



**FACULTAD DE INGENIERÍA, ARQUITECTURA Y
URBANISMO**

**ESCUELA ACADÉMICO PROFESIONAL DE INGENIERÍA
DE SISTEMAS**

TESIS

**ESTIMACIÓN DE LA SIMETRÍA DE LA PALTA HASS EN
IMÁGENES TOMOGRAFICAS**

**PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO
DE SISTEMAS**

Autor:

Bach. Sernaque Céspedes Benjamin Augusto

Asesor:

Mg. Samillán Ayala Alberto Enrique

**Línea de Investigación:
Ciencias de la computación**

**Pimentel, Perú
2017**



ESTIMACIÓN DE LA SIMETRÍA DE LA PALTA HASS EN IMÁGENES TOMOGRAFICAS

Aprobación de la Tesis

Ing. Mejía Cabrera Heber Iván
Presidente del Jurado de Tesis

Ing. Fuentes Adrianzen Denny Jhon
Secretario del jurado de Tesis

Dr. Ramos Moscol Mario Fernando
Vocal del Jurado de Tesis

**Pimentel, Perú
2017**



INFORMACIÓN GENERAL

1.1 Título del Informe de Investigación:

ESTIMACIÓN DE LA SIMETRÍA DE LA PALTA HASS EN IMÁGENES TOMOGRAFICAS

1.2 Línea de Investigación:

Ciencias de la computación

1.3 Autor:

Sernaque Cespedes Benjamin Augusto

1.4 Asesor Metodólogo:

Mg. Samillán Ayala Alberto Enrique

1.5 Tipo y diseño de investigación.

Tipo experimental, metodología cuantitativa.

1.6 Facultad y Escuela Académico Profesional:

Facultad de Ingeniería, Arquitectura y Urbanismo
Escuela Profesional de Ingeniería de Sistemas

1.7 Periodo: 2017-II

1.8 Fecha de inicio y término del proyecto:

Abril – Diciembre 2017

1.9 Firma de los autores del proyecto:

Sernaque Cespedes Benjamin

AUTOR

1.10 Aprobado:

Mg. Samillán Ayala Alberto Enrique

ASESOR METODÓLOGO

Ing. Mejía Cabrera Heber Iván

ASESOR ESPECIALISTA

1.11 Fecha de Presentación:

Diciembre del 2017



DEDICATORIA

El presente proyecto de investigación se lo dedico a Dios, ya que sin el nada sería posible en esta vida.

A mis padres que siempre me apoyaron y nunca me dejaron solo a pesar de las diversas situaciones que se me presentaron durante mi etapa de formación estudiantil como profesional.

A mis asesores que me brindaron su apoyo incondicional durante todo el tiempo que conllevo a realizar la presente investigación.

EL AUTOR

AGRADECIMIENTO

Primeramente, a Dios por haber permitido llegar con bien a este momento tan importante en mi vida.

A mis padres por depositar toda su confianza en mí y por nunca dejarme solo; A mi familia que siempre confió en mí y me dieron su apoyo incondicional para poder seguir a delante y nunca rendirme.

A todas las personas que me apoyaron para el desarrollo de este proyecto de investigación como es el docente de área de Proyecto de Tesis de la USS, el ingeniero Víctor Alexci Tuesta Montesa, a mi asesor especialista, el ingeniero Mejia Cabrera Heber Ivan, y al doctor Manuel Forero Vargas que gracias a su ayuda y experiencia hicieron posible el desarrollo satisfactorio de este proyecto de investigación.

EL AUTOR



INDICE

DEDICATORIA.....	4
AGRADECIMIENTO.....	5
RESUMEN	13
ABSTRACT	15
INTRODUCCIÓN	16
CAPITULO I: PROBLEMA DE INVESTIGACIÓN.....	17
1.1. Situación Problemática.....	17
1.2. Formulación del problema	22
1.3. Delimitación de la Investigación	22
1.4. Justificación e importancia de la investigación	23
1.5. Limitación de la Investigación.....	23
1.6. Objetivos de la Investigación.....	24
1.6.1 Objetivo general	24
1.6.2 Objetivo específico.....	24
CAPITULO II: MARCO TEÓRICO	25
2.1. Antecedentes de Estudios.....	25
2.2. Estado del Arte.....	26
2.3. Base teórica científicas	29
2.3.1. Simetría.....	29
2.3.2. Geometría Computacional	30
2.3.3. Imagen Digital	31
2.3.4. Tipos de Imágenes.....	31
2.3.5. Procesamiento digital de imágenes.....	32
2.3.6. Morfología de imágenes.....	33
2.3.7. Modelos de Color	34
2.3.8. Correlación.....	35
2.3.9. Vectores propios y valores propio	36
2.3.10. Análisis de Componentes Principales (PCA)	36
2.3.11. Momentos estadísticos:	38
2.3.12. Métodos de Enfriamiento.....	38



2.3.13.	Imágenes Médicas Digital:.....	39
2.3.14.	Tipos de imágenes médicas	39
2.3.15.	Formato Dicom (Digital Imaging and Communications in Medicine)	40
2.4.	Definición de términos básicos	41
CAPITULO III:	MARCO METODOLÓGICO	44
3.1.	Tipo y Diseño de Investigación.....	44
3.3.1.	Tipo de Investigación	44
3.3.2.	Diseño de Investigación.....	44
3.2.	Población y Muestra.....	44
3.3.	Hipótesis	44
3.4.	Variables	45
3.5.	Operacionalización.....	45
3.6.	Abordaje metodológico, técnicas e instrumentos de recolección de datos	45
3.6.1.	Abordaje metodológico	45
3.6.2.	Técnicas de recolección de datos.....	46
3.6.3.	Instrumentos de recolección de datos.....	46
3.7.	Procedimientos para la recolección de datos	47
3.8.	Análisis estadísticos e interpretación de los datos.....	47
3.9.	Principios éticos	48
3.10.	Criterios de rigor científico	49
CAPITULO IV:	Análisis e interpretación de resultados	50
4.1.	Resultados en Tablas y Gráficos.....	50
4.1.1.	Pre procesamiento de Imágenes:.....	50
4.1.2.	Alineación del stack de imágenes	50
4.1.3.	Fidelidad de la simetría	51
4.1.4.	Variabilidad de los resultados de simetría	55
4.2.	Discusión de resultados	55
CAPITULO V:	PROPUESTA DE INVESTIGACIÓN	57
5.1.	Introducción	57
5.2.	Desarrollo.....	57
5.2.1.	Identificación de las características de las paltas	57



5.2.2.	Construir un prototipo para ambiente controlado de adquisición de imágenes digitales de la palta Hass.....	59
5.2.3.	Obtención de Tomografías de las paltas	61
5.2.4.	Peso de la pulpa, cascara y pepa de la palta	63
5.2.5.	Obtención de imágenes fotográficas a las paltas	64
5.2.6.	Recorte de las paltas en la tomografía	65
5.2.7.	Conversión de una imagen RGB a escala de grises	71
5.2.8.	Aplicando la técnica de segmentación para la palta	72
5.2.9.	Segmentando la pepa en las imágenes recortadas.....	80
5.2.10.	Aplicando la técnica de PCA.....	81
5.2.11.	Calculo de los ángulos directores de las componentes principales del stack de la palta	89
5.2.12.	Alienación del stack de imágenes de la palta	92
5.2.13.	Hallando el contorno de la palta en la pila de imágenes	95
5.2.14.	Obtendiendo la nube de puntos del objeto.....	100
5.2.15.	Entendiendo el método de los momentos estadísticos	101
5.2.16.	Validación del método de simetría	105
5.2.17.	Método de puntos equidistante.....	109
5.2.18.	Deteccion rápida de simetría	111
5.2.19.	Aplicación del método de simetría en las imágenes tomograficas de la palta Hass	115
1.	Método de los momentos estadísticos.....	115
2.	Método de puntos equidistantes.....	117
3.	Detección de simetría rapida.....	118
CAPITULO VI: Conclusiones y Recomendaciones.....		120
6.1.	Conclusiones.....	120
6.2.	Recomendaciones	121
Referencias		122



INDICE DE FIGURAS

Figura 1: Crecimiento anual de la exportación de la palta Hass en el Perú.....	18
Figura 2: Exportación de palta 2016-2017.....	19
Figura 3: Imágenes de muestra de simetría.....	30
Figura 4: Etapas del procesamiento digital de imágenes.....	33
Figura 5: Diagrama Cromático RGB	35
Figura 6: Ejemplos de Correlación	36
Figura 7: Ejemplo de una Componente Principal	37
Figura 8: Esquema de toma radiográficas	39
Figura 9 Estructura de un Archivo DICOM	41
Figura 10: Imagen de la palta Hass (a) y su pepa (b).....	59
Figura 11: Prototipo para la toma fotográfica de la palta Hass	60
Figura 12: Prototipo final para la toma de fotografías de la palta Hass.....	61
Figura 13: Prototipo para el transporte de la palta Hass para las Tomografías.....	62
Figura 14: Visualización de las tomografías	63
Figura 15: Balanza Utilizada para pesar la Palta	64
Figura 16: Formato de nombre de carpeta para guardar las paltas.....	65
Figura 17: Formato para guardar las paltas.....	65
Figura 18: Visualización del corte coronal de la tomografía	66
Figura 19: Apilación de los cortes de la tomografía.....	67
Figura 20: Visualización del Stack de imágenes	68
Figura 21: Visualización completa de las paltas en la tomografía.....	68
Figura 22: Selección de la palta en la Tomografía.....	69
Figura 23: Identificación de fotograma de la palta.....	70
Figura 24: Generación de la pila de imágenes de la palta	70
Figura 25: Palta Hass en escala RGB.....	71
Figura 26: Palta en escala de Grises	72
Figura 27: Código de conversión de RGB a escala de grises.....	72
Figura 28: Imagen tomográfica de la palta	74
Figura 29: Histograma de la Palta	75
Figura 30: Ubicación del fotograma de la palta para la segmentación	77
Figura 31: Valores para la segmentación.....	78
Figura 32: Segmentación de la palta.....	79
Figura 33: Aplicación de relleno de regiones en la palta 2.....	80
Figura 34: Segmentación de la pepa de la palta 2.....	81
Figura 35: Ejemplo de pila de imágenes para la aplicación de PCA.....	81
Figura 36: Implementación de la media en código java.....	83
Figura 37: Implementación de la matriz de covarianza	85
Figura 38: Vectores Propios de la Matriz de Covarianza	88
Figura 39: Vector 1 en el plano X,Y.....	89



Figura 40: Ángulo director en el eje X.....	90
Figura 41: Ángulo director en el eje Y.....	90
Figura 42: Ángulo director en el eje Z.....	91
Figura 43: Implementación del código para calcular el ángulo director.....	91
Figura 44: Aplicación del método PCA y cálculo de los ángulos directores en la palta 2.....	92
Figura 45: Palta Nro. 2 con sus ángulos directores.....	93
Figura 46: Rotación de la imagen con TransformJ en el eje Z (a) stack original, (b) stack rotado, (c, d, e) imágenes que conforman el stack de la imagen rotada.....	94
Figura 47: Palta 2 con sus nuevos ángulos directores.....	94
Figura 48: Rotación de la Imagen en el eje X (a) imagen rotada en el eje Z (b) imagen rotada en el eje X.....	95
Figura 49: Proceso para hallar el pixel Central.....	96
Figura 50: Código para encontrar el pixel central de la imagen.....	98
Figura 51: Imagen del pixel central.....	98
Figura 52: Las 8 direcciones del plano cartesiano.....	98
Figura 53: Palta 2 con su representación matricial para la detección del borde.....	99
Figura 54: Desplazamiento para encontrar la nueva dirección a moverse.....	99
Figura 55: Borde de la palta Hass en el stack de imágenes.....	100
Figura 56: Nube de puntos del contorno de la palta : a) imagen del contorno de la palta b) imagen hallando el borde cada 5° c) imagen de la nube de puntos.....	101
Figura 57: Imagen de ejemplo para hallar los momentos estadísticos.....	102
Figura 58: Imágenes sintéticas de objetos no simétricos: 1-10 imágenes normales y la 11 stack de 4 imágenes.....	106
Figura 59: Imágenes sintéticas de objetos simétricos.....	106
Figura 60: Imagen de ejemplo de la Palta Hass con los puntos equidistantes la punto central del objeto en estudio: a) palta segmentada b) punto central del objeto c) distancias entre puntos de cada eje X e Y.....	111
Figura 61: Transformada de Hough de un punto.....	112
Figura 62: Votación de pixel por línea de simetría con parámetros R y θ	113
Figura 63: Rotacion de pixeles del borde con respecto al centro de la imagen con un angulo θ , donde las coordenadas horizontales de los pixeles que han sido girados son insertados en la matriz Rot....	114
Figura 64: Algoritmo de Deteccion de Simetria.....	115
Figura 65: Imágenes de las paltas Hass.....	116
Figura 66: Imagen de la palta Hass segmentada.....	117
Figura 67: Separacion de stack de imágenes a imágenes individuales.....	118
Figura 68: Resultado despues de aplicar el algoritmo: a) Representa la línea de simetria arrojado por algoritmo, b) y d) Representan los puntos arrojados por el algoritmo, c) Representa el punto central de la imagen.....	119
Figura 69: Selección de tipo de proyecto a crear.....	130
Figura 70: Ubicación del archivo fuente XML.....	131
Figura 71: Selección de archivos para los plugins e interface grafica de ImageJ.....	132



Figura 72: Selección de JDK para el proyecto	133
Figura 73: Configuración de archivo XML.	134
Figura 74: Compilación del proyecto.....	134
Figura 75: Selección de Classpath del proyecto.	135
Figura 76: Agregando Jar para los plugin.....	136
Figura 77: Ejecución del programa.	136
Figura 78: Creación de clase a crear.....	137
Figura 79: Código a implementar para nuevo plugin.	138
Figura 80: Compilación de plugin.....	139
Figura 81: Modificación de línea para no crear un nueva carpeta al compilar el plugin.....	139
Figura 82: Verificación de plugin creado en el menú de Plugin de ImageJ	140



INDICE DE TABLAS

Tabla 1	Tabla de tiempos promedio empleado por las técnicas utilizadas	50
Tabla 2	Cantidad de iteraciones por cada palta.....	51
Tabla 3	Resultado de evaluación de imágenes simétricas y no simétricas	52
Tabla 4	Tabla de resultado de simetría de cada palta por eje	52
Tabla 5	Cantidad de paltas simétrica y no simétricas por cada eje X e Y	53
Tabla 6:	Resultados de simetría en eje X e Y así como sus diagonales	54
Tabla 7	Variabilidad de Resultados de Simetría	55
Tabla 8	Época en el que se siembra la palta Has en el Norte del Perú.....	58
Tabla 9	Calibres de exportación a los EE. UU	58
Tabla 10	Tabla de resultados de frecuencia de intensidad de pixeles	74
Tabla 11	Probabilidad de Intensidad	76
Tabla 12	Cálculo de la media de clases	77
Tabla 13	Valores en x e y de la figura 57 y aplicación del primer momento estadístico	102
Tabla 14	Aplicación del segundo momento estadístico	103
Tabla 15	Aplicación de tercer momentos estadístico.....	104
Tabla 16	Asimetría de los datos	105
Tabla 17	Cuadro de resumen después de aplicar los 3 momentos estadísticos	105
Tabla 18	Resultado de imágenes asimétricas	107
Tabla 19	Resultados de imágenes simétricas	107
Tabla 20	Tabla de resultados de simetría en figuras asimétricas	108
Tabla 21	Tabla de resultados de simetría en figuras simétricas.....	108
Tabla 22	Tabla de resultados de simetría de cada palta evaluada	116
Tabla 23	Tabla de resultados de simetría en cada palta evaluada	117



RESUMEN

El presente trabajo de investigación se estima la simetría de la palta Hass en imágenes tomográficas, la cual ayudará a mejorar el modelo de transferencia de calor para que se adapte mejor a este tipo de fruto con semilla.

Las empresas dedicadas a la exportación de frutas emplean métodos de conservación como es el método de enfriamiento para mantener en buen estado el producto; este método de conservación toma como referencia la forma geométrica que presenta el producto para poder estimar correctamente el tiempo de enfriamiento que necesita.

En tal sentido, existen muchos modelos matemáticos, los cuales se basan en formas geométricas como es la esfera, elipses, cilindros, para poder calcular el tiempo de enfriamiento que requiere el producto.

La palta es una fruta que posee una variada forma geométrica y además tiene una pepa en su interior, lo cual hace que, el método de enfriamiento no sea el adecuado causando daños en la presentación física del producto.

Por ello, en la presente investigación se ha utilizado imágenes tomográficas de la palta del tipo Hass, las mismas que han pasado por una etapa de pre procesamiento para poder aplicar la técnica de PCA y hallar los ángulos directores para hacer la rotación de las imágenes y tener alineados los ejes X, Y, Z de las imágenes.

Se utilizó el código implementado por el Dr. Manuel Forero Vargas para hallar el contorno del objeto en la imagen y generar la nube de puntos para luego aplicar el método de los momentos estadísticos y así poder estimar los valores de simetría que presenta la palta del tipo Hass.

Para poder validar este método se utilizó 11 imágenes sintéticas simétricas con el fin de saber si los datos arrojados son los correctos; una vez que se validó el método se procedió a aplicarlo al stack de imágenes de la palta Hass dando como resultado que, la palta presenta una simetría promedio de 0.0080533 en el eje X y -0.00026485 en el eje Y; por lo que se puede afirmar que, la palta es simétrica.



Palabras claves: procesamiento de imágenes, imágenes tomográficas, pre procesamiento, simetría, ángulos directores, momentos estadísticos.



ABSTRACT

The present work of investigation esteems the symmetry of the Hass avocado in tomographic images, which will help to improve the model of heat transfer so that it adapts better to this type of fruit with seed.

The companies dedicated to the export of fruit use conservation methods such as the cooling method to keep the product in good condition; This method of conservation takes as a reference the geometric shape that the product presents in order to correctly estimate the cooling time it needs.

In this sense, there are many mathematical models, which are based on geometric shapes such as the sphere, ellipses, cylinders, to calculate the cooling time required by the product.

The avocado is a fruit that has a varied geometric shape and also has a pepa inside, which means that the cooling method is not adequate causing damage to the physical presentation of the product.

Therefore, in the present investigation, tomographic images of the Hass avocado have been used, the same ones that have gone through a pre-processing stage in order to apply the PCA technique and find the principal angles to rotate the images and have aligned the X, Y, Z axes of the images.

The code implemented by Dr. Manuel Forero Vargas was used to find the outline of the object in the image and generate the point cloud to then apply the method of statistical moments and thus be able to estimate the symmetry values presented by the Hass type avocado.

In order to validate this method, 11 symmetric synthetic images were used in order to know if the data thrown are correct; Once the method was validated, it was applied to the image stack of the Hass avocado resulting in that the avocado presents an average symmetry of 0.0080533 on the X axis and -0.00026485 on the Y axis; so it can be said that the avocado is symmetrical.

Keywords: image processing, tomographic images, pre-processing, symmetry, director angles, statistical moments.



INTRODUCCIÓN

La palta es una fruta que tiene múltiples usos por ejemplo en la nutrición del ser humano ya que proporciona nutrientes como vitamina B6, fibra, grasa entre otros; en la cosmetología como producto de belleza para la piel.

En este sentido, varios países como Perú, México, Colombia, EE. UU entre otros exportan y producen este tipo de fruto; el Perú desde que firmó el tratado de libre comercio ha tenido un crecimiento constante en la exportación y producción de esta fruta, llegando a ocupar el 3er lugar como país productor de este fruto en el año 2013.

Las empresas actualmente emplean métodos de conservación como es el enfriamiento o congelación; estos métodos de conservación toman en cuenta la forma geométrica que presenta el fruto para poder estimar el tiempo y la temperatura que requiere un producto para que conserve su calidad; en el caso de frutas con semillas como es la palta del tipo Hass, al aplicar un método de enfriamiento puede causar daños físicos debido a la semilla que lleva internamente, lo cual impide que, la transferencia de calor no se la adecuada causando daños en la presentación del producto.

En este aspecto, es necesario modificar el modelo de transferencia de calor para que se adecue a este fruto y así poder conservar su calidad; por ello en esta investigación se estima que tan simétrica es la palta Hass en imágenes tomográficas; en tal sentido se ha empleado diversas técnicas como PCA, vectores y valores propios de una matriz, entre otras con el fin de alinear los ejes X, Y, Z en la imagen para luego hacer uso de los momentos estadísticos como método para estimar la simetría.

Para saber si los datos arrojados por el método de los momentos estadísticos son correctos, se ha hecho uso de 11 imágenes simétricas y asimétricas; estas imágenes permiten obtener datos verdaderos de alta precisión debido a la uniformidad del objeto.

Los Resultados que arroje el método de los momentos estadísticos se compararán con la expresión $x=0$ para saber si el objeto es simétrico o no; por ello los datos deben encontrarse próximos a 0.



CAPITULO I: PROBLEMA DE INVESTIGACIÓN

1.1. Situación Problemática

Según el Ministerio de Agricultura y Riego (MINAGRI-DGPA, 2015), en su informe “La Palta Peruana, Producto Estrella de Exportación”, afirma que la palta aporta un alto grado de ácidos grasos mono insaturado, grasa, fibra, vitamina B6, potasio, calorías, ácidos grasos poliinsaturados y agua, siendo fundamental para la nutrición del ser humano. La palta es un fruto que tiene múltiples usos, por ejemplo, la pulpa se utiliza en la cosmetología, entre otros usos.

Dada la importancia de este fruto, en la actualidad varios países producen y exportan la variedad Hass, en el Perú después que se firma el Tratado del Libre Comercio(TLC) con los países de gran economía mundial como lo es Estados Unidos; en el año 2006, el Perú logró ocupar el 7mo lugar como país productor y exportador (MINAGRI, 2008).

