Sinergia Mecatrónica, Capítulo 37, pp.457 – 482, 2019.
ISBN: 978-607-9394-17-2.

Comparación de Imágenes para Reconstrucción con Línea Láser Discriminando Canales para Reducción de Ruido

[1]Díaz Cano Carlos Daniel, [2]Pedraza Ortega Jesús Carlos, [2]Ramos Arreguín Juan Manuel, [2]Aceves Fernández Marco Antonio y [2]Vargas Soto José Emilio

[1]Asociación Mexicana de Mecatrónica A.C.
[2]Universidad Autónoma de Querétaro, Facultad de Ingeniería
carlosd.dc.it@gmail.com

Resumen

En este artículo se muestra un método para reducción de ruido, la discriminación de ruido, el cuál es aplicado para línea láser. Desde hace tiempo se ha intentado reproducir objetos físicos en modelos digitales por medio de reconstrucción. El método de triangulación es uno de los principales en línea láser, el cual a través de un procesamiento de imágenes, permite obtener la silueta de una línea proyectada, que sacando el centro de la luz, se consiguen los puntos que representan el espacio 3D del objeto escaneado. Por medio de algunas pruebas pre-eliminares comparando tres diferentes colores de línea láser al discriminar cada canal y luego ver de forma individual cada canal. Posteriormente se ven los histogramas de cada canal. Se llega a concluir que el verde es la solución media con el cual se parte a implementar un algoritmo para reconstrucción. Este se implementó en Python con librerías para procesamiento de imágenes y modelado de datos. Midiendo el tiempo de ejecución y los puntos generados al aplicar el método de discriminación contra la no aplicación de este, demuestra que no adiciona tiempo considerable, así como una cierta pérdida de información. Se concluye que el método de discriminación en efecto reduce el ruido a compromiso de pérdida de información sin carga adicional a todo el proceso.

Palabras clave: Reconstrucción 3D, línea láser, triangulación, reducción de ruido, procesamiento de imágenes digitales.

1. Introducción

Desde hace varios años se ha buscado recrear objetos físicos de manera digital para tareas variadas que van desde la ingeniería inversa [1], preservación de objetos históricos [2], modelos para prótesis, y desarrollo de calzado [3]. A lo cual han surgido varios métodos de reconstrucción al punto que existe toda una taxonomía, teniendo métodos de contacto o destructivo y sin contacto o no destructivo.

Los métodos sin contacto usan sistemas que hacen uso de dispositivos que no requieren de algún aditamento que requiera tocar la superficie del objeto, resultando en otras subcategorías que consisten en reflexión y transmisión. De entre esos dos el de reflexión se divide en métodos ópticos que dependen mucho del uso de cámaras que capturen el objeto al cual se le adicione algún medio que pueda reflejar la forma del objeto.

Para ello existen los métodos como luz codificada, línea láser, luz estructurada y cambio de fase. Siendo la línea láser uno de los más empleados e investigados y que forma parte de los métodos de triangulación, que como su nombre lo indica hace uso de la figura geométrica para poder calcular la forma del objeto al medir distancias y ángulos desde un punto emisor al receptor.

Algunos trabajos recientes con línea láser buscan producir diseños para femorales [4], modelos tipo yeso para muñecas [5] y de ortodoncia [6]. Lo que indica un uso en la actualidad, por lo que mejoras son requeridas para obtener modelos con mayor exactitud a su contraparte física.

Entre las propuestas de mejora, se encuentran métodos interesantes como sumergir el objeto en agua y proyectar el láser en el agua [7]. Así como un posterior intento para reproducir el color con el uso tres láseres de colores diferentes [8]. También se intenta mejorar la obtención del centro de la luz con el uso de un kernel circular [9].

En esta ocasión se ha propuesto mejorar la reducción del ruido al hacer una discriminación de canales para el método de triangulación láser. La reducción de ruido es importante en el campo de reconstrucción, ya que existen métodos que dependen mucho de la carencia de este pues algunos asumen que la información empleada es libre de ruido [10].

Existen varios métodos de reducción de ruido, algunos tratan de reducir el ruido por medio de programación en paralelo [11], usar algoritmos de mínimos cuadrados rápidos estables numéricamente (NS-FRLS, siglas de: Numerically Stable Fast Recursive Least Squares) [12], aplicar un filtro Kalman segmentando imágenes por textura usando coeficientes locales de variación [13], difusión anisotrópica [14-15], y también existen trabajos que trabajan con imágenes a color usando filtros adaptativos de componentes del color [16].

Para este proyecto se emplea una plataforma giratoria sobre la cual se coloca el objeto a reconstruir, frente al objeto se coloca una cámara fotográfica digital y adyacente a este, un rayo láser para poder capturar la silueta proyectada sobre el objeto en un ambiente oscuro. Teniendo el conjunto de imágenes se procede a realizar el procesamiento de imágenes para encontrar los puntos sobre los cuales el láser recae sobre el objeto. Esto se planea hacer con láseres de diferentes colores, con el fin de emplear en un paso intermedio, la discriminación de un canal para encontrar cual es el que reduce el ruido y si este muestra una mejora a usar los tres canales. Como siguientes pasos está el obtener una serie de puntos que permita reconstruir el objeto y este sea proyectado.

Se empleó el lenguaje de programación Python para la realización de este proyecto, aprovechando el uso de algunas librerías que permitieran la carga de imágenes y visualización de los modelos.

La estructura de este capítulo empieza con las pruebas iniciales en la sección 2 para determinar en una primera instancia si es factible reducir el ruido por la discriminación de ruido a través de unas pruebas con las imágenes tomadas y tres colores de láseres dividiendo los canales y hacer las combinaciones posibles para observar la cantidad de luz capturada que se puede considerar como ruido y así determinar cuál de todos se empleará para procesar sus imágenes. La sección 3 es el procesamiento de imágenes donde se muestran algunos resultados cuando se aplica y cuando no la discriminación de canales para comprobar si el ruido es reducido, así como el algoritmo empleado para la reconstrucción. La sección 4 muestra los resultados de la reconstrucción, así como la obtención de las posiciones de los puntos en un espacio tridimensional. La sección 5 contiene el código en Python empleado así como una corta lista de las librerías requeridas para correr el programa mostrado. En la sección 6 se muestran y discuten los resultados finales de reconstrucción. Se finaliza en la sección 7 con las conclusiones llegadas a partir de todas las pruebas realizadas y que se podría hacer para combatir los problemas encontrados.

2. Pruebas iniciales

Se decide emplear el uso de tres láseres de diferentes colores, siendo estos rojo, verde y azul, los mismos colores básicos que componen la visión humana y los dispositivos de captura comunes. Las características de los láseres se muestran en la tabla 1.

Los objetos empleados para las pruebas fueron una cabeza de unicel pintada en negro, un cráneo de cerámica, una campana de cerámica también pintada en negro y un cerdo de cerámica. Estos pueden ser vistos en la figura 1.

Una forma pre-eliminar de observar si la hipótesis es factible, fue aplicar la discriminación de canales con cada láser diferente y ver las imágenes obtenidas. Esto nos brinda una forma de observar el comportamiento de una imagen con respecto al color del láser y los canales que la componen. En una primera instancia con la eliminación de un único canal por cada láser, se encontró que en todos al eliminar el canal que equivale al color del láser, la reducción del ruido se vuelve aparente.

Tabla 1. Características de los láseres.

Característica	Rojo	Verde	Azul
Longitud de onda	650 nm	532 nm	405 nm
Energía emitida	3 mW	5 mW-	5 mW
Voltaje	3 V, CD	3 V, CD	3.2 V, CD
Clase	IIIa	IIIa	IIIa

(a) (b) (c) (d)

Figura 1: Objetos de prueba para reconstrucción. (a) Cabeza; (b) Cráneo; (c) Campana; (d) Cerdo

2.1 Comparaciones a dos canales

En los casos del rojo mostrados en la figura 2, durante la discriminación del canal R se puede apreciar una reducción del reflejo de la luz, comparado a los otros dos casos. En el caso del canal B discriminado, en la parte de la dentadura y sobre todo de la nariz, la línea de la silueta se confunde con el reflejo de la luz, por lo que el ruido es considerable, siendo la imagen original con el caso del canal G discriminado muy similares.

Mientras que en el azul, figura 4, no se alcanza a percibir diferencia aparente entre los canales que emplean el azul y la imagen original, siendo esto tan mínimos que es despreciable, sin embargo en el caso en el cual se discrimina el azul, se nota una gran diferencia, pero da la sensación de que existe pérdida de información, tal como podría ser por la parte inferior del hueco de la nariz, y en la parte superior de la frente.

En el caso del láser verde mostrado en la figura 3, su respectivo canal genera un reflejo de luz en un radio considerable que podría afectar al momento de buscar el punto específico sobre el cual el rayo de luz hace primer contacto con el objeto. En los casos tanto el R como B discriminados, se muestran similares a la imagen original, siendo en el caso del canal G discriminado, en el cual el reflejo prácticamente imperceptible, con la zona de la nariz el único lugar donde se muestra un ruido aparente.

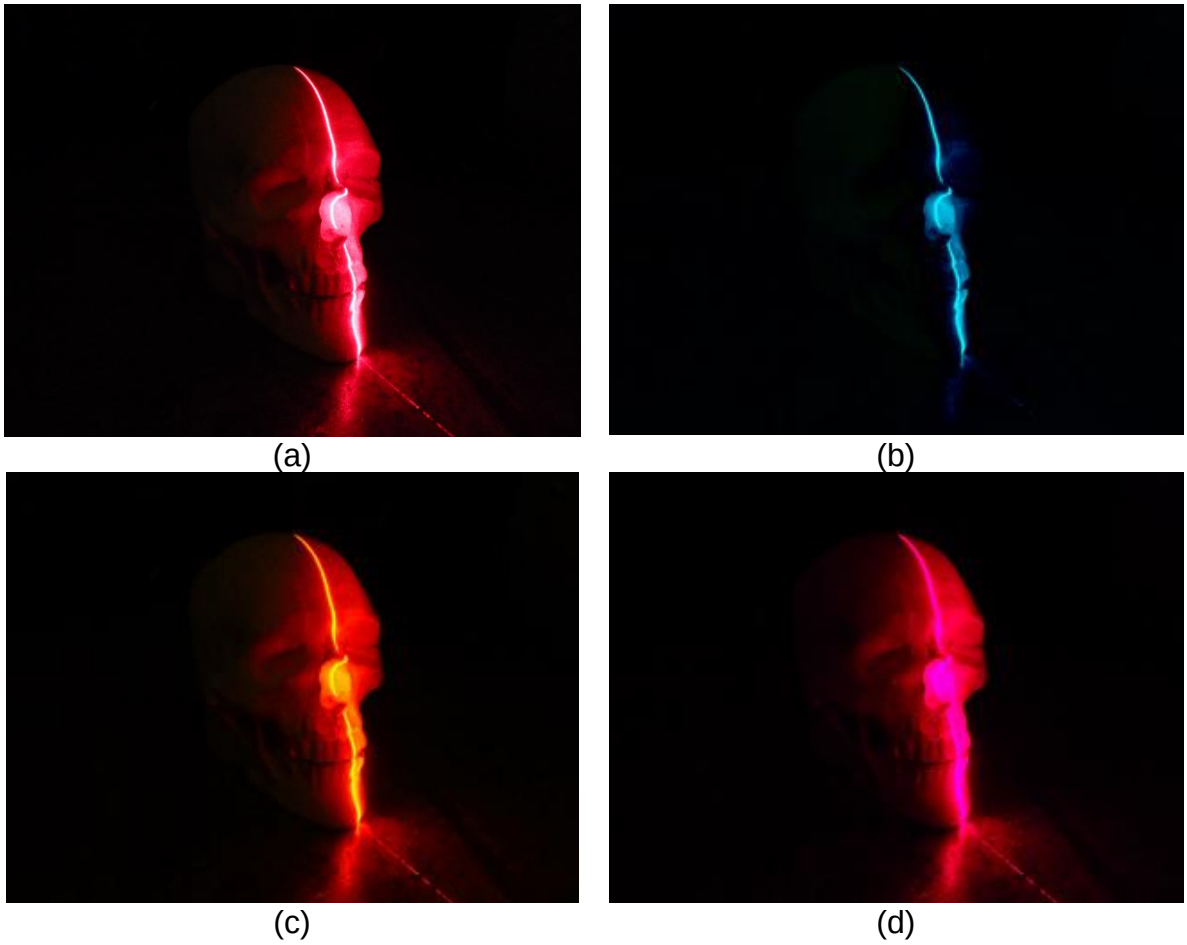
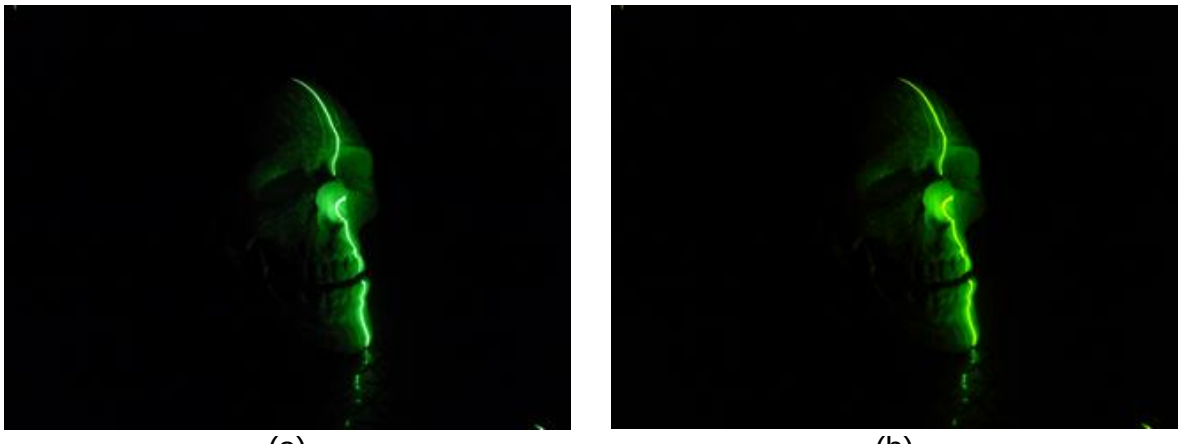
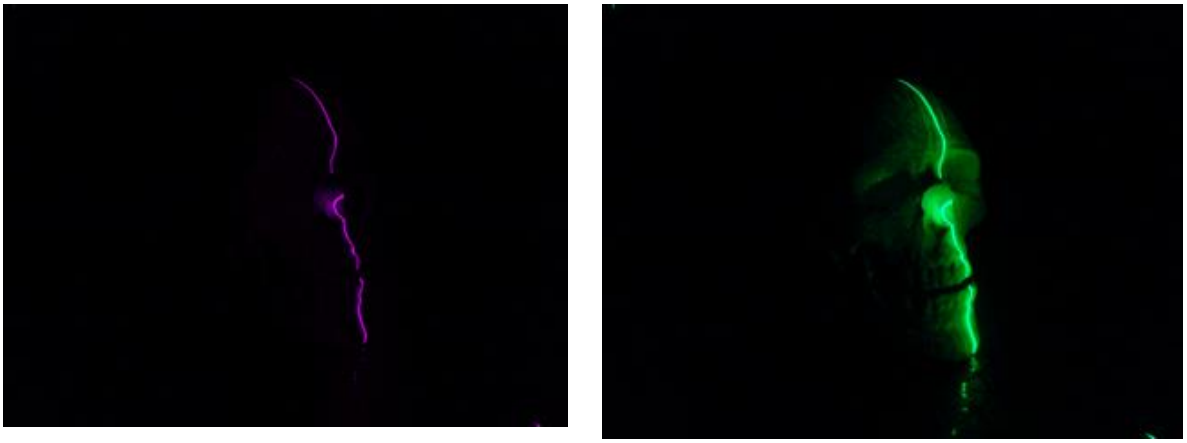


