.3.3 Luz axial difusa

*“Al provenir la luz desde el mismo eje de la cámara, la sombra proyectada no es perceptible por el objetivo debido a que se produce verticalmente. En superficies planas reflectantes si no se utilizará este método de iluminación, la cámara vería reflejado su propio objetivo. Por tanto, este equipo está indicado para inspeccionar superficies planas reflectantes y cavidades profundas (figura 2.3(c))”.[2]*

### 2.1.3.4 Campo oscuro (*Dark field*)

*“Luz directa de alta intensidad que se hace incidir sobre el objeto con un ángulo muy pequeño respecto a la superficie donde descansa. De esta manera conseguimos que sobre superficies que presentan hendiduras o elevaciones, éstas interfieran el la trayectoria de la luz produciéndose zonas brillantes (figura 2.3(e)). Las aplicaciones más típicas en las que se utiliza esta técnica son aquellas en las que hay que verificar grabados, (láser, troquel..), defectos superficiales, etc..”[2].*

#### 2.1.3.5 Campo brillante

*“La fuente de luz es uniforme y difusa (suave) e incide con un pequeño ángulo sobre la superficie reflectante del objeto a inspeccionar. La cámara se coloca con el mismo ángulo de forma que obtenga una imagen reflejada de la fuente de luz en la superficie el objeto inspeccionado. Una posible imperfección en el mismo producirá una mancha oscura en la imagen captada por la cámara. La mancha oscura que se apreciará en la cámara es debida a la variación del ángulo de reflexión que produce la imperfección con respecto al ángulo de la luz incidente, perdiéndose el reflejo de fuente de iluminación (figura 2.3(f)). Indicado para detectar rayas, fisuras y deformaciones en objetos con superficies planas y brillantes”. [2]*

#### 2.1.3.6 Día nublado

*“El objeto es iluminado desde todas las direcciones con luz difusa, eliminando las sombras y reflejos, suavizando las texturas y minimizando la influencia de las rayas, el polvo y los relieves y curvaturas que pueda tener el objeto inspeccionado (figura 2.3(f)). Indicado para la detección de marcas de diferentes colores, caracteres y detección de todo lo que suponga un cambio de color tanto en superficies lisas, esféricas, rugosas o con brillo”. [2]*

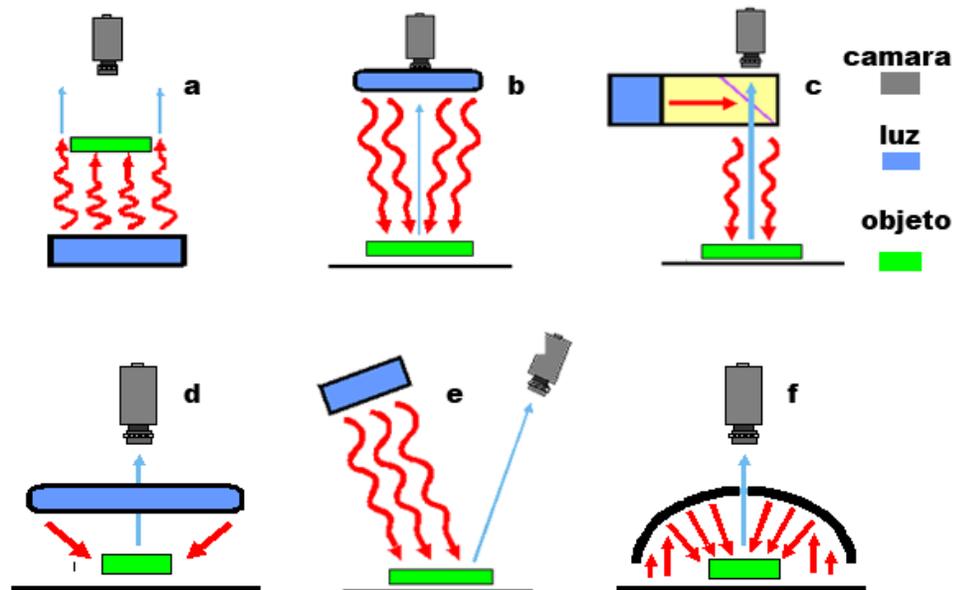
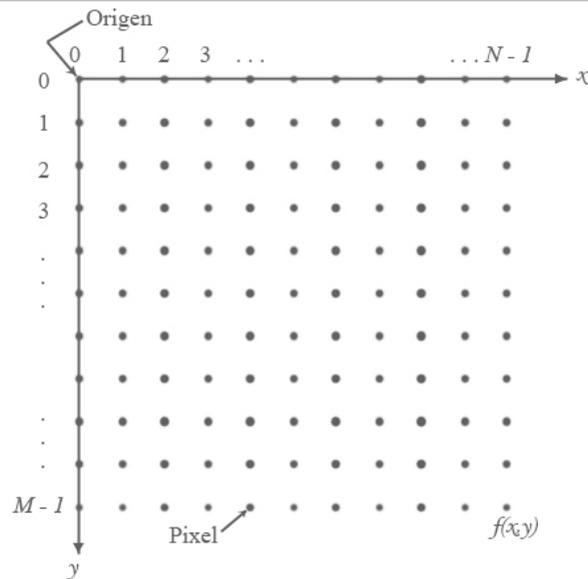


Figura 2.3 Diversos tipos de iluminación. (a) Luz trasera difusa, (b) Luz radial difusa, (c) Luz axial difusa, (d) Campo oscuro, (e) Campo brillante, (f) Día nublado.

#### 2.1.4 Representación de una imagen digital

Tenemos que una imagen  $f(x,y)$  es muestreada y como resultado de este proceso es una imagen digital de M filas y N columnas. Los valores de las coordenadas  $(x,y)$  ahora son cantidades discretas. Estos valores de coordenadas poseen un origen en el punto  $(x,y) = (0,0)$ . Los siguientes valores de coordenadas a lo largo de la primera fila de la imagen es representado como  $(x,y) = (0,1)$  y de esa forma sucesivamente (Figura 2.4). Es importante tener en cuenta que este tipo de notación representa la significancia del segundo muestreo a lo largo de la primera fila y no significa que esa sea el valor actual de los valores de las coordenadas físicas cuando la imagen fue muestreada.



**Figura 2.4 Convención de coordenadas para representar imágenes digitales.**

La forma matricial de representar una imagen se expresa de la siguiente forma, siendo la simbología mostrada sólo un ejemplo de las muchas que se pueden encontrar en las diversas literaturas.

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \dots & f(M-1,N-1) \end{bmatrix}$$

**Figura 2.5 Imagen digital M x N en forma matricial.**

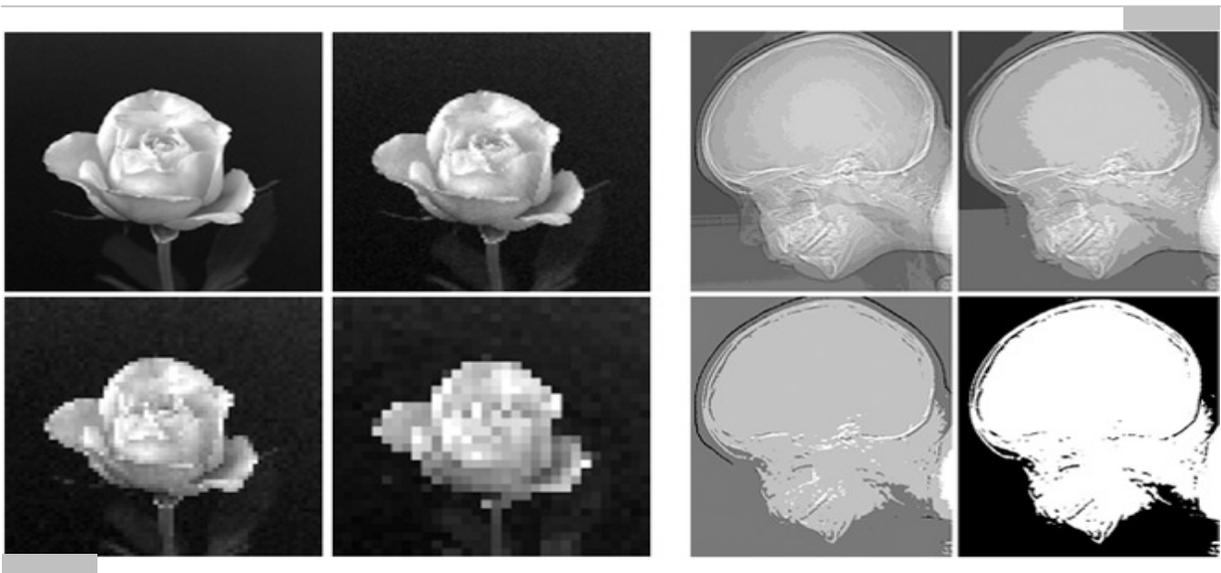
El proceso de digitalización requiere de determinar los valores de N y M al cual llamamos *muestreo*, pero también se necesita un tercer valor denotado por L el cual

representa los valores de grises de cada píxel denominado como *cuantización*. Se considera una cantidad positiva y entera cuyo valor es típicamente una potencia de 2:

$$L = 2^k$$

Con la anterior expresión nosotros asumimos que los valores de grises en una imagen se encuentran en el rango  $[0, L-1]$ . Y denotamos a  $k$  con el valor de bits que posee la imagen, por ejemplo decimos que una imagen de 8 bits posee 256 posibles valores de niveles de grises para cada píxel

El número de filas y columnas que posee una imagen también es conocido como *resolución espacial*, la cual se define básicamente como los detalles más pequeños distinguibles en la imagen, mientras que la resolución de *nivel de gris* nos da los cambios más pequeños en los niveles grises.



**Figura 2.6** Imagen con diferente tipo de resolución espacial (izquierda), imagen con diferente tipo de resolución de nivel de gris (derecha).

### 2.1.5 Tipos de imágenes

En la sección anterior hablábamos que una imagen depende tanto de su resolución espacial como la de nivel de gris que posea, así que clasificaremos al tipo de imágenes en tres grupos principales tomando como base lo anterior.

- Imágenes binarias:

Independiente de su valor de resolución espacial, sólo poseen dos posibles valores de intensidad de gris, los píxeles son blancos 1's o son negros 0's (Figura 2.7(a)). Gran parte del procesamiento implementado en este trabajo se realiza sobre este tipo de imágenes

- Imágenes en escala de grises:

La diferencia con las anteriores es que aquí hay un mayor rango de intensidad de gris, el cual es determinado por la ecuación antes mostrada  $L = 2^k$ . Por lo general se da un valor de  $k = 8$  con lo cual tenemos un rango de 256 posibles valores de intensidad de gris (Figura 2.7(b)).

- Imágenes a color:

Es en este tipo de imágenes donde se da una gran diferencia con las anteriores, pues este tipo de imágenes está compuesta por una matriz espacial de tres dimensiones, es decir tiene tres matrices en escala de grises, cada una representando una intensidad de color RGB (rojo, verde y azul). El modelo de color RGB no es el único existente, pero es el que se aplicará para el presente trabajo (Figura 2.7(c)). Otros modelos de color son CMY, CMYK y HSI.

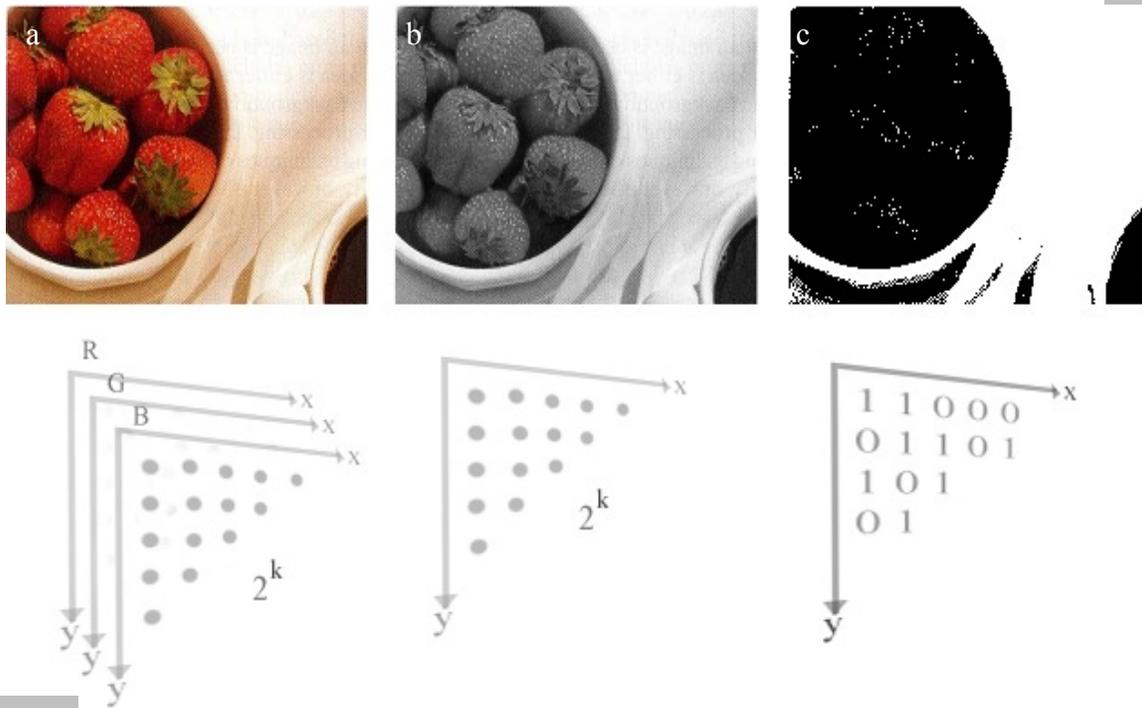


Figura 2.7 Tipos de imágenes. (a) Imagen a color con su representación matricial, (b) imagen en escala de grises, (c) imagen binaria.

## 2.2 Relaciones básicas entre píxeles

Como vimos antes, definimos como píxel en una imagen digital, al punto definido por los valores en  $x$  y  $y$  de la función bidimensional  $f(x,y)$ . Por comodidad nos referiremos con la letra  $p$  a un píxel en particular de la imagen en las secciones posteriores.

### 2.2.1 Vecinos de un píxel

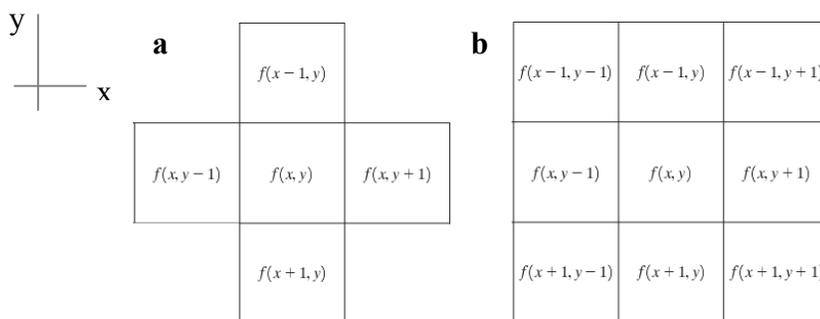
Un píxel  $p$  con coordenadas  $(x,y)$  tiene cuatro píxeles vecinos de manera horizontal y vertical cuyas coordenadas son dadas por:

$$(x+1,y), (x-1,y), (x,y+1), (x,y-1)$$

Este grupo de píxeles es llamado "4-píxeles vecinos" de  $p$  (figura 2.8(a)) representado por  $N_4(p)$ , con cada píxel a una unidad de distancia de  $(x,y)$ . Los cuatro vecinos en diagonal de  $p$  tienen por coordenadas:

$$(x+1,y+1), (x+1,y-1), (x-1,y+1), (x-1,y-1)$$

Representados por  $N_D(p)$ . Estos puntos junto con los "4-píxeles vecinos" son llamados los "8-píxeles vecinos" de  $p$  (figura 2.8(b)) representado por  $N_8(p)$ . En cada uno de los anteriores casos los píxeles vecinos en ocasiones llegan a quedar fuera de la imagen debido a sus bordes.



**Figura 2.8 Vecinos de un píxel. (a) 4-píxeles vecinos, (b) 8-píxeles vecinos.**

### 2.2.2 Adyacencia, Conectividad, Regiones y Límites

Para establecer si dos píxeles están conectados se debe determinar si son vecinos y si su nivel de gris está dentro de un criterio de similitud, por ejemplo en una imagen binaria con valores 0 y 1 dos píxeles pueden ser *4-píxeles vecinos*, pero sólo estarán conectados si tienen el mismo valor. Digamos que  $V$  representa el conjunto de niveles de gris usado para definir la adyacencia, en una imagen binaria  $V = \{1\}$  si nosotros nos referimos a una adyacencia de píxeles con valor 1. En una imagen en escala de grises es lo mismo, pero el conjunto representado por  $V$  puede contener más elementos. Al igual que los píxeles vecinos la adyacencia también se encuentra ubicada dentro de “4-adyacentes”, “8-adyacentes” y “adyacencia mixta”.

Digamos que  $S$  representa un subconjunto de píxeles en una imagen, dos píxeles estarán conectados entre sí, si existe un camino entre ellos compuesto por píxeles dentro del conjunto  $S$ . Para cualquier píxel  $p$  que está conectado dentro del conjunto en  $S$ , es llamado componente de conexión.

Si  $R$  es un subconjunto de píxeles en una imagen, será una región de la imagen si los componentes de  $R$  están dentro de un conjunto conectado. Los límites, también llamados contornos de una región  $R$  son el conjunto de píxeles que poseen uno o más vecinos que no se encuentran en  $R$ . Si  $R$  representa a una imagen completa entonces los píxeles que se encuentran en las primeras y últimas filas y columnas pasarían a ser los límites, esto es porque en los bordes de la imagen no existen píxeles vecinos.