En el año 2012, el Perú logró alcanzar el 6to lugar como país con más áreas cosechadas de palta a nivel mundial y como país productor de esta fruta, así mismo, en el año 2013, el Perú alcanzó el 3er lugar como país productor de este fruto y el 1er lugar como país proveedor, estos datos permiten conocer el crecimiento sostenible que ha tenido el Perú en el mercado mundial después que se firmó el TLC. (MINAGRI-DGPA, 2015)

Asimismo, en el año 2018, el Perú alcanzó posicionarse como 2do país proveedor a nivel mundial de este fruto, al obtener un ingreso de US\$ 580 millones. (MINAGRI,2018)

Al ver el crecimiento constante de la exportación de este fruto, las empresas peruanas comenzaron a exportar más este tipo de fruta como se puede observar en la figura 1,2 y 3, y en su afán de conservarlas en condiciones óptimas para maximizar su exportación, emplean métodos de conservación como es el enfriamiento o congelación, lo que permite que esta fruta esté en buen estado, en ese aspecto, las temperaturas que se le apliquen al producto,



juegan un papel muy importante en cuanto a la calidad y buena presentación del fruto, es por ello que antes de elegir un método de conservación se debe tomar en cuenta las características organolépticas de los productos y seleccionar el que se adecue mejor (Gómez Sánchez, Cerón Carrillo, Rodríguez Martínez, & Aguilar Vázquez, 2007)

Exportación de la palta Hass entre el año 2007-2013



Figura 1: Crecimiento anual de la exportación de la palta Hass en el Perú
Fuente: INEI (2014) Exportación de paltas creció en promedio 20,4% por año

Exportación de la palta a nivel internacional

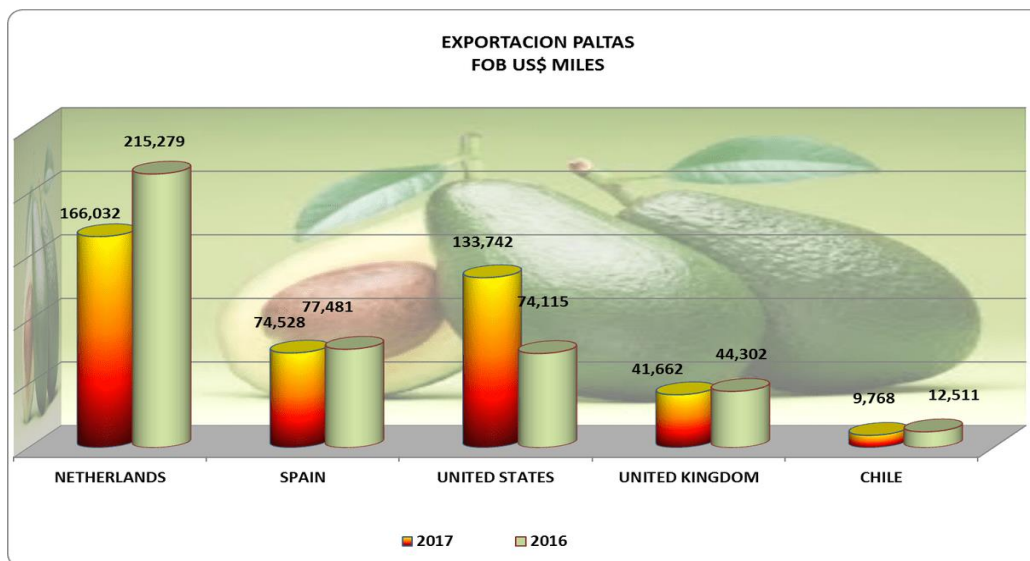


Figura 2: Exportación de palta 2016-2017

Fuente: AGRODATAPERU (2017) Exportación de la palta

Para Barboza-Corona, Vázquez-Acosta, Salcedo-Hernández, & Bautista-Justo, (2004), una de las causas del deterioro en las frutas se da por contaminantes no intencionales, por ejemplo, los fertilizantes, pesticidas que se le agrega durante el proceso producción, y por contaminantes intencionales, en los cuales, se le agrega algún químico al producto para su correcta conservación. Otra de las causas que produce el deterioro de las frutas, es por su variada forma geométrica y de tamaño variado que presenta cada una, ya sea del mismo tipo o no, como por ejemplo la palta del tipo Hass, lo que impide que se le aplique una enfriamiento rápido y adecuado. Gómez Sánchez et al., 2007.

En este aspecto, Machado Velasco & Vélez Ruiz (2008), mencionan que se han realizado varios estudios durante los últimos 20 años sobre la congelación de alimentos, como ejemplos de investigación presenta a: Navarro y col. (1995), quienes analizaron como la velocidad de congelación influye en las pastas alimenticias. Agnelli y Mascheroni (2002) estudiaron los efectos que causa la congelación en algunos alimentos, tales como camarón, fresas y moras, etc., para lograr ello, utilizaron ciertos parámetros de calidad, color y textura.

Ante esta situación Reyes Pérez & Sosa Morales, (2013), plantean 3 métodos de conservación que se basan en la transferencia de calor: la conducción, convección y radiación, los cuales se le aplica a alimentos con tratamiento térmico.

Cuando se le aplica el método de enfriamiento o congelación a un producto alimenticio sólido, el mecanismo predominante de transferencia de calor es el llamado conducción, en donde la transferencia de calor se dispersa progresivamente debido a la agitación molecular en el producto alimenticio, esto genera que el método mencionado anteriormente sea un sistema de transferencia interno lento. Al aplicar este tipo de sistema a un producto conservado, se hace indispensable que las industrias alimentarias analicen los



equipamientos y hagan los cálculos necesarios de calor o frío que se le va a aplicar al producto con valores de necesidad real. (Virsedá & PINAZO, 1998) Virsedá & PINAZO, (1998) explican que, para poder calcular la transferencia de calor, existen modelos aplicables a procesos de enfriamiento sin y con cambio de fase. Dentro de los modelos sin cambio de fase encontramos a los métodos numéricos, y los métodos analíticos. Los métodos numéricos fueron estudiados desde 1978 hasta 1995, donde se utilizó el método de los factores de respuesta, para la elaboración de programas informáticos, los cuales se utilizaban en productos agroalimentarios basados en formas geométricas simples (cilindros y esferas).

En los modelos de transferencia de calor aplicables a procesos de enfriamiento con cambio de fase, el problema de enfriamiento o congelación de alimentos ha sido ampliamente estudiado y se ha planteado numerosos modelos matemáticos para poder determinar los tiempos de enfriamiento o congelamiento, que se adecuen al producto en conservación, basados en su mayor parte en el cálculo de diferencia finita y elemento finito, aunque se ha desarrollado también métodos basados en la integración de la ecuación diferencial ordinaria para cambio de fase. Aunque el modelo físicamente más realista para la congelación de materiales biológicos sólidos es el de la transmisión de calor con propiedades térmicas variables. (Comini, Del Guidice, Lewis, & Zienkiewicz, 1974) citado por Virsedá & PINAZO (1998)

Para poder aproximar el tiempo de enfriamiento, se utiliza la ecuación de Planck la cual fue desarrollada para sistemas ideales en donde se emplean constantes, que dependen de la dimensiones y forma del producto a enfriar. (Heldman & Hartel, 1997) citado por Gómez Sánchez et al., (2007), establecieron valores estándares para las constantes, basándose en tres formas de productos más comunes como son: El plato infinito, el Cilindro Infinito y la Esfera. En donde se puede observar que los productos que poseen forma más esférica, necesitarán menores tiempos de enfriamiento que los productos con forma cilíndrica; y este



último obtendrá valores más bajos de tiempos de enfriamiento que los productos con forma de placa. (Gómez Sánchez et al., 2007).

Diversos investigadores propusieron sus modelos basándose en la forma del producto, por ejemplo (Pham, 1991), creó un modelo para calcular tiempos de enfriamiento tomando como referencia la forma elipsoidal del producto. Así mismo (Hossian, Cleland, & A.C., 1992) crearon una expresión para factor geométrico con formas elipsoidales aplicable a soluciones semi-analíticas de cambio de fase, ampliándolo posteriormente a tres dimensiones. (Virveda & PINAZO, 1998)

Una técnica muy utilizada para simular la transferencia de calor y/o de masa es la utilización del método analítico, el cual fue desarrollado por Josep Fourier (1822 - 1830), con la limitación de uso exclusivamente para formas regulares como (placa infinita, cilindro infinito, esfera) aunque se pueden obtener formas finitas como rectángulos y cubos mediante la intersección de dos y tres placas infinitas respectivamente y cilindros finitos, mediante la intersección de una placa infinita y un cilindro infinito.

Debido a que los métodos de conservación se basan en las formas geométricas de las frutas, estos pueden afectar la calidad de los productos, en este sentido, la visión artificial ha cobrado gran importancia en el campo de la industria agroalimentaria, como por ejemplo las industrias cárnicas, de conservación, permitiendo a través del procesamiento y análisis de imágenes, maximizar la valoración de la calidad de los alimentos, dando así, la posibilidad de aplicar la ingeniería artificial en múltiples tareas, como son, la clasificación por forma, tamaño. (Sánchez, Arizcuren, Abril, & Casp, 2004).

Como se puede apreciar, los modelos que se utilizan para poder estimar los tiempos de congelamiento/enfriamiento toman como base principal la forma que tiene el producto, lo cual hace que, los cálculos sean estimaciones aproximadas, debido a que, se basan en figuras geométricas como esferas, elipses, cilindros, por lo que, los resultados son aproximaciones. Se sabe que



los productos agroindustriales en especial, la palta en su variedad Hass, se asemeja a figuras geométricas conocidas, pero presenta múltiples variantes en sus formas, incluso esta fruta tiene una pepa en su interior, por lo que un cálculo de enfriamiento sin tener en cuenta las características de este fruto, no estaría siendo el más adecuado, es por ello, que para poder ser más exactos en la simulación de los procesos que involucren transferencia de calor en este fruto, surge la necesidad de analizar y evaluar las formas geométricas que presenta la palta en su variedad Hass, para solucionar este problema se propone la evaluación de la simetría de la palta Hass en imágenes tomográficas mediante la técnica de análisis de componentes principales.

1.2. Formulación del problema

¿De qué manera se podrá estimar adecuadamente la simetría de la palta Hass en imágenes tomográficas?

1.3. Delimitación de la Investigación

- El proyecto de investigación se realiza en los ambientes del laboratorio de investigación en sistemas inteligentes y seguridad informática (LABSIS) perteneciente a la escuela profesional de ingeniería de sistemas.
- Se tomarán 18 fotos a 19 paltas, cada fotografía será obtenida con ángulos de 10 grados de diferencia, bajo ambiente controlado en un prototipo diseñado.
- Se utilizarán técnicas de pre procesamiento para procesar las imágenes tomográficas de las paltas Hass.
- La investigación es un trabajo en conjunto con la universidad de Ibagué de Colombia, con la finalidad de plantear un modelo de transferencia de calor para frutos con pepa utilizando imágenes Tomográficas.
- El periodo de desarrollo del tema de investigación se lleva a cabo desde abril hasta diciembre del 2017.



1.4. Justificación e importancia de la investigación

La presente tesis es pertinente debido a que, en la actualidad las empresas que exportan la palta del tipo Hass en el Perú emplean el método de enfriamiento para mantener en buen estado el producto; este método de conservación toma como referencia la forma geométrica del producto para calcular el tiempo que se necesita de transferencia de calor.

La palta por ser un fruto de variada forma geométrica y con pepa en su interior hace que este método de conservación no sea el adecuado causando daños en la misma, por lo que es necesario hacer una modificación al modelo de transferencia de calor para que se adecue este tipo de fruto; por ello la estimación de la simetría de la palta Hass en imágenes tomográficas permitirá hacer los ajustes necesarios al modelo de transferencia de calor lo que beneficiara a reducir las pérdidas en la exportación de la palta.

Es viable debido a que existe información acerca del tema investigado, existen algoritmos y programas que ayudan al procesamiento de imágenes tomográficas, se cuenta con las herramientas matemáticas para hacer cálculos y gráficos

1.5. Limitación de la Investigación

Una limitación principal que se tuvo es la de conseguir las paltas del tipo Hass debido a que solo las venden las empresas exportadoras de este fruto y los fondos que proveen las mismas; otra de las limitaciones para esta investigación es que no se contó con el equipo necesario para realizar las tomografías de las paltas, lo cual conlleva a realizar las tomas en los centros de salud; este tipo de imagen solo está permitido para las personas y no para frutas haciendo que el coste de este tipo de imagen sea más elevado; otra limitación es la falta de conocimiento en la estimación de la simetría mediante el método planteado.



1.6. Objetivos de la Investigación

1.6.1 Objetivo general

Estimar la simetría de la palta Hass en imágenes tomográficas

1.6.2 Objetivo específico

- a) Identificar las características de la palta Hass
- b) Construir un prototipo para ambiente controlado de adquisición de imágenes digitales de la palta Hass.
- c) Construir la base de datos de imágenes a partir de las 12 paltas Hass, a ángulos de 10 grados y 12 imágenes tomográficas.
- d) Pre procesar las imágenes tomográficas de la palta Hass
- e) Aplicar la técnica de PCA
- f) Validar el método de simetría.
- g) Aplicar el método de los momentos estadísticos.



CAPITULO II: MARCO TEÓRICO

2.1. Antecedentes de Estudios

López Del Alamo (2014), en su tesis para optar por el grado de doctor en ciencias de la computación “Análisis de cuerpos no rígidos usando algoritmos de difusión para la detección de simetría en modelos 3D”, propuso 3 algoritmos que ayudan a la detección de la simetría; el primero fue para determinar patrones de simetría en cuerpos no rígidos sin la necesidad de depender de la postura y escala, el segundo algoritmo estuvo basado en la difusión de calor y ondas para hallar los componentes clave (key components) en una imagen 3D y el tercero para hallar los puntos clave(key points) basándose en el análisis de las mallas triangulares; con el fin de cumplir con su objetivo utilizó la ecuación de Head kernel Signature(HKS) y la de Wave Kernel Signature(WKS) para poder hallar los key component y el uso de análisis de triángulos para hallar los puntos de interés; para probar la eficiencia de estos métodos, utilizó la base de datos de benchmark, la cual está compuesta por 3 modelos (Humano, perro y caballo), obteniendo como resultado que, el algoritmo propuesto para hallar los key point consume menos recursos computacionales, simplifica la realización de cálculos complejos y los resultados son similares al método de 3D Harris y el HKS, en el caso del algoritmo para hallar key component arrojó que, cuando el algoritmo se utiliza junto al WKS, este último es más robusto que el HKS para hallar las key component sin la necesidad de depender del uso de la transformación; por último, para evaluar el algoritmo de detección de simetría utilizó la misma base de datos usada en los algoritmos anteriores y trabajó con otra base de datos de modelos de rostros donde se observó que, el método de la distancia de difusión calcula la simetría en modelos con simetría extrínseca mientras que el método de la distancia geodésica no detecto bien la simetría; el autor llegó a la conclusión que, sus métodos son robustos frente a otros métodos propuestos en la literatura consumiendo menos recursos computacionales.



Agustí Melchor (2016), en su tesis doctoral “Análisis y clasificación de imágenes repetitivas mediante técnicas de simetría computacional” propone una Arquitectura de Sistema de recuperación de imágenes por contenido basado en el uso de la simetría computacional; con el fin de lograr la determinación de la existencia de simetrías dependiendo de la escala de observación, las cuales presentan un margen de error en caso de no determinar su forma absoluta; para lograr esto, tuvo que identificar los tipos y patrones de simetría que pueden haber en una imagen, evaluar los diferentes clasificadores de grupos de simetría; teniendo como resultado que, el modelo propuesto para la clasificación de imágenes frente a los métodos que hay en la literatura no genera ambigüedades.

2.2. Estado del Arte

Li, Zhang, & Kleeman (2006) en su investigación “Real Time Detection and Segmentation of Reflectionally Symmetric Objects in Digital Images”, desarrollaron un nuevo algoritmo para encontrar líneas de simetría en objetos con el fin de solucionar el problema de segmentar objetos sin la necesidad de utilizar los modelos de objetos; para lograr solucionar ese problema, utilizaron la transformada de Hough como modelo base para la detección del objeto y rotación de la imagen, usaron la votación para discernir si el borde de la imagen es simétrico o no, la programación dinámica para hallar contornos continuos y simétricos en la imagen; para validar su algoritmo usaron 4 fotos, las cuales contenían uno o más objetos en diferentes posiciones, en donde se pudo comprobar que, el algoritmo propuesto pudo segmentar correctamente los objetos a base al color, forma, y la línea de simetría; cuando hay sombras y reflejos en los objetos, el algoritmo tiende a dejar espacios vacíos, esto permitió llegar a conclusión que, el algoritmo propuesto muestra resultados de óptimos de tiempo para la detección de objetos simétricos, la programación dinámica consiguió segmentar objetos en presencia de píxeles de borde ruidosos causados por sombras, desorden de fondo y textura de objeto.



Hauagge & Snavely (2012), en su trabajo “Image Matching using Local Symmetry Features”, plantean un detector de características y un descriptor diseñado principalmente para escenas arquitectónicas basado en la puntuación de simetrías locales en todas las ubicaciones y escalas en una imagen; para lograr ello propusieron una medida simple de simetría local basada en el análisis de las diferencias de imagen a través de ejes de simetría; hicieron uso de la puntuación para probar lo propuesto, usaron 48 pares de fotos teniendo como resultado que, su propuesta de detección de simetría ofrece mejores resultados bajo los tipos de variación presentado en el conjunto de datos y que el descriptor DoG supera al detector que plantean cuando las imágenes tienen poca simetría como por ejemplo las imágenes de graffiti; llegando a la conclusión que, el método propuesto proporciona características más repetibles y proporcionan información complementaria en imágenes que no son SIFT; sin embargo, el conjunto de datos usado, sigue siendo extremadamente difícil para técnicas de coincidencia de imágenes, y por lo que creemos que las características de imagen siguen siendo un área abierta.

Santo (2012), en su investigación “Principal Component Analysis applied to digital image compression”, propone el uso de PCA como herramienta para reconocer patrones y comprimir la imagen, para lograr lo propuesto vectorizó una imagen llevándola a una matriz MXN para luego calcular la covarianza a los datos, luego procedió a sacar las componentes principales y seguir los pasos que demanda la técnica PCA; para comprobar lo propuesto utilizó una imagen médica de 512×512 píxeles, donde se pudo apreciar que, cuando la tasa de compresión es más alta el número de componentes es bajo degradando la imagen y cuando la imagen médica tiene una textura densa ocasiona diferentes resultados, lo cual llevó a la conclusión que la cantidad de componentes principales que se utilicen en la compresión influye en la recuperación de la imagen original de la imagen compactada; permitiendo así un ahorro significativo de espacio de almacenamiento.



Akbar, Hayat, Haq, & Bajwa, (2014), en su trabajo de investigación sobre “Bilateral Symmetry Detection on the Basis of Scale Invariant Feature Transform”, plantean un método para hallar la simetría bilateral mediante el uso de SIFT, la comparación del gradiente y la orientación de los puntos clave con el fin de detectar la simetría bilateral en imágenes médicas reales de un solo objeto; para lograr alcanzar lo propuesto, utilizaron un detector de características SIFT, uso de puntos clave, coordenadas polares y pares simétricos, teniendo como resultado que, el método propuesto muestra que, cuando se llega a un Ángulo de 45° en el plano polar con respecto al eje polar, se puede calcular los pares simétricos opuestos y se puede hallar los puntos medios para calcular la línea de simetría; se recogió una muestra de 1209 imágenes reales donde 788 imágenes sin desorden de fondo, eran simétricas y 310 eran asimétricas, dando al método un 94% de aceptación, lo cual permitió llegar a la conclusión que, el método propuesto es más que óptimo cuando en la imagen de entrada solo existe un solo objeto y en el caso de los objetos asimétricos, el método los clasifica cada vez mejor y muestra error en algunos objetos que tienen fondo distorsionado.

Vieyra (2013), en su investigación “Análisis de componentes principales en imágenes de teledetección”, muestra como la técnica Análisis de componentes principales (PCA) ayuda al algoritmo k mean para la clasificación de imágenes; para lograr su cometido hizo uso de la correlación de una matriz de covarianza, el cálculo de los eigen vectores y los eigen valores de modo que desglosó el algoritmo PCA logrando así modificar una parte matemática de dicho algoritmo, lo que ayudó a reducir el uso computacional que genera el algoritmo de k means en la clasificación de imágenes; lo cual permitió llegar a la conclusión que, la técnica PCA tiene un amplio uso interdisciplinario que facilita la interpretación de los datos y el análisis cualitativo de la información obtenida.



La manera en la que elimina la redundancia conservando la estructura original de los datos permite usar esta herramienta como una alternativa para disminuir el costo computacional de la aplicación de otras técnicas multivariadas.

Agustí Melchor (2016), en su tesis doctoral “Análisis y clasificación de imágenes repetitivas mediante técnicas de simetría computacional propone una Arquitectura de Sistema de recuperación de imágenes por contenido basado en el uso de la simetría computacional”, para lograr esto, tuvo que identificar los tipos y patrones de simetría que puede haber en una imagen, evaluar los diferentes clasificadores para grupos de simetría, teniendo como resultado que el modelo propuesto para la clasificación de imágenes frente a los métodos que hay en la literatura no genera ambigüedades.

Soto Hidalgo, Chamorro, Palomares, Carlos Gámez, & Olivares, 2013, en su investigación “Aplicación de PCA y técnicas bayesianas a la clasificación de píxeles basada en color (2013)”, proponen un método para la clasificación de píxeles en base a su color, este método consta del uso de PCA para obtener variables características y el uso de estas variables como patrón para un clasificador bayesiano, teniendo como resultado un 95% de aceptación en la clasificación de los píxeles por color llegando a la conclusión que para garantizar incorrección e independencia de las componentes cromáticas del espacio de color RGB se debe aplicar PCA; el método propuesto permitió detectar zonas en imágenes según el color.

2.3. Base teórica científicas

2.3.1. Simetría

Para Ana María Sandoval Poveda (Guía de trabajo-2013-u. Estatal a distancia), afirma que, la simetría es la correspondencia exacta de dos partes o puntos de un cuerpo o figura con relación a un centro o a un eje.

Según Real Academia Española (2017) define la simetría como:



“la Correspondencia exacta en forma, tamaño y posición de las partes de Un todo. En la geometría: correspondencia exacta en la disposición regular de Las partes o puntos de un cuerpo o figura con relación a su centro”.

Liu, (2008) En su libro “Computational Symmetry in Computer Vision and Computer Graphics”, afirma que, la simetría es un fenómeno que está presente en el medio natural y artificial, de manera que un objeto esta hecho de múltiples copias del mismo que son intercambiables de alguna manera.

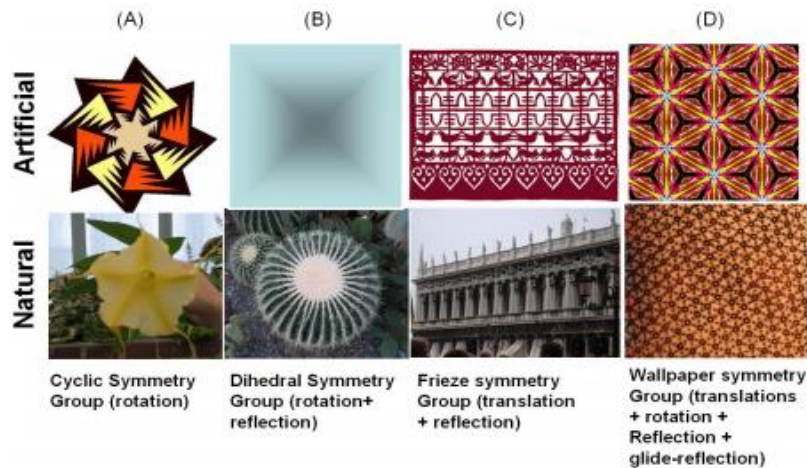


Figura 3: Imágenes de muestra de simetría

Fuente: (Liu, 2008) Computational Symmetry in Computer Vision and Computer Graphics

2.3.2. Geometría Computacional

En el documento escrito por (Mendoza, 2006) sobre la Geometría Computacional, define que:

“La geometría computacional como aquella que se ocupa del diseño y análisis de algoritmos de computación, para resolver problemas de tipo geométrico” (pag.5).



2.3.3. Imagen Digital

Según el apunte impartido escrito por Escalante Ramírez, (2006), Una imagen digital se define como $f(x, y)$, donde esta ha sido discretizada en coordenadas espaciales y en luminosidad. Una imagen digital se puede considerar como una matriz en donde cada índice de fila y columna identifican un punto (en el espacio bidimensional) en la imagen.

Gonzalez & Woods (2008), en su libro digital image processing, mencionan que:

“Una imagen digital puede definirse como una función bidimensional. $f(x, y)$ donde x e y son coordenadas en el plano y la amplitud de f en cualquier par de coordenadas (x, y) se denomina intensidad o nivel de gris de la imagen en ese punto”. (pag,20)

2.3.4. Tipos de Imágenes

2.3.4.1. Imagen Vectorial

Para Alcalá & navarro (2008), afirma que son imágenes construidas a partir de unos objetos que se generan matemáticamente a través de vectores, en donde los puntos que definen al vector están unidos por una curva llamada Bézier, dicha curva se emplea en la creación de ilustraciones y este tipo de imagen tiene la ventaja de ocupar poco espacio en disco y permitir extender el objeto sin distorsionarlo en la imagen.

2.3.4.2. Imagen de mapa de Bits

Según Alcalá & navarro (2008), afirma que, este tipo de imagen a diferencia de la vectorial no puede escalarse debido que es compuesta por una rejilla rectangular de pixeles, donde esta rejilla representa la imagen en cualquier dispositivo que sea capaz de leer este tipo de imagen y al intentar escalar este tipo de imagen hay que tener en cuenta la resolución y la profundidad de bits.