Figura 2: Captura con láser rojo y discriminación de canales. (a) Imagen original; (b) Canal R discriminado; (c) Canal G discriminado; (d) Canal B discriminado.

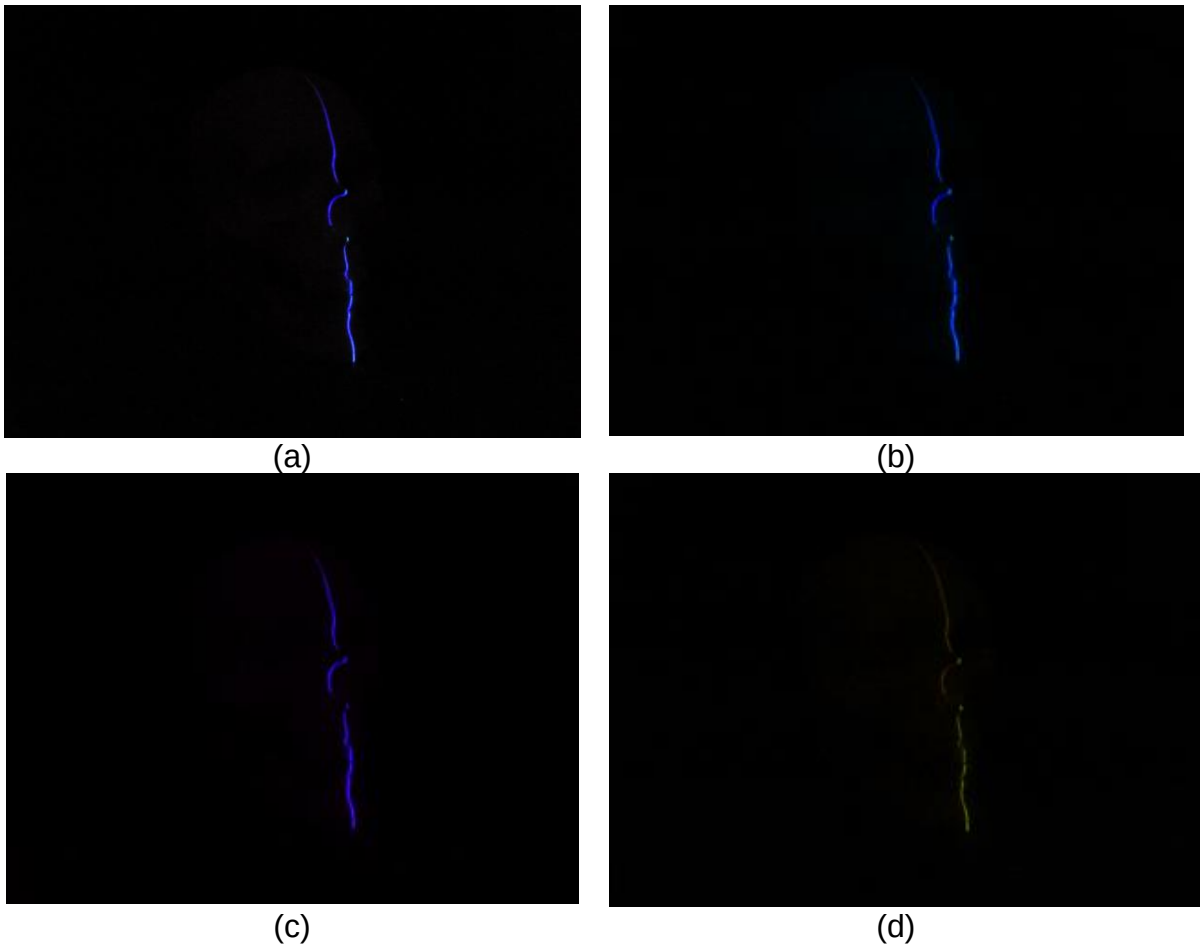


Sinergia Mecatrónica, Capítulo 37, pp.457 – 482, 2019.
ISBN: 978-607-9394-17-2.



(c) (d)

Figura 3: Captura con láser verde y discriminación de canales. (a) Imagen original; (b) Canal R discriminado; (c) Canal G discriminado; (d) Canal B discriminado.



(a) (b)

(c) (d)

Figura 4: Captura con láser azul y discriminación de canales. (a) Imagen original; (b) Canal R discriminado; (c) Canal G discriminado; (d) Canal B discriminado.

2.2 Comparaciones a un canal

Una segunda prueba consiste en tomar un único canal que sirva de referencia para conocer la cantidad de información almacenada en cada uno. En el caso del láser rojo, figura 5, es evidente que el su canal correspondiente excede la cantidad de información en comparación a los otros dos, por lo que se puede considerar que en este como poseedor de una gran cantidad de ruido, mientras que en los otros canales, se puede apreciar la línea que corresponde a la silueta del objeto correspondiente a la vista. De entre las los canales, el canal G es el que muestra menos ruido y la línea sigue presente para poder obtener la información deseada.

En el caso del láser verde mostrado en la figura 6, el canal G que corresponde al verde, muestra un exceso de información, la mima situación que ocurre con el láser rojo. En cuanto a los otros dos canales, sorpresivamente son prácticamente idénticos, siendo el B con una intensidad ligeramente mayor a R. Igualmente en los canales no correspondientes al color del láser, la línea de la silueta es visible, por lo que es perfectamente útil para intentar generar la nube de puntos.

Para el caso del láser azul, no se aprecia mucho ruido, incluso parece que en este caso la mejor opción es emplear el canal B, tal como se muestra en la figura 7, en el caso del canal de color correspondiente hay una línea notable. Sin embargo existe en ocasiones que se presente una ligera pérdida de información por la baja intensidad obtenida las imágenes, sobre todo en los dos canales restantes donde la línea no se tan clara como con los láseres rojo y verde.

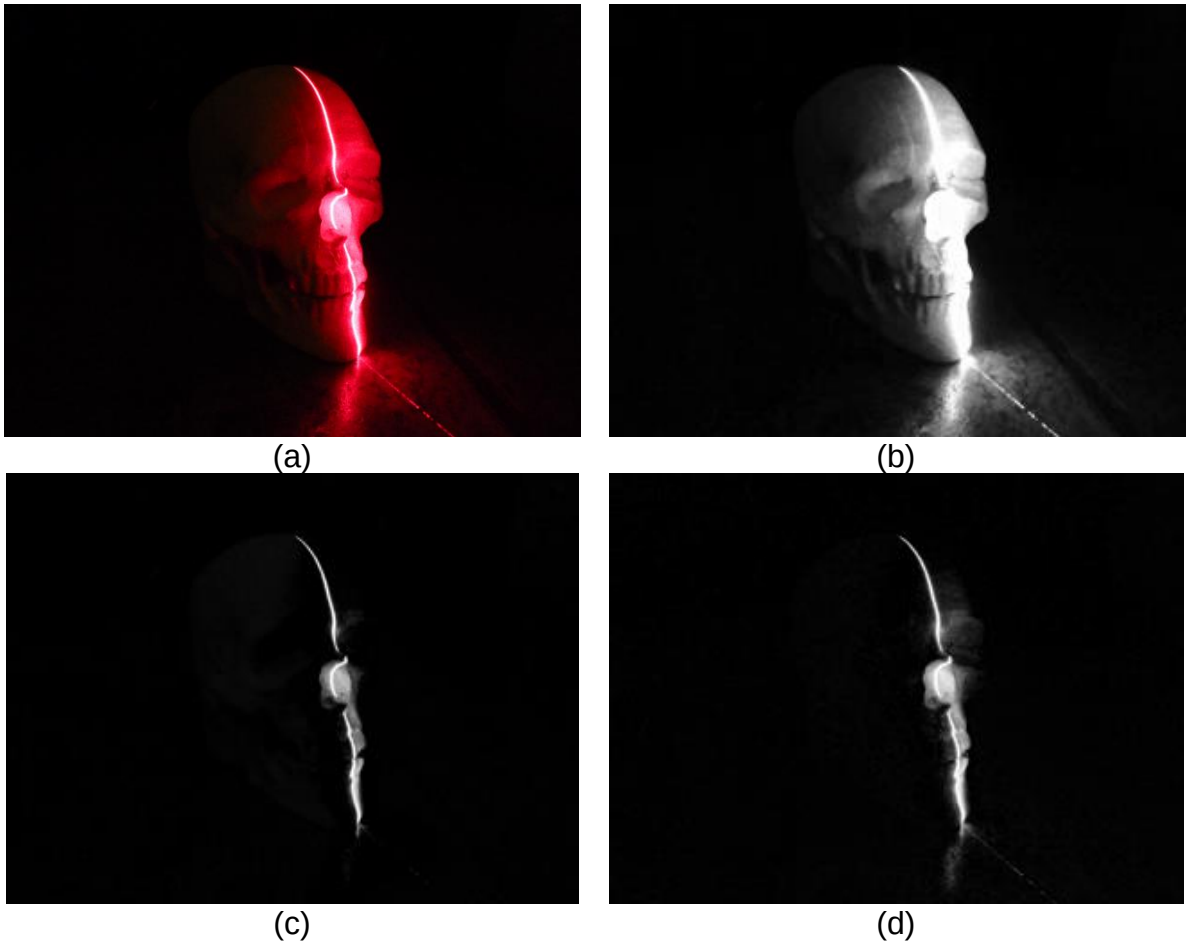


Figura 5: Canales individuales de la imagen con láser rojo. (a) Imagen original, (b) Canal R, (c) Canal G, (d) Canal B.