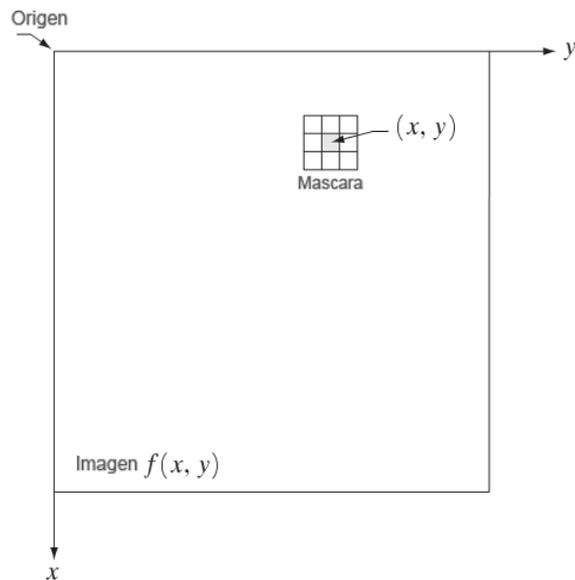
## 2.3 Dominio espacial

El término dominio espacial se refiere al plano de la imagen en sí mismo, y a la manipulación directa de los píxeles en la misma. La siguiente expresión matemática define el proceso en el que consiste el dominio espacial.

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

Donde  $f(x,y)$  es la imagen de entrada,  $g(x,y)$  es la imagen procesada y T es un operador en  $f$ , definido por los píxeles vecinos de  $(x,y)$ . La mejor manera de acercarnos a una forma de saber como se trabaja en el dominio espacial es ubicarnos en un punto  $(x,y)$ , usar una subimagen de forma cuadrada o rectangular, centrada en el punto  $(x,y)$ , el centro de la subimagen se desplaza píxel a píxel empezando de la esquina superior izquierda. El operador T es aplicado en cada localidad  $(x,y)$ , para darnos la salida  $g$  en esa misma localidad, este proceso sólo utiliza los píxeles en el área de la imagen que son abarcados por los píxeles vecinos de la subimagen. La subimagen es llamada filtro, *kernel*, máscara o ventana, siendo los tres primeros los principales nombres usados

La forma más simple que tiene el operador T es cuando sólo está compuesto de un solo píxel 1 x 1, en este caso  $g$  solo depende del valor de  $f$  en  $(x,y)$ , T se convierte en una intensidad o nivel de gris



**Figura 2.9** Máscara ubicada en el dominio espacial de la imagen.

## 2.4 Operaciones morfológicas

La palabra morfológico comúnmente se usa en la rama de la biología que describe la forma y estructura de animales y plantas, en el procesamiento de imágenes esta palabra se usa como una herramienta para extraer componentes de una imagen que son útiles en la representación y descripción de regiones.

En el lenguaje de las matemáticas la morfología es una teoría de conjuntos. Un conjunto en la morfología matemática representa un objeto de la imagen. Por ejemplo, el conjunto de todos los píxeles negros en una imagen binaria es una completa descripción morfológica de una imagen.

### 2.4.1 Teoría de conjuntos (Conceptos básicos)

Digamos que  $A$  es un conjunto en  $Z^2$ . Si  $a = (a_1, a_2)$  es un elemento de  $A$ , entonces escribimos.

$$a \in A$$

De manera similar, si  $a$  no es un elemento de  $A$ , escribimos

$$a \notin A$$

Si el conjunto no tiene elementos entonces lo llamamos nulo o conjunto vacío representado por el símbolo  $\emptyset$ . Un conjunto es especificado por estar contenido dentro de dos llaves  $\{.\}$ . Los elementos a los cuales nos referimos son las coordenadas de los píxeles los cuales representan objetos u otras presentaciones de la imagen.

Si cada elemento de  $A$  es también otro elemento de otro conjunto llamado  $B$  entonces decimos que es un subconjunto de  $B$ .

$$A \subseteq B$$

La unión de dos conjuntos  $A$  y  $B$  se representa.

$$C = A \cup B$$

De manera similar la intersección de dos conjuntos  $A$  y  $B$ .

$$D = A \cap B$$

Dos conjuntos  $A$  y  $B$  se dice que son mutuamente exclusivos si no tienen elementos en común.

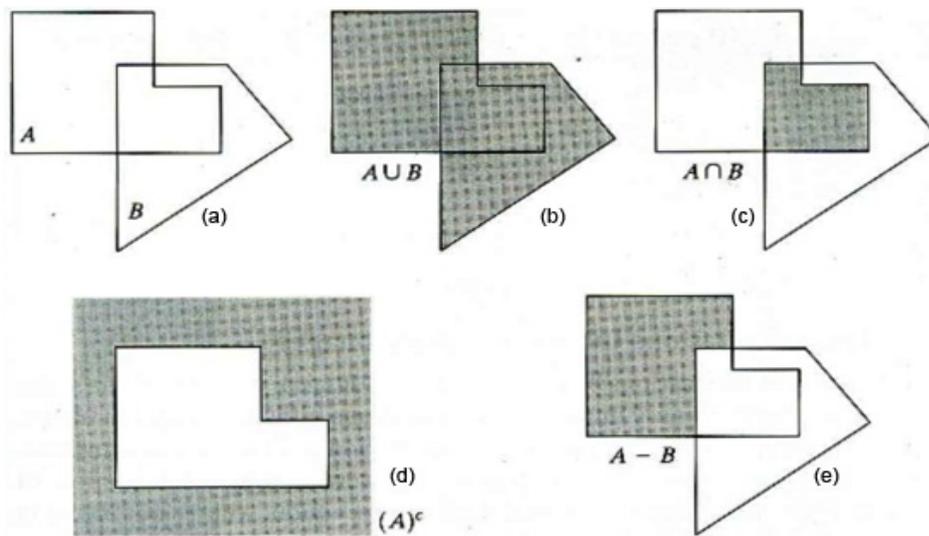
$$A \cap B = \emptyset$$

El complemento de un conjunto  $A$ , es el grupo de elementos que no se encuentran en  $A$ .

$$A^c = \{\omega \mid \omega \notin A\}$$

La diferencia de dos conjuntos  $A$  y  $B$  representada por  $A - B$  es definida como

$$A - B = \{\omega \mid \omega \in A, \omega \notin B\} = A \cap B^c$$



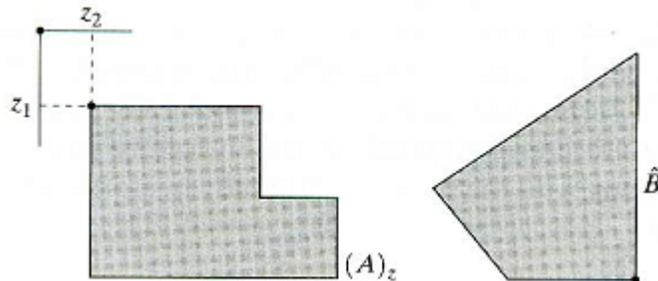
**Figura 2.10** Teoría de conjuntos. (a) Dos conjuntos  $A$  y  $B$ , (b) la unión de  $A$  y  $B$ , (c) la intersección de  $A$  y  $B$ , (d) el complemento de  $A$ , (e) la diferencia entre  $A$  y  $B$ .

Son necesarias dos definiciones más que no son tan extensamente usadas como las anteriores pero útiles para comprender conceptos mostrados más adelante. La reflexión de un conjunto  $B$  representada por  $\hat{B}$  y definido matemáticamente

$$\hat{B} = \left\{ \omega \mid \omega = -b, \text{ para } b \in B \right\}$$

La traslación de un conjunto  $A$  por el punto  $z = (z_1, z_2)$  definido con  $(A)_z$  y representado por

$$(A)_z = \left\{ c \mid c = a + z, \text{ para } a \in B \right\}$$



**Figura 2.11** La traslación de  $A$  por  $z$  (izquierda). Reflexión de  $B$  (derecha). El punto color negro representa el origen de los conjuntos mostrados.

#### 2.4.2 Operaciones lógicas con imágenes binarias

La mayoría de las aplicaciones basadas en procesos morfológicos involucran imágenes binarias. Las operaciones lógicas son simples por naturaleza y son un poderoso complemento en la implementación de los algoritmos morfológicos usados en procesamiento de imágenes. Las principales operaciones usadas son AND, OR y NOT. Estas operaciones son completamente funcionales en el sentido de que se pueden combinar para formar cualquier otra operación lógica.

Las operaciones lógicas son implementadas píxel a píxel entre dos o más imágenes, excepto en la operación NOT (Figura 2.12).

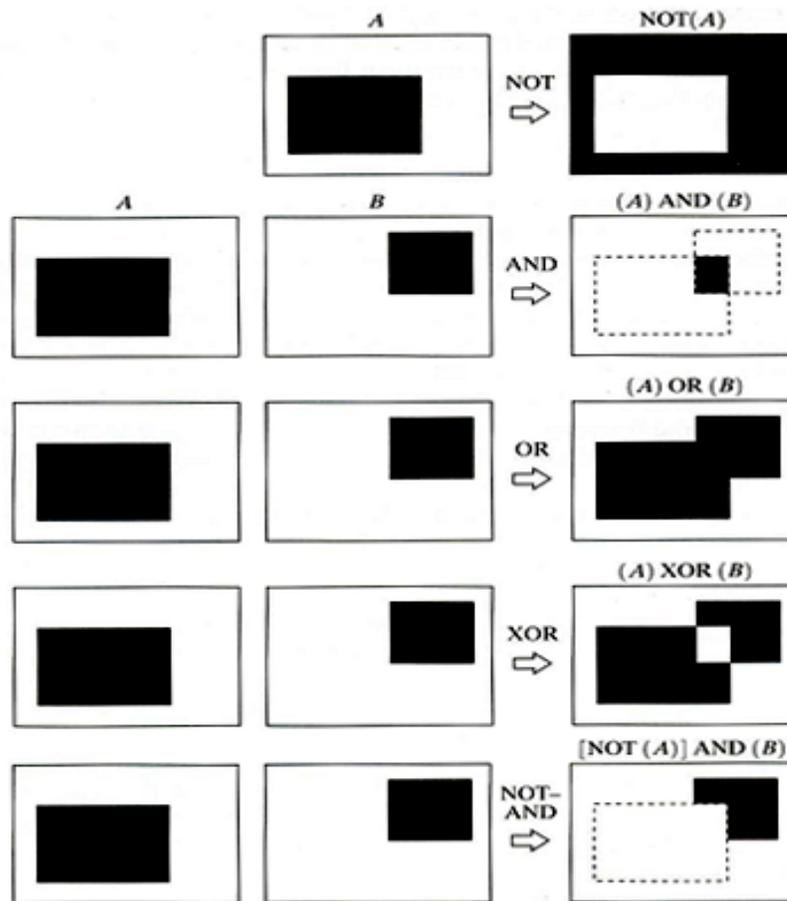


Figura 2.12 Operaciones lógicas binarias.

### 2.4.3 Dilatación y Erosión

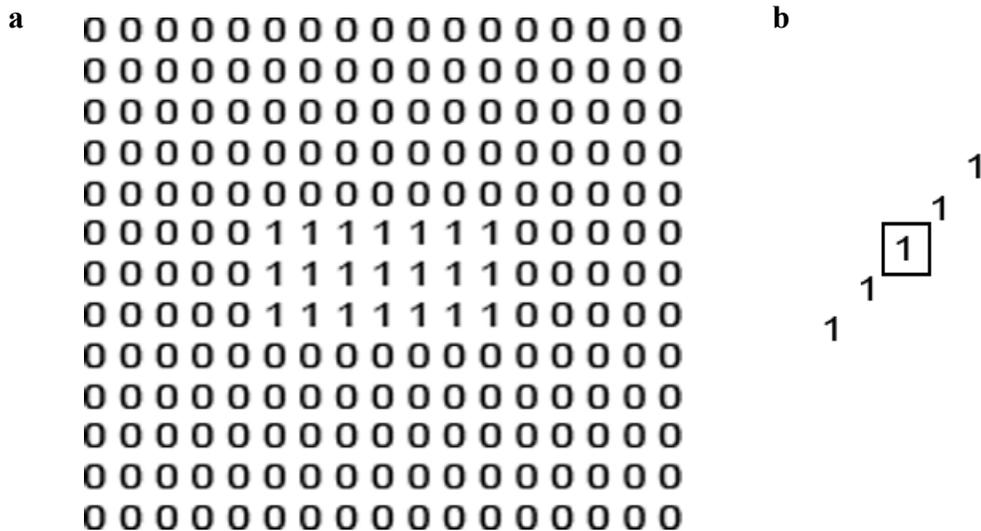
Para entrar más en detalles acerca de los procesos morfológicos, es necesario hablar acerca de dos operaciones que son fundamentales: dilatación y erosión, de hecho la mayoría de los procesos morfológicos están basados en estas dos primitivas operaciones.

### 2.4.3.1 Dilatación

Con A y B como dos conjuntos en  $Z^2$ , la dilatación de A por B queda representada por  $A \oplus B$  y definida por la expresión matemática.

$$A \oplus B = \left\{ z \mid (\hat{B})_z \cap A \neq \Phi \right\}$$

Esta ecuación está basada en la obtención de la reflexión de B en su origen y desplazando su reflexión por las coordenadas en los puntos de z. La dilatación de A por B entonces, es el conjunto de todos los desplazamientos en z de B reflejada, que interfecta en A. Nos referiremos al conjunto B como elemento de dilatación.

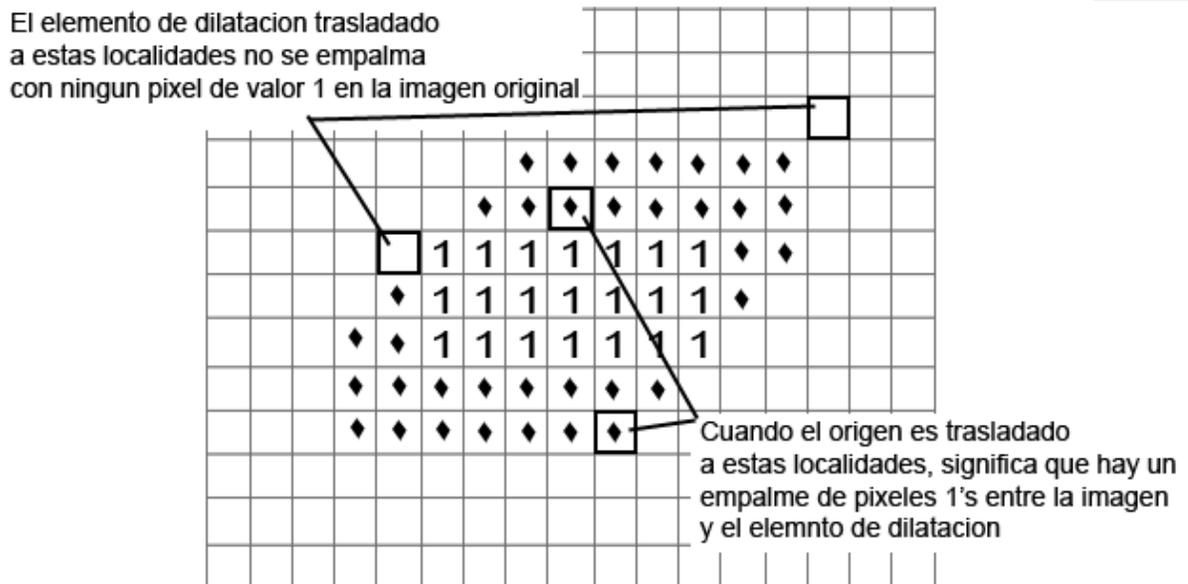


**Figura 2.13 Izquierda (a) imagen original con el objeto rectangular, derecha (b) elemento de dilatación, con su origen encerrado con un cuadrado.**

La anterior ecuación no es la única forma de definir la dilatación, para entender de mejor manera este proceso entendamos a dilatación como una operación que incrementa el tamaño de los objetos en una imagen binaria. La figura 2.13(a) muestra un objeto rectangular representado con 1's, mientras que la figura 2.13(b)

observamos la estructura del elemento de dilatación compuesta por cinco píxeles en forma de una línea diagonal (por conveniencia no se colocan los 0's faltantes para formar la matriz cuadrada), el origen de la estructura del elemento está señalada por estar encerrada en un cuadrado.

Observamos gráficamente (Figura 2.14) como el proceso de dilatación consiste en trasladar el origen del elemento de dilatación a través del dominio espacial de la imagen mientras revisa cuando existan empalmes con un píxel de valor 1 en la imagen:



**Figura 2.14** elemento de dilatación trasladado a diferentes puntos de la imagen.

La salida de la imagen es un 1, en cada localidad en que se encuentra el origen mientras exista un empalme de píxeles de valor uno entre la imagen y el elemento de dilatación (figura 2.15).

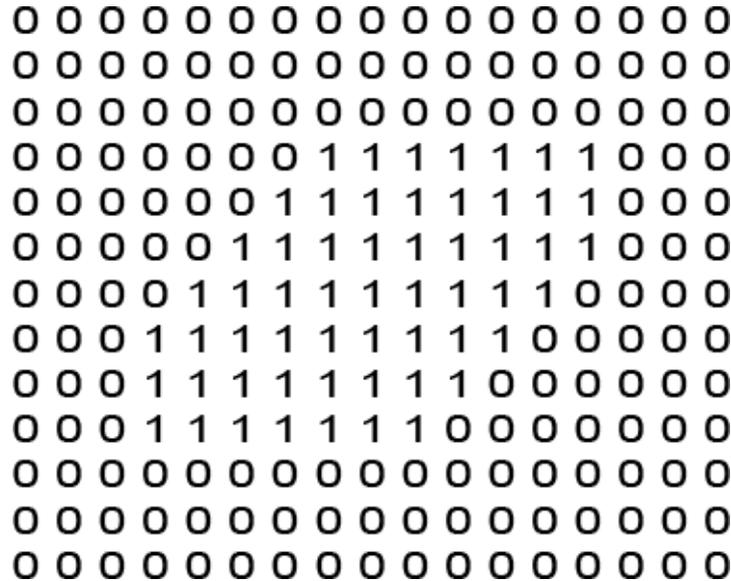


Figura 2.15 Imagen de salida.