2.3.5. Procesamiento digital de imágenes

Para Dr. Escalante Ramírez (2006), en sus apuntes impartidos afirma que, el procesamiento digital de imágenes es un campo de investigación interdisciplinario debido a que se relaciona con áreas como: las matemáticas, la computación, lo cual ayuda a generar nuevos conocimientos.

2.3.5.1. Adquisición de Imágenes

Tiene como finalidad convertir un objeto del mundo real a una imagen para ser posteriormente procesada; para hacer esta conversión se emplean sensores de imágenes como escáner, rayos X, una cámara fotográfica, donde las señales que se emiten son digitalizadas (La Serna Palomino & Román Concha, 2009).

2.3.5.2. Pre Procesamiento

Consiste en eliminar cualquier ruido o fallas que se hayan generado durante la adquisición de la imagen, para así mejorar sus características como pueden ser contornos, bordes, brillo, etc. (Mart, 2004)

2.3.5.3. Segmentación:

Tiene por fin reconocer, localizar y separar los objetos de interés del resto de la imagen utilizando técnicas como la umbralización, detección de discontinuidad, entre otras; esta parte finaliza cuando se haya detectado el objeto de interés, cabe recalcar que depende de esta parte el éxito de un proyecto de procesamiento de imágenes. (La Serna Palomino & Román Concha, 2009)

2.3.5.4. Extracción de Características

Consiste en extraer las características del objeto de estudio tales como forma, área u algún dato que sea útil y permita encontrar



información.

2.3.5.5. Reconocimiento e Interpretación

Según el autor La Serna Palomino & Román Concha, (2009) en su investigación afirma que:

“La etapa de reconocimiento es el proceso que asigna una etiqueta a un objeto basándose en la información proporcionada por sus descriptores. La interpretación implica asignar significado a un conjunto de objetos reconocidos” (pag,11).

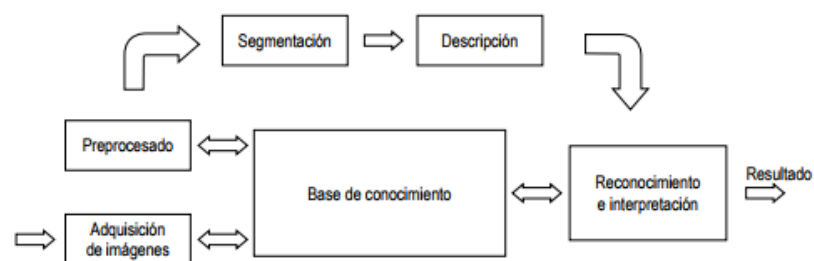


Figura 4: Etapas del procesamiento digital de imágenes
Fuente: (La Serna Palomino & Román Concha, 2009) Técnicas de segmentación en Procesamiento Digital de Imágenes.

2.3.6. Morfología de imágenes

2.3.6.1. Dilatación:

Aguilar (1995) menciona que, la dilatación es una operación matemática que fue propuesta por Minkowski y se basa en la unión de conjuntos abiertos (dispersos); la cual fue aplicada como una operación de suavización en imágenes binarias por investigadores contemporáneos, la implementación de esta técnica se hace en base a traslaciones de imagen aplicando la unión de traslaciones del conjunto A por los elementos del conjunto B y se denota con la siguiente formula :



$$A \oplus B = \bigcup_{b \in A} (A)_b \quad (1)$$

2.3.6.2. Erosión

El autor González González (2010), afirma que, la erosión es el resultado de saber si el elemento estructurante Y pertenece conjunto X ; en caso de que no tengan elementos en común el resultado es el conjunto vacío y la ecuación que define a la erosión es la siguiente.

$$\varepsilon_y(X) = \{X | Y_x \subseteq X\} \quad (2)$$

2.3.7. Modelos de Color

2.3.7.1. Modelo RGB

En el libro escrito por Sucar & Gomez (2011) acerca de la visión Computacional manifiesta que, el modelo RGB se basa en los tres sensores humanos, es decir, la combinación de tres colores básicos o primarios: R (rojo), G (verde), B (azul). Generalmente los componentes se normalizan, obteniendo:

$$r = \frac{R}{R+G+B} \quad g = \frac{G}{R+G+B} \quad b = \frac{B}{R+G+B} \quad (3)$$

En donde los colores se pueden visualizar dentro de un triángulo en donde los vértices de dicho triángulo están compuestos por los componentes primarios R, G, B y en el centro del triángulo se encuentra el color blanco, como ejemplo de uso de este modelo, los autores muestran a la televisión y a la pantalla de computador



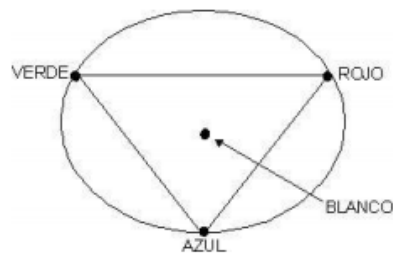


Figura 5: Diagrama Cromático RGB

Fuente: (La Serna Palomino & Román Concha, 2009) Técnicas de Segmentación en Procesamiento Digital de Imágenes.

2.3.7.2. Modelo HSI

Este modelo se basa en la tonalidad, intensidad y saturación para poder diferenciar un color del otro; en donde la tonalidad se asocia con la longitud predominante en la mezcla de ondas de luminosidad permitiendo así reconocer el tono del color, la intensidad para la iluminación que tiene el objeto y la saturación que nos permite diferenciar un color intenso de un pálido (Báez Rojas & Alonso Pérez, 2008).

2.3.8. Correlación

Para Javier Gorgas García, Cardiel López, & Zamorano Calvo, (2011), en su libro “Estadística Básica para Estudiantes de Ciencias”, (2011), testifica que, la correlación estudia el grado de asociación o dependencia entre las dos variables, en otras palabras, la correlación es la medida significativa de la dependencia de una variable con la otra.



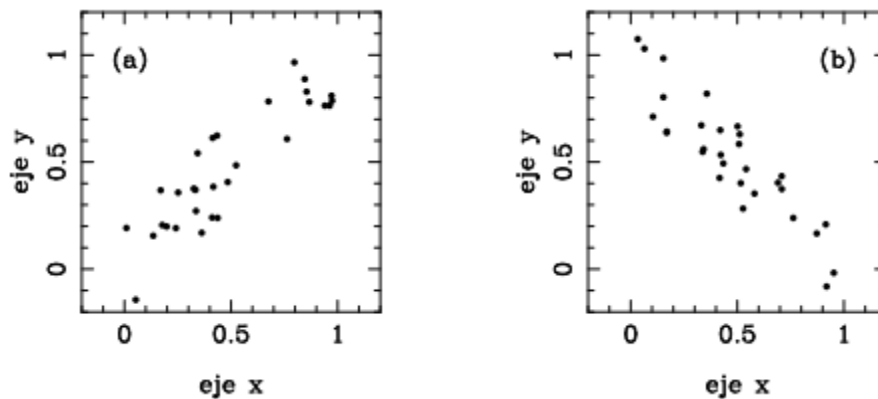


Figura 6: Ejemplos de Correlación

Fuente: (Gorgas García, Cardiel López, & Zamorano Calvo, 2011) Estadística Básica para Estudiantes de Ciencias

2.3.9. Vectores propios y valores propio

Los valores propios son aquellos valores de una matriz que al ser transpuestos o cambiados de coordenadas no varían como puede ser la determinante de una matriz y los vectores propios representan las direcciones características de una matriz; estas direcciones no se modifican al multiplicar la matriz por un escalar, para su cálculo se puede aplicar la siguiente fórmula: $(A - \lambda I)u = 0$ donde **A** es la matriz, λ es el escalar y **u** es el vector propio de la matriz. (Peña, 2002)

2.3.10. Análisis de Componentes Principales (PCA)

El autor Jolliffe (2002), en su libro “Principal Component Analysis, Second Edition” afirma que, la idea principal del análisis de componentes principales (PCA) es reducir la dimensionalidad que presenta un conjunto de datos de un gran número de Variables, manteniendo la máxima variación posible en el conjunto de datos. Esto se logra transformando los datos a un nuevo conjunto de variables llamado Componentes principales (PC), en donde los datos no están correlacionados y se ordenan los datos según la variación que tengan de mayor a menor.



En el libro análisis de datos multivariantes Peña (2002) afirma que, El análisis de componentes principales tiene como objetivo: dadas n observaciones de p variables, se analiza si es posible representar esta información con un número menor de variables, donde estas nuevas variables son combinaciones lineales de las originales.

La técnica tiene utilidad doble:

1. Permite la representación óptima en un espacio de dimensión pequeña de p -dimensional; en este aspecto la definición de componentes principales es el primer paso para poder identificar las posibles variables “latentes” o no observadas, las cuales generan variabilidad entre los datos.
2. Permite las transformaciones de variables originales correladas, en nuevas variables incorreladas, para así facilitar la interpretación de los datos.

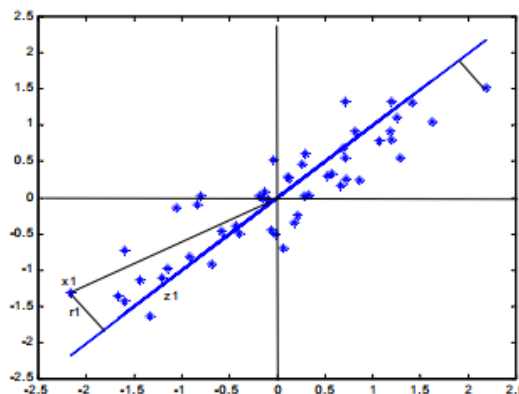


Figura 7: Ejemplo de una Componente Principal

Fuente:(Peña, 2002) ANÁLISIS DE DATOS MULTIVARIANTES



2.3.11. Momentos estadísticos:

El autor Julie y Billy (2007) menciona que:

“Los momentos de una variable aleatoria son valores esperados de algunas funciones de la variable aleatoria y constituyen a una colección de medidas descriptivas con las que se puede caracterizar de manera única a su distribución de probabilidad”

Por ello existen dos momentos de una variable aleatoria; el primero es conocido como momentos con respecto al origen y el otro como momentos respecto a la media. (Ruíz Muñoz, 2004)

2.3.12. Métodos de Enfriamiento

2.3.12.1. Refrigeración

Umaña Cerros (2010) afirma que, la conservación de los productos a bajas temperaturas evita el crecimiento de los microorganismos que dañan al alimento.

De manera que, cuando hay una refrigeración entre -1°C y 8°C , se consigue que el valor nutricional y las características propias de los productos casi no se diferencien del producto fresco.

2.3.12.2. Congelación

La congelación a diferencia de la refrigeración, reduce la multiplicación de microorganismos basándose en temperaturas más bajas que las ideales; al aplicar una congelación por debajo de cero grados centígrados a los alimentos, el agua que posee el alimento se solidifica produciendo una significativa conservación. Hay que destacar que, en la refrigeración, se producen menos modificaciones en los alimentos de forma que después de la descongelación, los alimentos mantienen sus propiedades como materia prima. (Umaña Cerros, 2010)



2.3.12.3. Pasteurización

La pasteurización tiene por objeto destruir los agentes patógenos que causan la descomposición del alimento; este tratamiento térmico debe ser manejado a temperaturas menores a 100°C para así poder prolongar la vida del alimento; este método se emplea en leche, huevos líquidos, hortalizas, etc. (Romero, 2014)

2.3.13. Imágenes Médicas Digital:

Este tipo de imagen actualmente se considera como un instrumento fundamental en el campo de la medicina ya que permiten detectar o extraer información confiable y subjetiva sobre el estado de salud del paciente, permitiendo a los médicos saber las causas o los síntomas que genera cierta enfermedad; existen diversos dispositivos electrónicos y técnicas que ayudan a la captura de este tipo de información a través de imágenes. (Roa Martínez, Borsetti Gregorio Vido, & Vicentini Jorente, 2016)

2.3.14. Tipos de imágenes médicas

2.3.13.1. Radiología

El autor Passariello & Mora (1995) afirma que, la radiología es un procedimiento que se ha difundido en la generación de imágenes debido a que sus ondas atraviesan los tejidos del cuerpo humano y permite generar imágenes médicas diagnosticables haciendo uso de proyecciones frontales del objeto.

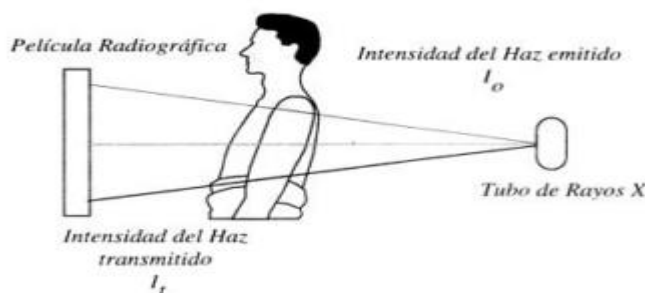


Figura 8: Esquema de toma radiográfica



Fuente: Passariello & Mora (1995) Imágenes medicas

2.3.13.2. **Angiografía**

Es un procedimiento radiológico similar a la radiografía, la cual se usa para observar los vasos sanguíneos del cuerpo y estudiar la irrigación del cerebro; se aplica cuando se quiere obtener una imagen de un órgano en movimiento que realiza un flujo torrente sanguíneo.

En este tipo de imagen se necesita un equipo de rayos X con una cámara de cine de 35mm para la captura de la imagen y rollos de película.(Passariello & Mora, 1995)

2.3.13.3. **Tomografía computarizada:**

El autor (Rastrepo V., 1998), afirma que :

“Este tipo de imágenes permite obtener cortes bidimensionales de cualquier parte del organismo humano sobre la pantalla de un tomógrafo y en una película e blanco y negro.”

2.3.15. **Formato Dicom (Digital Imaging and Communications in Medicine)**

Es un mecanismo estándar de codificación y almacenamiento de imágenes, el cual fue propuesto y administrado por la National Electrical Manufacturers Association(Nema), el cual garantiza que la imagen no se dañe hasta el momento en que es impresa en papel radiográfico. (Trujillo, Rivera, & Serna, 2010)

2.3.14.1. **Formato de un archivo Dicom**

Este tipo de archivo está compuesto por dos partes: una cabecera y un dataset; la cabecera es un preámbulo el cual consta de 128 bytes y almacena información como el nombre del paciente, el tipo de exploración, dimensión de la imagen,



entre otras cosas de importancia, el dataset contiene la imagen especificada; el mismo que se construye con data element que son etiquetas que contiene un tag, valor representativo, un valor de longitud y un valor de campo. (Trujillo et al., 2010)

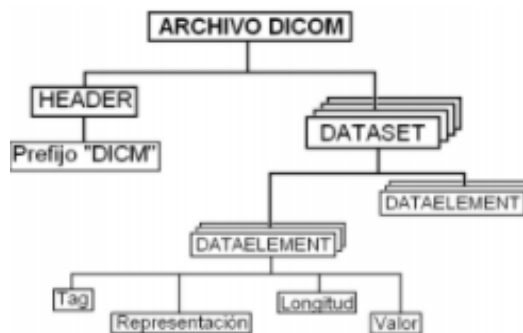


Figura 9 Estructura de un Archivo DICOM

Fuente: (Trujillo, Rivera, & Serna, 2010)

2.4. Definición de términos básicos

a) **Imagen:**

Es una representación visual del mundo real, compuesta por diferentes intensidades luminosa, que manifiesta la apariencia visual de un objeto real o imaginario.

b) **Resolución**

Permite tener una mayor o menor nitidez o calidad visual de la imagen, así mismo cuando más sea la resolución, obtendremos más detalles de la imagen.

c) **Pixel:**

El píxel es la unidad más pequeña de la imagen, representa el tono y la profundidad del color de la imagen en un punto, el píxel toma valores entre 0 y 255.



d) **Algoritmo**

Método que describe la solución de un problema computacional, siguiendo una serie de pasos precisos, ordenados y finitos.

e) **Método**

Procedimiento utilizado para llegar a un fin.

f) **Procesamiento de imágenes**

Tiene como objetivo mejorar el aspecto de la imagen y hacer más evidentes los detalles que esta presenta, sin importar de donde fue conseguida la imagen.

g) **Umbral**

Es aquel número que va a permitir decidir los pixeles que forman parte de un objeto o no, en una imagen digital; depende de este número el nivel de fiabilidad de la segmentación.

El umbral puede ser función de la posición $p(r, c)$, de la vecindad $N(r, c)$ y de la intensidad $I(r, c)$ actual del pixel.

$$T = T((r, c), N(r, c), I(r, c)) \quad (4)$$

h) **Histograma**

Representa el nivel de intensidad que tiene cada pixel de la imagen, donde cada el nivel de intensidad se encuentra entre los valores de 0 a 255 en imágenes de escala de grises.

i) **Técnicas de segmentación**

a. **Binarización**

Este método realiza un barrido de la matriz de la imagen para calcular el umbral de cada pixel y en función de ello separa el objeto del fondo de la imagen a través de 0 y 1; donde 0 representa al blanco y 1 al negro, permitiendo así la reducción de información.

b. **Umbralización**



Es una técnica de segmentación que se basa en un umbral para poder separar los pixeles de una imagen y así separa el objeto del fondo de la imagen en dos categorías

c. Basados en Bordes:

Permiten determinar los límites de cada segmento en la imagen y así poder identificarlo bordes de un objeto.

j) Image J

Es un programa de procesamiento de imágenes de código abierto desarrollado por la National Institutes of Health diseñado para imágenes multidimensionales científicas; está escrito en lenguaje Java lo que permite que se despliegue en cualquier plataforma, cuenta con miles de complementos y scripts para realizar una amplia variedad de tareas y una gran comunidad de usuarios.

k) Estimacion:

Proceso de encontrar una aproximación sobre una medida, lo que se ha de valorar con algún propósito es utilizable incluso si los datos de entrada pueden estar incompletos, incierto, o inestables.



CAPITULO III: MARCO METODOLÓGICO

3.1. Tipo y Diseño de Investigación

3.3.1. Tipo de Investigación

El presente trabajo de investigación corresponde a una investigación de tipo de cuantitativa, debido a que, se trabajará con valores cuantificables como porcentajes, tasas, etc., los cuales permiten comprobar la estimación correcta de la simetría; además admite la realización de preguntas específicas como ¿Qué tan eficientes?, ¿En qué medida?, etc.

3.3.2. Diseño de Investigación

En el presente trabajo de investigación se estimará la simetría de la palta Hass en imágenes tomográficas mediante el método de los momentos estadístico; por ello es necesario aplicar un tipo de diseño investigación experimental del tipo cuasi experimental con un solo grupo y será post test, ya que este tipo de diseño investigación permite la manipulación de las variables independientes y observar las variaciones que genera esta sobre las variables dependientes.

3.2. Población y Muestra

La presente investigación utilizará 12 imágenes tomográficas de la palta Hass como población y 216 imágenes digitales de 12 paltas del tipo Hass de exportación en un ambiente controlado donde cada palta generará 18 imágenes RGB a 10°.

En este trabajo utilizara una muestra a conveniencia.

3.3. Hipótesis

Con el método de los momentos estadísticos se estima adecuadamente la simetría de la palta Hass en imágenes tomográficas.



3.4. Variables

Para este trabajo de investigación se utilizará como variable independiente el método de los momentos estadísticos y como variable dependiente estimación de la simetría de la palta Hass en imágenes tomográficas.

3.5. Operacionalización

Variable Independiente	Dimensiones	Indicadores	Fórmula	Ítems o Respuestas	Técnicas e instrumentos de recolección de datos
Momentos Estadísticos	Eficiencia	Tiempo de respuesta	$T=ti-tf$	tiempo	Observación
	Variabilidad de datos	Nivel de variabilidad	$\sigma^2 = \frac{\sum_{i=1}^n (x_i - \mu)^2}{N}$	Porcentaje de(%) variabilidad	Análisis documental
Variable dependiente	Dimensiones	Indicadores	Fórmula	Ítems o Respuestas	Técnicas e instrumentos de recolección de datos
Simetría de la palta Hass	Fidelidad de la simetría	Nivel de certeza	$X = 0$ $X > 0$ $x < 0$	Porcentaje (%)	Análisis documental

3.6. Abordaje metodológico, técnicas e instrumentos de recolección de datos

3.6.1. Abordaje metodológico

En la presente investigación, se empleará la investigación experimental y cuasi experimental, debido a que vamos a controlar la variable independiente para hallar mejores resultados.



3.6.2. Técnicas de recolección de datos

a) Análisis documental

Dulzaides Iglesias & Molina Gómez (2004) afirma que:

“El análisis documental es una forma de investigación técnica, un conjunto de operaciones intelectuales, que buscan describir y representar los documentos de forma unificada sistemática para facilitar su recuperación. Comprende el procesamiento analítico- sintético que, a su vez, incluye la descripción bibliográfica y general de la fuente, la clasificación, indización, anotación, extracción, traducción y la confección de reseñas” (pag,2)

b) Observación:

El autor Campos y Cobarrubias & Lule Martinez, (2013), establece que:

“La observación es la forma más sistematizada y lógica para el registro visual y verificable de lo que se pretende conocer; es decir, es captar de la manera más objetiva posible, lo que ocurre en el mundo real, ya sea para describirlo, analizarlo o explicarlo desde una perspectiva científica” (pag,5).

3.6.3. Instrumentos de recolección de datos

El instrumento que se utilizara para la presente investigación es la ficha de observación, en la cual podemos describir la relación que se da entre las variables ante un hecho y para el análisis documental se utilizará los datos arrojados por la aplicación ImageJ para después de aplicar el método de los momentos estadísticos, dichos datos arrojados servirán para hacer cálculos estadísticos y generar gráficos que nos ayuden a



interpretar los datos y finalmente deducir conclusiones sobre qué tan simétricas son las paltas en conjunto.

3.7. Procedimientos para la recolección de datos

- a. Conseguir las paltas
- b. Pesar cada palta e ir registrando el peso en una tabla de Excel
- c. Pesar la pulpa, la cáscara y la pepa en una tabla de Excel
- d. Tomar las fotos a cada palta a un Ángulo de 10°
- e. Sacar tomografías a las paltas
- f. Aplicar algoritmos de pre procesamiento
- g. Aplicar la técnica PCA
- h. Calcular los ángulos directores
- i. Rotación y Detecciones de bordes de la imagen
- j. Aplicar el método de simetría
- k. Pasar los datos arrojados a un Excel para su análisis
- l. Evaluar los resultados en conjunto de cada palta para discernir si son simétricas

3.8. Análisis estadísticos e interpretación de los datos

a) Tiempo de respuesta

Para el pre procesamiento la imagen de la palta, se analizará el tiempo que demanda calcular las componentes principales, ángulos directores y los momentos estadísticos, estos tiempos lo calcularemos con la siguiente fórmula.

$$tiempo = tiempo\ inicial - tiempo\ final \quad (5)$$

b) Cantidad veces ejecutadas para alinear la imagen

Se estimará el promedio de número de veces que se emplea para alinear la imagen de la palta has así mismo el tiempo que se necesita para la alineación.



$$\mu = \frac{\sum_{i=0}^t x}{n} \quad (6)$$

c) Fidelidad de la simetría:

Consiste en calcular y verificar que tan cercano al 0.00 son los resultados de simetría de cada palta para poder calcular el promedio general y ver si son simétricas en conjunto.

Se empleará herramientas como el Excel para hacer los cálculos y gráficos si fuese necesario para poder entender lo que pasa con los datos.