Sinergia Mecatrónica, Capítulo 37, pp.457 – 482, 2019.
ISBN: 978-607-9394-17-2.

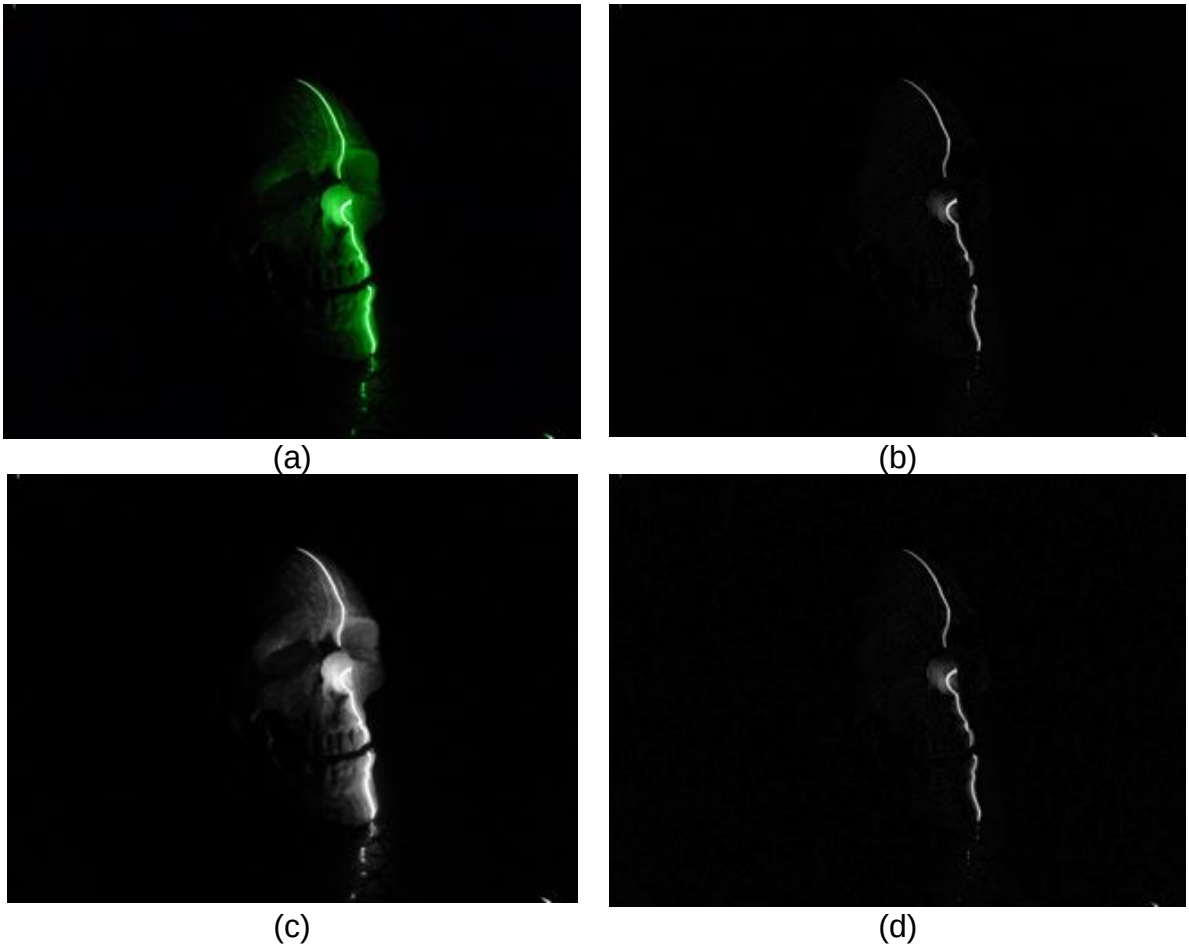
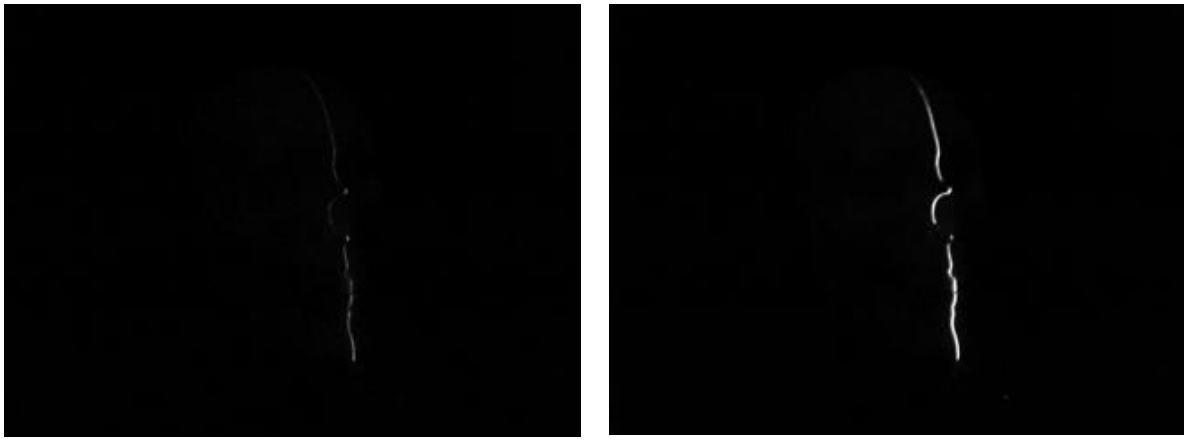


Figura 6: Canales individuales de la imagen con láser verde. (a) Imagen original, (b) Canal R, (c) Canal G, (d) Canal B.

Sinergia Mecatrónica, Capítulo 37, pp.457 – 482, 2019.
ISBN: 978-607-9394-17-2.



(c) (d)

Figura 7: Canales individuales de la imagen con láser azul. (a) Imagen original, (b) Canal R, (c) Canal G, (d) Canal B.

2.3 Revisión de histogramas

El histograma es una herramienta empleada comúnmente para la separación entre objeto de fondo, por lo que se procedió a también hacer una comparación de histogramas de canal para hacer una inspección que permita comprender un poco el comportamiento del ruido en cada caso de láser y canal.

El primer caso con el rojo, se puede ver muy claramente como en el canal R hay un considerable incremento en los niveles cercanos al blanco, si consideramos que el alto de la imagen es de 240 píxeles, significaría que se esperaría de manera ideal esa cantidad de píxeles en los niveles más altos, preferentemente a nivel de 255 que es valor máximo que podría tomar un píxel, esto asumiendo la línea cubriera todo el eje *y*. Sin embargo en la realidad eso casi nunca ocurre. De manera manual se estima que la línea deseada tiene 160 píxeles de altura, mientras que en el histograma los 10 valores más altos (246-255) suman un total de 1558, eso es casi 10 veces la cantidad deseada. Esto explicaría en gran medida el motivo del porque en la figura 5c, la línea se empieza a perder en las zonas de la nariz y dentadura. En cuanto a los canales G y B, como se observó en la figura 5, la línea es mucho más distinguible, sin embargo la suma de los 10 valores más altos fue de 15 y 46 respectivamente, eso significa que no es posible buscar la línea entre esos valores. Sin embargo visualmente es razonable considerar que la línea es aún es recuperable y los métodos que trabajan con el histograma pueden servir para nuestro propósito.

El siguiente caso, que es el láser verde, como se puede apreciar en la figura 10 el histograma correspondiente a G, no existe el incremento en los valores como sucedió con el láser rojo. Incluso al hacer la suma de los 10 valores más altos, da un total de 69, otra vez menor a lo que podría ser la línea, pues se esperaría un mínimo de 120 de altura. Al igual que con la figura 6, los canales R y B muestran un comportamiento similar. Las semejanzas hasta el momento con el histograma entre el láser rojo y verde, consisten en que sus respectivos canales R y G, tienen un ancho mayor en los niveles iniciales de los píxeles, comparado a los otros dos canales. No obstante, no se puede considerar esto como un indicativo de nivel de ruido sin un análisis a profundidad, ya que se debe tomar en cuenta las características de la cámara, láser y ambiente que pueden afectar la captura de la imagen. Siendo estas características conocidas como intrínsecas y extrínsecas.

Por último se tiene el caso del láser azul el cuyo comportamiento refleja un poco al láser verde con respecto a que la suma de los 10 valores más altos de píxel son muy bajos, llegando incluso a un total de un dígito, 8, incluso los últimos 50 son menor al dado por los últimos 10 del verde con un total de 51. Adicionalmente se vuelve notable como el canal B es más similar al R que al G, contrario a lo que se esperaría basado en la figura 7, pues se creería que el R y G fueran similares entre ellos. Esto

refuerza la noción que el láser verde genera más ruido, considerando que tanto el láser verde como el azul en sus canales predominantes, los de mayor nivel no presentan una cantidad mayor a 100 para una imagen de una dimensión de 240 x 320. En ambos casos, la suma de los 10 valores más altos en caso del láser rojo es 100 veces mayor al de los otros dos láseres. Sugiriendo que el rojo es un mayor productor de ruido en imágenes de línea láser en ambiente oscuro.

Al considerar las diferencias obtenidas por cada láser, se opta por usar el verde. Los motivos son porque es el que presenta un punto intermedio entre los tres. En el rojo como se pudo ver en la figura 5, produce resultados con fuerte intensidad, así como los resultados con los histogramas, sobre todo en el caso donde se muestra únicamente el canal R en el cual la línea no es tan visible por el ruido, perdiéndose por el reflejo producido y la gran cantidad de altos valores de píxel. Mientras el azul tienen una intensidad muy baja en el cual los otros canales, tal como se muestra en la figura 7 y sus histogramas, no se muestra la silueta tan clara como en los otros láseres y muy baja cantidad de altos niveles de píxel. Ya que el verde mostró que en su canal correspondiente, figura 6, se produce un cierto ruido pero la línea se puede diferenciar comparado al caso con el láser rojo, y sus dos canales restantes, la línea se muestra claramente sin pérdida en comparación al caso del láser azul.

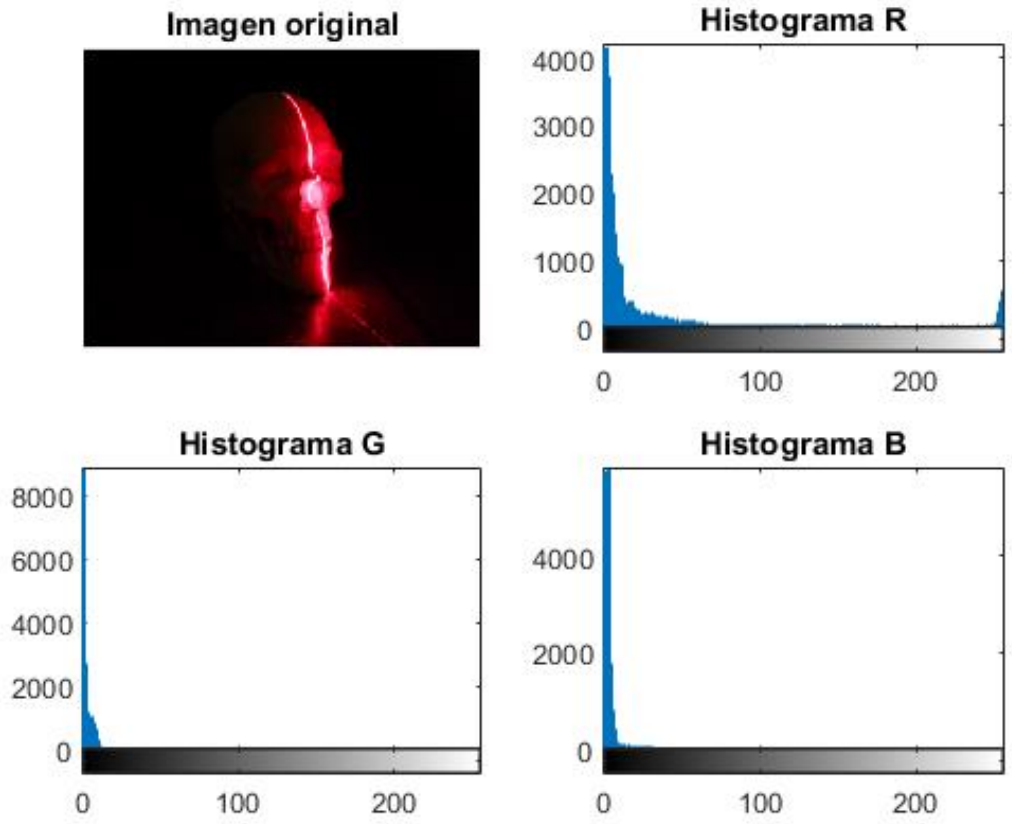


Figura 8: Histograma de láser rojo.