La dilatación es conmutativa, es decir podemos afirmar  $A \oplus B = B \oplus A$  aunque por conveniencia en procesamiento de imágenes dejamos al primer operando en este caso  $A$  como la imagen y al segundo  $B$  como el elemento de dilatación, el cual es usualmente más pequeño que la imagen.

Otra propiedad es la de asociación, esto es  $A \oplus (B \oplus C) = (A \oplus B) \oplus C$ . Suponga que el elemento de dilatación  $B$  puede representarse como dos elementos separados  $B_1$  y  $B_2$ :

$$B = B_1 \oplus B_2$$

Entonces la ecuación queda  $A \oplus B = A \oplus (B_1 \oplus B_2) = (A \oplus B_1) \oplus B_2$  lo que significa que la dilatación de  $A$  por  $B$  es lo mismo que decir  $A$  dilatado por  $B_1$  y el resultado volver a dilatarlo con  $B_2$ . Podemos decir que un conjunto  $B$  puede descomponerse en dos elementos de dilatación  $B_1$  y  $B_2$ . Esto hace de la propiedad de asociatividad de gran importancia, debido a que el tiempo de cálculo requerido por la operación de dilatación es proporcional al número de píxeles 1's que tenga el elemento de dilatación. Tomemos de ejemplo el siguiente arreglo de una matriz 5 x 5 como elemento de dilatación:

$$\begin{array}{ccccc} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & [1] & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{array}$$

Esta estructura puede ser descompuesta en un elemento de 5 filas 1 columna y otro elemento de 5 columnas 1 fila:

$$[1 \ 1 \ [1] \ 1 \ 1] \oplus \begin{bmatrix} 1 \\ 1 \\ [1] \\ 1 \\ 1 \end{bmatrix}$$

El número total de elementos en el arreglo original era de 25 mientras que la suma de los elementos en los dos arreglos separados apenas suma los 10, esto hace que la dilatación por medio de la propiedad de asociación sea 2.5 veces más rápido.

### 2.4.3.2 Erosión

La definición matemática de la erosión es similar a la de dilatación. Siendo dos conjuntos A y B pertenecientes a  $Z^2$ , la erosión de A por B queda definida:

$$A \ominus B = \{z \mid (B)_z \subseteq A\}$$

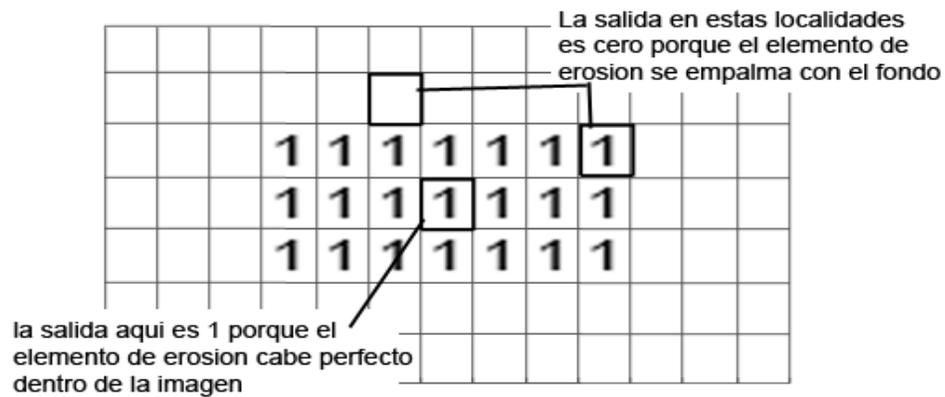
En otras palabras la erosión de A por B es el conjunto de todos los puntos de origen del elemento de dilatación donde la traslación de B no tenga un empalmamiento con el fondo de A.

Erosión es encoger o empequeñecer objetos en una imagen binaria que al igual que la dilatación se realiza mediante un elemento que para este caso es de erosión. La figura 2.16 ilustra una imagen con un objeto rectangular formado por el arreglo de 1's (a) y un elemento de erosión en forma vertical con solo 3 píxeles (b)



**Figura 2.16** Izquierda imagen original (a). Derecha elemento de erosión (b).

Gráficamente trasladamos el elemento de erosión a través del dominio de la imagen, mientras se observa si se traslapa completamente con el objeto rectangular de la imagen (figura 2.17).



**Figura 2.17** Elemento de erosión trasladado a diferentes localidades de la imagen.

La imagen de salida (figura 2.18) tiene un 1 en cada una de las localidades del origen del elemento de erosión sólo cuando se empalmaban completamente con los píxeles de valor 1 en la imagen.



**Figura 2.18** Imagen de salida

Dilatación y erosión son dos operaciones duales con respecto a su complementación y reflexión expresada como:

$$(A \ominus B)^c = A^c \oplus \hat{B}$$

Dicho de otra manera la erosión de un objeto puede ser la dilatación de su fondo y a su vez la dilatación del objeto es la correspondiente erosión de su fondo (considerando que los píxeles con valor 0 es el fondo de una imagen).

#### 2.4.4 Rellenado de regiones

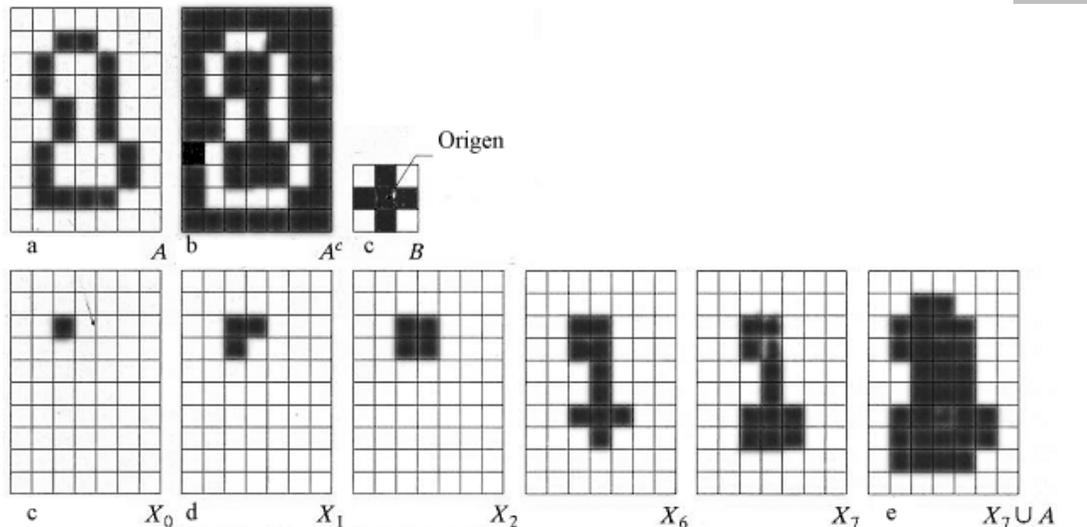
En el tema anterior revisamos las dos principales operaciones básicas en procesos morfológicos, existen en más operaciones pero cada una de ellas se basa en una serie de combinaciones de las dos anteriores, y tomando en cuenta los conceptos que también se mencionaron anteriormente en cuanto a la relación de adyacencia, conectividad, regiones y límites entre píxeles.

El proceso morfológico de relleno de regiones consiste básicamente en una combinación de dilatación, complementación e intersección. Tomando como ejemplo la figura 2.19  $A$  supone un conjunto, cuyos elementos están conectados con una adyacencia de 8-píxeles vecinos. Empezamos con un punto inicial  $p$  dentro de los límites de la región, el objetivo es rellenar con 1's la trayectoria cerrada formada en la imagen, la siguiente ecuación nos muestra como rellenar una región con valores 1's:

$$X_k = (X_{k-1} \oplus B) \cap A^c \quad k = 1, 2, 3, \dots$$

Donde  $X_0 = p$  y  $B$  es el elemento de dilatación, este algoritmo significa que debemos dilatar de manera iterada a  $p$  pero limitando estas dilataciones con la intersección del complemento de  $A$ , las iteraciones terminarán cuando  $X_k = X_{k-1}$ , entonces la unión de los conjuntos  $X_k$  y  $A$  formará la región rellena junto con sus límites.

El uso del complemento de  $A$  nos permite que el resultado de la dilatación sólo ocurra dentro de la región de nuestro interés. Este es un ejemplo de cómo los procesos morfológicos pueden ser condicionados y de esta forma poder utilizar una gran combinación de ellos para formar otros nuevos.



**Figura 2.19** Rellenado de regiones. (a) conjunto  $A$ , (b) complemento de  $A$ , (c) elemento de dilatación  $B$ , (d) diferentes dilataciones iteradas, (e) unión de  $A$  y  $X_7$ .

## 2.5 Segmentación de imágenes

La segmentación consiste en dividir una imagen en sus regiones u objetos. El nivel al cual deseamos llegar en la segmentación dependerá del problema que queramos resolver, dicho de otra manera el proceso de segmentación deberá detenerse cuando los objetos de nuestro interés estén totalmente aislados.

Los algoritmos de segmentación de imágenes se basan por lo general en dos propiedades básicas de intensidad de valores: discontinuidad y similitud. En la primera categoría la segmentación se realiza basada en los cambios abruptos de la intensidad de gris, tomados como bordes o fillos de la imagen. En la segunda

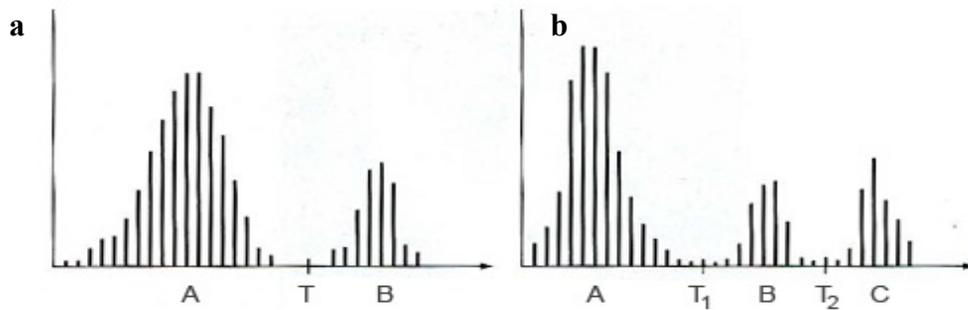
categoría esta basada en particionar una imagen en regiones que son similares, de acuerdo a un conjunto de criterios previamente definidos.

Detección de umbral, detección de borde, crecimiento de regiones, son algunos ejemplos de algoritmos de segmentación de los cuáles sólo el primero será discutido en el presente proyecto debido a su implementación y además de profundizar en el tema de segmentación no es el verdadero propósito de este trabajo. Por último también será comentado un último proceso de segmentación que hace uso extensivo de los procesos morfológicos llamado transformada *watershed* que también se aplica para este trabajo.

### 2.5.1 Detección de umbral

Debido a sus propiedades intuitivas y a la simplicidad de su implementación, la detección de umbral de imágenes disfruta de una gran popularidad en el procesamiento de imágenes. Suponga que el histograma de la figura 2.20 corresponden a una imagen  $f(x,y)$ , compuesta por objetos iluminados y un fondo oscuro, estos dos grupos tienen diferentes niveles de gris los cuales podemos apreciar fácilmente. Una manera de separar los objetos representados por B y el fondo A en la figura 2.20(a) es usar un umbral  $T$  que separa a estos dos grupos. Para cualquier punto  $(x,y)$  en el que  $f(x,y) > T$  es llamado un punto de objeto u objeto de fondo.

Existen casos en los que hay más de un objeto en la imagen en el que queremos detectar su umbral, como observamos en la figura 2.20(b) tenemos dos objetos A y B además del fondo C, aquí tenemos que usar detección de umbral multinivel la cual clasifica un punto  $(x,y)$  si éste se encuentra a lo largo del objeto  $T_1 < (x,y) \leq T_2$  o en el objeto  $f(x,y) > T_2$  o finalmente si pertenece al fondo  $f(x,y) \leq T_1$ .



**Figura 2.20** Niveles de gris en una imagen pueden ser divididos por simple detección de umbral (a) o por múltiple detección de umbral (b).

De forma general podemos decir que una imagen con detección de umbral  $f(x,y)$  es definida por:

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

Estos píxeles pueden ser etiquetados con un 1 para representar los objetos o algún otro nivel de gris que sea conveniente, y será un 0 para aquellos píxeles que representen el fondo o cualesquier otro nivel de gris que no se haya asignado a los objetos.

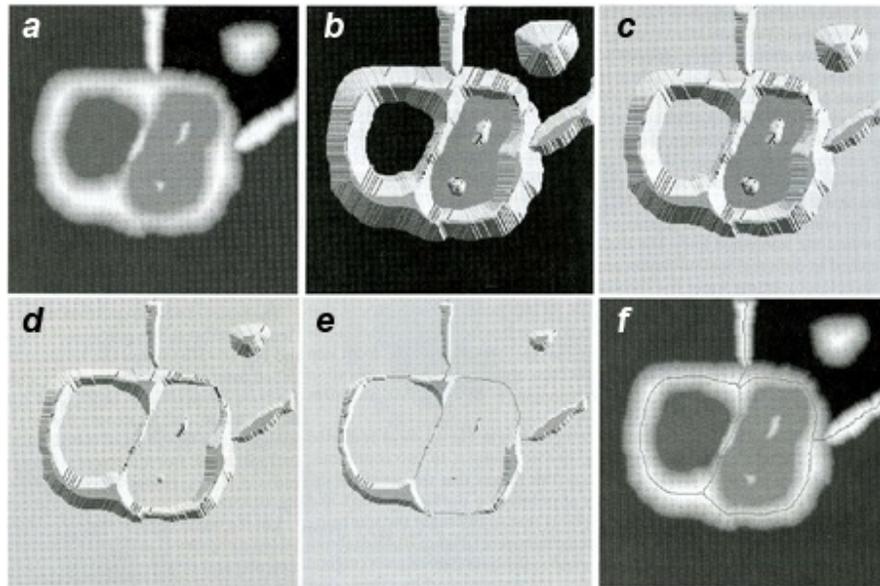
Cuando  $T$  dependa de un sólo valor en la imagen  $f(x,y)$  el umbral es llamado global, pero si depende además de un valor particular en algún punto  $f(x,y)$  es llamado local, si además de lo anterior también depende de coordenadas  $x$  y  $y$  es llamado umbral dinámico o adaptativo.

### 2.5.2 Transformada watershed

El concepto de *watershed* o transformada de inundación está basado en la visualización de una imagen en tres dimensiones: dos de ellas son las coordenadas espaciales y una tercera sería el nivel de gris. Tal como si fuera una interpretación topográfica:

- puntos a lo largo de una región mínima,
- puntos en los cuales se pueda verter “agua” llamados *watershed*
- puntos en los cuales el agua llega a tener el mismo nivel que otras regiones mínimas inundadas, líneas *watershed*.

El principal objetivo de los algoritmos de segmentación basados en este concepto es el de encontrar líneas *watershed*. La idea básica es simple, suponga que un pequeño agujero es hecho en cada una de las regiones mínimas de una imagen y que a través de esos agujeros las regiones empiezan a ser inundadas de manera uniforme, cuando llega un momento en el que el nivel del agua de una región se empieza a desbordar en otra región en ese mismo momento se construye una presa que impide el desborde hacia otra región, de esta misma forma el nivel del agua seguirá aumentando hasta que llegue el momento en que sólo sean visibles las líneas que forman las presas, éstas corresponden a las líneas *watershed*. Por lo tanto tenemos los límites de las regiones extraídos mediante el algoritmo de transformada *watershed*.

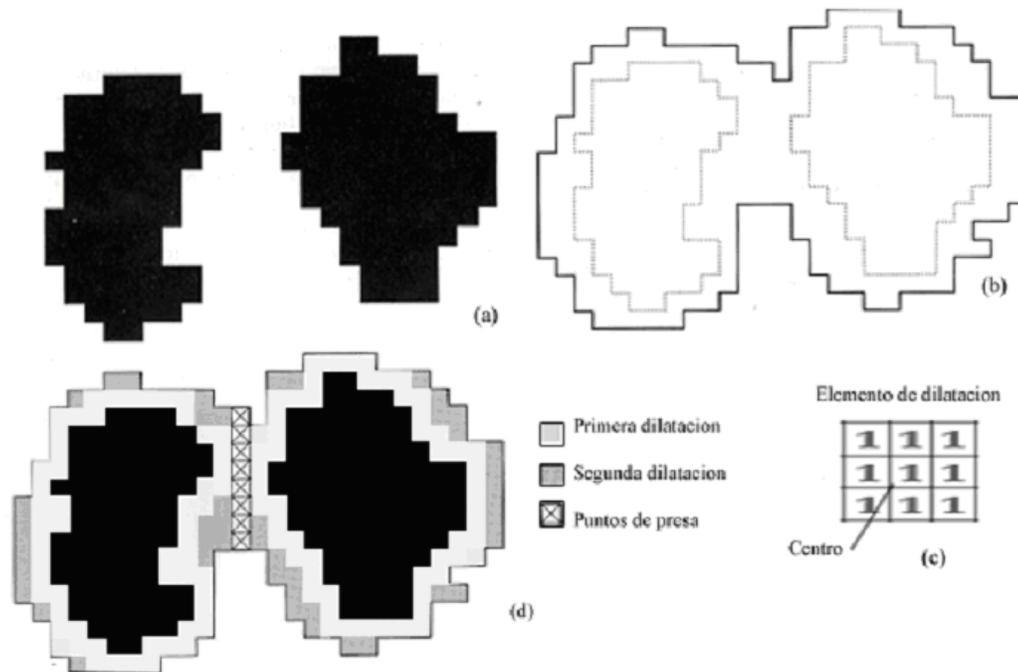


**Figura 2.21** Proceso de inundación (a) Imagen original, (b) vista topográfica, (c), (d) diferentes niveles de inundación, (e) construcción de las presas, (f) líneas *watershed* sobrepuestas en la imagen.