3.9. Principios éticos

Los criterios éticos que se han tomado en la presente investigación, están relacionados con cualquier suceso en el cual se pueda ver involucrado esta investigación. Por ello se han tomado los siguientes criterios éticos:

a. Criterio de confiabilidad

La información y los datos personales que se requieren para la presente investigación, deben ser obtenidos de manera legal y con profesionalismo para evitar causar daños a las personas involucradas; tal como lo expresa:

La ley N° 29733: “Ley de protección de datos personales”, en su Título IV: “Obligaciones del titular y del encargado del banco de datos personales” En su Artículo 28 señala: Que recopilar datos personales por medios ilícitos, fraudulentos o desleales está penado por el estado peruano.

b. Criterio de Confirmabilidad

Los resultados y afirmaciones que se puedan suscitar como producto de la investigación serán confirmados y validados por un profesional especialista en el tema tal como lo enuncia: El Código Deontológico del Colegio de Ingenieros del Perú en su Capítulo III “Faltas Contra la Ética Profesional y Sanciones”, en el Sub Capítulo II “De la Relación con El Público” en su Artículo 105 señala:



Los ingenieros serán objetivos y veraces en sus informes y declaraciones, y expresarán opiniones en temas de ingeniería solamente cuando ellas se basen en un adecuado análisis y conocimiento de los hechos, detalles técnicos y científicos suficientes y veraces.

3.10. Criterios de rigor científico

a) **Consistencia**

Los datos que han sido recolectados para este trabajo de investigación son de carácter científico y formal.

El análisis realizado a los datos está hecho con total profesionalidad aplicando habilidades, técnicas y conocimientos de la ingeniería y de la investigación para mantener la consistencia de los datos y resulte en información consistente y útil.

b) **Validez**

Los datos obtenidos de la técnica de PCA y el método de los momentos estadísticos serán correctamente evaluados y analizados para lograr dar un resultado válido que verifique la hipótesis.

c) **Fiabilidad**

Las técnicas utilizadas para la evaluación de la simetría de la palta Hass permitirá analizar qué tan fiable son los resultados obtenidos, estos resultados deben ser validado por los expertos en el campo.

d) **Objetividad**

Los datos que se obtendrán después de aplicar el método de los momentos estadísticos para estimar la simetría en imágenes digitales de la palta Hass están exentos de la influencia de la perspectiva de los investigadores.



CAPITULO IV: Análisis e interpretación de resultados

4.1. Resultados en Tablas y Gráficos

4.1.1. Pre procesamiento de Imágenes:

En esta parte se muestra el tiempo promedio en segundos que demando el uso de cada algoritmo hasta encontrar la simetría del objeto en la imagen, dichos tiempos se muestra en la siguiente tabla:

Tabla 1 Tabla de tiempos promedio empleado por las técnicas utilizadas

Acción	Pca(s)	Ángulos directores(s)	Detección de borde(s)
Promedio	0.03008333	0.001333333	1.30858333

Fuente: Elaboración propia



Gráfico 1: Tiempo empleado de cada algoritmo por cada palta Hass

Fuente: Elaboración propia

Los datos mostrados en la tabla 1 corresponden al tiempo promedio que necesita cada algoritmo para ejecutarse.

Para ver más detalle del tiempo empleado por cada algoritmo ver anexo III

4.1.2. Alineación del stack de imágenes

En el caso de la rotación de imágenes, se calculó el promedio de todas las iteraciones de cada palta, tiendo como valor 10 iteraciones como promedio;



por ello en la tabla nro. 2 se muestra un resumen del número de iteraciones empleada por cada palta; para ver más detalle revisar el **anexo IV** donde se muestra más a detalle cada iteración con el ángulo empleado.

Tabla 2 Cantidad de iteraciones por cada palta

Palta	Cant. Rotación
2	8
3	12
4	14
5	16
8	14
12	9
13	10
14	7
15	8
17	6
18	12
19	5
promedio	10

Fuente Elaboración propia

En la tabla 2 se puede apreciar el número de veces que fue necesario rotar cada stack de imágenes hasta tener alineados los ejes X, Y, Z, lo que permite tener alienado cada imagen del stack de imágenes apuntando todas ellas a una misma dirección.

Se puede observar que, el stack de imágenes de la palta 19 necesitó menos número de veces de rotación; demostrando que sus ejes de este stack están mejor alienados que los demás.

4.1.3. Fidelidad de la simetría

1. Momentos estadísticos

Para saber la fidelidad de la simetría se aplicó el método de los momentos estadísticos a 11 imágenes sintéticas simétricas y asimétricas; teniendo como resultado la siguiente tabla:



Tabla 3 Resultado de evaluación de imágenes simétricas y no simétricas

	X	Y
Simétrica	10	9
No Simétrica	11	11

Fuente Elaboración propia

Como se puede observar el método de los momentos estadísticos identificó correctamente las imágenes asimétricas y en el caso de las imágenes simétricas solo en el eje Y la simetría no fue detectada correctamente en dos imágenes.

Después de haber verificado que el método del momento estadístico calcula bien la simetría, se aplicó este método a las imágenes del contorno de la palta; cabe recalcar que solo se utilizó los 3 primeros momentos estadísticos.

Tabla 4 Tabla de resultado de simetría de cada palta por eje

Nro. Palta	X	Y
Palta 2	0.01754839	0.01954839
Palta 3	0.01748485	0.02279606
Palta 4	-0.00876471	0.00037095
Palta 5	0.009328	0.00674806
Palta 8	0.00095366	-0.01286207
Palta 12	0.00708523	-0.00516217
Palta 13	0.02059646	-0.02047059
Palta 14	0.02273959	-0.00846941
Palta 15	0.00126509	-0.00125845
Palta 17	0.00723928	-0.0126
Palta 18	-0.00882761	0.00522053
Palta 19	0.02042424	0.00058092
Promedio	0.00892271	-0.00046315

Fuente Elaboración propia

Los datos arrojados en la tabla 3 se compararon con la siguiente condición:



$$x = 0 \quad x > 0 \quad x < 0 \quad (7)$$

Al utilizar la formula $x=0$ de la condición 7, se obtuvo la siguiente tabla con el número de paltas total que presentan simetría en el eje X, así como en el eje Y.

Tabla 5 Cantidad de paltas simétrica y no simétricas por cada eje X e Y

	X	Y	%	% Error
Simétrica	12	12	100%	0.08%
No Simétrica	0	0	0%	

Fuente Elaboración propia

Como se puede apreciar en la tabla nro. 5, el algoritmo detecto que, de las 12 paltas evaluadas, todas son simétricas, por lo que se concluye que la palta es simétrica en ambos ejes con un valor promedio de 0.0080533 en el eje X y -0.00026485 para el eje Y; en el gráfico nro. 2 se visualiza los resultados de la tabla nro. 5.



Gráfico 2: Cantidad de paltas simétricas y no simétricas por cada eje

Fuente Elaboración propia

2. Puntos equidistantes

Este método fue aplicado a 12 imágenes de la palta del tipo Hass tal y como se hizo en el punto anterior; obteniendo el siguiente resultado:



Tabla 6: Resultados de simetría en eje X e Y así como sus diagonales

Nro Palta	Prom.Sim XY	Prom.Sim DD
Palta 2	0.97	0.949
Palta 3	0.979	0.958
Palta 4	0.978	0.957
Palta 5	0.955	0.974
Palta 8	0.943	0.987
Palta 12	0.976	0.956
Palta 13	0.968	0.95
Palta 14	0.961	0.97
Palta 15	0.978	0.985
Palta 17	0.949	0.967
Palta 18	0.976	0.956
Palta 19	0.935	0.945
Promedio	0.964	0.962833333

Fuente: Elaboración propia

Como se puede apreciar en la tabla 6 donde se evalúan las 12 paltas del tipo Hass, todas presentan un valor muy cercano a 1 tanto en el eje X e Y así como en sus diagonales; por lo que se puede afirmar que, las paltas son simétricas en conjunto con un valor promedio de 0.964, un valor muy cercano a 1.

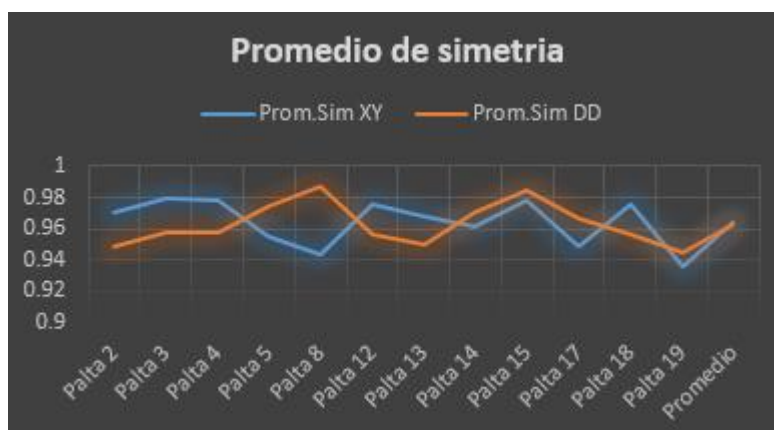


Grafico 3: Promedio de simetría ejes X e Y



Fuente: Elaboración propia

4.1.4. Variabilidad de los resultados de simetría

Para obtener la variabilidad de los resultados de simetría, se aplicó la fórmula de la varianza tanto al eje X, eje Y y a ambos ejes para ver que tanto varían los datos.

Tabla 7 Variabilidad de Resultados de Simetría

	VARIABILIDAD	%
X	0.000256908	0.03%
Y	0.000385043	0.04%
XY	0.000338273	0.03%

Fuente Elaboración propia

Como se puede observar, el nivel de variabilidad de los datos es mínimo representando un 0.03 % en el eje X, 0.04% en el eje Y y un 0.03% de variabilidad en ambos ejes; el grafico de estos resultados se puede observar en el gráfico nro. .3

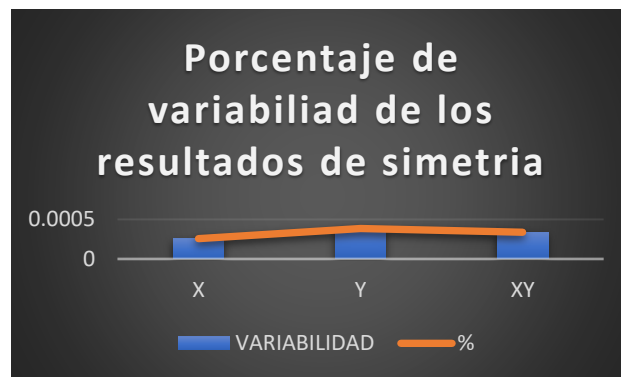


Gráfico 4: Variabilidad de los datos de simetría por eje y en conjunto

4.2. Discusión de resultados

- a) Para el pre procesamiento del stack de imágenes, se ejecutaron los algoritmos pertinentes para realizar cada tarea del pre procesamiento; se tuvo respuestas en un tiempo muy corto al momento de ejecutar los algoritmos en el stack de imágenes de la palta.



- b)** En la parte de la alineación de la imagen, se puede observar que fue necesario rotar la imagen varias veces para poder alinear los ejes x, y, z; esto conlleva a deducir que las imágenes de la palta cuando se hizo las tomografías no estuvieron bien alineadas.
- c)** Para saber si los resultados del método de simetría son los correctos, se utilizó 11 imágenes simétricas y no simétricas, donde el método de simetría acertó correctamente con los valores de simetría para cada asimétrica y en el caso de las simétricas solo hubo una pequeña variabilidad.
- d)** En el caso de la fidelidad de la simetría se pudo observar que, de las 12 paltas son evaluadas aplicando el método de los 3 momentos estadísticos para hallar la simetría todas fueron simétricas en ambos ejes con un valor promedio de 0.0080533 para el eje X y -0.00026485 para el eje Y; así mismo se puede apreciar que, al aplicar el método de puntos equidistante se obtiene un valor muy cercano a 1 (0.96); valor que nos permite deducir que, la palta es simétrica.
- e)** En el caso de la variabilidad de los resultados de simetría, estos fueron mínimos dando a entender que los datos están agrupados.



CAPITULO V: PROPUESTA DE INVESTIGACIÓN

5.1. Introducción

Como ya se mencionó en la parte de la situación problemática, las empresas que exportan fruta utilizan el método de enfriamiento o congelación como método de conservación, dicho método se basa en la forma geométrica que presenta el producto para poder estimar el tiempo de transferencia de calor que necesita el producto para que se mantenga en óptimas condiciones, pero la palta del tipo Has por ser un fruto que posee diferentes formas geométricas de forma irregular y más cuando en su interior tiene una semilla o pepa, hace que este método no sea el adecuado causando daños en la calidad del producto, es por ello que es necesario hacer algunos ajustes al modelo de transferencia de calor de dicho método; con el fin de lograr ello es necesario saber que tan simétricas son las paltas en conjunto para poder a estimar los ajustes necesarios en el modelo, por ello se ha estudiado el concepto de simetría.

en la presente tesis se pretende hallar la simetría de la palta Has.

Caber resaltar que los algoritmos y plagan fueron ejecutados en una maquina con las siguientes características: procesador Intel Core i5 de 2.60GZ, memoria RAM de 6GB.

5.2. Desarrollo

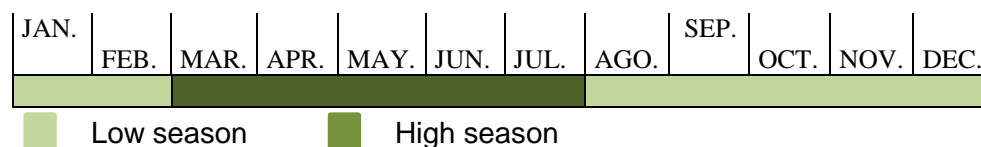
5.2.1. Identificación de las características de las paltas

La palta tiene origen mesoamericano debido a que sus restos fósiles de este fruto fueron hallados en la ciudad de Puebla de México; su nombre original de este fruto es Perseo americana; esta fruta es conocida con diferentes nombres según las diferentes lenguas existentes. (MINAGRI-DGPA, 2015)

La palta del tipo “Has” se cosecha entre los meses de marzo a julio en la costa del Perú y entre los meses de agosto a febrero es época de sembrío.



Tabla 8 Época en el que se siembra la palta Hass en el Norte del Perú



Fuente: AREX

Diversos organismos como el MINAGRI, AREX, entre otros, mencionan que el peso de la palta oscila entre 140 a 350 gramos, la misma que presenta forma oval con piel rugosa de color verde.

El AREX en su FRESH AVOCADO Ficha técnica especifica los siguientes pesos para la exportación de esta fruta:

Tabla 9 Calibres de exportación a los EE. UU

USA	(gris.)
28	364-462
32-36	300-371
36-40	258-313
50	227-274
50	203-243
60	184-217
60	165-196
70	151-175
70	144-157
84	134-147
84	123-137
84	100-142

Fuente: AREX

Sabiendo las características de la palta de exportación, se adquirieron 12 paltas del tipo Hass en el del distrito de Olmos provincia de Chiclayo departamento de Lambayeque en Perú



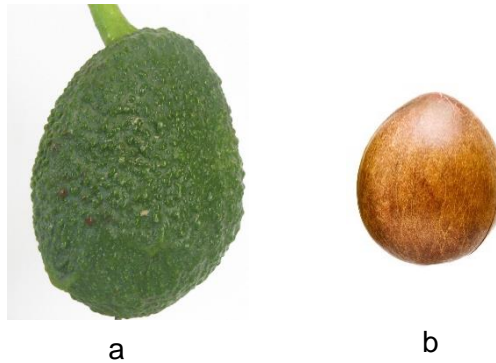


Figura 10: Imagen de la palta Hass (a) y su pepa (b)

Fuente: Elaboración propia

5.2.2. Construir un prototipo para ambiente controlado de adquisición de imágenes digitales de la palta Hass

Quando se construye un prototipo para ambiente controlado, permite configurar y hacer las modificaciones necesarias de los parámetros que mejor se adecuen a lo que se está trabajando; estas modificaciones se realizan en base a juicio de los expertos en el campo, lo que permite obtener mejores resultados.

En este aspecto, se construyó un primer prototipo para realizar las tomas fotográficas a las paltas con las siguientes características:

- a. Base de cartón de tamaño 40cm X 40cm con un espesor de 1cm.
- b. Un plato giratorio de 15 cm con un espesor de 0.5cm y con un punzón en el centro de 2cm, lo que permitió incrustar la palta.
- c. En el plato se dibujó 18 ángulos de 10° y en la base se dibujó un ángulo de 10° para verificar que los grados dibujados en el plato giratorio se mantengan.
- d. La base del plato es fue de 2.5cm.
- e. Para el fondo se utilizó tela blanca y dos parantes de 22 cm de alto que sostienen la misma.



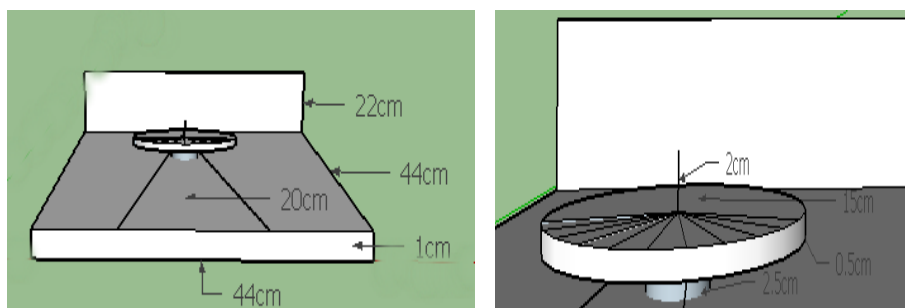


Figura 11: Prototipo para la toma fotográfica de la palta Hass

Fuente: Elaboración propia

En este prototipo se encontró muchas deficiencias, una de ellas es que no fue posible establecer una distancia y una altura en común, al momento de pasar a los 10 ° siguientes, el eje y cambiaba lo cual causaba que la palta en la imagen digital no tenga el mismo eje para todas.

En este aspecto se hizo las correcciones necesarias al prototipo teniendo en cuenta las deficiencias antes mencionadas, por lo que al final el prototipo quedo con las siguientes características:

- a) Se utilizó cartón prensado como base.
- b) Se añadió un brazo deslizante con un punzón en la parte final para que se adecue al tamaño de la palta y se pueda mantener el eje Y.
- c) Se añadió un riel con un largo de 22cm para establecer la distancia necesaria
- d) Otro riel de 22cm de altura con niveles para establecer la altura necesaria, la distancia entre cada nivel es de 1cm.
- e) Una tabla rectangular de 15cm X 8cm para asentar la cámara.

Al final de haber hecho las modificaciones necesarias, el diseño del prototipo para las tomas fotográficas quedo de la siguiente manera:



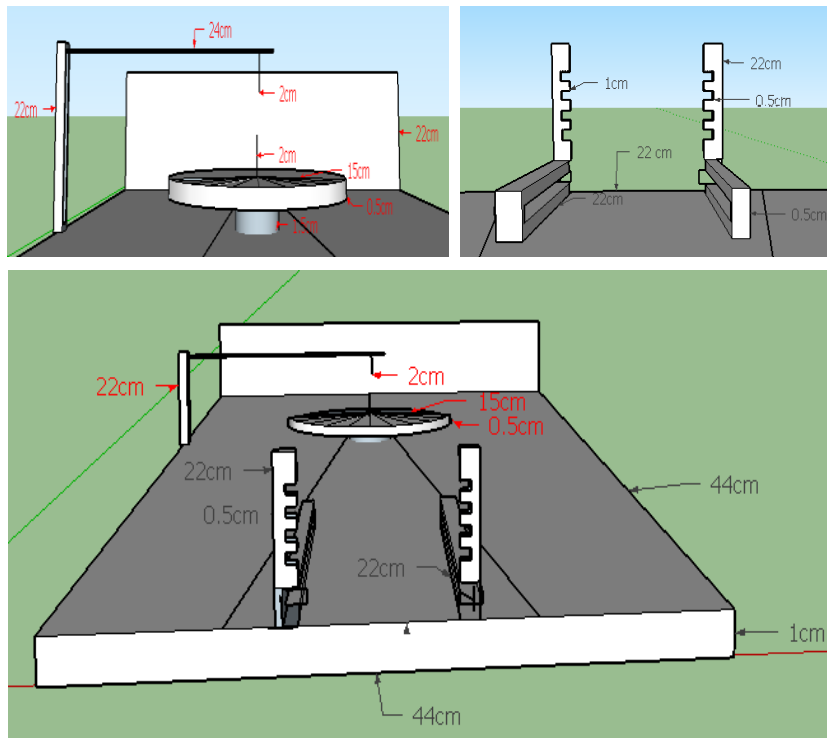


Figura 12: Prototipo final para la toma de fotografías de la palta Hass

Fuente: Elaboración propia

Cabe recalcar que, el uso de los 10° fue validado por expertos en el campo como lo es el Dr. Manuel Forero Vargas

5.2.3. Obtención de Tomografías de las paltas

Según el autor Hofer(2008) en su libro Manual Práctico Tomografía Computarizada (TC): Introducción a la TC , afirma que, las tomografías computarizadas realizan cortes axiales, donde el grosor de cada corte se predefine al principio del estudio, lo cual origina una serie de imágenes bidimensionales del cuerpo con espacios iguales; los sistemas modernos realizan aproximadamente 1400 proyecciones en 360°.

Por ello en esta investigación se utilizó imágenes tomográficas de frutas con pepa, como es el caso de la palta Hass.



Con el fin de mantener el control de las paltas al momento de sacar las tomografías, se construyó un prototipo a base de cartón grueso para transportar las paltas; cuyas características son las siguientes

- a. Tamaño de 20 cm cuadrados, con una altura de 2cm
- b. 9 divisiones de 6cm cuadrados

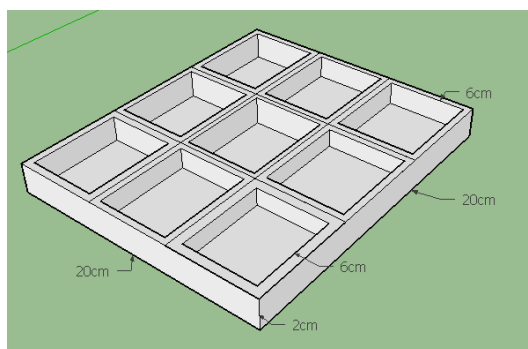


Figura 13: Prototipo para el transporte de la palta Hass para las Tomografías

Fuente: Elaboración propia

Una vez construido el prototipo, se procedió a realizar el etiquetado a las paltas para poderlas identificar en la tomografía.

Las tomografías se obtuvieron de 12 paltas Hass; las cuales presentan 3 cortes: sagital, coronal y axial, con 91 cortes transversales.

Estas tomografías se realizaron en el hospital Essalud "HEYSEN"- Chiclayo-Lambayeque.



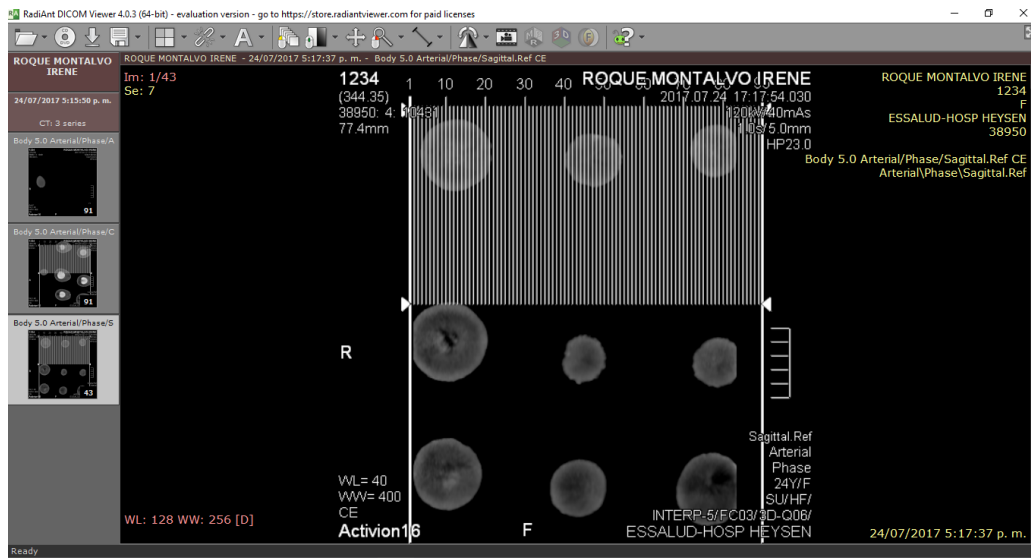


Figura 14: Visualización de las tomografías

Fuente: Elaboración propia

5.2.4. Peso de la pulpa, cascara y pepa de la palta

Realizar este paso va a permitir más adelante validar el peso y el volumen de la que genere la representación gráfica 3d de la palta; por ello esta parte se realizó en el laboratorio de agrobiotecnología de la facultad de agro industrial de la Universidad señor de Sipan, en donde se procedió a pesar las mismas en una balanza electrónica de marca A&D modelo GR-200, con una capacidad máxima de 210g calibrada; los datos arrojados por la balanza se guardaron en un archivo Excel manteniendo el número de la palta para tener el control de los mismas; para más detalle ver **anexo I**. Para sacar la cáscara se empleó un cuchillo de tamaño mediano y se procedió a hacer el peso respectivo.