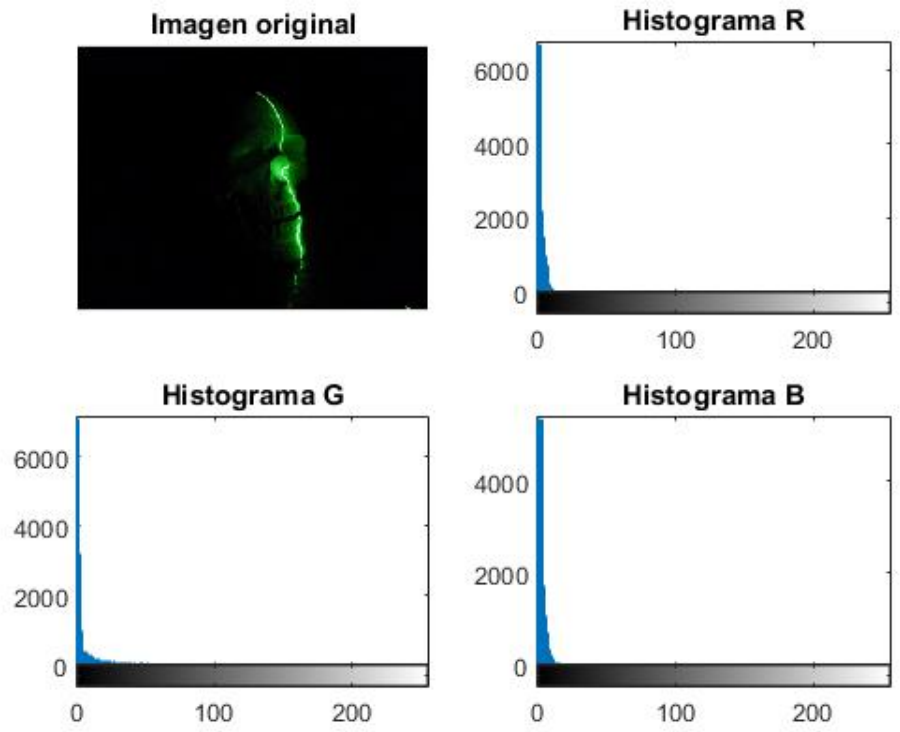


Figura 9: Histograma de láser verde.

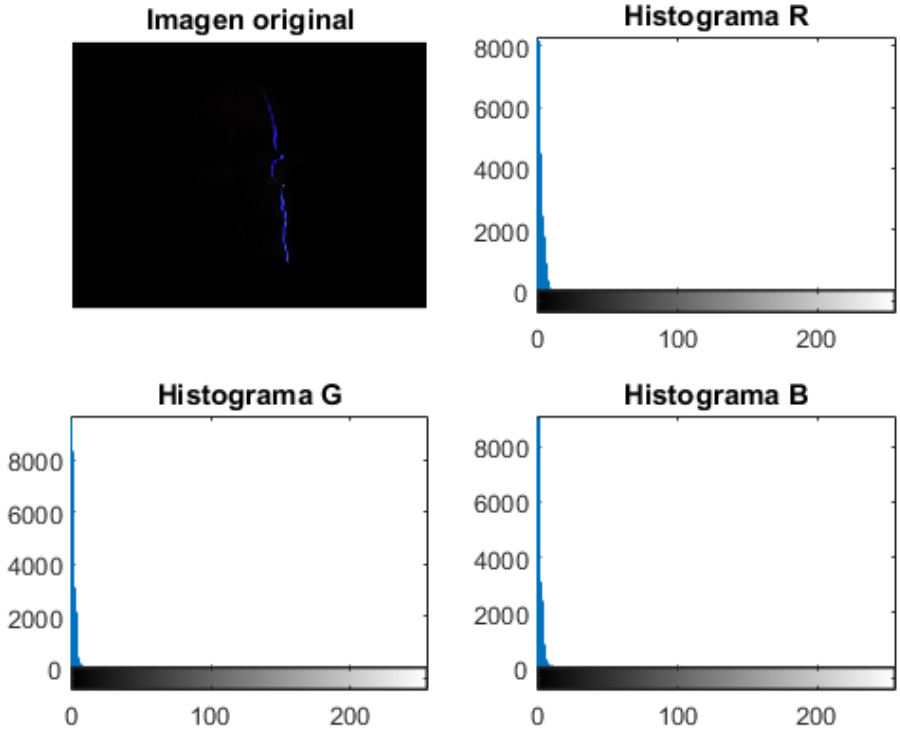


Figura 10: histograma de láser azul.

Sinergia Mecatrónica, Capítulo 37, pp.457 – 482, 2019.
ISBN: 978-607-9394-17-2.

3. Procesamiento de las imágenes

El proceso para obtener los puntos que indican la forma del objeto se realiza siguiendo el algoritmo mostrado en la figura 11.

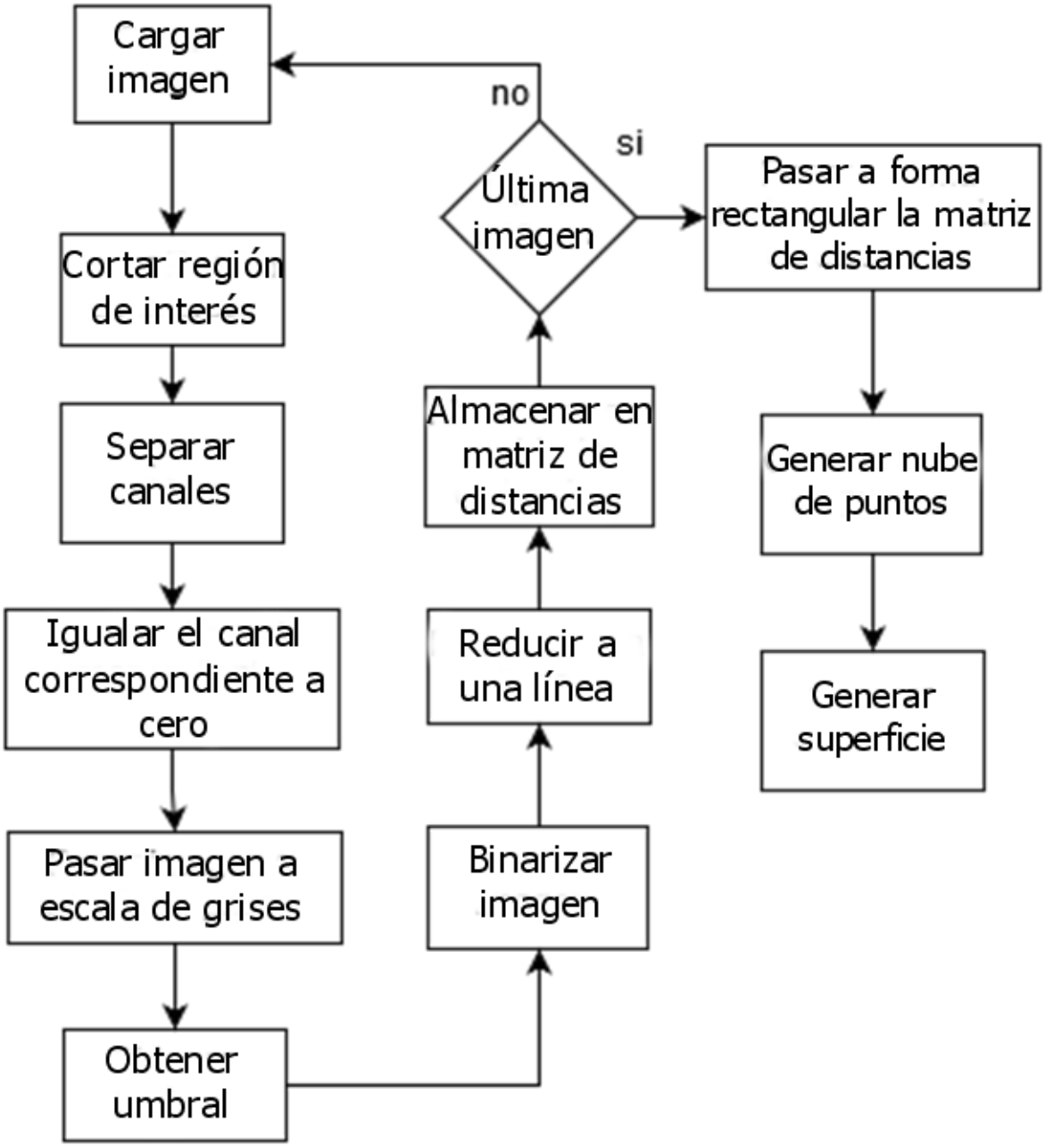


Figura 11: Algoritmo para reconstrucción empleado.

Aquí se presenta un algoritmo ligero que puede ser cargado en dispositivos de gama baja, una propuesta también para ofrecer un sistema de reconstrucción económico. El primer paso es cargar la imagen, en este caso se realizó por medio de Opencv [17], que se puede aplicar a varios lenguajes de

programación. Después es el corte de región de interés, en el cual se trata de determinar el centro del objeto y su distancia máxima con la superficie del objeto, así como la altura del objeto. Acto seguido se empieza a emplear el método de discriminación propuesto que es el tener las matrices que determinan el canal, R para rojo, G para verde y B para azul, para luego establecer el canal correspondiente al color del láser en una matriz de ceros. Un proceso bastante ligero que no afecta enormemente el desempeño de algún algoritmo de procesamiento de imágenes.

El siguiente paso consiste en convertir la imagen a color en escala de grises, que se obtiene al promediar los valores cada píxel en la misma posición de cada matriz. Quedando entonces la ecuación (1):

$$Gr = \frac{(R, G, B)}{3} \tag{1}$$

Quedando entonces una matriz *Gr* que corresponde al valor promedio de las matrices *R*, *G* y *B*. La imagen generada por está matriz debe pasarse a binario, el cual como lo indica su nombre, solo puede tener dos valores, lo que significa que se obtendrá una imagen en blanco y negro. Para poder hacer esto, se debe determinar un valor sobre el cual se determina si un píxel pasará a ser blanco o negro. Existen varios métodos para obtener dicho valor, de los cuales se utilizará el de Otsu [18], el cual es usado aún hoy día [19-24], considerándolo como un método suficientemente fiable que ha sido incluido en diversas librerías [25], incluyendo OpenCV, y otros lenguajes de programación.

Dicho método utiliza el histograma y se obtienen las probabilidades o momentos acumulativos de cada tono en la escala de grises, y las divide en dos grupos de forma iterativa hasta el máximo valor de la imagen. Una vez obtenidos los valores de las probabilidades de cada imagen para cada caso, se obtiene su varianza máxima, el cual será considerado como el umbral óptimo. Su implementación se describe en las ecuaciones (2) al (7):

$$\omega_0 = \sum_{i=1}^{t} p_i \tag{2}$$

$$\omega_1 = 1 - \omega_0 \tag{3}$$

$$\mu_0 = \frac{\sum_{i=1}^{t} i * p_i}{\omega_0} \tag{4}$$

$$\mu_1 = \frac{\left(\sum_{i=t+1}^{N} i * p_i\right) - \mu_0}{\omega_1} \tag{5}$$

$$\sigma_w^2(t) = \omega_0 \omega_1 (\mu_1 - \mu_0)^2 \tag{6}$$

$$U = \max(\sigma_w^2(t)) \tag{7}$$

En donde:

- *t*: es el umbral siendo evaluado.
- *N*: es el valor máximo que puede tomar un píxel.
- *p*: los valores probables obtenidos a partir del histograma de la imagen.
- ω_0: es la probabilidad acumulada desde 1 hasta t.
- ω_1: es la probabilidad acumulada desde t+1 hasta N.
- μ_0: Los momentos cumulativos del histograma desde 1 hasta t.

- μ_1: Los momentos cumulativos del histograma desde t+1 hasta N.
- σ_w^2: La varianza de t, donde la máxima es la óptima.
- *U*: Es el umbral óptimo.

Una vez obtenido el umbral como se muestra en la figura 13, se pasa la imagen a blanco y negro generando una matriz *bn*. Requiriendo de seguir la regla de la ecuación (8) por cada píxel p:

$$bn(x,y)\begin{cases}0, & p(x,y) < U\\ 1, & p(x,y) \geq U\end{cases} \tag{8}$$

Con la imagen binaria se procede a adelgazar el grosor de la línea para obtener la silueta del objeto requiriendo que solo exista un único píxel por fila, el método más común es el del centro de la masa, el cual es descrito en la ecuación (9):

$$md(x,y) = \frac{\sum p(x,y)x}{\sum p(x,y)} + c \tag{9}$$

Donde:

- md: es la matriz de distancia que almacenará el valor de la distancia del píxel desde el centro hasta el punto donde la línea es registrada.
- $p(x,y)$: es el valor del píxel siendo sumado.
- *x*: es la posición en el eje x.
- *c*: es un valor constante que sirve para rectificar un posible error en el centro del objeto al capturar la imagen.