### 2.5.2.1 Construcción de presas

La manera más simple de construir presas que separan conjuntos binarios es mediante la implementación del proceso morfológico de la dilatación. En la figura 2.22(a) tenemos dos porciones de regiones mínimas que están siendo inundadas, continuamos el proceso de inundación, usando el elemento de dilatación mostrado en la figura 2.22(c). Este proceso de dilatación está sujeto a dos condiciones:

- (a) La dilatación sólo puede ser llevada a cabo siempre y cuando el centro del elemento de dilatación se encuentre dentro de los límites de la figura.
- (b) La dilatación no puede llevarse a cabo en los puntos en los que exista un empalme con la dilatación de una región contigua.

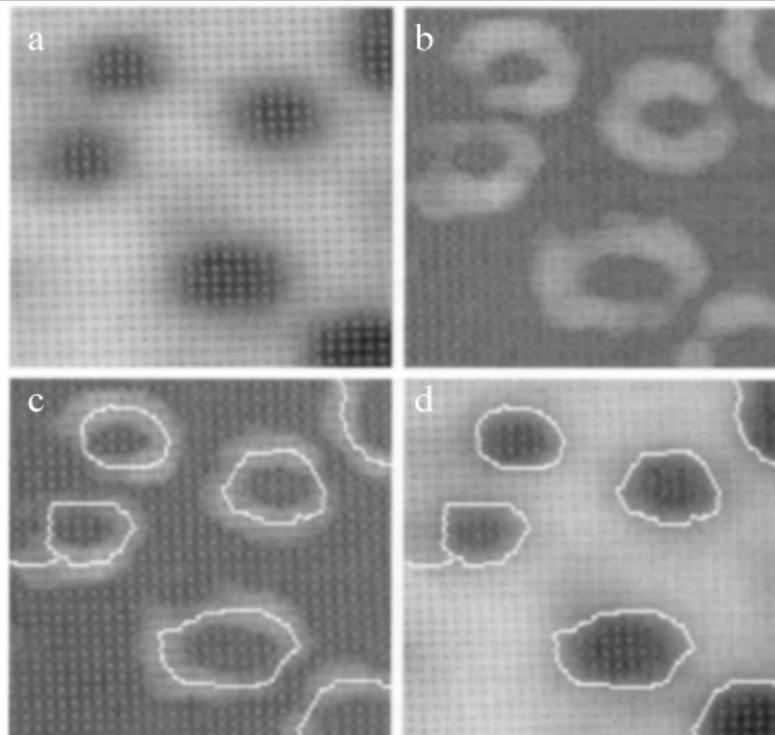


**Figura 2.22 Construcción de presas (a) Dos regiones mínimas a un primer nivel de inundación, (b) una sola figura cuando ha sido inundada por completo, (c) elemento de dilatación, (d) resultado de la dilatación y las líneas de presa.**

En la figura 2.22(d) podemos observar con un color gris claro como el primer paso de la dilatación, y como en este paso la primera condición se cumple para cada punto del proceso, mientras que la segunda condición no se aplica. En la segunda dilatación mostrada con gris oscuro, algunos puntos no cumplieron con la condición uno, mientras si cumplían con la condición dos por lo que el perímetro no queda de manera continua. Se hace evidente que en los únicos puntos en los que se cumplen las condiciones son aquellos marcados por las cruces, las cuales representan las presas que mantienen separados los dos conjuntos. La altura de las líneas de presa poseen un valor 1 más que el máximo valor de gris presentado en la imagen, esto evita que el agua se desborde antes de que el proceso de inundación sea completado y también evita que la imagen posea líneas de segmentación rotas.

### 2.5.2.2 El uso del gradiente

El gradiente de magnitud es usado generalmente en el preprocesamiento de una imagen antes de usar transformada *watershed*. El gradiente nos señala los cambios bruscos de niveles de gris en la imagen, colocando píxeles de valor alto en los bordes de las regiones y píxeles de valor bajo en el resto de la imagen. Idealmente con este concepto se generarían líneas *watershed* a lo largo de los bordes de los objetos.



**Figura 2.23** Uso del gradiente (a) Imagen original, (b) gradiente de la imagen, (c) líneas *watershed* en el gradiente, (d) imagen original segmentada.

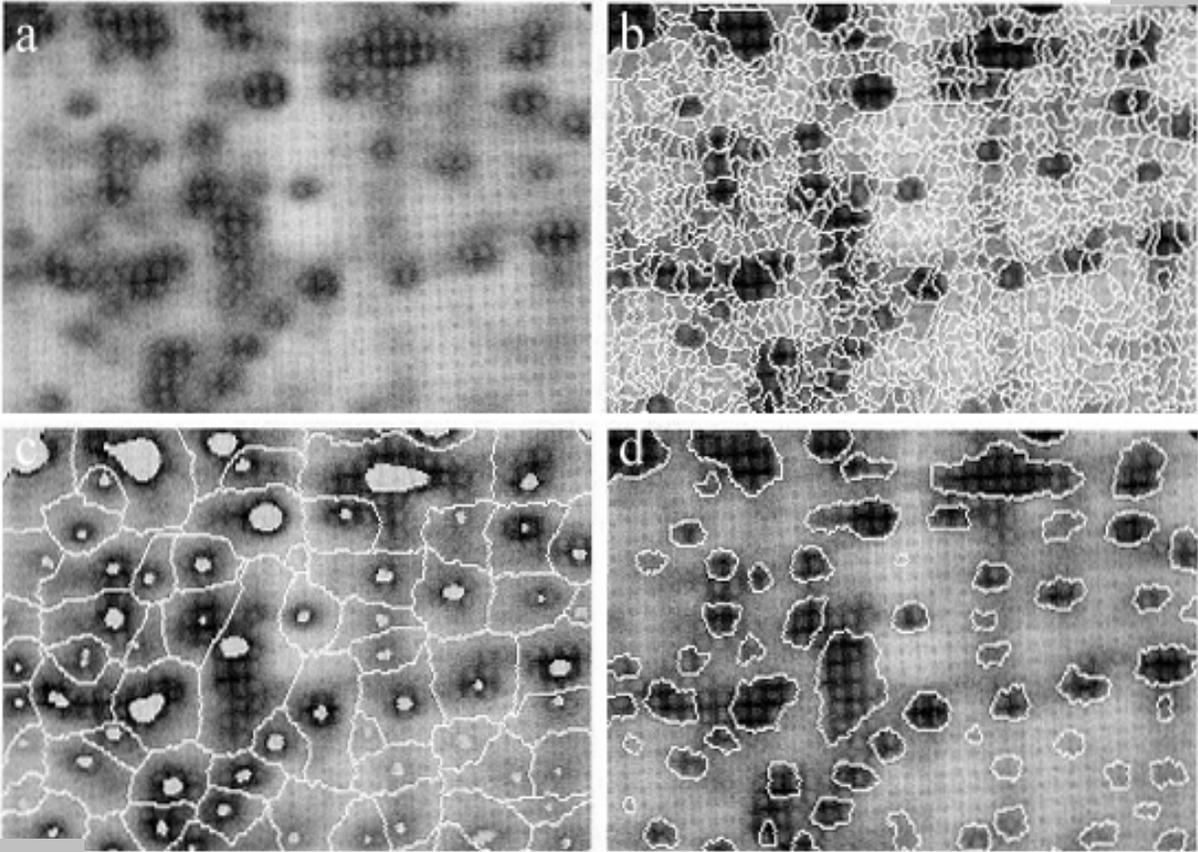
La figura 2.23(a) muestra objetos en un tono oscuro mientras el fondo es en un tono más claro, aplicamos el gradiente y podemos observar como se forman regiones de tonos más altos en donde estaban los bordes de los objetos (figura 2.23(b)), se aplica transformada *watershed* al gradiente y se superponen las líneas *watershed* a

la imagen original para mostrar como quedan identificados los objetos de interés (figura 2.23(d)).

### 2.5.2.3 El uso de marcas

La aplicación directa de la transformada *watershed* sobre la imagen o su gradiente, generalmente causa sobre segmentación debido en parte al ruido en la imagen y algunas otras irregularidades. Lo que quiere decir que un gran número de regiones son segmentadas haciendo casi inútil la utilización de este algoritmo. Una solución práctica a este problema es incorporar una etapa de preprocesado en la que se limite el número de posibles regiones a ser segmentadas.

La manera en la que se logra mayor control sobre la segmentación es mediante la utilización de marcas. Una marca es un componente conectado a lo largo de la imagen. Tenemos marcas internas, asociadas con los objetos de interés, y marcas externas, asociadas con el fondo. Un procedimiento típico de marcas consiste en dos pasos principales: (1) preprocesado y (2) determinar un conjunto de criterios para establecer las marcas. Tomemos como ejemplo la figura y observamos el resultado de aplicar transformada *watershed* directamente en la imagen, la sobre segmentación se presenta debido a la gran cantidad de regiones mínimas. Una forma de eliminar este efecto es aplicando un filtro que suavice la imagen y elimine los detalles irrelevantes. Con esto concluiría la etapa del preprocesado.



**Figura 2.24** Uso de marcas. (a) imagen original, (b) imagen sobre segmentada, (c) marcas internas y externas, (d) imagen segmentada correctamente.

Definiremos en este caso como una marca interna a una región rodeada por puntos de mayor altitud tal que esos puntos forman un componente conectado en el cual todos sus puntos tienen un mismo nivel de gris. Podemos observar las marcas en la figura 2.24(c) como los puntos en gris claro. El siguiente paso fue aplicar transformada *watershed* a la imagen suavizada bajo la restricción de que las marcas internas serán usadas como regiones mínimas. La figura 2.24(c) muestra las líneas *watershed* resultantes como marcas externas, las cuales pasan a través de los puntos más altos entre las marcas vecinas.

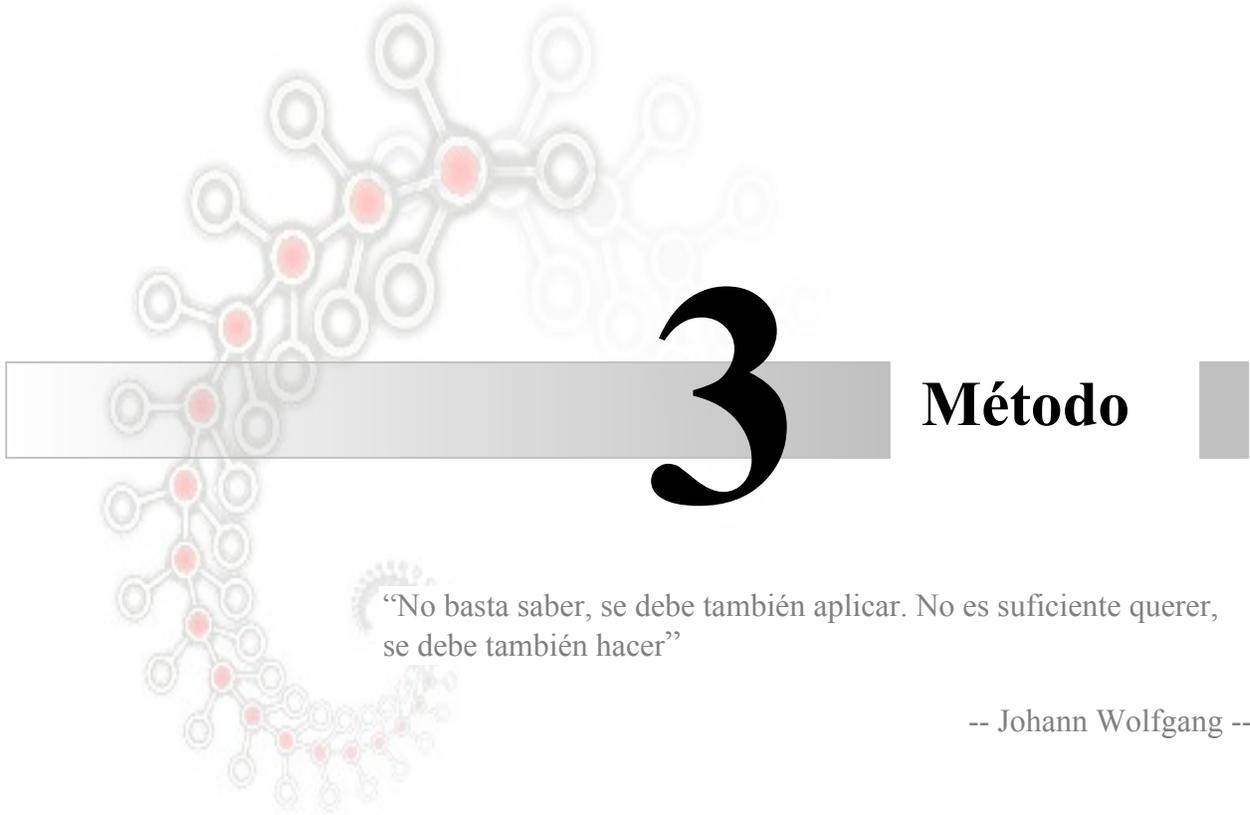
Las marcas externas particionan la imagen en regiones, cada región contiene un simple objeto y una parte del fondo con lo cual el problema se reduce a aplicar transformada *watershed* por separado al gradiente de cada una de las regiones en las que fue dividida la imagen suavizada, el resultado es mostrado en la figura 2.24(d)

La selección de marcas abarca un gran rango de procedimientos que van desde el más simple, basado en los niveles de grises y conectividad hasta procedimientos más complejos los cuales involucran tamaño, contorno, distancias relativas, texturas, etc. El principal objetivo es que las marcas brinden una base de conocimiento para aplicar transformada *watershed*, sin ocasionar el problema de sobre segmentación.

## 2.6 Bibliografía

[1] Rafael C. Gonzalez,, Richard E. Woods, Digital Image Prossecing, Pretince Hall, 2002.

[2] DCM Sistemas, Tipos de iluminación,  
[http://www.dcmsistemas.com/tipos\\_de\\_luz.html](http://www.dcmsistemas.com/tipos_de_luz.html), 2007



# 3

## Método

“No basta saber, se debe también aplicar. No es suficiente querer, se debe también hacer”

-- Johann Wolfgang --

El proyecto consiste en la elaboración completa de un prototipo de sistema de visión artificial, debido a que éste abarca la captura de la imagen y su procesado, dividiéndose así en dos partes, una enfocada directamente al hardware y otro al software.

### 3.1 Hardware

El equipo que se cuenta para realizar la captura de la imagen es el siguiente

- Cámara de web con una resolución máxima de 640x420 píxeles
- Lámpara de neón circular
- Filtro de contraste
- Bastidor de madera en el cual se fijan la cámara y la iluminación

Se realizaron pruebas para determinar la distancia focal de la cámara, para que ésta capture en su totalidad el objeto a fotografiar (la caja de Petri).

El siguiente punto fue ubicar la iluminación, la cual nos ayudará a obtener una imagen con mejores condiciones para realizarle el procesamiento, debido a que el objeto es de un material traslúcido como el vidrio, éste se comporta de manera especial ante una fuente de iluminación, ya que en los bordes de nuestra caja de

Petri se presentan brillos indeseables. Existen dos puntos importantes que fueron determinantes en nuestra fuente iluminación:

1. *Forma.*- El utilizar una fuente puntual de luz nos limitaba a mantener la iluminación en el centro de la caja de Petri, lo que nos imposibilitaba la colocación céntrica de la cámara, y el mantener la fuente de luz a un costado de la cámara nos generaba una correcta iluminación en el extremo opuesto del objeto, pero un brillo excesivo se generaba en el costado de la caja de Petri donde se encontraba la luz. Esto nos llevó a deducir que la iluminación tendría que estar presente en todos los lados de la caja de Petri, y debido a su forma circular, la iluminación también debía poseer esta forma.
2. *Distancia.*- Aun cuando se posea una iluminación uniforme alrededor de nuestro objeto, el brillo aún está presente en los bordes, y para darle solución a este problema se aleja del objeto a una distancia que nos permita obtener una cantidad de luz suficiente para distinguir nuestro objeto, y que el brillo en sus bordes no sea exagerado.

Al ser las colonias de microorganismos de un color claro que va desde el blanco hasta algunos tonos de rojo, necesitamos un color de contraste que tenga un punto intermedio para resaltar a las colonias del fondo, así como no resaltar demasiado la iluminación. Se optó por escoger un color negro como fondo, presentando éste los mejores resultados al momento de las pruebas realizadas.