Figura 15: Balanza Utilizada para pesar la Palta

Fuente: Elaboración propia

5.2.5. Obtención de imágenes fotográficas a las paltas

En esta parte se hizo uso del prototipo descrito en el punto 5.2.2 y se utilizó una cámara fotográfica con las siguientes características:

- a. Marca: canon
- b. Resolución: 50 mgpx
- c. Formato de imagen: Jpeg

Para establecer la distancia y la altura correcta se incrusto una palta entre el punzón del plato giratorio y el punzón del brazo a modo de prueba; lo que permitió establecer diferentes distancias y alturas que mejor localicen el objeto en el centro de la imagen, quedando al final con una distancia de 12cm del plato giratorio y una altura de 4cm.

Una vez establecido la distancia y la altura necesaria, se procedió a hacer la fotografía respectiva partiendo de los 0° avanzando 10° en cada giro del plato hasta llegar a los 180°; esto permite tener la mitad de la palta, lo cual permitirá estimar la simetría.

Una vez ya llegado a los 180°, se guardó las imágenes en un directorio con el siguiente formato: **palta Nro** y dentro de él se puso las imágenes de cada palta con el siguiente formato: **Nro. de palta _Nro. de grado**; ejemplo:



Google Drive > Google Fotos > fotos > fotos

Nombre	Fecha de modifica...	Tipo	Tarr
palta 1	31/08/2017 3:33 p....	Carpeta de archivos	
palta 2	1/08/2017 10:50 p....	Carpeta de archivos	
palta 3	1/08/2017 10:43 p....	Carpeta de archivos	
palta 4	1/08/2017 10:39 p....	Carpeta de archivos	
Palta 5	1/08/2017 8:06 p. m.	Carpeta de archivos	
palta 6	1/08/2017 8:04 p. m.	Carpeta de archivos	
palta 7	1/08/2017 8:01 p. m.	Carpeta de archivos	
palta 8	1/08/2017 7:58 p. m.	Carpeta de archivos	

Figura 16: Formato de nombre de carpeta para guardar las paltas

Fuente: Elaboración propia

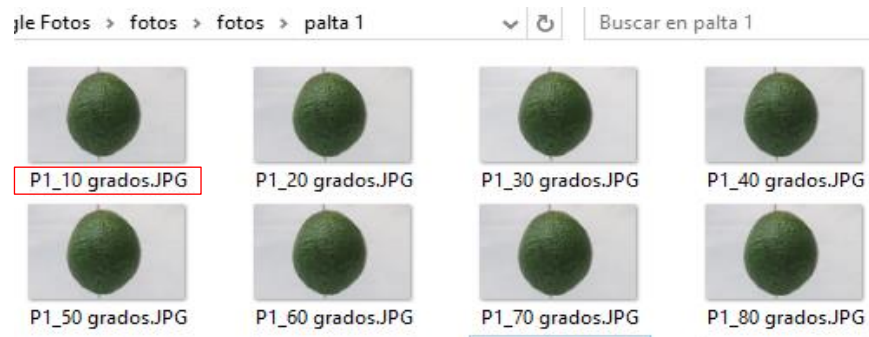


Figura 17: Formato para guardar las paltas

Fuente: Elaboración propia

Esta parte se realizó en las instalaciones del laboratorio de investigación en sistemas inteligentes y seguridad informática(LABSIS), el cual pertenece a la carrera profesional de ingeniería de sistemas de la UNIVERSIDAD SEÑOR DE SIPÁN.

5.2.6. Recorte de las paltas en la tomografía

Debido a que las tomografías se guardan en archivos. Dicom; estas representan un stack de imágenes según la cantidad de cortes que se hayan hecho, por ello fue necesario utilizar una herramienta o programa que permita la visualización del contenido y a la vez permita hacer el recorte de las paltas de forma manual; para ello se ha empleado el programa ImageJ, que es un programa gratuito dedicado al



procesamiento de imágenes y nos brinda gran facilidad para realizar este paso.

Por ello se siguió los siguientes pasos:

- a. Se seleccionó del archivo **dicom** la carpeta que tiene el corte coronal en la tomografía; para ello se hizo uso de la herramienta **RadiAnt** que es un software gratuito que permite visualizar archivos. dicom, el mismo que ordena los archivos según el corte y muestra la información necesaria de la tomografía.

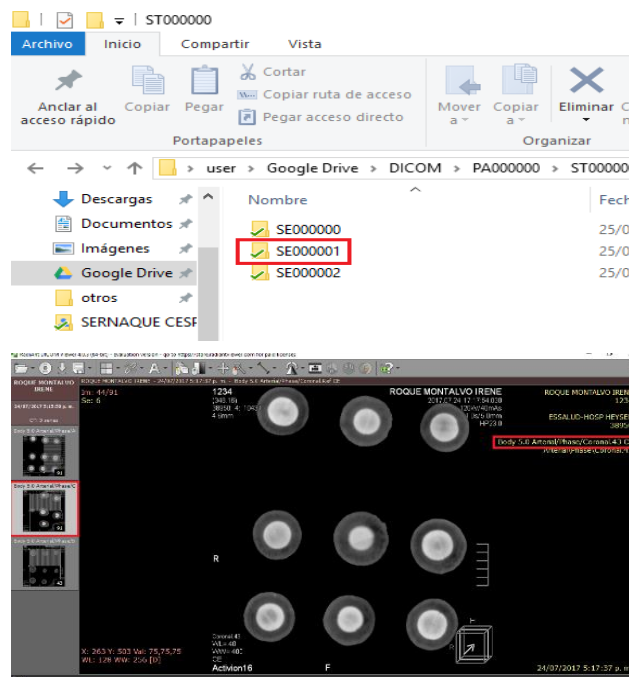


Figura 18: Visualización del corte coronal de la tomografía

Fuente: Elaboración propia

- b. Una vez que se ha identificado la carpeta del corte coronal, todos los archivos que se encuentran dentro de fueron arrastrados hacia ImageJ, en donde se abrió una colección de imágenes; esta colección fue convertida en una pila de imágenes para poder visualizar las imágenes como un video; para ello se utilizó la opción **Image to stack** que proporciona ImageJ, donde se



ingresó la orientación en que se van a insertar las imágenes en la pila y un nombre para la misma.

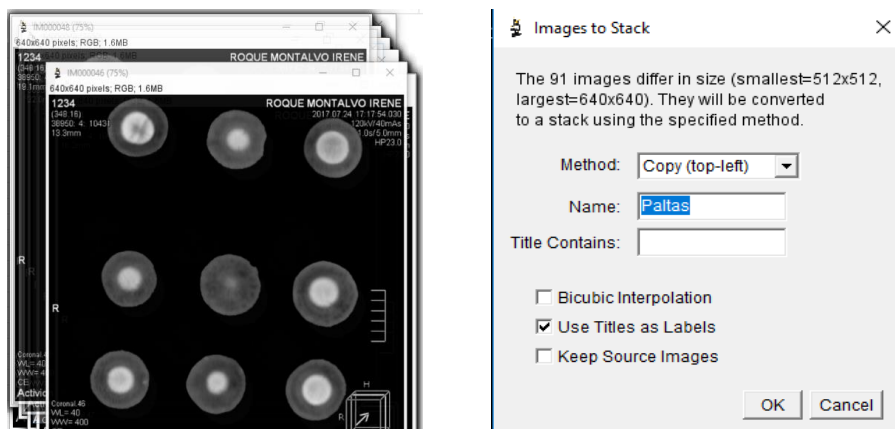


Figura 19: Apilación de los cortes de la tomografía

Fuente: Elaboración propia

- c. Una vez que se apiló las imágenes, se verificó que la pila de imágenes se comporte como un video y a partir de aquí ya se puede hacer la selección de las paltas una por una.



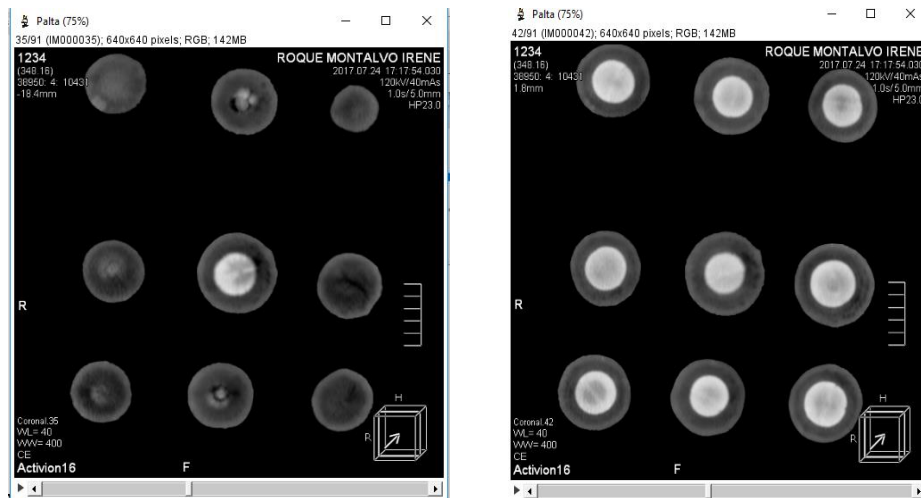


Figura 20: Visualización del Stack de imágenes

Fuente: Elaboración propia

- d. Se identificó el fotograma en la pila de imágenes donde se visualice completamente las paltas; para ello se desplazó la pila de imágenes con la barra de desplazamiento que aparece en la parte inferior.

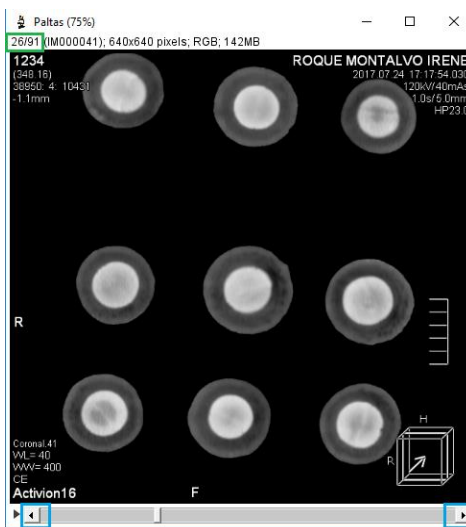


Figura 21: Visualización completa de las paltas en la tomografía

Fuente: Elaboración propia



- e. Una vez que se ha identificado el fotograma, con la herramienta rectángulo que ofrece ImageJ se procedió a seleccionar el área donde se encuentra la palta; el tamaño de esta selección fue uniforme para todas las paltas, para así asegurar que estas tengan las mismas dimensiones lo que ayudará a tener mejores resultados a futuro; en este aspecto **imageJ** nos da la facilidad de poder ingresar el tamaño de la selección haciendo uso a la opción **specify**.

Para esta investigación se trabajó con una altura y un ancho de 140 px.

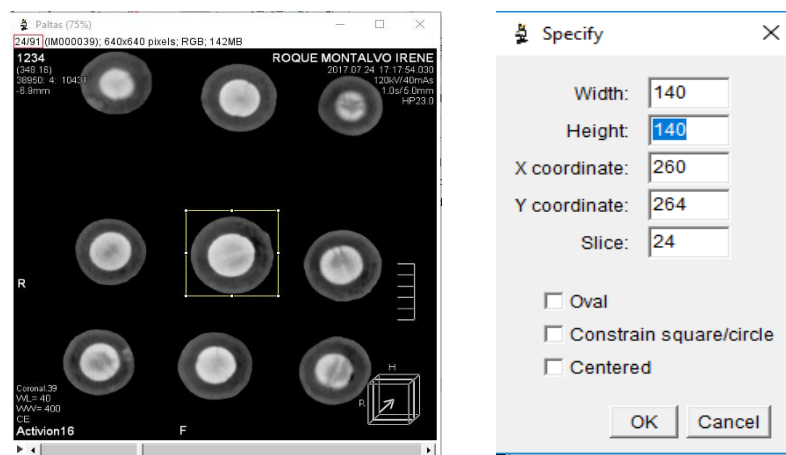


Figura 22: Selección de la palta en la Tomografía

Fuente: Elaboración propia

- f. Después se seleccionó el área de la palta a recortar, se ubicó los fotogramas de inicio y fin en donde aparece la palta seleccionada para luego hacer el recorte en base a estos fotogramas.



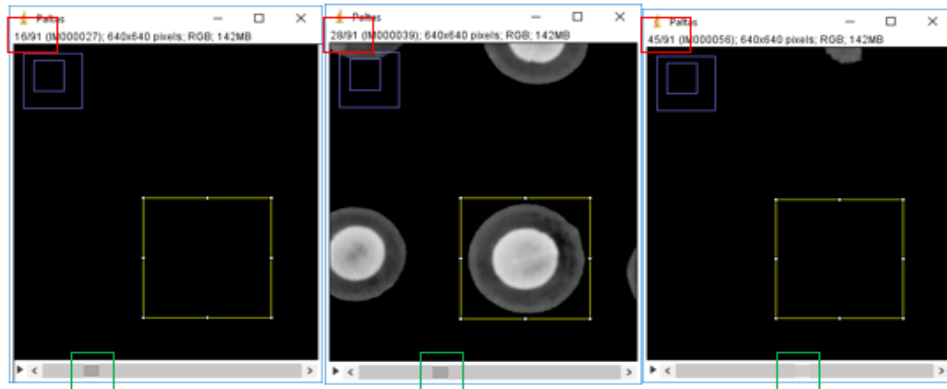


Figura 23: Identificación de fotograma de la palta

Fuente: Elaboración propia

- g. Después que se identificó los fotogramas de inicio y fin, se procedió a duplicar la sección seleccionada, para ello se hizo uso la opción duplicate de ImageJ donde se indicó el rango de fotogramas donde aparece la palta y se le asignó un nombre, en este caso se puso número de palta que es.

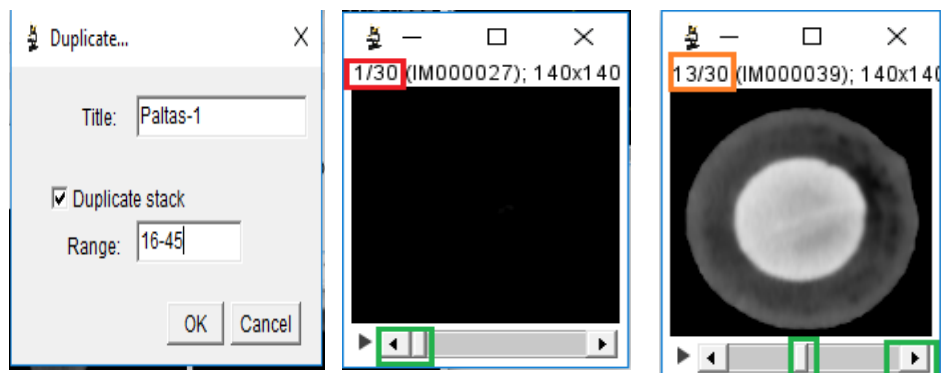


Figura 24: Generación de la pila de imágenes de la palta

Fuente: Elaboración propia

- h. Finalmente se guardó la pila de imágenes en formato **TIFF** para mantener la calidad y se le asignó un formato de nombre: **Nro. de Palta**; para poder identificarlas y mantener un orden.



5.2.7. Conversión de una imagen RGB a escala de grises

El autor Chavez Burbano (2015) menciona que:

“Las imágenes en escala de grises son arreglos unidimensionales. La escala de grises tiene un total de 256 intensidades, donde el 0 equivale al color negro y 255 equivale al blanco”.

Así mismo Jimy & Cortés (2011) testifica que, la conversión de una imagen a escala de grises facilita la extracción de características propias del objeto en estudio; Según el autor Jiménez (2016) afirma que, una imagen en RGB genera 3 matrices con intensidades diferentes según el color y para llevarla a escala de grises es necesario calcular el promedio de las intensidades de los 3 colores(Rojo, verde y azul) y para ello se aplica la siguiente formula:

$$I = Round \left\{ \frac{1}{3} + (R + G + B) \right\} \tag{8}$$

Para entender el funcionamiento de esta fórmula, se puso como ejemplo una imagen en RGB en la cual se aplica la formula (8) para convertirla en escala de grises.

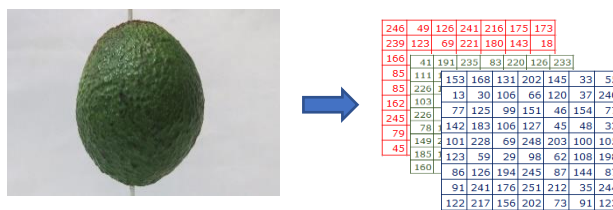


Figura 25: Palta Hass en escala RGB

Fuente: Elaboración propia

Par ello se seleccionó 2 intensidades de la imagen al azar y a estas se le aplico la formula, la cual arroga un numero entre 0 y 255

$$I_{11} = Round \left\{ \frac{1}{3} + (246 + 43 + 153) \right\} = 147$$

$$I_{31} = Round \left\{ \frac{1}{3} + (166 + 226 + 77) \right\} = 156$$



Al aplicar la formula a toda la matriz; se obtiene la siguiente imagen convertida en escala de grises

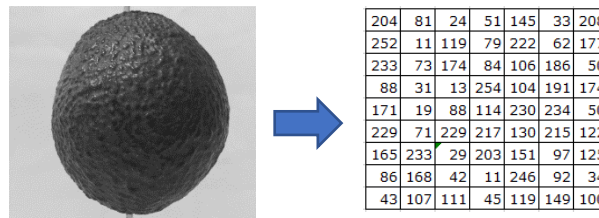


Figura 26: Palta en escala de Grises

Fuente: Elaboración propia

Su algoritmo en el lenguaje de programación Java para hacer este proceso es el siguiente:

```
BufferedImage imagen_gris = new BufferedImage(400, 400,
    BufferedImage.TYPE_INT_RGB);
for (int i=0;i<imagen_escal.getWidht();i++){
    for (int j=0;j<imagen_escal.getHeight();j++){
        int color=imagen_escal.getRGB(i, j);
        Color color_obtenido=new Color(color);
        int color_rojo=color_obtenido.getRed();
        int color_verde=color_obtenido.getGreen();
        int color_azul=color_obtenido.getBlue();
        int color_gris=(color_rojo+color_verde+color_azul)/3;
        Color color_convertido=new Color(color_gris,color_gris,color_gris);
        imagen_gris.setRGB(i,j,color_convertido.getRGB());
    }
}
```

Figura 27: Código de conversión de RGB a escala de grises

Fuente: Elaboración propia

5.2.8. Aplicando la técnica de segmentación para la palta

Los autores Dolgis, Ortega, Miguel, & Benítez (2015) manifiestan que, los métodos de segmentación de un objeto depende de la aplicación a realizar y el tipo de imagen entre otras cosas; en el caso de la segmentación de imágenes médicas afirman que, no existe un método de segmentación eficiente que alcance resultados aceptables.



En tal sentido el autor Nariño (2016) menciona que, el método Otsu es uno de los métodos de segmentación más populares en la literatura debido que se basa en la varianza entre clases para encontrar el umbral óptimo para la imagen.

En ese sentido para este trabajo se escogió el método Otsu como método de segmentación para la fruta de la palta

5.2.8.1. Entendiendo el método Otsu para la segmentación de la palta

Debido a que el método Otsu calcula automáticamente el umbral que mejor se adecue a la imagen en función de la varianza entre clases, fue necesario saber cómo se calcula el umbral y para ello se sigue los siguientes pasos que forman parte de este método:

a. Calculo del Histograma

El histograma se emplea para poder graficar las estadísticas que presenta una imagen; una de las estadísticas que se puede hallar es la frecuencia de los valores de intensidad que presenta una imagen.(Burger & Burge, 2009)

En ese aspecto se identifica los niveles de gris que presenta una imagen y para ello se aplica la siguiente fórmula matemática para poder calcular el histograma:

$$p(i) = n/NP \quad (9)$$

donde:

- N: es el número de pixeles de intensidad i
- NP: el n° total de pixeles de la imagen
- P(i): es la probabilidad de ocurrencia de intensidad i

Para entender como funciona la formula(9), se tomó la siguiente imagen de la palta en escala de grises de resolución 10x10 pixeles para el ejemplo:



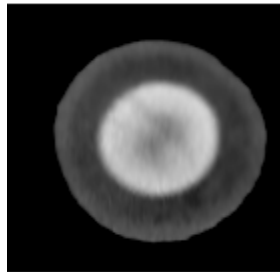


Figura 28: Imagen tomográfica de la palta

Fuente: Elaboración propia

Se asumió que la imagen tiene 4 niveles de gris entre [1-4], donde 1 es el negro y 4 el blanco por lo que se tiene 10 pixeles de nivel de intensidad 1, 20 pixeles de intensidad 2, 30 de intensidad 3 y 40 de intensidad 4, al aplicar la formula (9) se obtiene la siguiente tabla:

Tabla 10 Tabla de resultados de frecuencia de intensidad de pixeles

Nivel	P(i)
1	$P1=10/100=0.1$
2	$P2=20/100=0.2$
3	$P3=30/100=0.3$
4	$P4=40/100=0.4$

Fuente: Elaboración propia

A Partir de dichos valores se construye una gráfica sobre el plano cartesiano donde el eje de las abscisas representa el nivel de intensidad y el eje de coordenadas representa la cantidad de veces que aparece un valor de intensidad dentro del rango de la profundidad de la imagen.



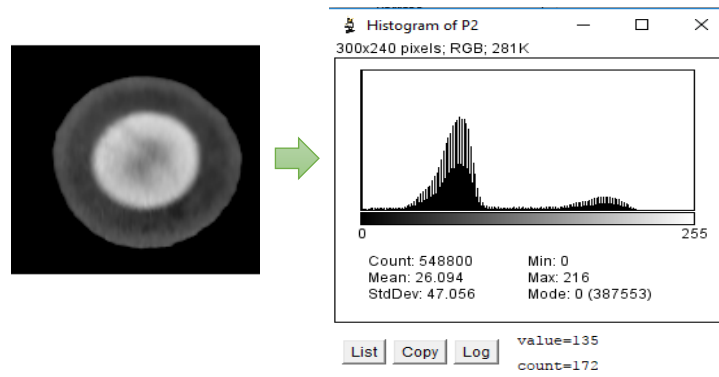


Figura 29: Histograma de la Palta

Fuente: Elaboración propia

b. Cálculo de la probabilidad de niveles de gris

Como en una imagen se presenta diferentes niveles de intensidad, es necesario dividir los pixeles en dos clases C1 y C2, de manera que C1 contiene los niveles de gris [1,..., t]; y C2 los niveles de gris [t+1,..., L]. Otsu definió la distribución de la probabilidad de los niveles de gris con la siguiente fórmula matemática.

$$C_1 = \frac{p_1}{\omega_1(t)}, \dots, \frac{p_t}{\omega_1(t)} \quad (10)$$

$$C_2 = \frac{p_{t+1}}{\omega_2(t)}, \frac{p_{t+2}}{\omega_2(t)}, \dots, \frac{p_L}{\omega_2(t)} \quad (11)$$

Donde:

$$\omega_1(t) = \sum_{i=1}^t p_i \quad (12)$$

$$\omega_2(t) = \sum_{i=t+1}^L p_i \quad (13)$$

Pi: probabilidad de intensidad

W(t)= es la distribución de probabilidad



Para entender mejor esta parte, se continuó con el ejemplo del paso 5.2.8.1 donde se hizo el cálculo del histograma de la imagen de la palta; se consideró un umbral $t = 2$ para calcular la distribución de intensidades aplicando la fórmula matemática (12,13).