De esta manera se tiene la imagen final que permitirá generar los puntos en un espacio tridimensional que representan el objeto físico.

Como se comentó en un principio, este algoritmo se aplicó con la variación en el paso que precede a la conversión a escala de grises. Esto sirviendo para reducir el ruido, tal como se muestra en la figura 12.

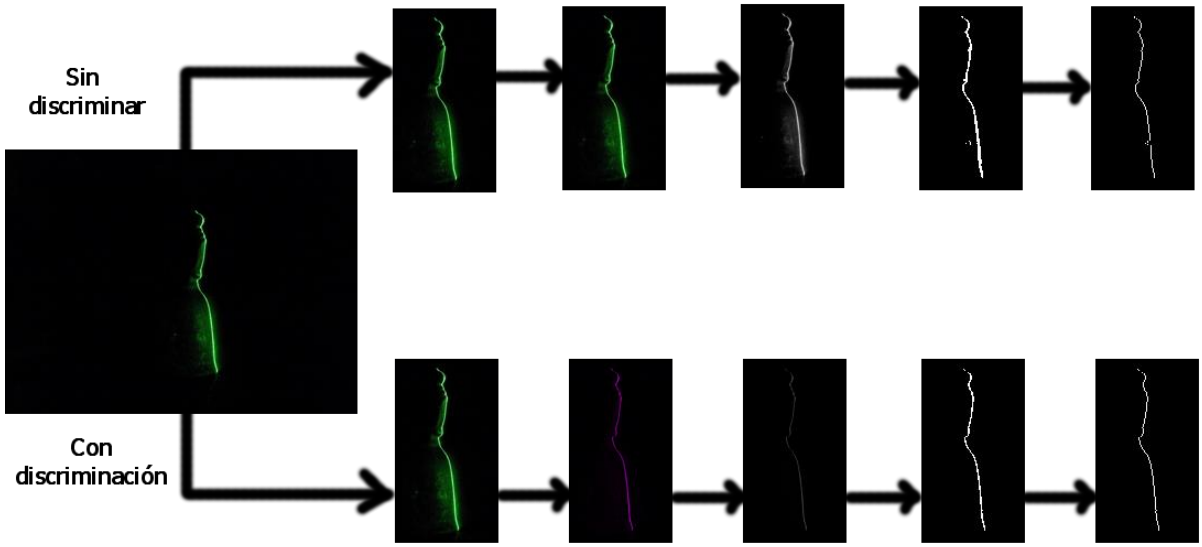
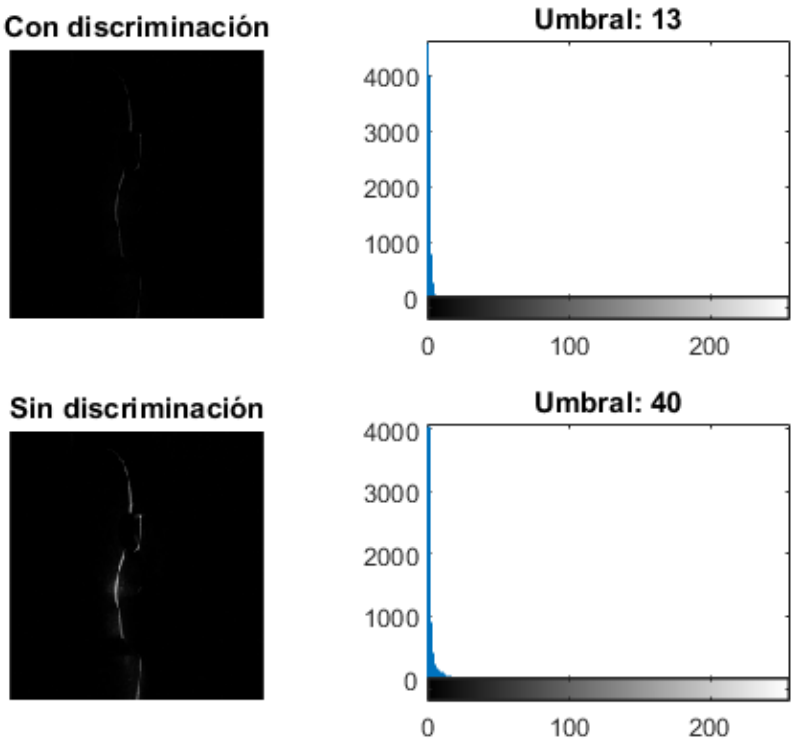
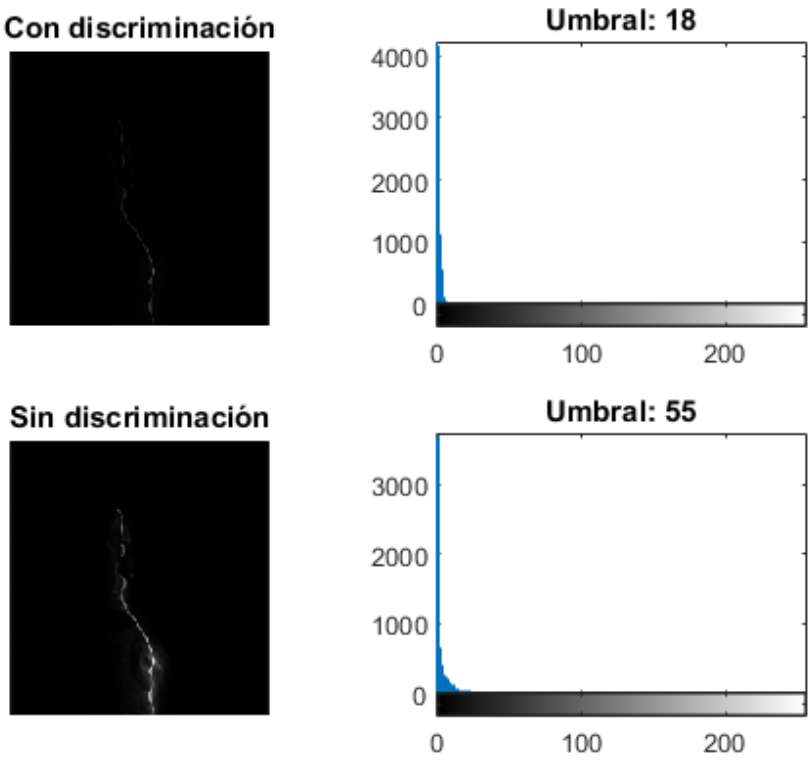


Figura 12: Procesamiento de imágenes con y sin discriminación.

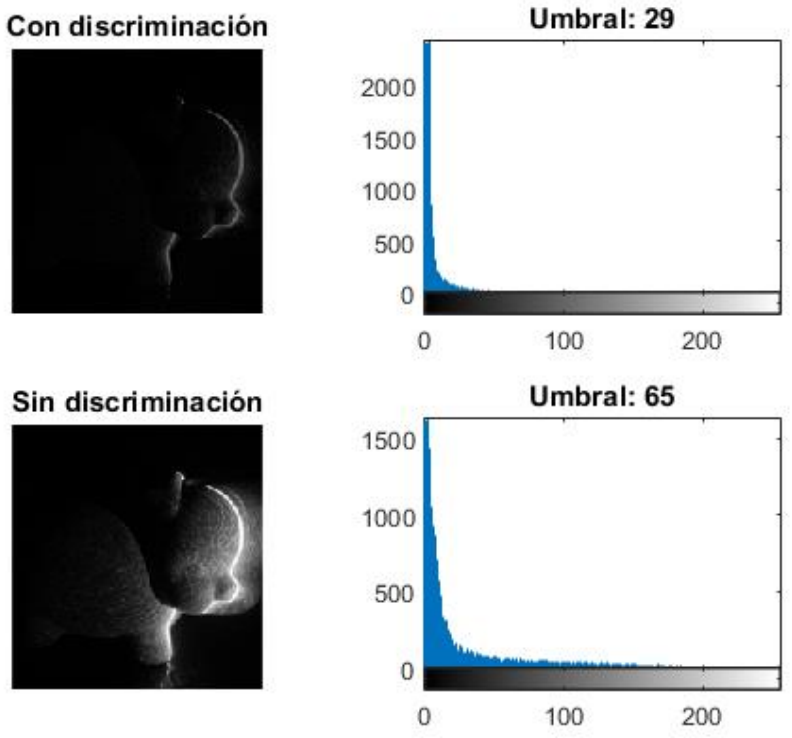
Sinergia Mecatrónica, Capítulo 37, pp.457 – 482, 2019.
ISBN: 978-607-9394-17-2.



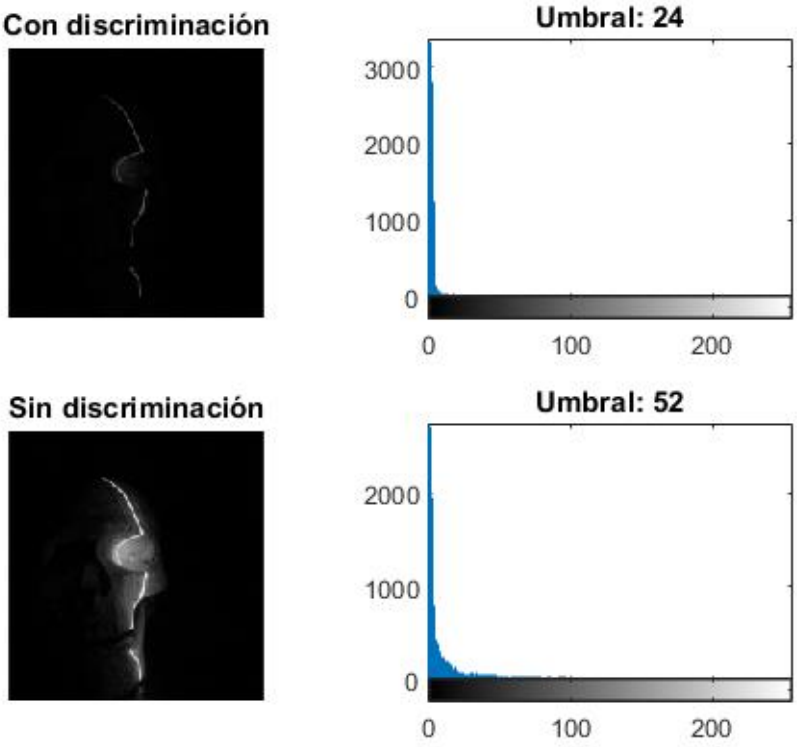
(a)



(b)