Establecidas ya las distancias se procede a construir una base en donde todo esto quede fijo y con lo cual la parte correspondiente al hardware. (Figura 3.1)

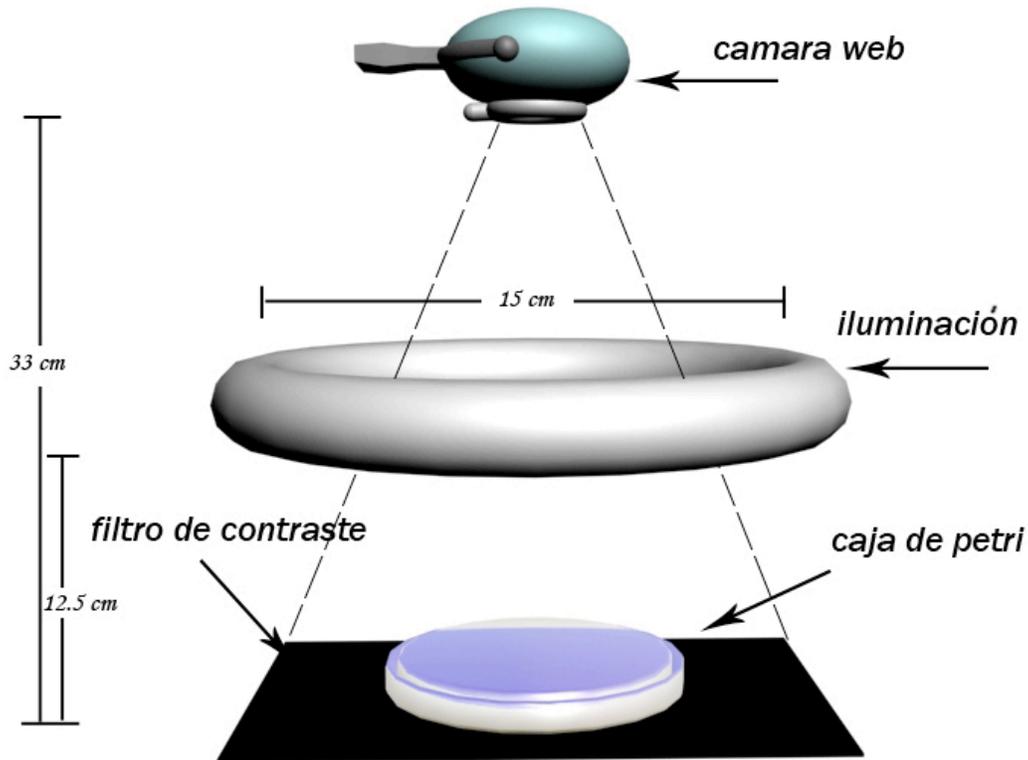


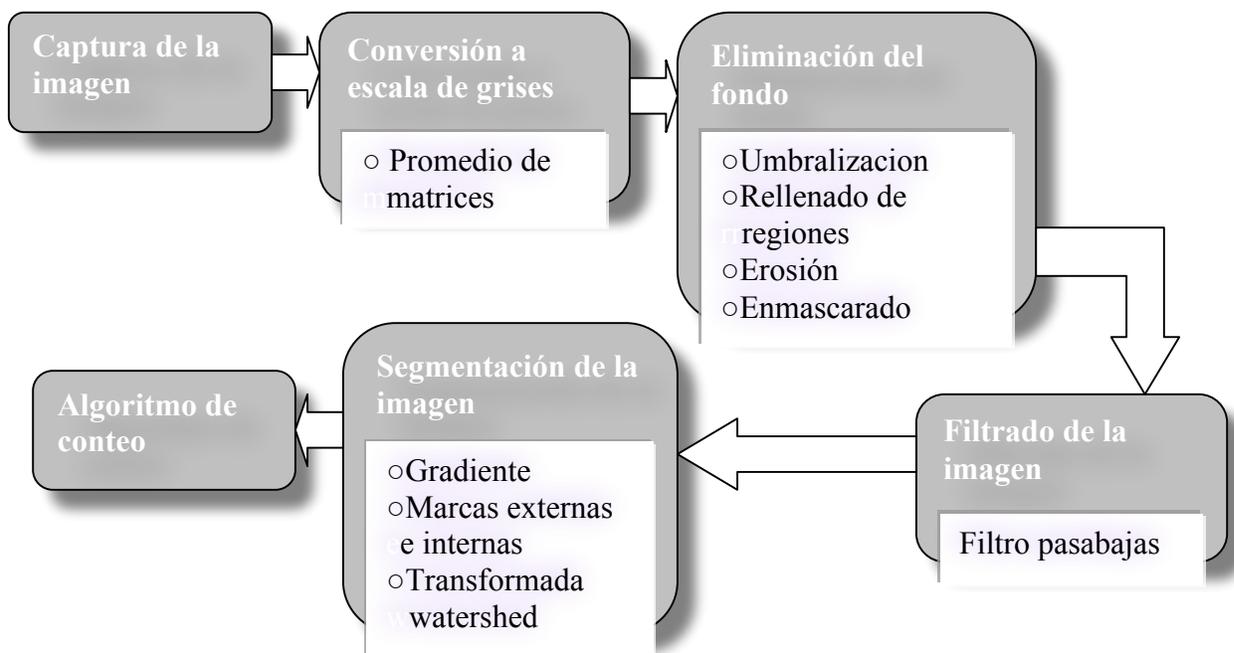
Figura 3.1 Esquema del sistema de adquisición de imágenes.

## 3.2 Software

El desarrollo de algoritmos se realizará usando el toolbox de procesamiento de imágenes de Matlab, el cual nos permitirá diseñar y probar un algoritmo en la imagen que de no ser el que mejor resultados nos da, podemos cambiarlo por otro sin la necesidad de tener que deducir muy a fondo su funcionamiento.

### 3.2.1 Desarrollo de los algoritmos

Los algoritmos que a continuación se presentan son los utilizados para el desarrollo de esta investigación, y que llevó a dar resultados exitosos en una prueba inicial hecha a una sola fotografía. Haciendo uso de las funciones preestablecidas en el *toolbox* de procesamiento de imágenes de Matlab, y en algunas ocasiones utilizando programación propia de Matlab. El siguiente diagrama a bloques ilustra el proceso completo así como los algoritmos implementados.



#### 3.2.1.1 Captura de imagen

La captura se realiza mediante el software propio que viene con la cámara a una resolución máxima de 640x480 píxeles, la cual se almacena en formato BMP, esto se realiza así debido a que no se cuenta con un control active-x que nos permita controlar la cámara mediante Matlab, posteriormente se guarda la fotografía en el directorio a trabajar.

Ya desde Matlab se selecciona el directorio, donde está la imagen y lo que sigue es almacenar esta imagen en una variable, la cual nos representará la imagen como una matriz de 3 dimensiones en donde cada una de ellas nos representa la intensidad en escala de grises de los colores Rojo, Verde y Azul que contiene la imagen, esto se realiza mediante la siguiente sintaxis:

```
a = imread('imagen.bmp');    % Almacena imagen en variable "a"  
figure, imshow(a);          % Muestra la figura
```

**Figura 3.2 Código de captura de imagen.**

La segunda línea es sólo para mostrar la imagen (figura 3.3)



**Figura 3.3 Imagen a color.**

### 3.2.1.2 Conversión a escala de grises

Los algoritmos que vamos a aplicar requieren que tengamos nuestra imagen en escala de grises, debido a esto, es preciso obtener un promedio de las tres matrices, rojo, verde y azul. El primer paso consiste en separar las matrices lo cual se realiza asignando en nuevas variables cada una de las matrices (figura 3.4).

```
R = a(:,:,1);    % Seleccionamos la matriz de rojos
B = a(:,:,2);    % Seleccionamos la matriz de azules
G = a(:,:,3);    % Seleccionamos la matriz de verdes
```

**Figura 3.4 Código para separar las matrices.**

Como nuestras matrices están representadas como números enteros (*uint8*) que van de 0 a 255, necesitamos transformarlas a un tipo de valores que puedan manejar números de coma flotante y así poder obtener un promedio. La función “double()” nos permite realizarlo (figura 3.5).

```
Rd = double(R);  % Nueva matriz en coma flotante
Gd = double(G);  % Nueva matriz en coma flotante
Bd = double(B);  % Nueva matriz en coma flotante
```

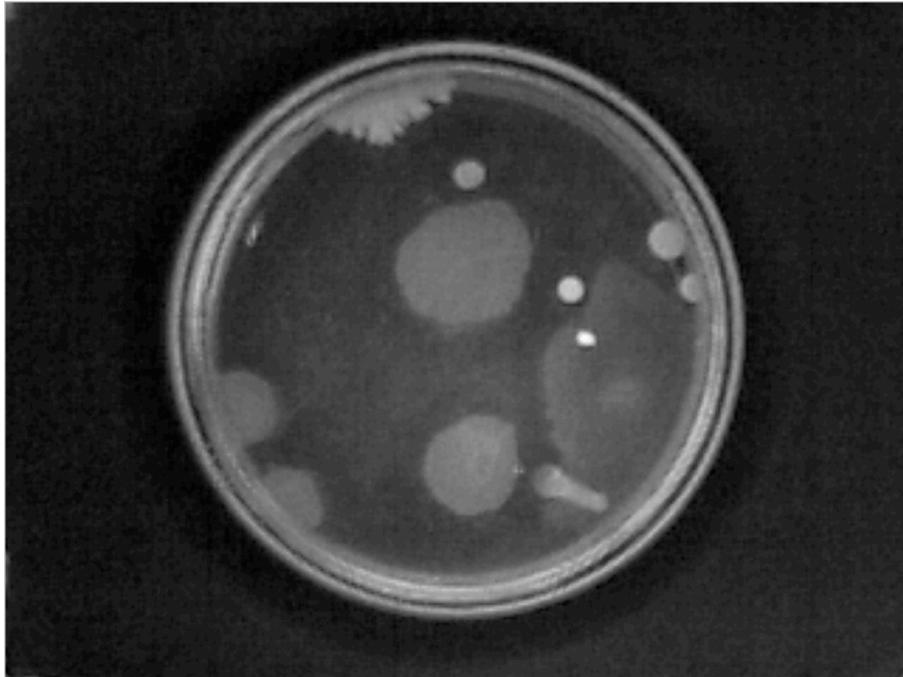
**Figura 3.5 Código para pasar las matrices a coma flotante.**

Una vez transformadas las matrices con valores flotantes podemos entonces sacar un promedio multiplicando por valores fraccionarios preestablecidos y sumando sus matrices resultantes, como son matrices, la multiplicación debe ser por “producto punto”. El resultado de la operación es almacenado en la variable “gray” y mostrado en la figura 3.7.

Los valores por los cuales se multiplicaron las matrices no obedecen a ninguna cantidad estándar preestablecida, por simplificación se eligieron estas cantidades y se le da una mayor ponderación a la matriz de verdes por que el ojo humano capta con mayor intensidad los tonos de verdes.

```
gray = 0.3*Rd+0.3*Bd+0.4*Gd;    % Promedio de las matrices  
figure,imshow(gray, [ ]);        % Muestra la figura
```

**Figura 3.6 Código para promediar las tres matrices.**



**Figura 3.7 Imagen en escala de grises.**

### 3.2.1.3 Eliminación del fondo

Con la imagen en escala de grises, el siguiente paso consiste en retirar todo aquello que no necesitamos en la imagen, para esto nos basamos en el brillo exterior de la caja de petri, con este brillo podemos crear una máscara por medio de detección de umbral. Mandamos aquellos píxeles que tengan valores menores de 80 a un valor 0 (color negro) y aquellos que sean mayores los hacemos 1 (blanco), de esta forma obtenemos una imagen binaria (figura 3.8).

Como se puede apreciar en la figura 3.9, tenemos una imagen poco coherente respecto a lo que estamos haciendo, pero lo que realmente nos importa es que la forma en la que quedó se aprecia una circunferencia perfecta alrededor, ocasionada por el borde exterior de la caja de Petri, los píxeles en color negro que quedaron de manera esparcida dentro de la circunferencia son debidos a que el centro también existen regiones igual de oscuras a las que tiene el fondo.

```
[ren,col] = size(gray)           %Dimensiones de nuestra matriz
molde = zeros(ren,col);         % Creamos una matriz vacía
for j = 1:ren,                  % Ciclo for el cual se encarga de
    for i = 1:col,              % barrer la matriz de la imagen
        if gray(j,i) > 80 & gray(j,i) < 255 % Para cada pixel mayor de 80
            molde(j,i) = 1;           % ese mismo pixel en molde valdrá 1
        else molde(j,i) = 0;         % y si no sera 0
        end
    end
end
end
figure, imshow(molde);
```

Figura 3.8 Código para generar una imagen con umbral.



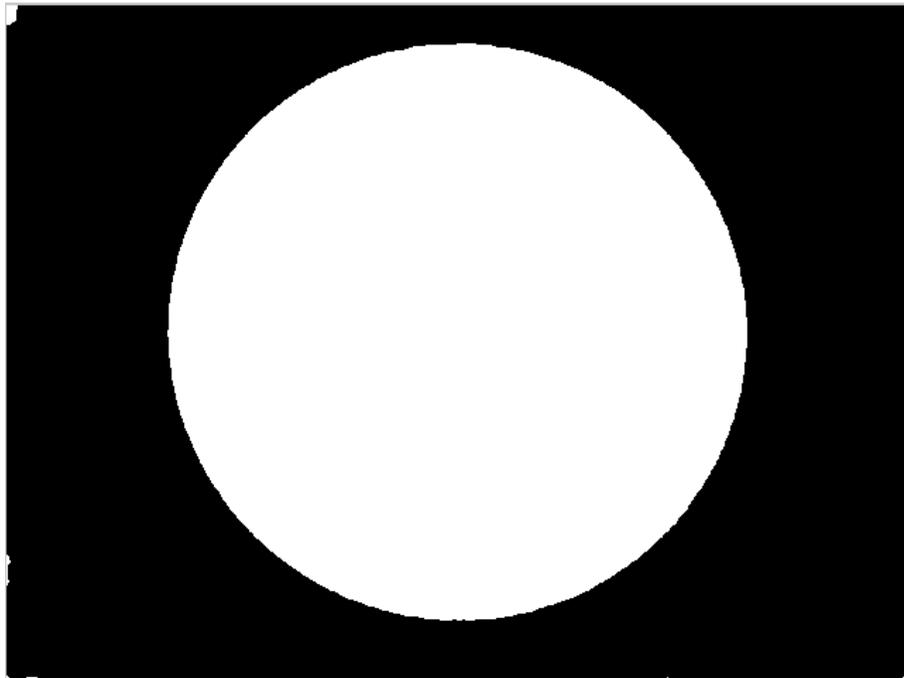
Figura 3.9 Imagen con umbral

Para que la imagen nos sirva de molde, tenemos que convertir los píxeles en negro que están encerrados dentro de la circunferencia de píxeles blancos, esto lo logramos en MATLAB escribiendo una de las funciones que se incluyen dentro del toolbox de procesamiento de imágenes (figura 3.10).

```
molde2 = imfill(molde, 'holes');           % Rellenar "hoyos" en la figura  
figure, imshow(molde2);                   % Muestra la figura
```

**Figura 3.10** Código para el relleno de regiones.

La función se llama "imfill" y le indicamos la imagen a procesar, además del tipo de relleno que le haremos *holes* (figura 3.11).



**Figura 3.11** Máscara rellena.

El molde queda a la medida de la caja de Petri, mas sin embargo aparte del fondo es necesario retirar los bordes de la caja de petri que producen un brillo similar al de las colonias, el cual debemos eliminar.

Para solucionar este problema tenemos que hacer el molde más pequeño en la misma forma aplicando un proceso de erosión, el cual consiste en ir restando la parte de color blanca de nuestra imagen original con otra imagen de la misma forma, con un radio de acuerdo a la cantidad que queremos erosionar. Con la función *strel* creamos la matriz con la que vamos a erosionar (figura 3.12).

```
se = strel('disk', 34);      % Elemento de erosión tipo disco y radio 34
```

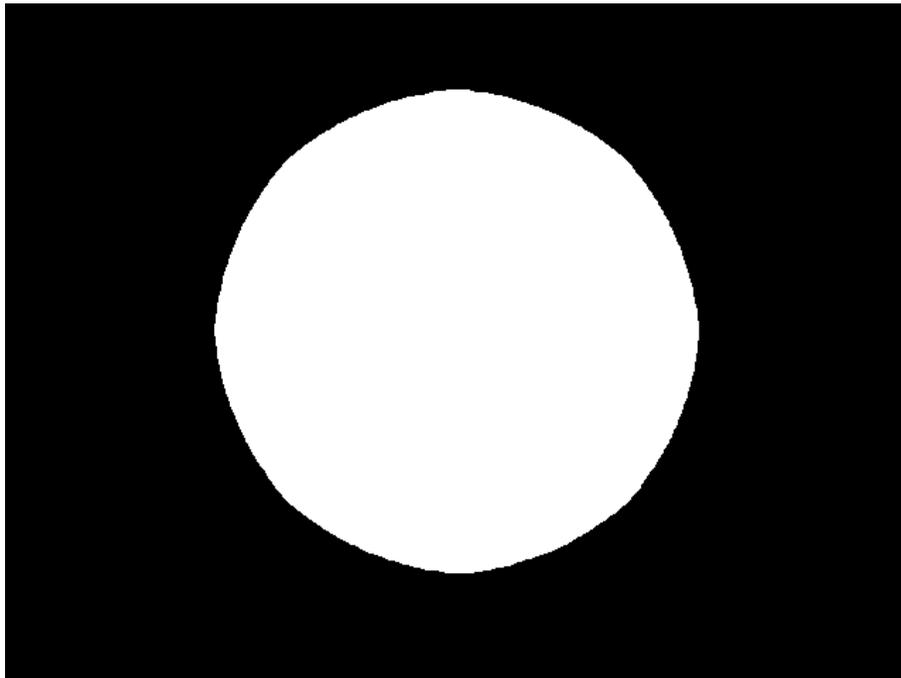
**Figura 3.12 Código para generar una matriz con forma circular de 1's.**

Los parámetros que incluye esta función es el tipo de matriz que usaremos que en este caso es en forma de disco “disk” y con que radio se creará, “34”. Usando la función “imerode”, aplicaremos la erosión (figura 3.13).

```
moldef = imerode(molde2, se);      % Erosionamos la imagen  
figure, imshow(moldef);           % Muestra la figura
```

**Figura 3.13 Código para erosionar la imagen umbralizada.**

La figura del molde la almacenamos en la variable “moldef” (figura 3.14)