Tabla 11 Probabilidad de Intensidad

Nivel	P(i)	distribución de la probabilidad
1	$P1=10/100=0.1$	$\omega_1(t) = 0.1 + 0.2 = 0.3$
2	$P2=20/100=0.2$	
3	$P3=30/100=0.3$	$\omega_2(t) = 0.3 + 0.4 = 0.7$
4	$P4=40/100=0.4$	

Fuente: Elaboración propia

Se puede comprobar que $\omega_1(t) + \omega_2(t) = 1$

c. Cálculo de media para las clases

El autor Garcia Santillán (2016), menciona en su investigación que, para calcular la media de las clases se aplica la siguiente fórmula:

$$\mu_1 = \sum_{i=1}^t \frac{i * p_i}{\omega_1(t)} \tag{14}$$

$$\mu_2 = \sum_{i=t+1}^L \frac{i * p_i}{\omega_2(t)} \tag{15}$$

$$\mu_t = \varpi_1 u_1 + \varpi_2 u_2 \tag{16}$$

Siguiendo con el ejemplo descrito anteriormente, se calcula la media a los datos calculados anteriormente con la fórmula (14,15), quedando la siguiente tabla:



Tabla 12 Cálculo de la media de clases

Nivel	Dist. de la probabilidad	Calc. Media
1	$\omega_1(t) = 0.1 + 0.2 = 0.3$	$\mu_1 = \frac{1 * 0.1 + 2 * 0.2}{0.3} = 1.67$
2		
3	$\omega_2(t) = 0.3 + 0.4 = 0.7$	$\mu_2 = \frac{3 * 0.3 + 4 * 0.4}{0.7} = 3.57$
4		

Fuente: Elaboración propia

d. Calculo de la varianza entre clases

$$\sigma_B^2 = \varpi_1(\mu_1 - \mu_T)^2 + \varpi_2(\mu_2 - \mu_T)^2 \tag{17}$$

Donde:

ϖ : Representa el promedio general

μ : Representa la media de cada clase

μ_t : Media general

σ_B^2 : Representa la varianza entre clases

e. Verificación del umbral optimo

$$t^* = Max\{\sigma_B^2(t)\} \quad 1 \leq t \leq L \tag{18}$$

5.2.8.2. Aplicando el método Otsu

Una vez comprendido cómo funciona el método otsu, se hizo uso de la herramienta imageJ para aplicar este método, para ello se ubicó el fotograma en la pila de imágenes que se había duplicado donde se visualice la palta completa

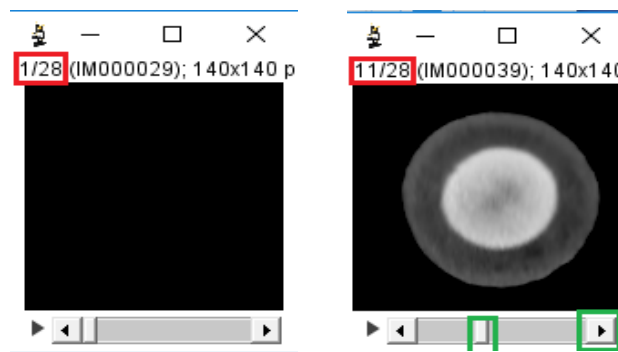


Figura 30: Ubicación del fotograma de la palta para la segmentación



Fuente: Elaboración propia

Después se procedió a aplicar la segmentación haciendo uso de la opción Threshold en imageJ, la misma que abre una nueva ventana donde se muestra la lista de métodos de segmentación que trae imageJ, donde se seleccionó el método Otsu como método de segmentación y por último se habilito el check del stack histogram para que calcule primero el histograma y en base a este calcule el umbral óptimo para la imagen.

	<p>1: Grafica de valores para el umbral.</p> <p>2: Establece el valor mínimo para el umbral.</p> <p>3: establece el valor máximo para el umbral.</p> <p>4: Permite seleccionar el método de segmentación.</p> <p>5: Permite establecer el modo de visualización.</p> <p>6: si está marcado, se calculará primero el histograma de la imagen y en función de él,</p>
<p>7: Permite ver si las funciones son más claras que el fondo.</p> <p>8: El botón auto, aplicara el mejor método que se adecue a la imagen, el apply, aplicara los datos que hemos introducido, el reset cancela cualquier modificación y el set permite ingresar valores a para el umbral.</p>	

Figura 31: Valores para la segmentación

Fuente: Elaboración propia



Al dejar los valores como se muestra en la figura 31, se obtuvo el siguiente resultado:

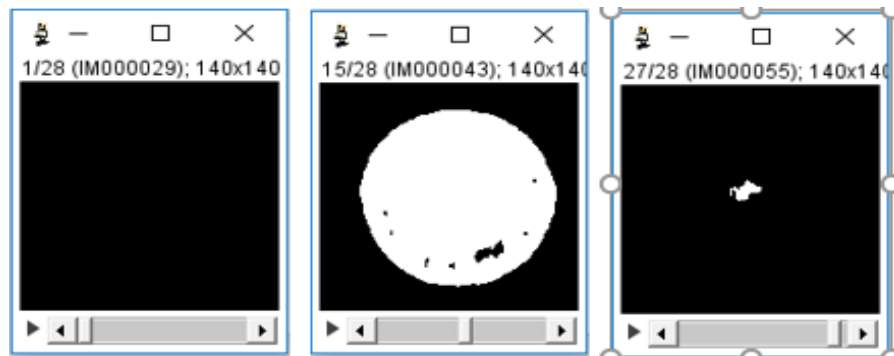


Figura 32: Segmentación de la palta

Fuente: Elaboración propia

Como se puede apreciar, a pesar de que se ha segmentado la palta correctamente en la pila de imágenes, está en su interior contiene puntos negros.

Para dar solución al problema, se aplicó la operación morfológica relleno de regiones, la cual se basa en la dilatación, complementación e intersecciones de conjuntos donde se establece un punto P con valor 1 con el fin de rellenar toda la región con este número y se realiza la operación de dilatación con el elemento B, de manera que esta se comienza a expandir y se interseca con la imagen A^c para mantener los puntos dilatados dentro del borde de la imagen original. El proceso de relleno termina cuando $X_k = X_{k-1}$; es decir cuando ya no haya cambios en la imagen; esta operación morfológica tiene la siguiente expresión matemática.

$$X_k = (X_{k-1} \oplus B) \cap A^c \quad (19)$$



Una que se comprendido como funciona esta operación morfológica, se procedió a realizar la aplicación de esta fórmula sobre la imagen de la palta; como el programa ImageJ ya tiene implementado esta fórmula matemática, no hay necesidad de implementarla; por ello con la opción Fille Holes que proporciona ImageJ se pudo hacer el relleno de regiones, con lo cual se obtuvo el siguiente resultado:

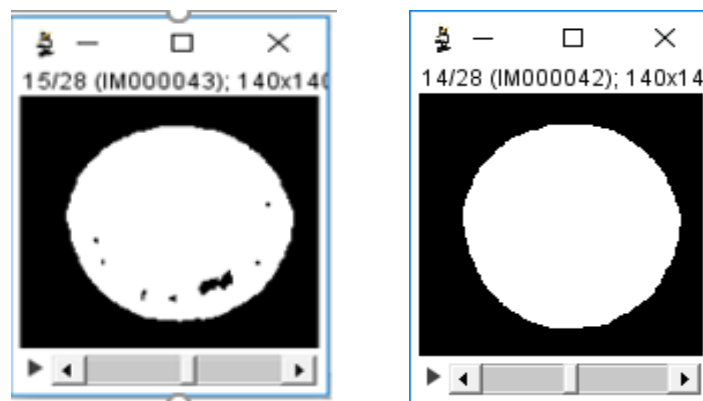


Figura 33: Aplicación de relleno de regiones en la palta 2

Fuente: Elaboración propia

5.2.9. Segmentando la pepa en las imágenes recortadas

Al igual que en la segmentación de la pulpa de la palta, se aplicó el método otsu para la segmentación de la pepa con la diferencia que se desmarco la opción del cálculo del histograma en la ventana de imageJ para que calcule el umbral a partir del fotograma donde se visualiza con claridad la pepa, lo cual permitió la segmentación de la pepa de la palta tal como se muestra en la figura 35.



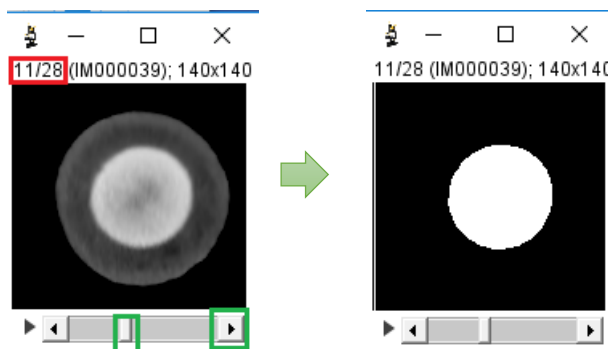


Figura 34: Segmentación de la pepa de la palta 2

Fuente: Elaboración propia

En este caso no hubo la necesidad de aplicar el relleno de regiones debido a que, la segmentación no genero manchas dentro de la pepa.

5.2.10. Aplicando la técnica de PCA

5.2.10.1. Calculó la media del stack de las imágenes

En esta parte se calculó la media a toda la pila de imágenes donde se visualiza la región de interés; para así poder calcular las medias para los ejes X, Y y Z, por ejemplo, se ha calculado la media de una pila de 4 imágenes.

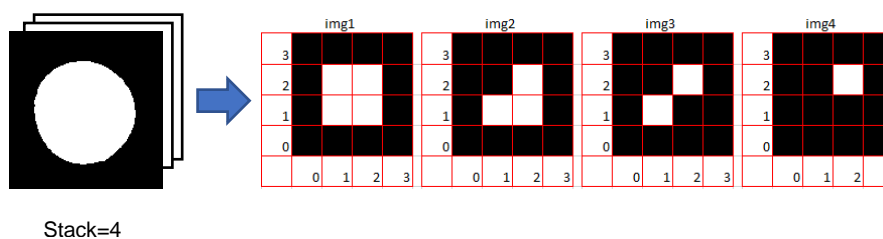


Figura 35: Ejemplo de pila de imágenes para la aplicación de PCA

Fuente: Elaboración propia

Se halló la ubicación donde se encuentra los píxeles de interés en cada imagen; debido a que, la imagen es una representación matricial entonces a través de las coordenadas de los píxeles de



interés se puede hallar los valores para los ejes X e Y y para el eje Z los valores es el número de imagen.

$$\text{Img1} = \{(1,1); (1,2); (2,1); (2,2)\}$$

$$\text{Img2} = \{(1,1); (2,1); (2,2)\}$$

$$\text{Img3} = \{(1,1); (2,2)\}$$

$$\text{Img4} = \{(2,2)\}$$

Una vez hallado las coordenadas de los pixeles; se aplica la fórmula de la media para cada eje; por lo cual se tiene el siguiente resultado:

$$\mu = \frac{\sum_{i=1}^n x_i}{N} \quad (20)$$

Donde:

μ : representa la media de la sumatoria

N: la cantidad de elementos

$\sum_{i=1}^n$: Esa la sumatoria de X_i

$$\text{Media para el eje X} = (1+1+2+2+1+2+2+1+2+2) / 10 = 1.6$$

$$\text{Media para el eje Y} = (1+2+1+2+1+1+2+1+2+2) / 10 = 1.5$$

$$\text{Media para el eje Z} = ((1+1+1+1) + (2+2+2) + (3+3) + 4) / 10 = 2$$



```

int nSlides=sStack.getSize();
int width=sStack.getWidth();
int height=sStack.getHeight();
int size=width*height;
double sumX=0, sumY=0, sumZ=0;
int contPoints=0;

for (int z = 0; z <nSlides; z++) {
    byte[] pixeles=(byte[])sStack.getProcessor(z+1).getPixels();

    for (int i = 0; i < size; i++) {

        if(pixeles[i]!=0){
            sumX+=(i%width);
            sumY+=(i/width);
            sumZ+=z;
            contPoints++;
        }
    }
}
sumX=sumX/contPoints;
sumY=sumY/contPoints;
sumZ=sumZ/contPoints;

```

Figura 36: Implementación de la media en código java

Fuente: Elaboración propia

5.2.10.2. Hallando la varianza para los ejes X, Y e Z

La varianza mide la variabilidad de los datos con respecto a su media; en ese sentido al momento de emplear la técnica de PCA sirve como criterio para seleccionar las componentes principales.(Vieyra, 2013)

$$\sigma^2 = \frac{\sum_{i=1}^n (x_i - \mu)^2}{N} \tag{21}$$

Debido a que, ya se ha calculado los valores para los ejes X, Y e Z con su media respectiva por cada eje, se procede a aplicar la fórmula de la varianza; para ello primero se calcula la varianza individual para luego hacer la sumatoria:

$$v = (x_i - \mu)^2 \tag{22}$$

Aplicando la formula (22) a todos los valores de X, Y y Z

$$v_x = (1 - 1.6)^2 = 0.36$$

$$v_y = (1 - 1.5)^2 = 0.25$$



$$v_z = (1 - 2)^2 = 1$$

Una vez que se ha calculado la varianza por cada elemento de cada eje; el siguiente paso es sumar cada valor individual para obtener la varianza; para ver los cálculos de todos los valores del ejemplo, se recomienda ver el **anexo II**.

Entonces se tiene los siguientes valores de varianza para cada eje:

Varianza del eje X: 0.24

Varianza del eje Y: 0.25

Varianza del eje Z: 1

5.2.10.3. Hallando la matriz de covarianza

Para poder hallar la matriz de covarianza fue necesario calcular los valores de la covarianza de cada elemento de cada eje; para ello se hace uso de la siguiente fórmula de la covarianza:

$$S_{xy} = \frac{1}{n} \sum_i^n (x_i - \mu_x)(y_i - \mu_y) \tag{23}$$

Debido a que, se tiene tres ejes X, Y y Z, se calcula la covarianza de X con Y, X con Z y Y con Z; aplicando la fórmula 23, se tiene los siguientes resultados:

$$S_{xy} = \frac{1}{10} \sum (1 - 1.6)(1 - 1.5) + \dots + (2 - 1.6)(2 - 1.5) = 0.1$$

$$S_{xz} = \frac{1}{10} \sum (1 - 1.6)(1 - 2) + \dots + (2 - 1.6)(4 - 2) = 0.1$$

$$S_{yz} = \frac{1}{10} \sum (1 - 1.5)(1 - 2) + \dots + (2 - 1.5)(4 - 2) = 0.1$$

Una vez que se tiene los valores de la covarianza, se procede a crear la matriz de covarianza y para ello se toma en cuenta los valores de la varianza de los ejes x, y, e z y los valores de covarianza de cada eje; los valores de la varianza de cada eje irán en la diagonal principal de la matriz y los valores de la covarianza a su alrededor; por lo que se tendría la siguiente matriz:



$$S = \begin{pmatrix} 0.24 & 0.1 & 0.1 \\ 0.1 & 0.25 & 0.1 \\ 0.1 & 0.1 & 1 \end{pmatrix}$$

```
double corXX=0,corYY=0,corZZ=0, corXY=0, corXZ=0,corYZ=0;
double [][]mCovar=new double[3][3];

for (int z = 0; z <nSlides; z++) {
    byte[] pixeles=(byte[])sStack.getProcessor(z+1).getPixels();
    for (int i = 0; i < size; i++) {
        if(pixeles[i]!=0){
            double x=i*width;//Coordenada X
            double y=i/width;//coordenadaY

            corXX+=(x-sumX) * (x-sumX);
            corYY+=(y-sumY) * (y-sumY);
            corZZ+=(z-sumZ) * (z-sumZ);
            corXY+=(x-sumX) * (y-sumY);
            corXZ+=(x-sumX) * (z-sumZ);
            corYZ+= (y-sumY) * (z-sumZ);
        }
    }
    //Diagonal
    mCovar[0][0]=corXX/(contPoints-1);
    mCovar[1][1]=corYY/(contPoints-1);
    mCovar[2][2]=corZZ/(contPoints-1);
    //directa
    mCovar[1][0]=corXY/(contPoints-1);
    mCovar[2][0]=corXZ/(contPoints-1);
    mCovar[2][1]=corYZ/(contPoints-1);
    //Inversa
    mCovar[0][1]=corXY/(contPoints-1);
    mCovar[0][2]=corXZ/(contPoints-1);
    mCovar[1][2]=corYZ/(contPoints-1);
}
```

Figura 37: Implementación de la matriz de covarianza

Fuente: Elaboración propia

5.2.10.4. Calculando los eigen vectores y los eigen valores

Una vez que se ha encontrado la matriz de covarianza, se procede a calcular los eigen vectores o vectores propios y los eigen valores o valores propios de una matriz; por lo que primero se calcula los valores propios de la matriz mediante la siguiente formula:

$$(A - \lambda I) = 0 \tag{24}$$

Donde:

A= Matriz original

λ= Número real

I= matriz identidad



Lo primero a desarrollar es resolución de la resta de las matrices; para luego hallar los eigen vectores y eigen valores

$$\begin{pmatrix} 0.24 & 0.1 & 0.1 \\ 0.1 & 0.25 & 0.1 \\ 0.1 & 0.1 & 1 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = 0$$

$$\begin{pmatrix} 0.24 - \lambda & 0.1 & 0.1 \\ 0.1 & 0.25 - \lambda & 0.1 \\ 0.1 & 0.1 & 1 - \lambda \end{pmatrix}$$

Una vez que se ha resuelto la resta de las matrices, se calcula su determinante; por ello el autor Hernandez Cano (2012) menciona que:

“El concepto de determinante o de volumen orientado fue introducido para estudiar el número de soluciones de los sistemas lineales de ecuaciones” (p.48)

En tal aspecto se hace uso de la regla de sarrus, la cual consiste en duplicar las dos primeras filas y colocarlas en la parte inferior de la última fila, para luego multiplicar las diagonales principales y sumar las mismas y luego poder restarlas con las diagonales secundarias.

$$|A| = \begin{vmatrix} 0.24 - \lambda & 0.1 & 0.1 & 0.24 - \lambda & 0.1 & 0.1 \\ 0.1 & 0.25 - \lambda & 0.1 & 0.1 & 0.25 - \lambda & 0.1 \\ 0.1 & 0.1 & 1 - \lambda & 0.1 & 0.1 & 1 - \lambda \\ 0.24 - \lambda & 0.1 & 0.1 & 0.24 - \lambda & 0.1 & 0.1 \\ 0.1 & 0.25 - \lambda & 0.1 & 0.1 & 0.25 - \lambda & 0.1 \end{vmatrix}$$

Entonces al realizar las operaciones necesarias; se tiene el siguiente valor para la determinante

$$|A| = (-\lambda^3 + 1.49\lambda^2 - 0.55\lambda + 0.06) - (0.0149 - 0.03\lambda)$$

$$|A| = (-\lambda^3 + 1.49\lambda^2 - 0.52\lambda + 0.0471)$$

Como la ecuación que arroja es de 3er grado, se utilizó el método de Ruffini para dar solución a la ecuación:



$$\begin{array}{c|cccc}
 & -1 & 1.49 & -0.52 & 0.0471 \\
 1.029 & \downarrow & -1.029 & 0.474 & 0.0471 \\
 \hline
 & -1 & 0.461 & -0.456 & 0
 \end{array}$$

Después de que se ha aplicado el método de Ruffini; este mismo arroja los siguientes valores que dan solución a la ecuación de 3er grado:

$$\lambda = 1.029 \quad \lambda = 0.315 \quad \lambda = -0.144$$

Donde los valores de λ son los eigen valores o valores propios de la matriz de covarianza; con los cuales se procedió a calcular los vectores propios de la matriz y para ello se aplica la siguiente formula:

$$(A - \lambda I)v = 0 \tag{25}$$

Donde:

A: es la matriz de covarianza

λ : es el valor propio de la matriz de covarianza

I: es la matriz de Identidad

v= es el vector propio de la matriz de covarianza

Como ya se ha encontrado los valores propios de la matriz; estos se reemplazan en el valor de λ , para luego multiplicarlo por la matriz identidad, lo que generaría la siguiente matriz:

$$\left(\begin{bmatrix} 0.24 & 0.1 & 0.1 \\ 0.1 & 0.25 & 0.1 \\ 0.1 & 0.1 & 1 \end{bmatrix} - 1.029 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \begin{bmatrix} x \\ y \\ z \end{bmatrix} = 0$$

$$\begin{bmatrix} -0.789 & 0.1 & 0.1 \\ 0.1 & -0.779 & 0.1 \\ 0.1 & 0.1 & -0.029 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = 0$$



Resolviendo la operación de multiplicación de matrices con valores que nos den 0 como resultado; se cuenta con los siguientes valores para x, y, e z del vector 1:

$$x = 0.726 \quad y = -0.686 \quad z = -0.004$$

Después de sustituir todos los valores de λ ; se tiene los siguientes valores para los vectores propios:

$$v_1 = (-0.671, -0.712, 0.202)$$

$$v_2 = (0.726, -0.686, -0.004)$$

$$v_3 = (0.142, 0.143, 0.979)$$

Cabe recalcar que los valores tanto para los vectores propios como para los valores propios se trabajó en base a 3 decimales sin aproximación; los vectores antes encontrados generan la siguiente gráfica:

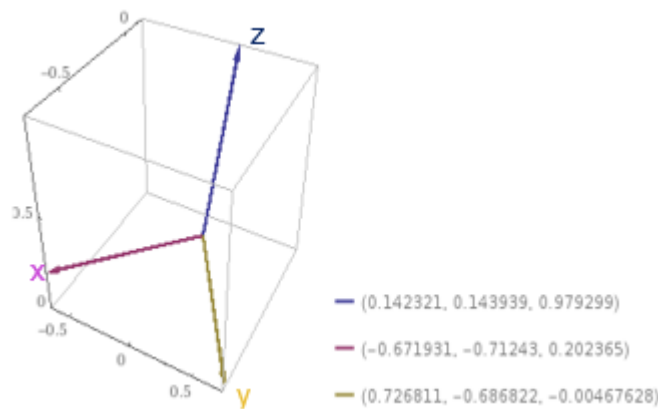


Figura 38: Vectores Propios de la Matriz de Covarianza

Fuente: Elaboración propia

Estos vectores Representan las componentes X, Y, Z

El código de la implementación de esta parte se encuentra en el **anexo VIII**



5.2.11. Cálculo de los ángulos directores de las componentes principales del stack de la palta

Vallejo Ayala & Zambrano(2010), en su libro “Física Vectorial” menciona que:

“Los ángulos directores son aquellos ángulos que forman el vector con los ejes positivos x e y del sistema de coordenadas rectangulares, y varían entre 0° y 180°” (pág. 23,24).