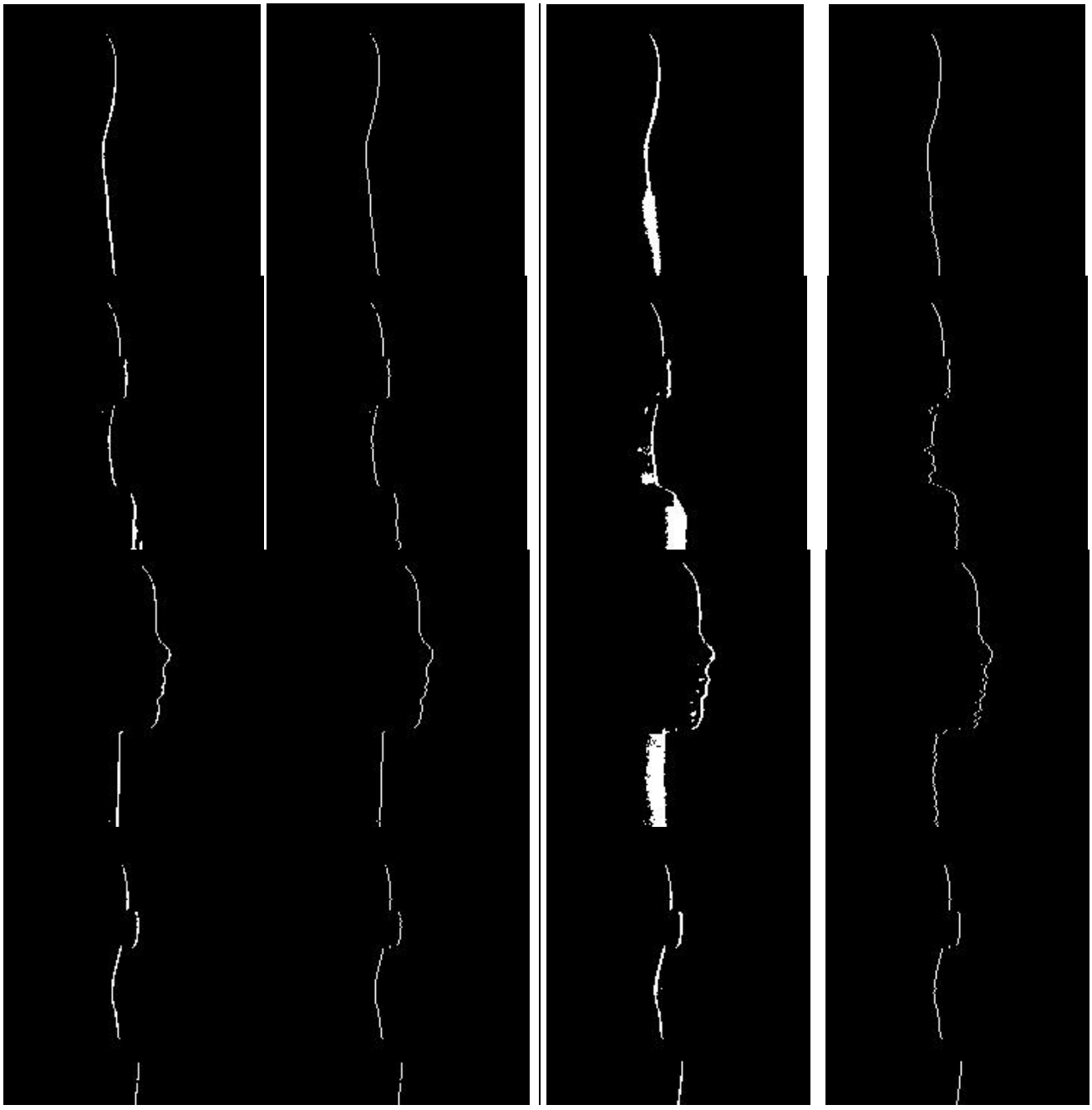
(c)



(d)

Figura 13: Histogramas y su umbral con Otsu. (a) Cabeza; (b) Campana; (c) Cerdo; (d) Cráneo.

En general, se encuentra una mejoría de las imágenes al aplicar la discriminación, ya que en varias ocasiones se encuentra una distorsión en la línea delgada, así como ciertas manchas en la imagen binaria que requerirán de un proceso adicional para eliminarlos y no influyan en la obtención de la silueta. Es así que se siguieron las pruebas con los demás objetos, de los cuales en la figura 14 se muestran algunas comparaciones cuando se aplica y cuando no, la discriminación de canales con la cabeza.



(a) (b)

Figura 14: Comparaciones de aplicar discriminación de canales. (a) Con discriminación; (b) Sin discriminación

Sinergia Mecatrónica, Capítulo 37, pp.457 – 482, 2019.
ISBN: 978-607-9394-17-2.

4. Proceso de reconstrucción

Siguiendo el método de triangulación, es posible determinar la profundidad de un objeto basándose en los ángulos de giro de la plataforma giratoria. El primer paso para el proceso de reconstrucción es del pasar la posición de los puntos de su forma angular a la forma polar, empezando por obtener el ángulo de desplazamiento que resulta de dividir la cantidad de imágenes más uno y dividirlo por un giro completo de 360°, este paso queda descrito en la ecuación (10).

$$\theta(i) = \frac{n+1}{360°} \tag{10}$$

En donde:

- n: es el número de imágenes utilizadas.
- θ: es el ángulo de desplazamiento en grados.

Sin embargo es requerido el ángulo en radianes θ', que se consiguen fácilmente con la ecuación (11).

$$\theta'(i) = \frac{\theta(i)}{180}\pi \tag{11}$$

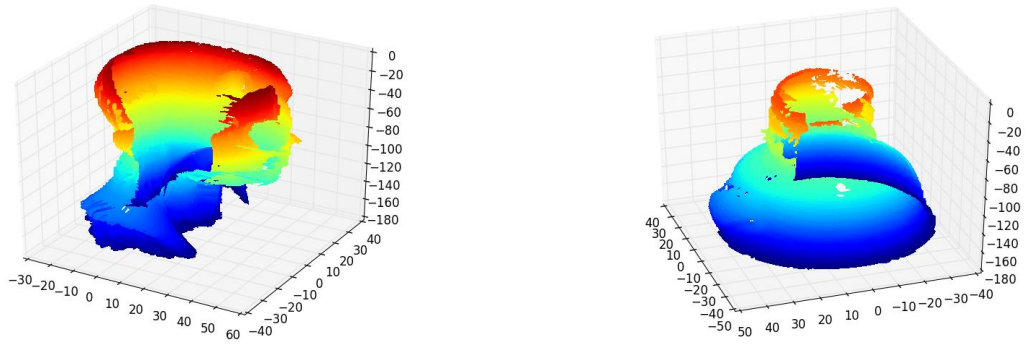
Para obtener la forma polar se emplea las ecuaciones (12) y (13) para poder obtener la matriz X de la posición x y la matriz Y de la posición y del punto en el espacio. Mientras que la matriz Z es fácilmente deducible con (14) usando la posición y en cada iteración.

$$X(x,y) = md(i,j)cos\theta'(i) \tag{12}$$

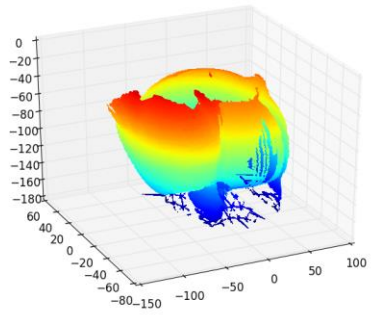
$$Y(x,y) = md(i,j)sin\theta'(i) \tag{13}$$

$$Z(x,y) = j \tag{14}$$

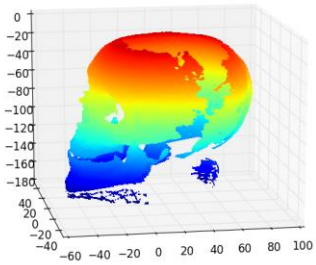
De esta forma se tiene la nube de puntos que representa el objeto en un plano de tres dimensiones. Para esto se usa la función plot_surface de la librería matplotlib para Python, cuyas reconstrucciones se muestran en la figura 15.



(a) (b)



(c)



(d)

Figura 15: Objetos reconstruidos con discriminación de canales. (a) Cabeza; (b) Campana; (c) Cerdo; (d) Cráneo

5. Código Python

En esta sección se comparte el código para reconstrucción desarrollado en Python versión 2.7. Para su funcionamiento se requerirán de las librerías indicadas en la tabla 2.

Tabla 2: Librerías requeridas.

Librería	Versión
numpy	2.2.4
opencv	2.4.13.6
matplotlib	1.16.3

```
1:    import numpy as np
2:    import cv2
3:    from matplotlib import pyplot as plt
4:    from matplotlib import cm
5:
6:    def tic():
7:    #Homemade version of matlab tic and toc functions
8:        import time
9:        global startTime_for_tictoc
10:       startTime_for_tictoc = time.time()
11:
12:   def toc():
13:       import time
14:       if 'startTime_for_tictoc' in globals():
15:           print "Elapsed time is " + str(time.time() - startTime_for_tictoc) + "
      seconds."
16:       else:
17:           print "Toc: start time not set"
18:
19:   nImg = 255#Cantidad de imagenes
20:   ext='.jpg'#Extesion de archivo a leer
21:   #RDI (Region de interes)
22:   x1=100
23:   y1=10
24:   x2=280
```

```
25:   y2=200
26:   r0=-55#Ajuste en caso de que no este centrado el objeto
27:   c0=1#Canal de color a discriminar
28:   cp=0#Contador para puntos generados
29:   size = nImg+1, y2-y1#Tamano de matriz de distancia
30:   matDis=np.empty(size) #Matriz de distancia
31:   matDis[:]=np.NaN
32:   savFig=str(nImg+1)+'_images,'+'_umb'+str(u)+'Disc' #Nombre para figura
33:   tic()
34:   for i in range(0,nImg):
35:       #print 'i:'+i
36:       root_dir='D:/Programas/Python/Reconstrusuper/ReconstruccionLaser/Cabeza/'
37:       readimg=root_dir+'img'+str('{num:03d}'.format(num=i+1))+ext
38:       saveurlbin=root_dir+'binaria/img'+str('{num:03d}'.format(num=i))+ext
39:       saveurlgray=root_dir+'gris/img'+str('{num:03d}'.format(num=i))+ext
40:       saveurlcut=root_dir+'corte/img'+str('{num:03d}'.format(num=i))+ext
41:       saveurl2ch=root_dir+'dobleCanal/img'+str('{num:03d}'.format(num=i))+ext
42:       saveurlredpix=root_dir+'centromasa/img'+str('{num:03d}'.format(num=i))+ext
43:       saveurlint=root_dir+'interpolacion/img'+str('{num:03d}'.format(num=i))+ext
44:       img=cv2.imread(readimg)#Carga una imagen
45:       #cv2.imshow("Imagen original", img)
46:       #img=cv2.cvtColor(img,cv2.COLOR_BGR2LAB)
47:       #cv2.imshow("Imagen original", img)
48:       img=img[y1:y2,x1:x2]#Recorta la region de interes
49:       lineinterp=np.zeros([190,180,3],dtype=np.uint8)
50:       #cv2.imshow("Region de interes", img)
51:       #cv2.imwrite(saveurlcut,img)
52:       #Convierte a 2 espacios de color
53:       img[...,c0] = 0
54:       #cv2.imshow("2 Canales", img)
55:       #cv2.imwrite(saveurl2ch,img)
56:       #Pasa a escala de grises
57:       img=cv2.cvtColor(img,cv2.COLOR_RGB2GRAY)
58:       #cv2.imshow("Gris", img)
59:       #cv2.imwrite(saveurlgray,img)
60:       #Convierte en binario
61:       ret,img=cv2.threshold(img,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
62:       #cv2.imshow("Binario", img)
63:       #guardar los resultados en jpg
64:       #cv2.imwrite(saveurlbin,img)
65:       #Obtiene la posicion donde esta el objeto
66:       line1=np.zeros(np.shape(img))
67:       for posy in range(0,y2-y1):
68:           #suma de fila
69:           row_sum=0
70:           row_sump=0
71:           mean=0.0
72:           for posx in range(0,x2-x1):
73:               #print "(",posx,"),(",posy,")",img[posy][posx]
74:               row_sump+=img[posy,posx]
75:               row_sum+=img[posy,posx]*posx
76:               #print posx,np.float64(posx).dtype,matDis.dtype
77:               if row_sump>0:
78:                   mean=row_sum/row_sump
79:                   #print row_sum," ",row_sump," ",mean
80:                   mean=np.fix(mean)
81:                   matDis[i,posy]=mean+r0
82:                   cp+=1
83:                   line1[posy,int(mean)]=255
84:       #cv2.imwrite(saveurlredpix,line1)
85:       #cv2.imshow("single line", line1)
86:       #print matDis[i,:]
87:       #Espera se oprima esc para siguiente iteración
```