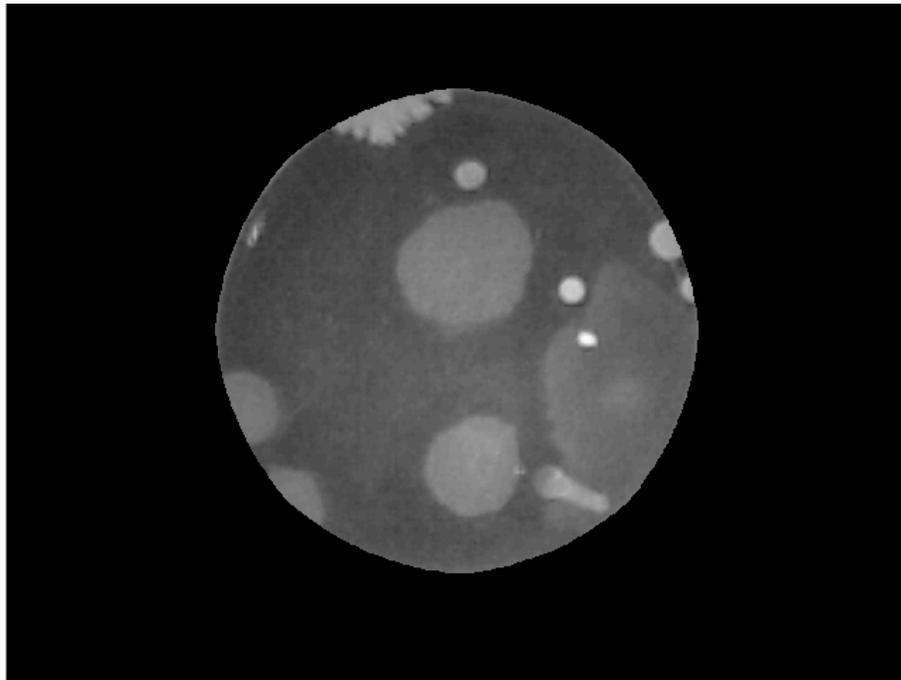


**Figura 3.14** Máscara erosionada.

Con esto obtenemos una máscara del tamaño deseado, la cual nos permitirá obtener en la imagen sólo el área en donde se encuentran las colonias. Como no hay una función en MATLAB que nos permita la aplicación de la máscara sobre la imagen en escala de grises, este proceso se deberá realizar aplicando un algoritmo que lleve en el proceso las dos matrices (figura 3.15), usaremos la matriz almacenada en molde para ir preguntando, “si el píxel en el que se encuentra es de color negro entonces vuelve negro el píxel con la mismas coordenadas pero en la imagen en escala de grises, de lo contrario si es de color blanco en la imagen *moldef*, entonces deja como estaba el píxel en la misma posición de la imagen en escala de grises”. El resultado se muestra en la figura 3.16.

```
for j = 1:ren, % Ciclo for encargado de recorrer la
  for i = 1:col, % imagen
    if molde(j,i) == 0 % Si el píxel del molde es negro has ese
      gray(j,i) = 1; % mismo píxel en la imagen gray negro
    else gray(j,i) = gray(j,i); % si no déjalo de su mismo valor
    end
  end
end
end
figure, imshow(gray, [ ]);
```

**Figura 3.15** Código para aplicar el molde a la imagen en escala de grises.



**Figura 3.16** Imagen enmascarada.

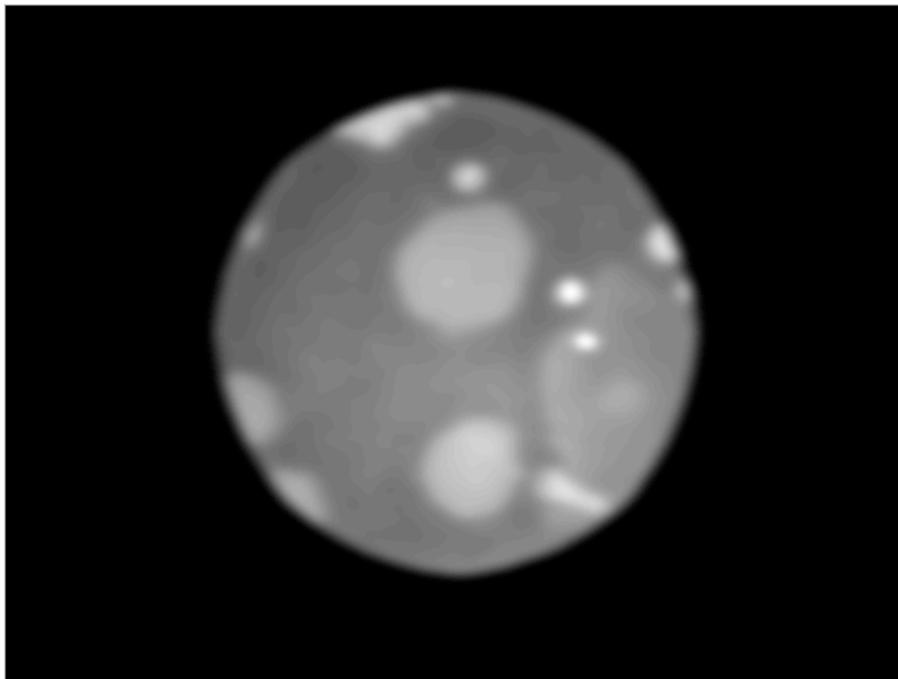
#### 3.2.1.4 Filtrado de la imagen

El siguiente paso en el proceso es filtrar la imagen con el propósito de eliminar píxeles aislados causantes de ruido. El filtro es del tipo pasabajas y se basa en la transformada rápida de Fourier discreta. Como resultado obtenemos una imagen

(figura 3.18) similar a un desenfoco ocasionado por la cámara, a este efecto lo llamaremos suavizado de imagen ya que si bien perdemos definición en los bordes de las colonias mejoramos en la uniformidad de las regiones, que al fin de cuentas lo que intentamos es contar regiones similares.

```
PQ = paddedsize(size(gray));           % Matriz vacia
D0 = 0.04*PQ(1);                       % Intensidad del filtro
H = lpfilter('gaussian', PQ(1), PQ(2), D0); % Filtro pasabajas tipo gaussiano
g = dftfilt(gray, H);                   % Transformada rápida de Fourier
figure, imshow(g, [ ]);                 % Muestra la imagen
```

**Figura 3.17 Código para suavizar la imagen.**



**Figura 3.18 Imagen suavizada.**

### 3.2.1.5 Segmentación de la imagen

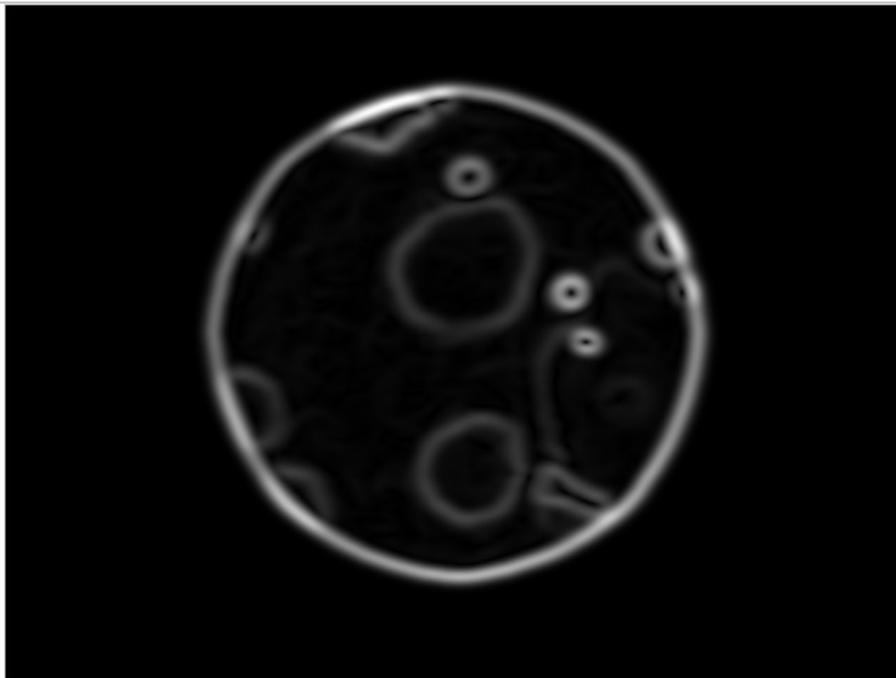
Segmentar una imagen es dividirla en regiones, y las regiones que para nosotros son de interés son las formadas por las colonias. La manera en la que se aplique la

transformada *watershed* puede variar según el objetivo, y el tipo de imagen que se tenga, para nuestro caso será de utilidad en la identificación de la forma que posean las colonias dentro de la imagen.

El primer paso consiste en sacar el gradiente de nuestra imagen (figura 3.19), el cual nos mostrará una imagen con los cambios de niveles de grises presentes en los bordes de cualquier región (figura 3.20).

```
h = fspecial('sobel'); %mascara para filtrado espacial
g1 = sqrt(imfilter(g, h, 'replicate').^2 + imfilter(g, h, 'replicate').^2); %raiz
%cuadrada del filtro de la imagen con la mascara
```

**Figura 3.19 Código para generar el gradiente de la imagen.**

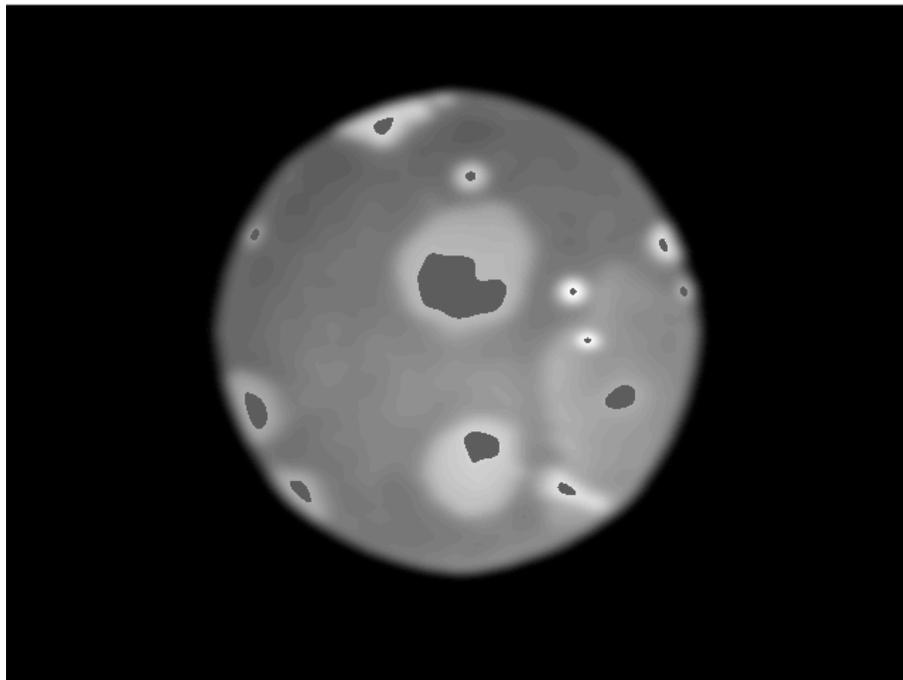


**Figura 3.20 Gradiente de la imagen.**

El segundo paso es identificar marcas internas en las regiones de interés, como las colonias son de un tono claro entonces nos enfocamos en las regiones máximas. (figura 3.22).

```
im = imextendedmax(g, 10);    %Localización de regiones máximas
fim = g;                      %sobre imposición de imágenes
fim(im) = 175;                %solo para visualización
figure, imshow(fim, [ ]);     %Muestra la imagen
```

**Figura 3.21** Código para la detección de marcas internas.

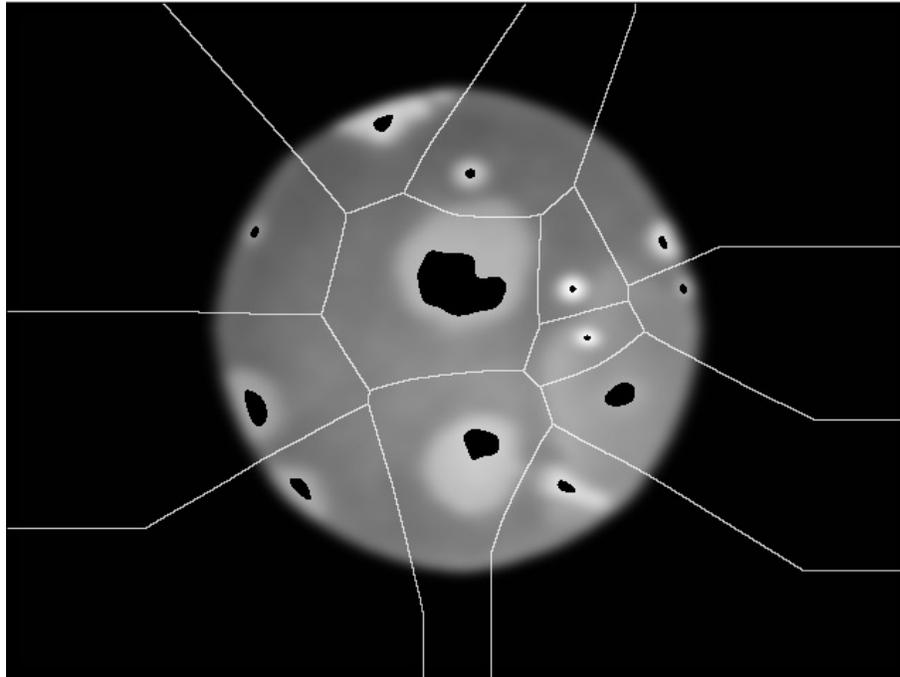


**Figura 3.22** Marcas internas detectadas.

Ya con las marcas localizadas utilizaremos transformada *watershed* por primera vez para encontrar las marcas externas, esto lo haremos únicamente utilizando la imagen binaria con las marcas. Aplicamos transformada *watershed* sobre la transformada de distancia, la cual es muy sencilla, su concepto es la distancia de cada píxel a su más cercano píxel de valor no cero, en una imagen binaria (figura 3.24).

```
lim = watershed(bwdist(im)); %Transformada watershed  
em = lim == 0; %Se invierte el color para visualizacion  
figure, imshow(em); %Muestra la imagen
```

**Figura 3.23 Código para detectar marcas externas.**



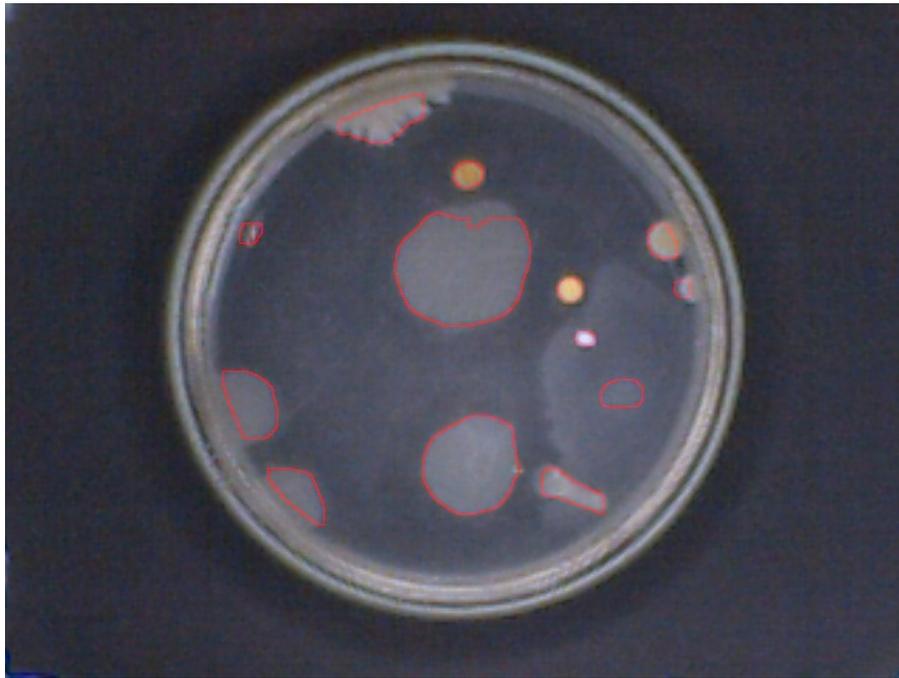
**Figura 3.24 Imagen con las marcas internas y externas.**

Aplicamos de nueva cuenta transformada *watershed*, pero esta vez al gradiente de la imagen limitada por las marcas externas y usando como base de inundación las marcas internas (figura 3.25).

```
g2 = imimposemin(g1, im | em); %Unimos marcas maximas y minimas  
L2 = watershed(g2); %Aplicación de la transformada watershed  
figure, imshow(L2); %Muestra la figura
```

**Figura 3.25 Código de segmentación final.**

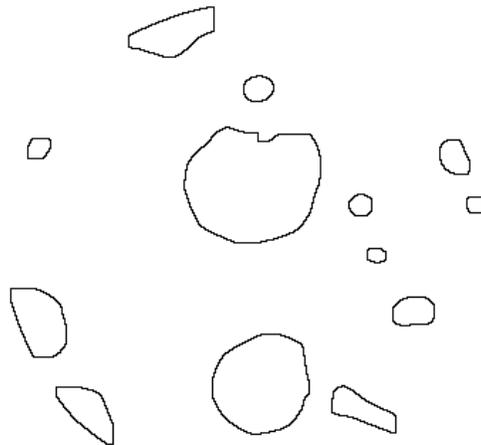
Observamos en la figura 3.26 como se nos muestran las líneas *watershed* finales sobrepuestas con la imagen original a color, de tal manera que nos podemos dar cuenta fácilmente que las colonias quedaron perfectamente identificadas.