Una vez calculado los vectores propios y sabiendo las componentes principales; se calcula los ángulos directores de cada componente y para ello se aplica la siguiente fórmula matemática:

$$\left. \begin{aligned} \alpha &= \arccos\left(\frac{x}{|v|}\right) \\ \beta &= \arccos\left(\frac{y}{|v|}\right) \\ \theta &= \arccos\left(\frac{z}{|v|}\right) \end{aligned} \right\} \text{ donde: } 0 \leq \alpha, \beta, \theta \leq 180^\circ \quad (26)$$

Para entender mejor esta fórmula; se ha graficado la primera componente para calcular sus ángulos directores:

$$v_1 = (-0.671, -0.712, 0.202)$$

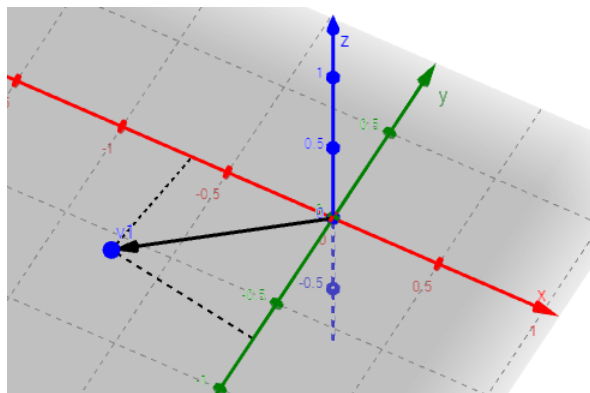


Figura 39: Vector 1 en el plano X,Y

Fuente: Elaboración propia

Lo primero a desarrollar es, encontrar la resultante del vector y para ello se aplica la fórmula de la resultante:



$$|v| = \sqrt{x^2 + y^2 + z^2} \tag{27}$$

Al aplicar la fórmula matemática 27; se tiene como resultante para el vector:

$$|v| = \sqrt{-0.671^2 + -0.712^2 + 0.202^2} = \sqrt{0.998}=0.999$$

Una vez que se ha obtenido la resultante, se empleó la fórmula de los ángulos directores, obteniendo los siguientes valores para los ángulos

$$\alpha = \arccos\left(\frac{-0.671}{0.999}\right) = 132.22^\circ$$

$$\beta = \arccos\left(\frac{-0.712}{0.999}\right) = 135.48^\circ$$

$$\theta = \arccos\left(\frac{0.202}{0.999}\right) = 78.45^\circ$$

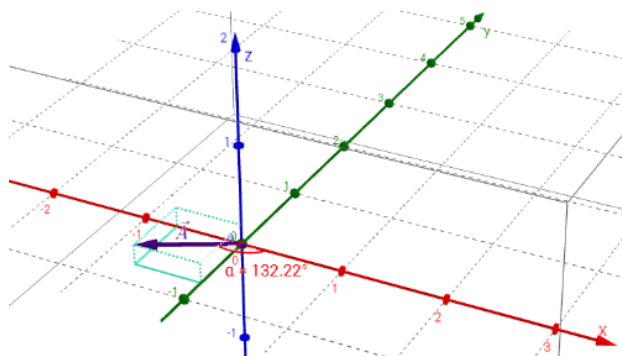


Figura 40: Ángulo director en el eje X

Fuente: Elaboración propia

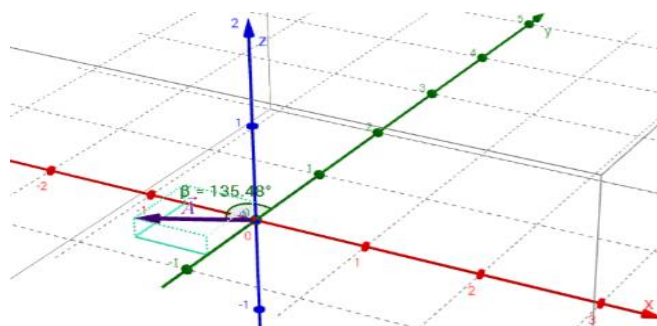


Figura 41: Ángulo director en el eje Y



Fuente: Elaboración propia

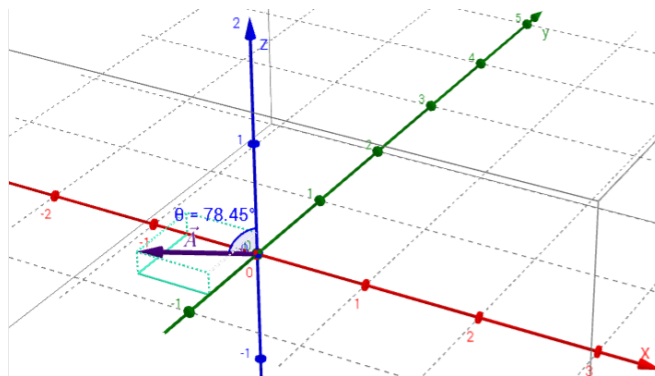


Figura 42: Ángulo director en el eje Z

Fuente: Elaboración propia

Una vez que se ha comprendido el funcionamiento de PCA y de los ángulos directores, se desarrolló un plugin el cual contiene todos los pasos descritos anteriormente; se recomienda revisar el **anexo VI y VII** donde se explica cómo Integrar el IDE de netbeans con imageJ y como crear e instalar plugins para imageJ.

```
public double[] angulosVector(double[] vector) {
    double[] respuesta=new double[3];

    double moduloVector=0;
    for (int i = 0; i < vector.length; i++) {
        moduloVector+=vector[i]*vector[i];
    }
    moduloVector=Math.sqrt(moduloVector);

    for (int i = 0; i < vector.length; i++) {
        double val=vector[i]/moduloVector;
        double anguloRadianes = Math.acos(val);
        respuesta[i] = Math.toDegrees(anguloRadianes);
    }

    return respuesta;
}
```

Figura 43: Implementación del código para calcular el ángulo director

Fuente: Elaboración propia



El código para realizar PCA y el cálculo de los ángulos directores se encuentra en el **anexo VIII** donde se ha programado todas las fórmulas que implican estos métodos; como ejemplo se tiene la siguiente imagen segmentada de la palta 2 en la cual, el plugin desarrollado muestra su matriz de covarianza, valores y vectores propios y sus ángulos directores

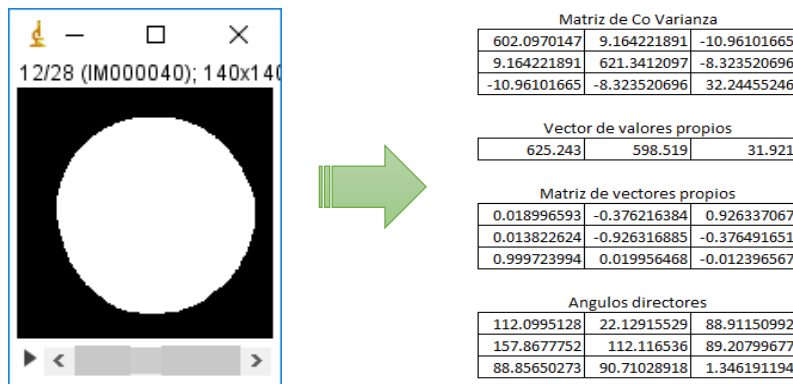


Figura 44: Aplicación del método PCA y cálculo de los ángulos directores en la palta 2

Fuente: Elaboración propia

5.2.12. Alienación del stack de imágenes de la palta

Después de que se halló los ángulos directores de las componentes de la imagen, se procedió centrar o alinear los ejes, de manera que, los ángulos que forma el objeto con los ejes X e Z se aproximen a 90° y en el caso del ángulo Y debe aproximarse a los 180° o a los 0°; para ello se restó 90° al eje Z por única vez y 90° en eje Z; el proceso de restar 90° al eje X finaliza cuando los ejes se han alineado.

Por ejemplo, se tuvo la imagen de la palta nro. 2, en la cual se alinea sus ejes.



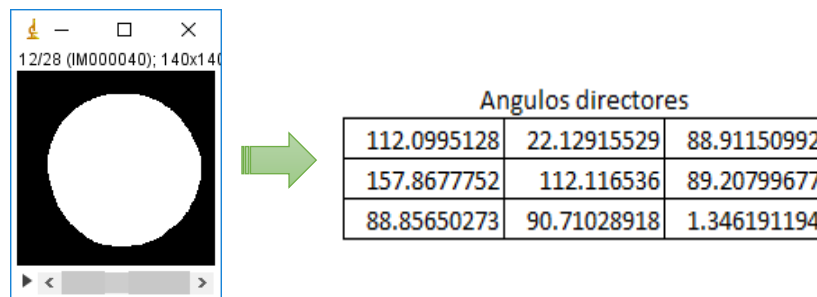


Figura 45: Palta Nro. 2 con sus ángulos directores

Fuente: Elaboración propia

De la figura 45 se tiene los siguientes ángulos para cada componente:

$$x = 112.09951277460992, 22.129155293455785, 88.91150992250272$$

$$y = 157.86777522433482, 112.1165360296299, 89.20799676919121$$

$$z = 88.85650272785098, 90.71028917859006, 1.3461911944813676$$

Se seleccionó el primer ángulo de las componentes x e z y con estos ángulos se hizo la rotación de la imagen; por ello se descargó el plugin TransformJ y se instaló en imageJ; este plugin permite ingresar el valor del ángulo en el eje z para así hacer la rotación de la imagen en ese eje; al utilizar TransformJ, se debe tener en cuenta lo siguiente:

Para ajustar el ángulo en el eje Z, se debe utilizar X-Angle, debido a que, en este plugins de IMAGEJ los resultados del cálculo del ángulo con PCA tienen la siguiente correspondencia.

$$X = Z\text{-Angle}$$

$$Y = Y\text{-Angle}$$

$$Z = X\text{-Angle}$$

Para hacer la rotación se trabajó con aproximación a 2 decimales después de restar los 90°.

Al hacer la rotación, este plugin genera una nueva imagen donde se puede visualizar la rotación el eje Z:



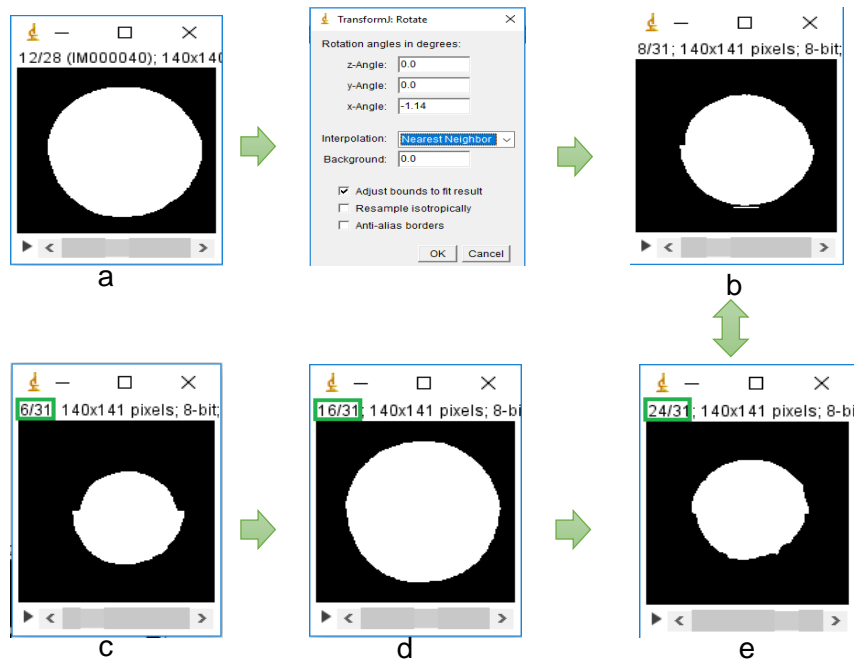


Figura 46: Rotación de la imagen con TransformJ en el eje Z (a) stack original, (b) stack rotado, (c, d, e) imágenes que conforman el stack de la imagen rotada

Fuente: Elaboración propia

Como se puede apreciar en la Figura 46, después de que se ha hecho la rotación a la pila de imágenes, esta ha sufrido cambios en todas las imágenes que conforman el stack de imágenes; a esta nueva imagen se le volvió a aplicar la técnica de PCA para calcular los nuevos ángulos directores de la nueva imagen y a hacer la rotación en el eje x.

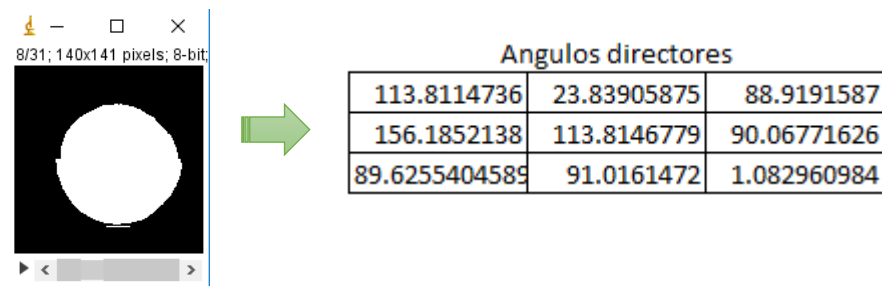


Figura 47: Palta 2 con sus nuevos ángulos directores

Fuente: Elaboración propia

Como ya se ha especificado anteriormente, se resta 90° a cada ángulo del primer ángulo director de la componente x:



$$x = 113.348 - 90 = 23.35^\circ$$

Mediante la opción rotate que ofrece ImageJ, se hizo la rotación en el eje x con el ángulo que da como resultado después de restar los 90°.

$$x = 113.348 - 90 = 23.35^\circ$$

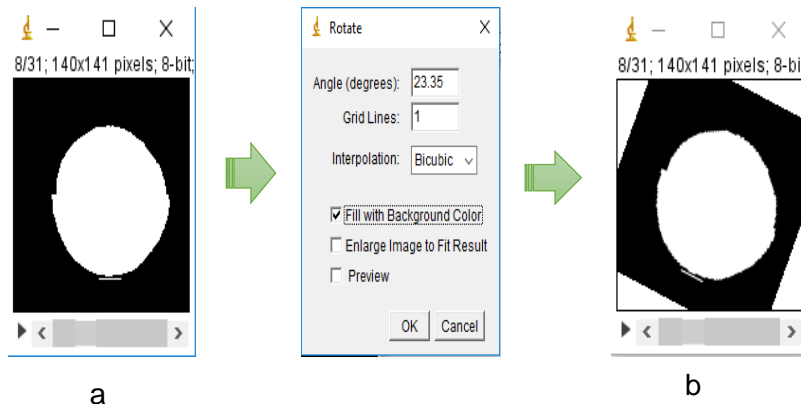


Figura 48: Rotación de la Imagen en el eje X (a) imagen rotada en el eje Z (b) imagen rotada en el eje X

Fuente: Elaboración propia

El proceso de rotación en el eje X se hizo hasta que las componentes X, Y, Z en sus primeros ángulos queden de la siguiente manera:

$$x = 90^\circ \quad y = 180^\circ \text{ o } 0^\circ \quad Z = 90^\circ$$

Esto indica que el objeto se ha alineado correctamente; cabe recalcar que, los cálculos de estos ángulos son aproximados.

En el **anexo IV** se puede visualizar los ángulos por cada rotación que se hizo en cada stack de imágenes de cada palta.

5.2.13. Hallando el contorno de la palta en la pila de imágenes

En esta parte se utilizó el método desarrollado por el Doctor Manuel Forero Vargas para imageJ **ver anexo IX**, el cual permite hallar el borde del objeto en estudio y sacar las coordenadas de los mismos en base a un ángulo dado; para ello seguimos el siguiente proceso:



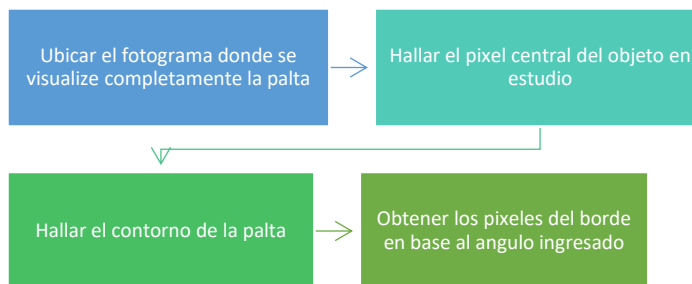


Grafico 5: Proceso para hallar el contorno de la palta Hass

Fuente: Elaboración propia

5.2.13.1. Hallar el pixel central del objeto en estudio

En esta parte se recorre todo el stack de imágenes y por cada imagen se obtiene el ancho y su alto, lo que permite obtener la cantidad de pixeles que presenta cada imagen, para así poder generar un vector que contenga estos pixeles.

Después se halla los valores para la coordenada en X e Y, para ello primero se halla el valor de X mediante el modulo entre el valor del pixel y el ancho de la imagen; en el caso de la coordenada Y se divide el valor del pixel entre el ancho de la imagen.

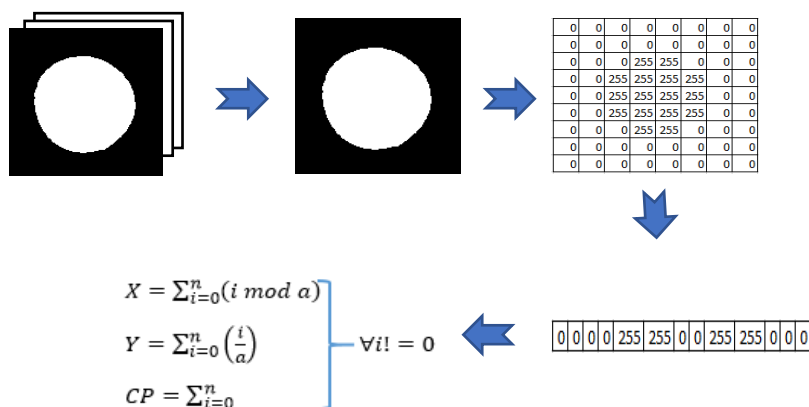


Figura 49: Proceso para hallar el pixel Central

Fuente: Elaboración propia

Donde:



i = pixel con valor 255

n= cantidad de pixeles de la imagen

a= ancho de la imagen

cp= cantidad de puntos

X, Y= representa la sumatorias en ambos ejes

Una vez calculado la sumatoria de los ejes X e Y y la cantidad de puntos, se halla la coordenada para el eje X e Y; para ello se aplica la siguiente fórmula matemática:

$$X = \frac{\sum x}{cp} \tag{28}$$

$$Y = \frac{\sum y}{cp} \tag{29}$$

Donde:

X e Y = son las coordenadas del pixel central del objeto en estudio

El código en para el plugin para calcular el pixel central fue el siguiente:

```
private int[] getCenterPixel(ImageStack sStack) {
    int nSlides = sStack.getSize();
    int width = sStack.getWidth();
    int height = sStack.getHeight();
    int size = width * height;
    double sumX = 0, sumY = 0, sumZ = 0;
    int contPoints = 0;
    int[] pixelCentral = new int[2];

    for (int z = 0; z < nSlides; z++) {
        byte[] pixeles = (byte[]) sStack.getProcessor(z + 1).getPixels();
        for (int i = 0; i < size; i++) {
            if (pixeles[i] != 0) {
                sumX += (i % width);
                sumY += (i / width);

                contPoints++;
            }
        }
    }
    sumX = sumX / contPoints;
    sumY = sumY / contPoints;
    pixelCentral[0] = (int) sumX;
    pixelCentral[1] = (int) sumY;
    return pixelCentral;
}
```



Figura 50: Código para encontrar el pixel central de la imagen

Fuente: Elaboración propia

Al ejecutar el pequeño código escrito en el lenguaje de java en la figura 50, se tiene el siguiente resultado

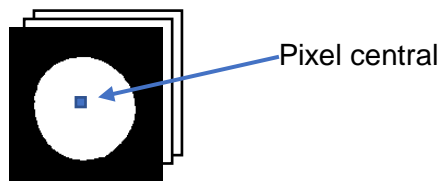


Figura 51: Imagen del pixel central

Fuente: Elaboración propia

5.2.13.2. Hallar el contorno de la palta

Para poder hallar el contorno de la palta es necesario tener en cuenta los 8 direcciones que se presenta en un plano cartesiano:

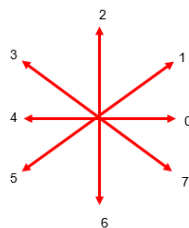


Figura 52: Las 8 direcciones del plano cartesiano

Fuente: Elaboración propia

Estas direcciones van a permitir saber si el pixel que se está recorriendo es un borde del objeto o no en cada una de las imágenes que conforman el stack de imágenes; para ello se escoge una dirección de las 8 direcciones que presenta el pixel central y se aplica la siguiente formula:

$$D = (D_m + 2) \bmod 8 \tag{30}$$

Por ejemplo, se tiene la siguiente imagen de la palta 2 con su pixel central:



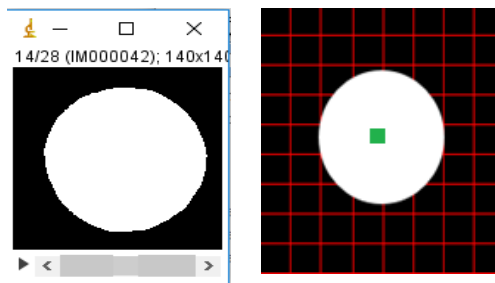


Figura 53 Palta 2 con su representación matricial para la detección del borde

Fuente: Elaboración propia

Se obtienen los valores de los pixeles vecinos (0 y 255) con respecto al pixel actual y dependiendo del valor del pixel vecino, se calcula la siguiente dirección a desplazarse en sentido horario y se ubica el nuevo pixel a posicionarse.

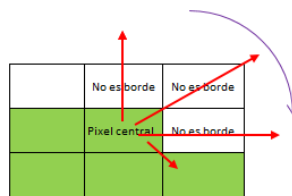


Figura 54: Desplazamiento para encontrar la nueva dirección a moverse

Fuente: Elaboración propia

Por ejemplo, en la figura 54 se visualiza el desplazamiento para encontrar el borde del objeto en la imagen.

Se recomienda revisar el **anexo V** donde se muestra más a detalle el proceso del método de detección de borde desarrollado por el DR. Manuel Forero Vargas; el código fuente está en el **anexo IX**.



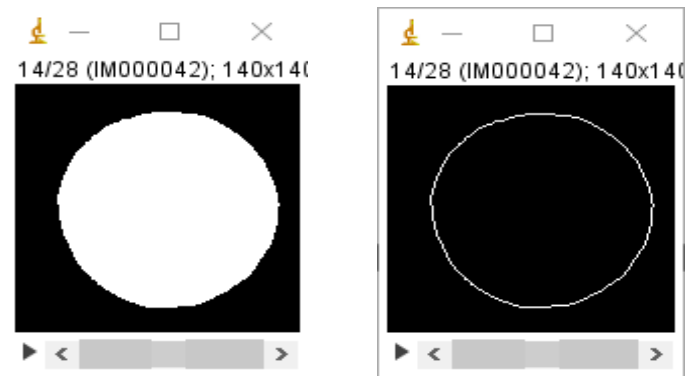


Figura 55: Borde de la palta Hass en el stack de imágenes

Fuente: Elaboración propia

5.2.14. Obteniendo la nube de puntos del objeto

La nube de puntos es una colección de puntos en coordenadas x e y fijadas de un elemento del mundo real; donde cada punto representa la distancia desde el origen hasta el punto donde fue capturado. (Heredia Favieri, 2015)

Así mismo HERNÁNDEZ CASTORENA, (2015) en su investigación menciona que, la nube de puntos representa la geometría y morfología de un objeto real; por lo que esta nube de puntos puede ser analizada e interpretada por varios programas tales como Matlab.

En tal sentido, se procedió a generar la nube de puntos del objeto en cada imagen que forma parte del stack de imágenes en base a un ángulo de 5°; luego se guardó estos puntos en un archivo de texto con las coordenadas de cada punto.