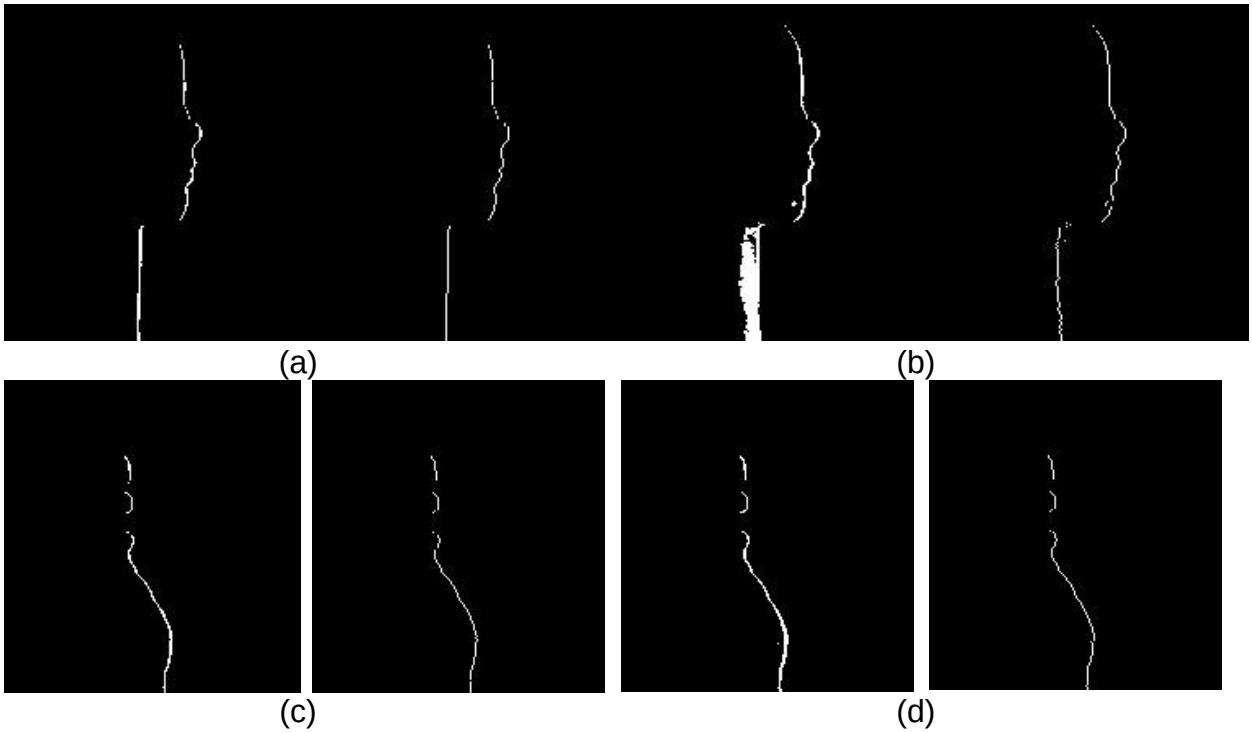
```
88:       cv2.waitKey(0)
89:   #Duplica el primer resultado de la matriz de distancia en la ulitma posicion
90:   matDis[nImg,:]=matDis[0,:]
91:   print "Matriz de distancias final: ",matDis,matDis.shape
92:
93:   #Obtenemos los angulos y se crea la un arreglo con los radianes
94:   degree=360.00/nImg
95:   radian=np.radians(degree)
96:   thetas=np.ones((nImg+1))
97:   for i in range(0,nImg+1):
98:       thetas[i]=radian*(i+1)
99:   thetas=thetas[np.newaxis]
100:  print "Radianes: ",thetas,thetas.shape
101:
102:  #Se crean las matrices para la generacion de la nube de puntos
103:  X=np.zeros((nImg+1,size[1]))
104:  Y=np.zeros((nImg+1,size[1]))
105:  Z=np.ones((nImg+1,y2-y1))
106:  C=np.ones((nImg+1,y2-y1))
107:  #print X,X.shape,thetas.shape
108:
109:  for i in range(0,nImg+1):
110:      for j in range (0,y2-y1):
111:          X[i,j]=(escala*(matDis[i,j]))*((np.cos(thetas[0,i])+dx))
112:          Y[i,j]=(escala*(matDis[i,j]))*((np.sin(thetas[0,i])+dy))
113:  for i in range(0,y2-y1):
114:      Z[:,i]=-(i+1)
115:  toc()
116:
117:  with file('matX.txt', 'w') as outfile:
118:      outfile.write('# Array shape: {0}\n'.format(X.shape))
119:      for data_slice in X:
120:          np.savetxt(outfile, data_slice, fmt='%-7.2f')
121:          outfile.write('# New slice\n')
122:
123:  with file('matY.txt', 'w') as outfile:
124:      outfile.write('# Array shape: {0}\n'.format(Y.shape))
125:      for data_slice in Y:
126:          np.savetxt(outfile, data_slice, fmt='%-7.2f')
127:          outfile.write('# New slice\n')
128:
129:  with file('matZ.txt', 'w') as outfile:
130:      outfile.write('# Array shape: {0}\n'.format(Z.shape))
131:      for data_slice in Z:
132:          np.savetxt(outfile, data_slice, fmt='%-7.2f')
133:          outfile.write('# New slice\n')
134:
135:  matDis2=matDis
136:  matDis2=matDis2[np.newaxis]
137:  np.savetxt('matrizDist.txt', matDis, fmt='%.2f', delimiter='|', newline='\n',
      header='', footer='', comments='# ')
138:  np.savetxt('matrizX.txt', X, fmt='%.2f', delimiter='|', newline='\n', head-
      er='', footer='', comments='# ')
139:  np.savetxt('matrizY.txt', Y, fmt='%.2f', delimiter='|', newline='\n', head-
      er='', footer='', comments='# ')
140:  np.savetxt('matrizZ.txt', Z, fmt='%.2f', delimiter='|', newline='\n', head-
      er='', footer='', comments='# ')

141:  fig=plt.figure()
142:  #fig.suptitle('256 images', fontsize=20)
143:  fig.canvas.set_window_title(savFig)
144:  ax=fig.gca(projection='3d')
145:  #X,Y=np.meshgrid(X,Y)
```

```
146: print "Tamanyo de matricez",X.shape, Y.shape, Z.shape
147: print "Numero puntos: ",cp
148: surf=ax.plot_surface(X,Y,Z, rstride=1, shade='interp', cstride=1, cmap=cm.jet,
149: linewidth=0, antialiased=False)
150:
151: plt.show()
152:
```

6. Resultados

Como resultados, las imágenes mostraron una menor cantidad de ruido en los casos donde se aplica la reducción del ruido. Algunos ejemplos se muestran en la figura 16, donde se aprecia en casos donde el ruido es reducido enormemente como en los casos de la cabeza, el cerdo y el cráneo, y en el caso de la campana llega a existir una ligera mejoría al eliminar un punto cerca de la protuberancia que aparece en la campana en la figura 1. Sin embargo no es del todo infalible, ya que el ruido no es completamente eliminado, además de que llega a existir una ligera pérdida de la información como en la nariz de la cabeza donde se notan algunos puntos faltantes que dejan algunas discontinuidades que no están presentes en la línea de la imagen que no aplicó la discriminación, así como la pérdida en la frente cercana a la punta de la cabeza.

Una posible explicación es que al aumentar la rapidez de los cambios de intensidad en el histograma como aparecen en la figura 13, esto es un decremento de mayor inclinación, facilita la distinción de características en una imagen ya que el contraste es mayor [26], por lo que el tratar de



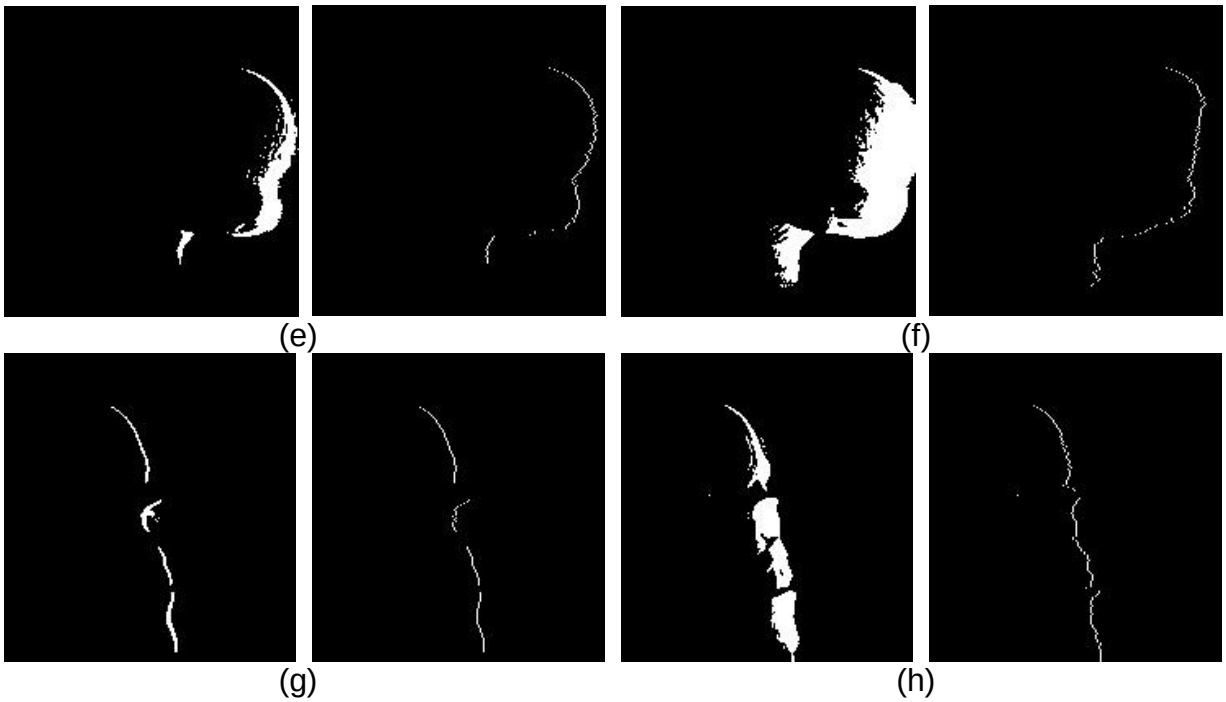
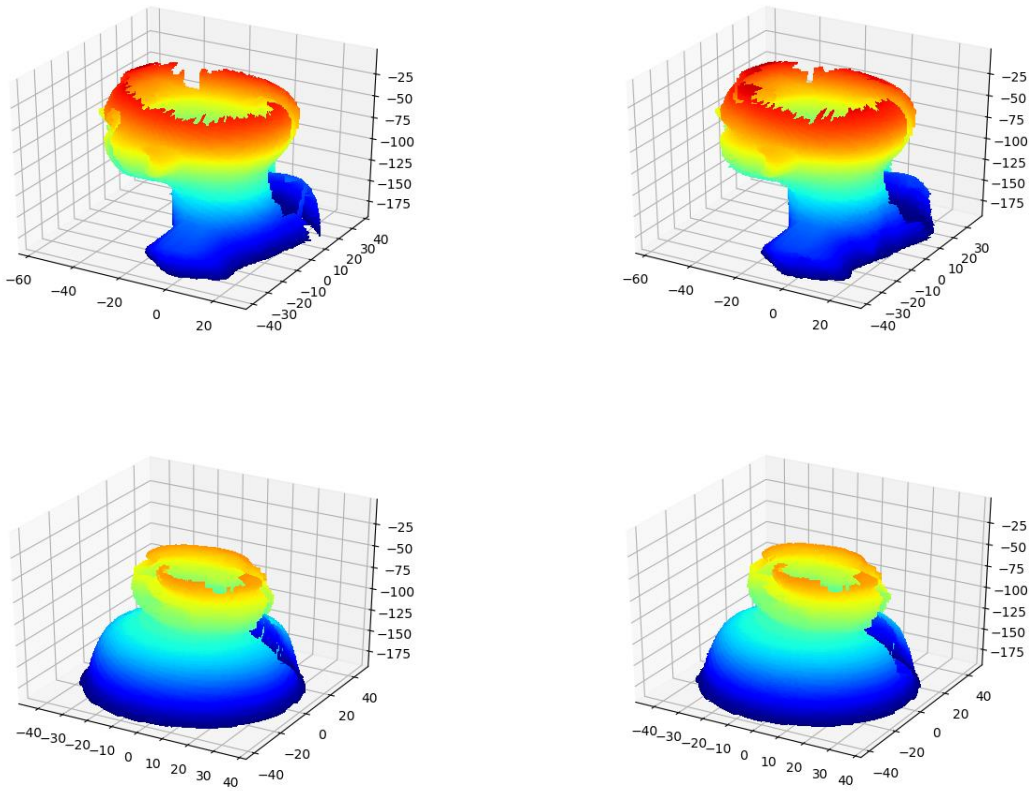


Figura 16: Comparación de binarización y reducción a un píxel. (a) Cabeza con discriminación; (b) Cabeza sin discriminación; (c) Campana con discriminación; (d) Campana sin discriminación; (e) Cerdo con discriminación; (f) Cerdo sin discriminación; (g) Cráneo con discriminación; (h) Cráneo sin discriminación.