**Figura 3.26 Imagen original segmentada mediante transformada *watershed*.**

#### 3.2.1.6 Algoritmo de conteo

El proceso de segmentación nos regresa una imagen binaria (figura 3.27) en donde el contorno de las colonias queda representado por líneas en color negro y un fondo color blanco. Con la imagen final totalmente lista, lo único que resta es aplicar un algoritmo el cual nos indique cuantos contornos color negro existen, el razonamiento del proceso es el siguiente.



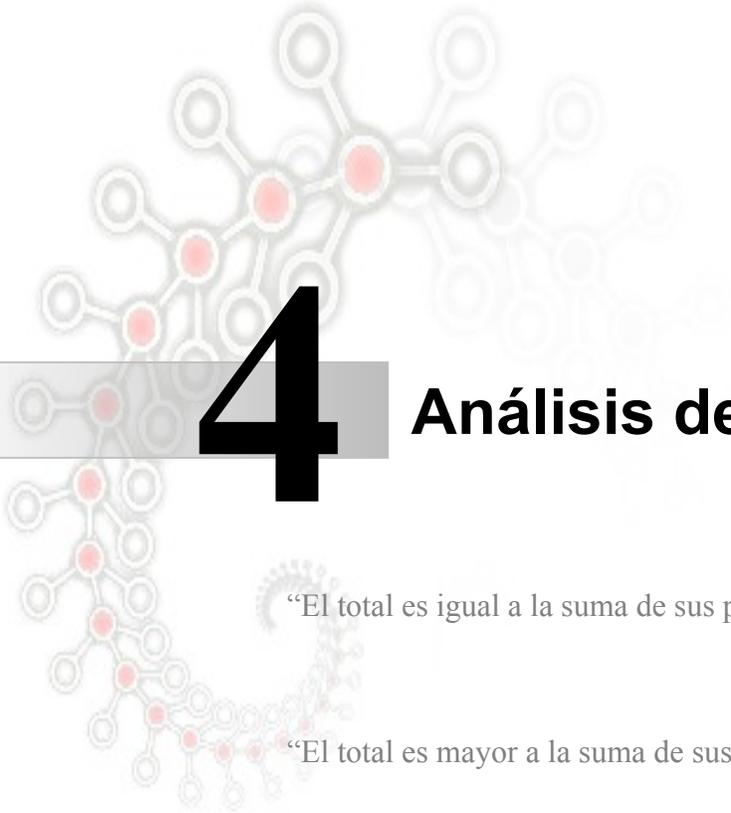
**Figura 3.27 Imagen lista para aplicar el algoritmo de conteo.**

Cada contorno mostrado en la imagen representa un conjunto conectado, por simplicidad decimos que su conectividad es de “8-pixeles vecinos”, de esta manera que a cada píxel origen vamos buscando su siguiente vecino y etiquetándolo como parte de un mismo conjunto, si no tiene vecinos entonces deducimos que es parte de otro conjunto conectado.

```
[L2c, num] = bwlabel(L2);    %Conteo de conjuntos conectados  
num;
```

**Figura 3.28 Código para contar el número de colonias**

La variable “L2c” almacena una matriz con los conjuntos etiquetados con su número correspondiente, mientras que en la variable “num” nos entrega el número final de contornos que representa el número de colonias en la imagen.



# 4 **Análisis de resultados**

“El total es igual a la suma de sus partes”

-- Euclides --

“El total es mayor a la suma de sus partes”

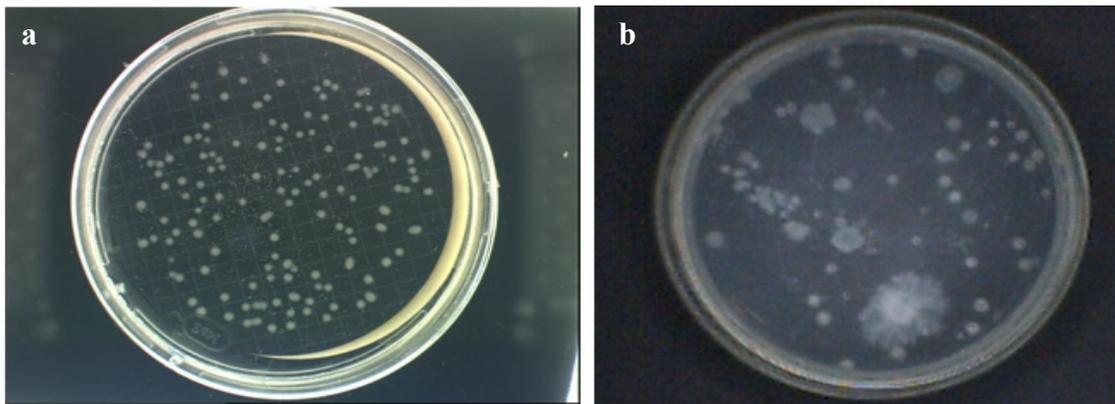
-- Max Wertheimer --

Los resultados obtenidos serán divididos tanto en hardware como en software, en lo referente al hardware nos enfocamos al comportamiento que tuvo el sensor utilizado con la fuente de iluminación y el objeto. En cuanto a la parte software, la efectividad del algoritmo desarrollado y como fue puesto a prueba en diferentes tipos de imágenes serán los principales puntos a reflejar en los resultados obtenidos.

## 4.1 Sensor

En lo referente al sensor utilizado, el cual fue una *webcam* iremos describiendo en este apartado el comportamiento que éste presentó ante las diferentes condiciones de iluminación del objeto y como fue que éste mostraba los resultados mas adecuados para la realización de este proyecto

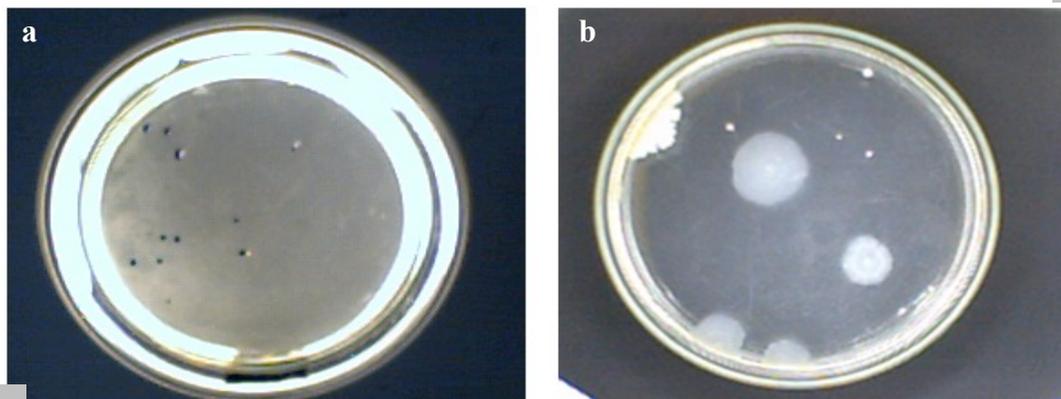
La figura 4.1 nos ilustra la diferencia en la calidad de las imágenes que se obtenían con la *webcam* y una cámara científica para este uso específico. Se puede observar fácilmente la diferencia en cuanto a calidad. Muy aparte de la resolución ya que la de la imagen de la cámara científica apenas era del doble.



**Figura 4.1** Imágenes con diferentes tipos de cámaras (a) Imagen obtenida de una cámara científica. (b) Imagen obtenida con una *webcam*.

#### 4.1.1 Tipo de iluminación

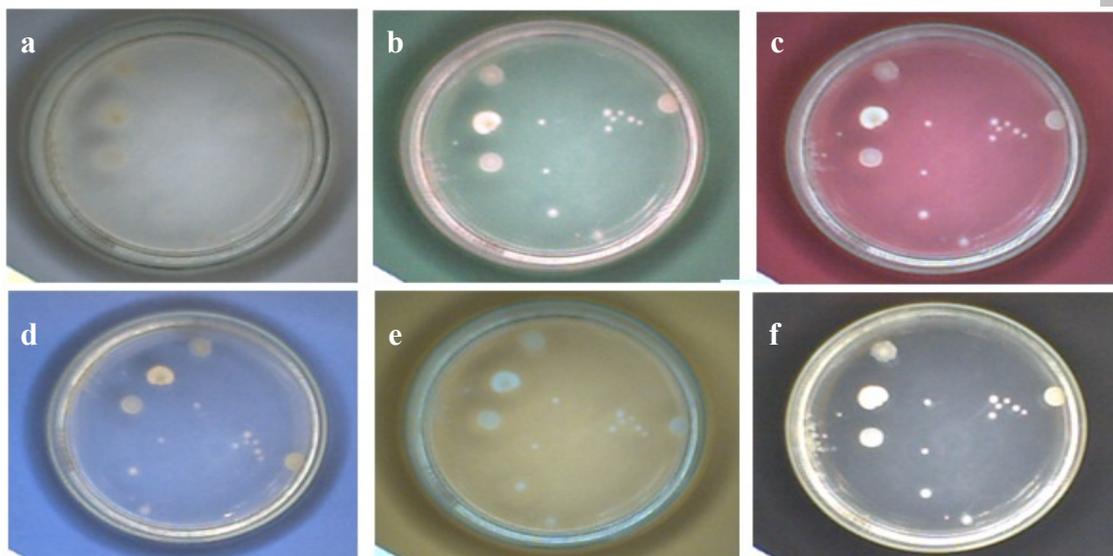
El tipo de iluminación final escogido fue el de tipo *Frontlight* el cual ubica la fuente de luz entre el objeto y el sensor, más sin embargo, se realizaron pruebas con el tipo de iluminación que implementaban los contadores manuales ya existentes la cual es de tipo *Backlight* que ubica la fuente de luz por debajo del objeto. La figura 4.2 nos muestra la diferencia existente entre los dos tipos de iluminación y como se nota una disminución de brillo en los bordes de la caja de Petri, razón principal por la cual el primer tipo de iluminación fue el escogido como definitivo. Al ser el brillo de un color blanco, éste se podía confundir fácilmente con una colonia o ser tan intenso que no nos dejaría distinguir cuando una colonia estuviera presente en las orillas de la caja de Petri.



**Figura 4.2** Caja de Petri iluminada(a) Iluminación *Backlight*. (b) Iluminación *Frontlight*.

#### 4.1.2 Filtro de contraste

Al ser el vidrio un material traslucido es necesario tener un medio que contraste las colonias que son de color claro.



**Figura 4.3** Colores de contraste (a) blanco, (b) verde, (c) rojo, (d) azul, (e) amarillo, (f) negro.

Se realizaron pruebas con diferentes colores los cuales van desde el color blanco (el cual presentó los mejores resultados en cuanto a la dispersión de luz en la caja de Petri pues era casi nulo el brillo presentado, más sin embargo el contraste entre las colonias era también casi nulo) hasta el color negro (fue el que mejores resultados dio en cuanto al contraste con las colonias pero también el que más problemas resultó tener en cuanto al brillo generado en los bordes de la caja de Petri). La figura 4.3 muestra diferentes colores de contraste utilizados para realizar las pruebas. El resto de los colores presentan resultados muy similares entre sí. Debido a que el procesamiento se realizó sobre imágenes en escala de grises, fueron muy poco relevantes los cambios en el contraste de las colonias con el fondo.

#### 4.1.3 *Intensidad de luz*

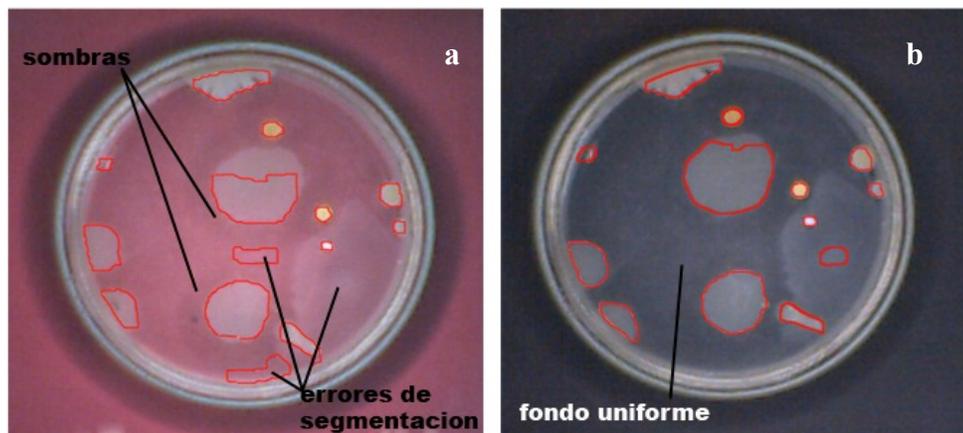
Teniendo un correcto control en cuanto al nivel de la intensidad de la luz incidente sobre el objeto, se puede usar el color negro como medio de contraste y obtener una imagen con un buen contraste y un nivel de brillo aceptable en los bordes de la caja de Petri. Sin embargo la cantidad de luz utilizada influye en la calidad de imagen obtenida por el sensor, más allá de tener una imagen más oscura o más clara, al tener poca luz la cámara *web* tiende a generar más ruido en la imagen. Ruido en la imagen significa pérdida de detalle y posibles errores al momento del procesamiento de la imagen.

El nivel de intensidad de luz obtenido para los mejores resultados de contraste-brillo, no resultó ser el mismo de manera uniforme para todas las imágenes obtenidas, debido a que en las pruebas realizadas se mostró que al tener más detalles (número de colonias) en la imagen se perdía calidad y nitidez. Además las cajas de Petri presentaban tonalidades de color verde que no era uniforme, ya que algunas eran más opacas que otras lo que ocasionaba imágenes con mayor o menor calidad de iluminación.

## 4.2 Algoritmo

### 4.2.1 Prueba de contraste

El desarrollo del algoritmo se hizo teniendo ya establecido el tipo de iluminación *Frontlight* y filtro de contraste color negro. Hasta este punto los resultados en cuanto a la calidad de la imagen obtenida por la cámara fueron juzgados únicamente por la apreciación de la visión humana, pero debido a que una máquina sería la encargada de juzgar la imagen, se realizaron pruebas con un fondo de color ligeramente más claro, como el rojo. Pues no necesariamente una buena imagen para el ojo humano es una buena imagen para una máquina.

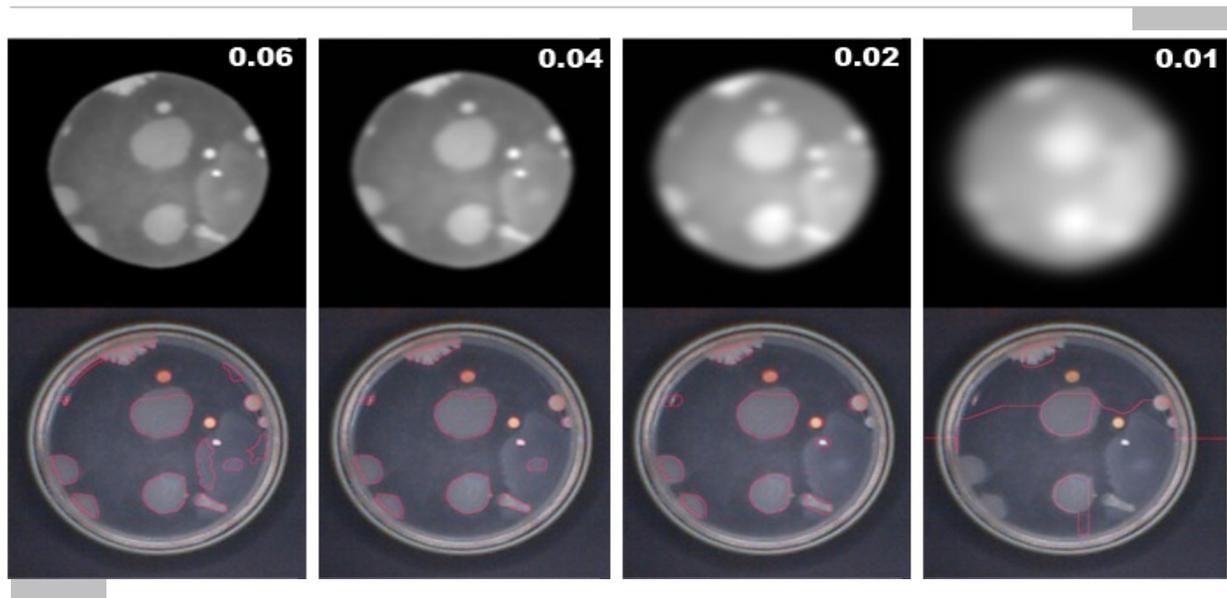


**Figura 4.4 Prueba de contraste. (a) Imagen con fondo color rojo. (b) Imagen con fondo color negro.**

Observamos como en la imagen con el fondo color rojo (figura 4.4) se forman sombras alrededor de las colonias, lo que ocasiona que el fondo no sea de un color uniforme y se presenten brillos ocasionales que provocan una mala segmentación. Esto no sucede con un fondo color negro pues las sombras al también ser oscuras no se distinguen obteniéndose así un fondo uniforme.

### 4.2.2 Intensidad del filtro

Una de las variables contenidas dentro del código del algoritmo se encontraba en la intensidad del filtro de suavizado, encargado de dar una imagen con menos ruido a cambio de una pérdida de nitidez en la imagen, la figura 4.5 nos muestra una imagen segmentada con diferentes grados de intensidad del filtro.



**Figura 4.5** Cuatro diferentes grados de intensidad en el filtro de suavizado.

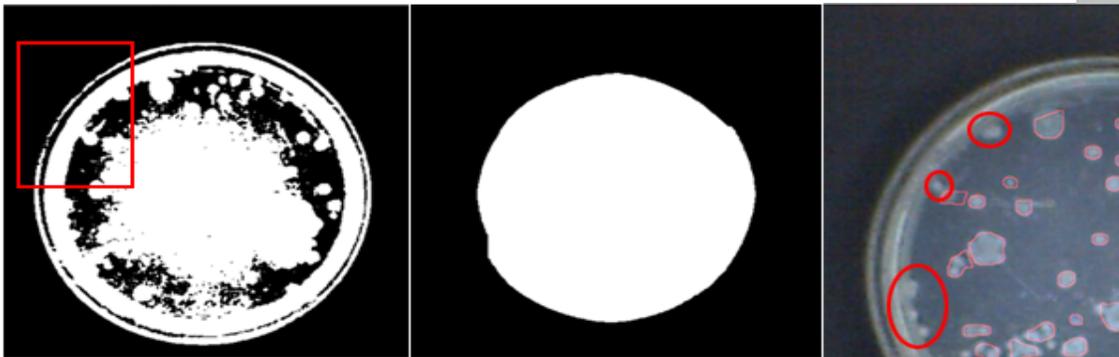
Podemos observar que si vamos aumentando la intensidad, las colonias de menor tamaño no son identificadas al momento de realizar la segmentación, por el contrario, al disminuir la intensidad el ruido en la imagen causa sobre segmentación. Podemos concluir que existe un límite con el tamaño mínimo de las colonias, aquellas que sean de un tamaño muy reducido no podrán ser identificadas debido a que el filtro encargado de suavizar la imagen puede llegar a difuminarlas junto con el fondo.

### 4.2.3 Pruebas de funcionamiento

Una vez mostrados resultados satisfactorios con la imagen de prueba anterior se procedió a realizar pruebas con tres diferentes imágenes que presentaban formación de colonias mucho más numerosas y de mayor variedad en forma y color. En estas pruebas se evidenció diversas fallas que poseía el algoritmo respecto principalmente con la iluminación y la calidad de las imágenes

#### 4.2.3.1 Error por detección de umbral

Parte fundamental del correcto funcionamiento del algoritmo es la eliminación de lo que no es necesario, el fondo y los bordes de la caja de Petri. Esta parte se basó en el brillo generado por los bordes de la caja de Petri (previamente explicado en el capítulo anterior), cuando la iluminación no era la correcta, este brillo tenía variaciones y presentaba irregularidades alrededor de la circunferencia, lo cual ocasionaba una mala detección de umbral (figura 4.6(a)), y por lo tanto un mal resultado al formar la parte correspondiente al molde (figura 4.7(b)), lo que a su vez causa que zonas de la imagen a las que se debería aplicar el algoritmo de conteo, quedarán sin un correcto procesado (figura 4.6(c)).

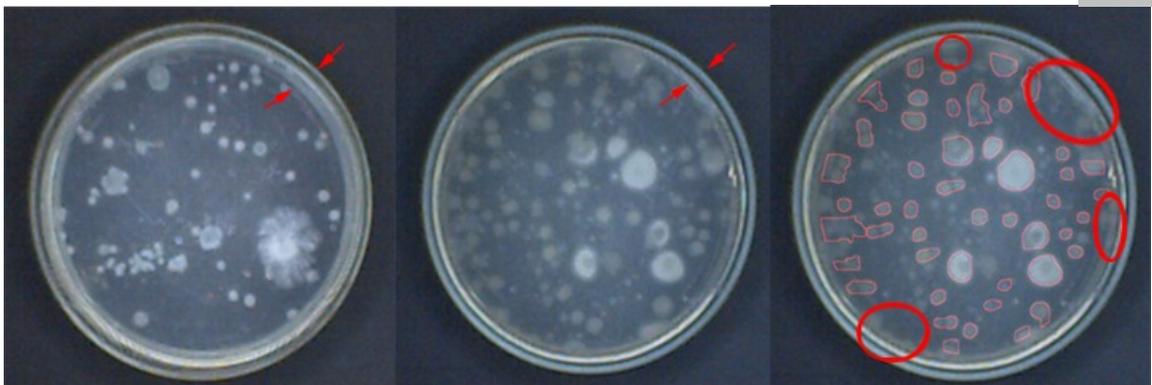


**Figura 4.6 Error por detección de umbral. (a) umbral incorrecto, (b) mala formación del molde, (c) colonias sin segmentar.**

#### 4.2.3.2 Error por erosión

Mediante detección de umbral y rellenado de regiones logramos eliminar el fondo, pero para poder eliminar el borde de la caja de Petri se tiene que hacer el molde un poco más pequeño mediante erosión. Establecer una cantidad de píxeles que se deban eliminar se realiza mediante previa inspección de la imagen a simple vista, culminando con un ensayo de prueba y error. Pero aunque las cajas de Petri poseen un diámetro estándar, no es así en su construcción ya que algunas poseen un borde más redondeado que otras, lo que provoca cambios en el brillo del borde exterior y una mayor área en la que las colonias pueden ser contadas es discriminada por una erosión excesiva.

En la figura 4.7(a) podemos ver el tipo de caja de Petri que se usó para desarrollar el algoritmo y como es el tipo de brillo que produce y debemos eliminar, mientras que en la figura 4.7(b) vemos como el brillo es más angosto y bien podría necesitar de una erosión menor. La figura 4.7(c) muestra el resultado de la segmentación la cual discrimina colonias en los extremos de la caja de Petri.

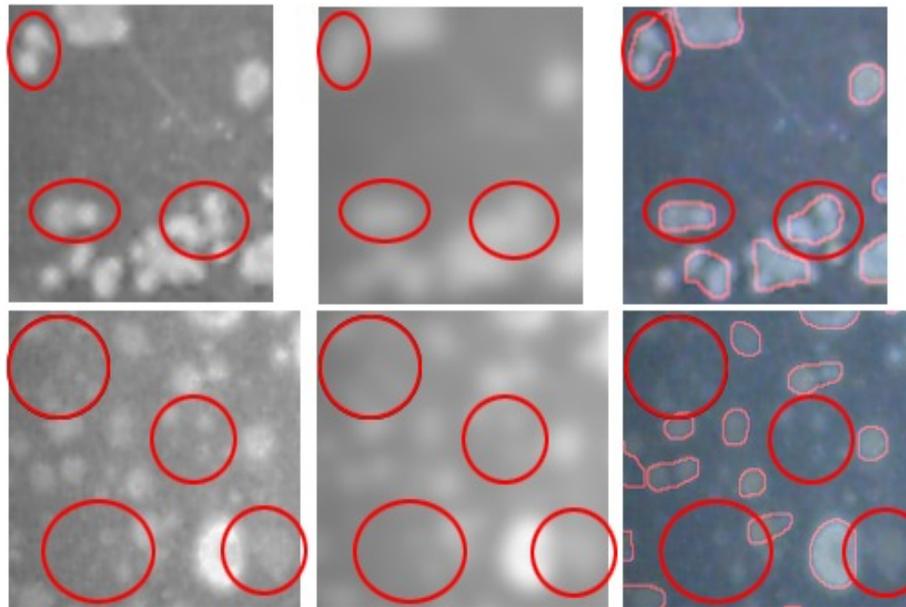


**Figura 4.7 Error por erosión. (a) Petri usado para desarrollar el algoritmo, (b) Petri con borde menos redondeados, (c) colonias sin segmentar.**

#### 4.2.3.3 Error por filtrado

Previamente en la sección 4.2.2 se describió como el filtro de suavizado en la imagen nos causaba una imposibilidad de contar colonias más pequeñas, aparte de lo anterior en las pruebas realizadas demostró tener más fallos, uno de ellos se presentó debido a la cercanía de dos o más colonias, pues no se realizaba una correcta segmentación al tomarlas como una sola colonia. El segundo caso se dio debido en parte a la mala iluminación, ya que colonias de un tono más oscuro estaban poco contrastadas, al momento de aplicar el filtro éste ocasionaba que estas colonias se difuminaran con el fondo y no fueran segmentadas. En la figura 4.8 podemos ver ambos errores tanto por cercanía de las colonias (a) como por poco contraste (b), se muestra en círculos rojos como al aplicar el filtro éste ocasiona que las colonias sean menos imperceptibles.

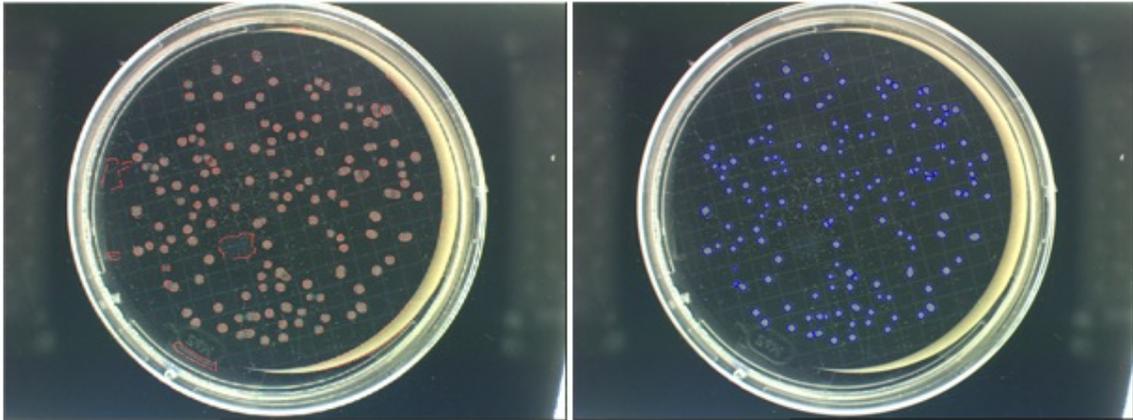
Cada uno de los errores antes mencionados tuvo solución al variar los valores de las variables de intensidad del filtro de suavizado, valor de umbral y erosión, pero cada uno era diferente para cada tipo de imágenes lo que implicaría en un futuro tener que trabajar con variables que puedan ser manipuladas por el usuario, lo que resultaría incomodo e impráctico, pues sería un proceso de ensayo y error.



**Figura 4.8** Error por filtrado. (a) Colonias muy cercanas, (b) colonias con poco contraste.

#### 4.2.4 Pruebas con imágenes de alta calidad

Las pruebas de funcionamiento también se pudieron realizar con una imagen de alta calidad tomada de un ejemplo que se mostraba en una página comercial. La imagen presentaba iluminación *Frontlight* y un fondo de contraste color negro, lo cual es exactamente la forma en la que se tomaron las demás imágenes, pero con la diferencia que ésta es tomada por una cámara científica. Podemos observar en la figura 4.9 que el algoritmo funciona pero aun persisten los errores anteriormente mencionados.



**Figura 4.9** Imágenes de alta calidad. (a) Imagen segmentada por transformada *watershed*, (b) imagen segmentada con lógica difusa.



# 5 Conclusiones

“No se fracasa si se tiene un comienzo”

-- Anónimo --

En el último capítulo de este trabajo se plantea las conclusiones a las cuales se llegaron durante la realización de esta investigación. Las conclusiones abarcarán cada uno de los temas tratados dividiendo en una parte teórica, comenzando con la introducción y una parte práctica, finalizando con lo que sería la parte más fuerte de las conclusiones, el análisis de los resultados.

## **5.1 Conclusiones obtenidas en la parte teórica**

Las conclusiones acerca de la necesidad que se tiene en el laboratorio de microbiología de contar con un equipo de conteo automático de colonias, se enfocan más acerca del valor económico que éstos poseen actualmente en el mercado. Su elevado costo se debe principalmente a que el desarrollo de la tecnología de procesamiento de imágenes es aún difícil, pues se debe tener a un experto no sólo en el campo de procesamiento de imágenes si no también en programación, iluminación y además alguien que tenga amplio conocimiento del problema a solucionar. Aunado a estos puntos anteriores el costo de las cámaras científicas es bastante elevado.

Seguir desarrollando esta investigación hasta que se pudiera crear un prototipo funcional no sólo pondría el camino para solventar el problema que se tiene en el laboratorio de microbiología, si no que además en el proceso de su desarrollo se conseguirá dar pauta para desarrollar más aplicaciones que requieran la tecnología que conlleva el procesamiento de imágenes. Subsecuentes trabajos aunque difieran en aplicación igualmente en el apartado del marco teórico serán de gran utilidad como base de datos para comprender más cosas sobre este tema. Es aquí donde cobra importancia este trabajo, así como otros similares que ya se han hecho.

El uso del paquete computacional Matlab demostró de ser de gran utilidad, pues es una manera rápida de probar teorías iniciales sobre como atacar un problema sin tener que irnos más a fondo. El seguir trabajando con este programa ayudaría en mucho a los subsecuentes proyectos, además de tener una mayor comprensión en cuanto a su *toolbox* de procesamiento de imágenes y al futuro entendimiento de cómo funcionan los comandos ya programados para el tratamiento específico de imágenes.

## 5.2 Conclusiones obtenidas en la parte práctica

Se pudo observar con los análisis en la anterior sección que gran parte de los errores se debieron principalmente a la mala calidad del sensor sumado a la mala iluminación. Si bien el algoritmo desarrollado funciona en una primera imagen con un número de colonias reducidas, esto no fue así cuando se sometieron imágenes con una mayor cantidad de colonias, ya que la iluminación y la cámara no captaban la imagen con mayor cantidad de detalle.

La figura 4.9 nos permite realizar una clara comparación entre el algoritmo desarrollado y uno ya implementado de manera comercial mediante lógica difusa.

Aunque de nueva cuenta observamos que nuestro algoritmo presentó las fallas anteriormente mencionadas, resaltaremos ciertos puntos, la intensidad del filtro se redujo, lo que prueba que una mayor calidad en la imagen facilita la segmentación y además hace notar que por obvias razones no podemos comparar el trabajo realizado con lo ya hecho por una empresa dedicada a este tipo de proyectos. Así que concluimos que el algoritmo a pesar de estar en una etapa muy preliminar de desarrollo se puede decir que nos da buenos resultados.

El núcleo principal de nuestro algoritmo es la detección de marcas, pues es en esto en donde se basa la transformada *watershed* para lograr la segmentación, como observamos en el anterior capítulo, sección 3.2.1.5, las marcas sólo se colocan en donde están las colonias, así que para propósitos de conteo tan sólo es necesaria la detección de marcas internas, la transformada *watershed* nos sirve más que nada para detectar las formas de las colonias.

Una mejora que a la larga se haría totalmente necesaria dentro de la programación del algoritmo es dotarlo de inteligencia, pues dentro del código existen variables que se deben de adaptar dependiendo de las muestras que se tomen, las cuales son la intensidad del filtro, la detección de marcas internas, el tamaño en la caja de Petri y los desajustes que puedan surgir en la iluminación. De esta manera nuestro algoritmo tendría más robustez.

### 5.3 Recomendaciones

- Mejorar la calidad de la iluminación es vital para seguir desarrollando este proyecto, debido a que la mala calidad de las imágenes complica el desarrollo de los algoritmos y por ende el tiempo de cálculo también. Una buena opción es la iluminación por medio de LEDs debido a que permiten un mejor manejo de la dirección de la luz, además de ser más compactos. Otro punto a favor de este tipo de iluminación es que ya existen empresas dedicadas a la venta específica para sistemas de visión artificial.
- La principal mejora en el algoritmo se debe dar en la parte de la detección de marcas internas pues esta es el núcleo de la transformada *watershed* y es la que nos da la pauta para lograr una segmentación exitosa. El *toolbox* de procesamiento de imágenes lo realiza mediante el comando de “transformada máxima extendida”, la cual a su vez la realiza mediante “transformada h-domo”. Un posterior trabajo de investigación dedicado a este tema sería una gran aportación para la culminación exitosa de este trabajo.
- Para cualquier sistema de procesamiento de imágenes se debe tener una base de conocimientos, para este caso en particular nuestra base de conocimientos, son las imágenes de los cultivos de colonias. El tener una gran cantidad de imágenes captadas por nuestro sistema de visión nos permitirá tener una mejor idea de cómo atacar los problemas que se presentaron.



## Glosario

**Algoritmo.-** Conjunto ordenado y finito de operaciones que permite hallar la solución de un problema. Método y notación en las distintas formas del cálculo.

**Cuantización.-** Procedimiento matemático para construir un modelo cuántico para un sistema físico a partir de su descripción clásica.

**CCD.-** Del inglés Charge-Coupled Device, "dispositivo de cargas (eléctricas) interconectadas". El CCD es el sensor con diminutas células fotoeléctricas que registran la imagen.

**Filtro digital.-** El término *filtro digital* lo entenderemos como cualquier procesamiento realizado en una señal de entrada digital.

**Microorganismo.-** Nombre genérico que designa los seres organizados solo visibles al microscopio; p. ej. las bacterias, los infusorios, las levaduras, etc.

**Muestreo.-** Acción de escoger muestras representativas de la calidad o condiciones medias de un todo.

**pH.-** Índice que expresa el grado de acidez o alcalinidad de una disolución. Entre 0 y 7 la disolución es ácida, y de 7 a 14, básica

**Psicrofílicos.-** Tipo de bacteria predominante en suelos no cultivados, lagos y agua estancada, también se pueden presentar en carnes y helados, su temperatura de crecimiento va de 0 a 5 °C

**Rayos-x.-** Radiación capaz de atravesar cuerpos opacos y de impresionar las películas fotográficas. La longitud de onda está entre 10 a 0,1 nanómetros, correspondiendo a frecuencias en el rango de 30 a 3.000 PHz.

**Sensor.-** Dispositivo que detecta una determinada acción externa, temperatura, presión, etc., y la transmite adecuadamente.

**Termofílicos.-** Tipo de bacterias que crecen en ambientes a temperaturas inusualmente altas por arriba de los 45 °C y e ocasiones hasta los 80 °C otras formas de vidas morirían al instante a estas temperaturas

**Textura.-** Estructura, disposición de las partes de un cuerpo

**Transmitancia.-** Es una magnitud que expresa la cantidad de energía que atraviesa un cuerpo

**Ultrasonido.-** Onda acústica cuya frecuencia está por encima del límite perceptible por el oído humano (aproximadamente 20 KHz). Algunos equipos utilizan ondas de ultrasonido para producir imágenes de las cavidades internas del cuerpo.