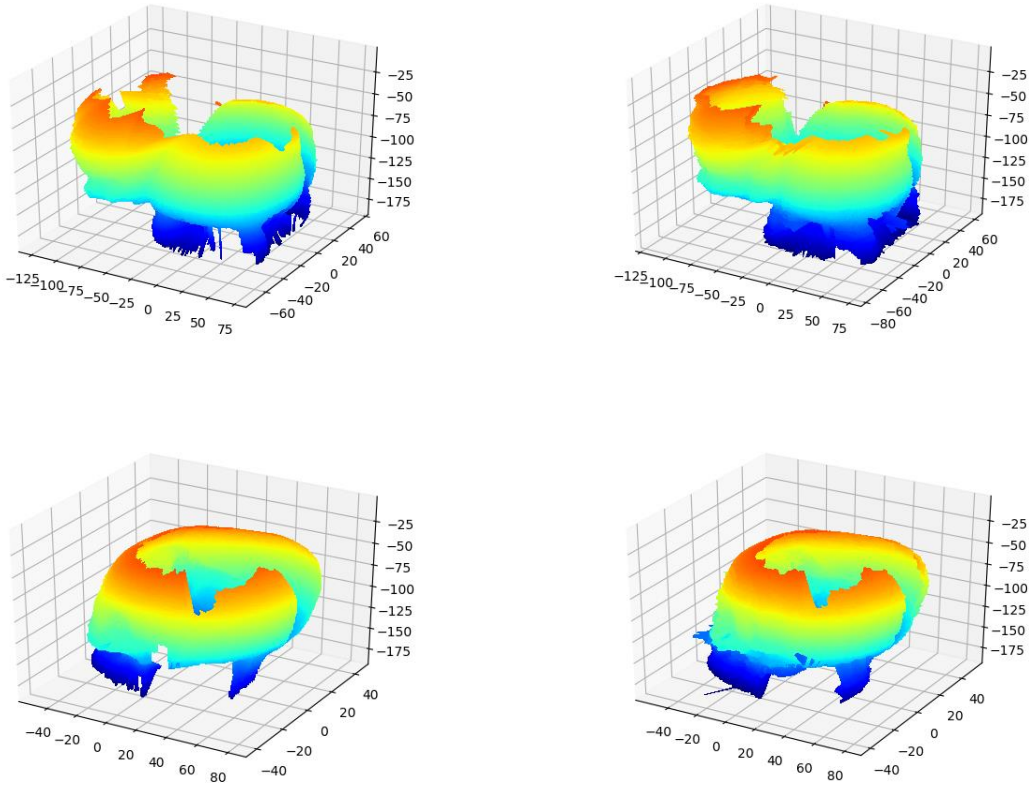


Figura 17: Resultados de reconstrucción. (a) Con discriminación; (b) Sin discriminación.

Las reconstrucciones mostraban digitalizaciones que se asemejan a su contraparte física, sin embargo se encontraron algunas limitaciones en referencia a la librería de Python para la visualización de los modelos 3D. La más notable es la dificultad para manipular la vista una vez reconstruido el objeto, pues este tiende a trabarse y ser muy sensible, por lo que al tratar de hacer una ligera rotación se torna en casi una vuelta, así como por el hecho de no ser instantáneo, no se puede tener un control en el manejo de las vistas, motivo por el cual las imágenes mostradas en la figura 17 son las que aparecen por defecto, mientras que las mostradas en la figura 15 son las mejores vistas que se pudieron obtener para muestra con el método de discriminación. Otro aspecto es que tiende a cortar las partes que estén directamente en el lado derecho de la vista frontal o en la misma vista central dependiendo del tamaño del objeto, por lo que llegan a dar la apariencia de no haber sido reconstruidas.

Tratando de verificar si la discriminación de canales genera un consumo extra que requiere de un poco de tiempo adicional, se midió el tiempo de reconstrucción, desde el momento de carga de imagen hasta el despliegue de la digitalización de las figuras 11 y 13. Y como adición, se contó el número de puntos generados en cada caso para comprobar que tanto podría ser la pérdida de información. Dichos resultados se presentan en la tabla 3.

Basado en los resultados, es posible asumir que hay una pérdida de información por la cantidad de puntos generados, aunque no es posible determinar con exactitud si algunos de los puntos son en realidad, ruido, tal como se muestra en el cráneo sin discriminación de la figura 17 donde se muestran algunas deformaciones no presentes en el objeto físico, notablemente en la base donde una línea está presente y en la parte frontal del cráneo se puede observar una mancha que estaría deformando el objeto si es visto en perfil. Por otra parte se puede encontrar en comparación a cada objeto en contra-

parte al método de discriminación, hay zonas en las que se ven algunos huecos más grandes en contra de las reconstrucciones sin reconstrucción.

Otro aspecto notable fue el tiempo que requirió completar todo el proceso de reconstrucción, con el cual se encontró que la discriminación no parece adicionar tiempo extra, únicamente en el caso de la campana se mostró un tiempo mayor por 5 segundos, sin embargo para el caso del cerdo el resultado fue invertido, mientras que en la cabeza y el cráneo la discriminación sigue el patrón de que requirió tiempo adicional, sin embargo este es bastante despreciable. Destacando que se procesaron un total de 256 imágenes en cada caso.

Tabla 2. Resultados de reconstrucción

	Con discriminación		Sin discriminación	
Objeto	**Tiempo (seg)**	**Puntos**	**Tiempo (seg)**	**Puntos**
Cabeza	34.2739999294	38853	33.8429999352	41236
Campana	40.5290000439	31358	35.3519999981	32281
Cerdo	31.117000103	26883	37.9660000801	32779
Cráneo	31.9400000572	27830	31.5649998188	34945

7. Conclusiones

El método de discriminación de canales produce imágenes con menos ruido, pero a compromiso de una pérdida de información como se puede concluir por la cantidad de puntos generados entre el aplicar el método contra no aplicarlo, sin embargo es de considerar que algunos puntos son inexistentes en el objeto y llegan a ser resultado del ruido mismo. Por ello se considera que se puede usar el canal discriminado para tratar de rellenar las discontinuidades, o bien implementar algún tipo de interpolación que se encargue de ese problema.

También se estaría haciendo pruebas con canales individuales como alternativas siguiendo las pruebas presentadas, así como seguir con las pruebas con el láser azul y encontrar si este es más efectivo.

Se encontró necesario emplear otra librería de visualización para Python o implementarlo en otro lenguaje que permita una mejor visualización de los resultados. También se consideraría exportar los resultados a un formato de modelos 3D como ply u obj y usar una aplicación que permita una mejor visualización de los resultados.

Se ha concluido que el método de discriminación de canales es un método que puede reducir el ruido sin generar carga considerable en el programa, y este se puede implementar fácilmente en cualquier sistema de reconstrucción. El problema de pérdida de información se puede tratar con un método extra que emplea el canal faltante o un método de interpolación.

Referencias

[1] Motavalli, S., Bidanda, B., *“A part image reconstruction system for reverse engineering of design modifications”, Journal of Manufacturing Systems,* Volume 10, Número 5, páginas 383-395, 1991.

[2] Domiter, V., Repnik, B., Žalik, B., Sadžak, A., y Rizvić, S., "Surface reconstruction algorithms in cultural heritage digital representation," *2009 XXII International Symposium on Information, Communication and Automation Technologies*, páginas 1-5, Bosnia, 2009.

[3] Shunwang, X., Zhiyong A., Hui, S., Wubin, F., Lijuan, L. y Shengcai, L., "Laser Line-Scanning Based Customized Shoe-Last System," *2010 International Conference on Electrical and Control Engineering*, páginas 1230-1233, Wuhan, 2010.
[4] Sukvichai, K., Wongsuwan, K., Loraksa, C., y Chantarachit, S., "Simple 3D splint reconstruction using a low cost smart phone line laser 3D scanner," *2018 International Conference on Electronics, Information, and Communication (ICEIC)*, páginas 1-4, Honolulu, HI, 2018.
[5] Yunardi, R. y Imandiri, A., "Design of The 3D Surface Scanning System for Human Wrist Contour Using Laser Line Imaging," *2018 5th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)*, páginas 1-5, Semarang, 2018.
[6] Hasanuddin, M., Permana, G., Akbar, I., y Wuryandari, A. "3D scanner for orthodontic using triangulation method," *2015 International Conference on Electrical Engineering and Informatics (ICEEI)*, páginas 360-364, Denpasar, 2015.
[7] Yang, Y., Zheng, B., Zheng, H., Wang, Z., Wu, G. y Wang, J. "3D reconstruction for underwater laser line scanning," *2013 MTS/IEEE OCEANS - Bergen*, páginas 1-3, Bergen, 2013.
[8] Yang, Y., Zheng, B., Kan, L., Yu, J., y Wang, J. *"3D color reconstruction based on underwater RGB laser line scanning system", Optik*, Volume 125, Número 20, páginas 6074-6077, 2014.
[9] Deng, J., Chen, B., Cao, X., Yao, B., Zhao, Z., y Yu, J. "3D Reconstruction of Rotating Objects Based on Line Structured-Light Scanning," *2018 International Conference on Sensing, Diagnostics, Prognostics, and Control (SDPC)*, páginas 244-247, Xi'an, China, 2018.
[10] Aganj, E., Keriven, R., y Pons, J., "Photo-consistent surface reconstruction from noisy point clouds," *2009 16th IEEE International Conference on Image Processing (ICIP)*, páginas 505-508, Cairo, 2009.
[11] Sánchez, M., Vidal, V., Arnal, J., y Vidal, A., "*Image Noise Removal on Heterogeneous CPU-GPU Configurations*", *Procedia Computer Science*, Volume 29, páginas 2219-2229, 2014.
[12] Ykhlef, F., Arezki, M., Guessoum, A., y Berkani, D., *Adaptive noise reduction using numerically stable fast recursive least squares algorithm*. *International Journal of Adaptive Control and Signal Processing*, 21, páginas 354 - 374, 2007.
[13] Tsuchida, M., Haseyama, M., y Kitajima, H., "*A Kalman filter using texture for noise reduction in SAR images*". *Electronics and Communications in Japan (Part I: Communications)*, 86, páginas 21 - 32. 10.1002/ecja.10065, (2003).
[14] Weickert, J., "*Coherence-Enhancing Diffusion Filtering*", *International Journal of Computer Vision*, Volumen 31, Número 2–3, páginas 111–127, 1999.
[15] Ding, Z., Gore, J., y Anderson, A. "*Reduction of noise in diffusion tensor images using anisotropic smoothing"*, *Magnetic Resonance in Medicine*, volumen 53, Número 2, páginas 485-490, 2005.
[16] Netravali, I., Holt, R., y Webb, C., "*Perceptual Denoising of Color Images*", *International Journal of Imaging Systems and Technology*, Volumen 20, Número 3, Páginas 215-222, 2010.
[17] https://opencv.org/
[18] Otsu, N. "*A threshold selection method from gray-level histograms*", *Automatica*, páginas 23-27, 1975.
[19] Danish, M., Ahmad T., Hashim, R., Said, N., Akhtar, M., Mohamad-Saleh, J., Sulaiman, O., "*Comparison of surface properties of wood biomass activated carbons and their application against rhodamine B and methylene blue dye*", *Surfaces and Interfaces*, Volumen 11, páginas 1-13, 2018.
[20] Wang, D., Khosla, A., Gargeya, R., Irshad, H., Beck, A. H., "Deep learning for identifying metastatic breast cancer", arXiv:1606.05718, 2016.
[21] Puri, M., Hoover, S., Hewitt, S., Wei, B., Adissu, H., Halsey, CHC., Beck, J., Bradley, C., Cramer, S., Durham, A., Esplin, D., Frank, C., Lyle, L., McGill, L., Sánchez, M., Schaffer, P., Traslavina, R., Buza, E., Yang, H., Lee, M., Dwyer, J., y Simpson, R. "*Automated Computational Detection, Quantitation, and Mapping of Mitosis in Whole-Slide Images for Clinically Actionable Surgical Pathology Decision Support*". *J Pathol Inform*. 2019 Feb 7;10:4. doi: 10.4103/jpi.jpi_59_18. PubMed PMID: 30915258; PubMed Central PMCID: PMC6396430.
[22] Tu, W., He, S., Yang, Q., y Chien, S. "Real-Time Salient Object Detection With a Minimum Spanning Tree", *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[23] Happy, S., y Routray, A., "Automatic facial expression recognition using features of salient facial patches," en *IEEE Transactions on Affective Computing*, vol. 6, no. 1, 1 Jan.-March, páginas 1-12 2015.
[24] Shi, Y., Li, X., Zhang, X., Wu, H., y Ma, B. "Reversible data hiding: Advances in the past two decades," en *IEEE Access*, vol. 4, páginas 3210-3237, 2016.
[25] Gehan, M., Fahlgren, N., Abbasi, A., Berry, J., Callen, S., Chavez, L., Doust, A., Feldman, M., Gilbert, K., Hodge, J., Hoyer, J., Lin, A., Liu, S., Lizárraga, C., Lorence, A., Miller, M., Platon, E., Tessman, M., y Sax, T. "*Plantcvv2: image analysis software for high-throughput plant phenotyping*" PeerJ 5 (2017). https://doi.org/10.7717/peerj.4088.
[26] Frei, W. "*Image enhancement by histogram hyperbolization*", *Computer Graphics and Image Processing*, Volumen 6, Número 3, páginas 286-294, 1977.