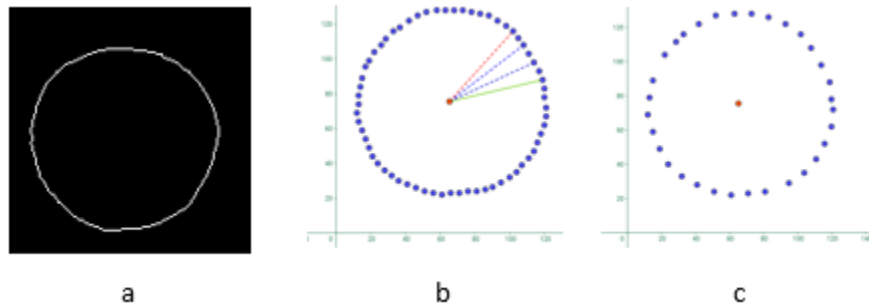


Figura 56: Nube de puntos del contorno de la palta : a) imagen del contorno de la palta b) imagen hallando el borde cada 5° c) imagen de la nube de puntos

Fuente: Elaboración propia

5.2.15. Entendiendo el método de los momentos estadísticos

En este apartado se aplica los momentos de una variable aleatoria discreta, por ello el autor Julie y Billy (2007) menciona que:

“Los momentos de una variable aleatoria son valores esperados de algunas funciones de la variable aleatoria y constituyen a una colección de medidas descriptivas con las que se puede caracterizar de manera única a su distribución de probabilidad”

5.2.15.1. Cálculo del primer momento estadístico

Este momento es conocido como la media aritmética; por ello Gorgas García, Cardiel López, & Zamorano Calvo,(2011), menciona que, la media pertenece a las medidas de centralización, lo que indica el promedio de los datos aplica la siguiente fórmula matemática:

$$m_1 = \frac{\sum(x_i - \mu)^1 * n_i}{n} \quad (31)$$

Donde:

x_i : es una variable discreta

n_i : la frecuencia

n : el total de números



Para entender mejor la fórmula 31, se ha utilizado una imagen del stack de imágenes de la palta 2 como muestra para hacer los diferentes cálculos:

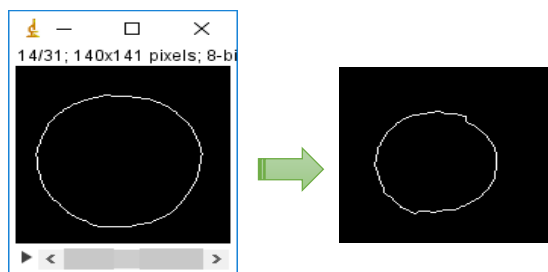


Figura 57: Imagen de ejemplo para hallar los momentos estadísticos

Fuente: Elaboración propia

Tabla 13 Valores en x e y de la figura 57 y aplicación del primer momento estadístico

X	y	Promedio		1er momento= $\sum x(x-\mu)=0$	
		Prom X	Prom Y	$(x-u)^1$	$(y-u)^1$
67	75	58.19	73.62	8.81	1.38
66	76	58.19	73.62	7.81	2.38
60	85	58.19	73.62	1.81	11.38
58	84	58.19	73.62	-0.19	10.38
56	83	58.19	73.62	-2.19	9.38
55	81	58.19	73.62	-3.19	7.38
55	79	58.19	73.62	-3.19	5.38
54	78	58.19	73.62	-4.19	4.38
53	77	58.19	73.62	-5.19	3.38
51	75	58.19	73.62	-7.19	1.38
53	73	58.19	73.62	-5.19	-0.62
54	71	58.19	73.62	-4.19	-2.62
55	70	58.19	73.62	-3.19	-3.62
56	69	58.19	73.62	-2.19	-4.62
57	68	58.19	73.62	-1.19	-5.62
58	67	58.19	73.62	-0.19	-6.62
60	67	58.19	73.62	1.81	-6.62
62	67	58.19	73.62	3.81	-6.62
63	67	58.19	73.62	4.81	-6.62
64	67	58.19	73.62	5.81	-6.62
65	67	58.19	73.62	6.81	-6.62

Fuente: Elaboración propia



5.2.15.2. Cálculo del segundo momento estadístico

Este momento se le conoce como la varianza de los datos, con lo cual se mide la dispersión que presentan estos datos con respecto la media; por ello se emplea la siguiente fórmula para su calculo.

$$m_2 = \frac{\sum(x_i - \mu)^2 * ni}{n} \tag{32}$$

Continuando con el ejemplo de la figura 57 del punto anterior, se aplicó la fórmula 32 a dicha figura; obteniendo la siguiente tabla:

Tabla 14 Aplicación del segundo momento estadístico

2do momento= $(\sum(x-\mu)^2)/n$	
$(x-u)^2$	$(y-u)^2$
77.6077097505669	1.90702947845805
60.9886621315193	5.66893424036281
3.2743764172336	129.52607709750600
0.0362811791383	107.76417233560100
4.7981859410431	88.00226757369610
10.1791383219955	54.47845804988660
10.1791383219955	28.95464852607710
17.5600907029478	19.19274376417230
26.9410430839002	11.43083900226760
51.7029478458050	1.90702947845805
26.9410430839002	0.38321995464853
17.5600907029478	6.85941043083901
10.1791383219955	13.09750566893430
4.7981859410431	21.33560090702950
1.4172335600907	31.57369614512470
0.0362811791383	43.81179138322000
3.2743764172336	43.81179138322000
14.5124716553288	43.81179138322000
23.1315192743764	43.81179138322000
33.7505668934240	43.81179138322000
46.3696145124717	43.81179138322000

Fuente: Elaboración propia

5.2.15.3. Cálculo del tercer momento estadístico

Este momento es considerado como momento de asimetría o sesgo de una distribución; por ello se aplica la siguiente fórmula:



$$g_1 = \frac{m_3}{s^3} \quad \text{donde } m_3 = \frac{\sum_{i=1}^k (x_i - \mu)^3 n_i}{N} \quad (33)$$

Donde se cumple la siguiente condición:

- $g_1 = 0$: La distribución es simétrica
- $g_1 > 0$: La distribución presenta asimetría positiva
- $g_1 < 0$: La distribución presenta asimetría negativa

Al igual que en los puntos anteriores, se aplicó la fórmula 33 a la figura 57, teniendo como resultado la siguiente tabla:

Tabla 15 Aplicación de tercer momentos estadístico

3er momento= $(\sum(x-\mu)^3)/n$	
$(x-u)^3$	$(y-u)^3$
683.686966850232	2.63351689882301
476.292409027103	13.49746247705430
5.925062088327	1474.13011553828000
-0.006910700788	1118.69474138862000
-10.510312061333	825.54508152467300
-32.476298455890	402.10290465392500
-32.476298455890	155.80358492603400
-73.585141993305	84.08249649065970
-139.836842673577	38.64712234099980
-371.768815462693	2.63351689882301
-139.836842673577	-0.23723140049671
-73.585141993305	-17.96512255695930
-32.476298455890	-47.40049670661920
-10.510312061333	-98.55015657056480
-1.687182809632	-177.41410214879600
-0.006910700788	-289.99233344131300
5.925062088327	-289.99233344131300
55.285606306015	-289.99233344131300
111.251592700572	-289.99233344131300
196.074721952273	-289.99233344131300
315.754994061117	-289.99233344131300

Fuente: Elaboración propia



Tabla 16 Asimetría delos datos

Asimetría= m_3/s^3	
x	y
0.00465386	0.00185669

Fuente: Elaboración propia

Los resultados de la tabla 15 se compararon con la condición de la fórmula 33, donde se puede ver que, los datos que muestra la tabla 15 cumplen con la condición $x=0$; por lo tanto, se afirma que el objeto es simétrico.

Tabla 17 Cuadro de resumen después de aplicar los 3 momentos estadísticos

Promedio		1er momento: media		2do momento: varianza		3er momento: sesgo o asimetría		Asimetría= m_3/s^3	
x	y	x	y	x	y	x	y	x	y
58.19	73.62	0.00	0.00	21.20181406	37.37868481	44.35395746	96.96425872	0.00465386	0.00185669

Fuente: Elaboración propia

5.2.16. Validación del método de simetría

Para la validación del método de simetría, se tomó como referencia el código descrito en el libro de Numerical Recipes The art of Scientific Computing escrito por (Press, A.Teukolsky, & William, 2007, pag 749), el cual está desarrollado en el lenguaje c y se implementó en lenguaje java por el autor de la presente investigación y el plugin Moment Calculator, el cual fue desarrollado por la University of Ottawa - Earth Sciences en el año 2007; tanto el plugin como el código descrito en el libro de Numerical Recipes The art of Scientific Computing emplean los tres momentos estadísticos.

Para saber si los resultados de simetría son los correctos por parte del plugin y del código descrito en el libro de Numerical Recipes The art of Scientific Computing, se escogieron 11 imágenes sintéticas simétricas y no simétricas tal como se muestra en la figura 58 y 59; estas imágenes presentan un grado de detalle bajo y medio, lo cual hace posible tener



mejores resultados debido a la uniformidad del objeto; en tal sentido el autor Plaza Miguel (2002), afirma que :

“La principal característica diferenciadora de las imágenes sintéticas con respecto a imágenes reales es la posibilidad de obtener datos de verdad terreno de alta precisión a partir de las mismas.”

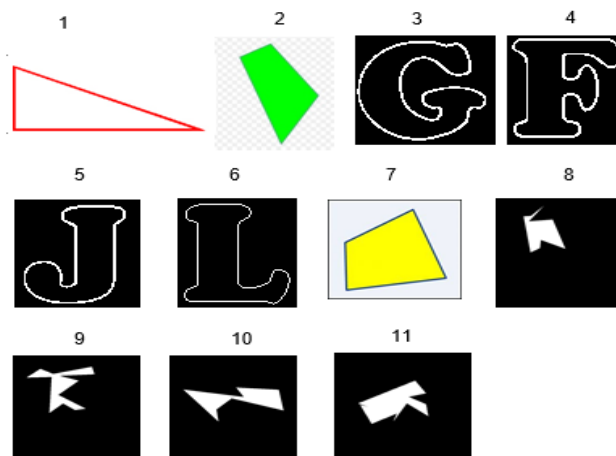


Figura 58: Imágenes sintéticas de objetos no simétricos: 1-10 imágenes normales y la 11 stack de 4 imágenes

Fuente: Internet y Elaboración propia

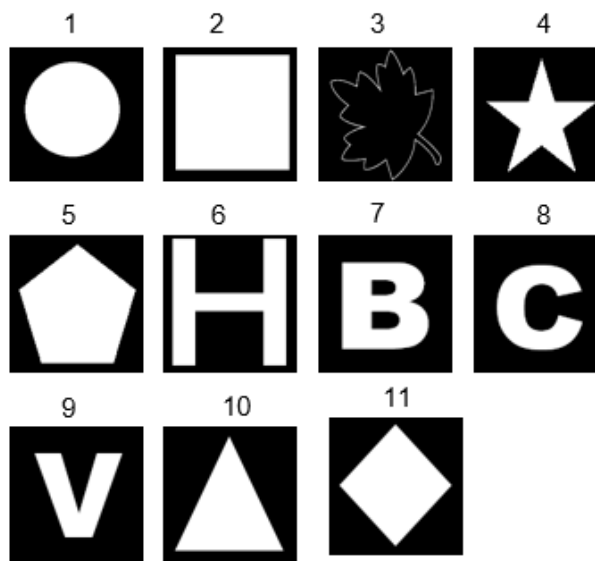


Figura 59: Imágenes sintéticas de objetos simétricos

Fuente: Internet y Elaboración propia



Las resoluciones de las imágenes de la figura 58 y 59 fueron variadas para ver la efectividad del plugin como la del código programado.

5.2.16.1. Probando el código descrito en el libro de Numerical Recipes The art of Scientific Computing

Para aplicar este código se generó la nube de puntos de cada imagen de la figura 58 y 59 y se procedió a ejecutar el código, obteniendo los siguientes resultados:

Tabla 18 Resultado de imágenes asimétricas

Evalu.	M. de libro	
	x	y
no_sim1	-0.192	-0.3558
no_sim2	0.0732	0.0803
no_sim3	0.1839	-0.1934
no_sim4	0.036	0.1472
no_sim5	-0.4582	-0.214
no_sim6	-0.0564	-0.2974
no_sim7	-0.0038	-0.1302
no_sim8	0.5154	-0.23
no_sim9	0.6761	0.0385
no_sim10	0.5653	0.5809
Pila_1	-0.0278	0.046

Fuente: Elaboración propia

Tabla 19 Resultados de imágenes simétricas

Evalu.	M. Estadístico	
	x	y
sim1	0.005	0.002
sim2	0.0126	0.006
sim3	4.717	0.0153
sim4	0.0142	0.0232
sim5	0.0021	-0.1029
sim6	0.0259	-0.0118
sim7	0.1056	-0.017
sim8	0.1144	0.1679
sim9	0.4269	-0.3954
sim10	0.0081	-0.5103
sim11	0.00021497	-0.00021383



Fuente: Elaboración propia

5.2.16.2. Probando Plugin Moment Calculator

Al utilizar el plugin en las figuras 58 y 59, se obtuvo los siguientes datos:

Tabla 20 Tabla de resultados de simetría en figuras asimétricas

Evalu.	M. Estadísticos	
	x	y
no_sim1	0.17	-0.679
no_sim2	0.125	0.064
no_sim3	-0.226	0.011
no_sim4	-0.068	0.256
no_sim5	-0.146	-0.277
no_sim6	0.24	-0.401
no_sim7	-0.036	-0.312
no_sim8	0.408	-0.045
no_sim9	-0.114	0.57
no_sim10	0.025	0.583
Pila_1	-0.1485	-0.214

Fuente: Elaboración propia

Tabla 21 Tabla de resultados de simetría en figuras simétricas

Evalu.	M. Estadístico	
	x	y
sim1	0	0
sim2	1.877E-17	-2.165E-17
sim3	0.061	-0.024
sim4	-0.0006698	0.047
sim5	0.0003814	-0.123
sim6	0	0
sim7	0.098	-0.002
sim8	-0.103	-0.012
sim9	-0.015	0.32
sim10	-2.236E-06	-0.588
sim11	0	0

Fuente: Elaboración propia

Como se puede observar tanto en la tabla 20 y 21, los valores de simetría arrojados por el plugin fueron más correctos que el método descrito en el libro para cada imagen, por lo que se



concluye que, el plugin reconoce bien las imágenes que son simétricas y las que no son simétricas.

5.2.17. Método de puntos equidistante

En esta parte se utilizó el método desarrollado por el Doctor Manuel Forero Vargas; el cual permite saber si un objeto es simétrico o no calculando la distancia que pueda haber entre dos puntos.

Este método consiste en calcular el pixel central (coordenada x, y) de toda la imagen para luego recorrer la coordenada Y hasta encontrar un punto en el cual su valor sea igual a 255 en la parte superior e inferior; una vez obtenido los puntos en la parte inferior y superior, se procede a calcular la diferencia de los valores en la coordenada Y; dando como respuesta la distancia total del objeto en el eje Y manteniendo el eje X constante.

En el caso del eje X, se calcula de la misma forma con la diferencia que la coordenada Y es constante y la distancia total resulta de la suma de las distancias del punto central hacia el lado derecho, así como el lado izquierdo.

Una vez que se ha obtenido las distancias totales en el eje X e Y del objeto, se procede a hacer uso de la división del eje menor con el eje mayor; con lo cual se obtendrá un valor, el cual debe aproximarse a 0 para deducir que el objeto es simétrico.

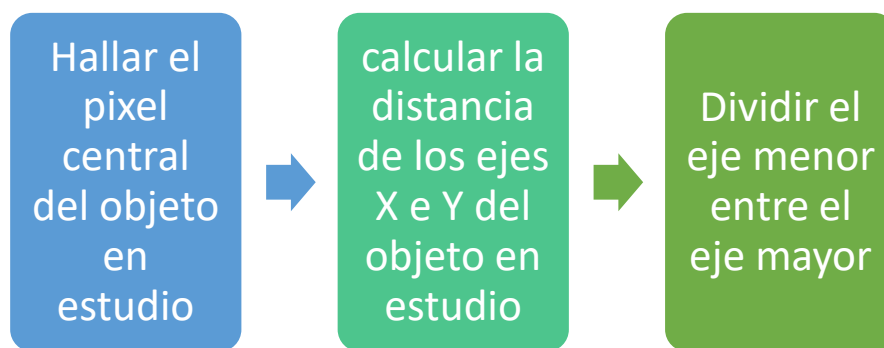


Grafico 6: Pasos del método de puntos equidistante.

Fuente: Elaboracion propia.



5.2.17.1. Hallar pixel central del objeto

Para este punto, se recomienda dirigirse al punto **5.2.13.1** donde se explica a detalle la obtención del pixel central.

5.2.17.2. Calcular la distancia de los ejes X e Y del objeto en estudio

En este punto se calcula la distancia total que se encuentra entre dos puntos del mismo eje; y para ello se aplica la siguiente fórmula matemática:

$$D = p_2 + p_1 \quad (34)$$

Donde:

D= distancia total

p_1 = es la coordenada y o x según del primer punto de donde se inicia el objeto.

p_2 = es la coordenada y o x según del segundo punto de donde se termina el objeto.

5.2.17.3. Dividir el eje menor entre el eje mayor

Como ya se mencionó anteriormente, en este punto se define que tan simétrico es el objeto; para ello se verifica que eje es el que tiene mayor distancia y se procede hacer la división con el eje menor; para lograra ello se utiliza la siguiente fórmula matemática.

$$S = \frac{e_1}{e_2} \quad (35)$$

Donde:

S : es el valor de simetría

e_1 : es la distancia total del eje menor

e_2 : es la distancia total del eje mayor

Donde se cumple la siguiente condición:

$S = 1$: *el objeto es simétrico*

$S > 1$: *El objeto presenta asimetría positiva*

$S < 1$: *el objeto presenta asimetría negativa*

Para comprender mejor el método se tomó como ejemplo un stack de imágenes para aplicar las formulas:



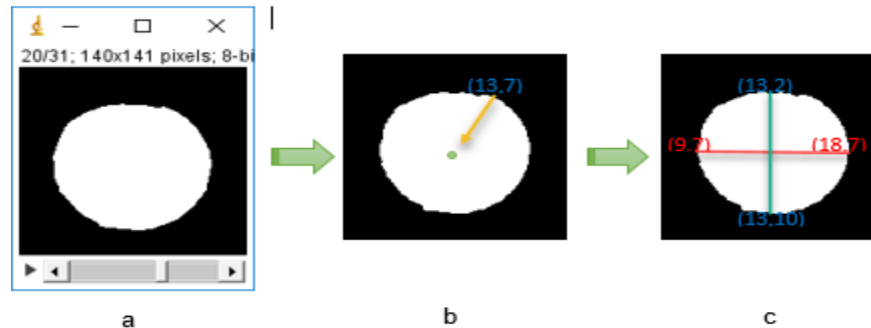


Figura 60: Imagen de ejemplo de la Palta Hass con los puntos equidistantes la punto central del objeto en estudio: a) palta segmentada b) punto central del objeto c) distancias entre puntos de cada eje X e Y.

Fuente: Elaboracion propia

Como se puede apreciar en la **figura 60**, se tiene los puntos p_1 , p_2 de cada eje X e Y, con los cuales al aplicar la fórmula **5.2.17.2** se obtiene las distancias totales en ambos ejes.

- a) Calculando distancia en el eje X
 $D = 18 - 9 = 9$
- b) Calculando distancia en el eje Y
 $D = 10 - 2 = 8$

Una vez que se obtuvo la distancia total, se procedió a verificar cuál de los ejes X e Y tenía la mayor distancia con el fin de aplicar la fórmula **5.2.17.3** para obtener el valor de simetría:

$$S = \frac{Y}{X} = \frac{8}{9} = 0.8$$

Como se puede observar, la imagen de ejemplo mostro que, según la condición especificada en el punto **5.2.17.3**, **se puede concluir que, el objeto presenta un valor de asimetría mínima positiva.**

5.2.18. Deteccion rápida de simetría

En esta sección, se describe el método propuesto por Li & Zhang (2005), para la detección de simetría de objetos en tiempo real; dicho método hace uso de la Transformada de Hough para detectar líneas de simetría a través de la parametrización polar; estas líneas son representadas por un ángulo y una distancia con respecto al centro de la imagen.



5.2.18.1. Transformada de Hough

La transformada de Hough se utiliza para detectar diversas formas, líneas o curvas en imágenes digitales haciendo uso de la transformación del plano cartesiano al espacio de parámetros o espacio de Hough (Canul-arceo, López-martínez, & Narváez-díaz, 2015).

Así mismo el autor (Rojas Z, Teddy V.; Sanz F., Wilmer; Arteaga, 2008) afirma que, en el espacio de Hough se determina la recta característica de una línea de puntos dados.

5.2.18.2. Formulación de coordenadas polares

El autor (Rojas Z, Teddy V.; Sanz F., Wilmer; Arteaga, 2008) menciona que, existe una fórmula de definición de parámetros con la cual se transforma cada punto (x,y) del plano cartesiano al espacio de Hough; dicha fórmula es la siguiente:

$$\rho = x \cos \theta + y \sin \theta \tag{36}$$

Donde:

- x: Representa la coordenada X de un punto en el plano cartesiano.
- y: Representa la coordenada Y de un punto en el plano cartesiano.
- θ: Es el ángulo que se forma con los puntos X e Y

Cuando se obtiene las líneas rectas coincidente que pasan por un punto en específico, se obtiene una curva sigmoïdal tal y como se muestra en la figura 61.

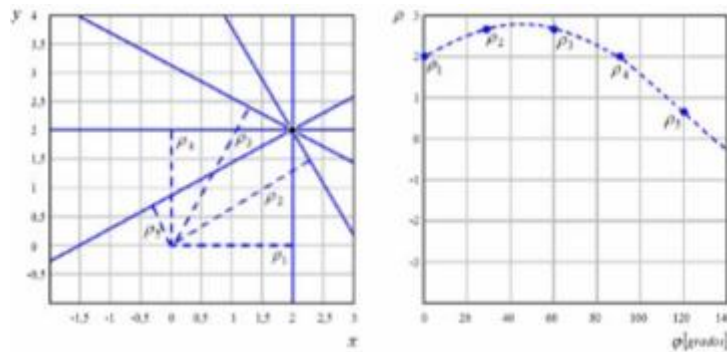


Figura 61: Transformada de Hough de un punto



Fuente: Rojas Z, Teddy V.; Sanz F., Wilmer; Arteaga, (2008) Sistema de visión por computadora para la detección de objetos esféricos a través de la transformada de Hough.

5.2.18.3. Descripción del método

El autor Li, Zhang, & Kleeman(2005), en su investigación para detectar simetría bilateral en tiempo real para aplicaciones roboticas menciona que, el primer paso a realizar es la detección de bordes del objeto en la imagen a través del filtro de Canny, lo que permite aplicar la Tranformada de Hough para la detección de líneas de simetría.

Estas líneas de simetría están representadas por un ángulo y distancia con respecto al centro de la imagen.

Los pixeles del borde son girados alrededor del centro de la imagen por cada ángulo discreto, estos bordes se cuantifican en una matriz 2D Rot; cada fila de esta matriz contiene los pixeles del borde que tienen la misma línea de exploración después de la rotación de la imagen.

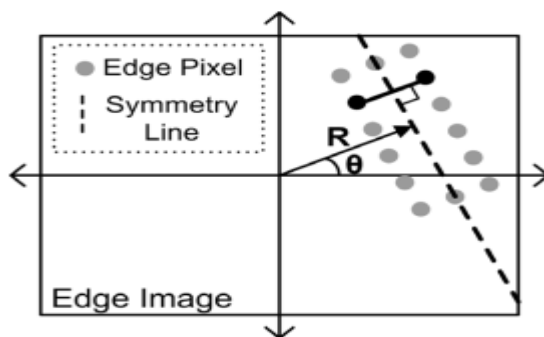


Figura 62: Votación de pixel por línea de simetría con parámetros R y θ .

Fuente:Li et al.(2005), Bilateral Symmetry Detection for Real Time

Robotics Applications

Una vez que se obtuvo los pixeles rotados en la misma línea de exploración, donde cada pixel corresponde a una línea de simetría con un ángulo θ , el algoritmo empareja la votación de cada pixel del borde en la misma línea de exploración, lo que



permite que, todos los votos que se realicen sean para un mismo ángulo, como se muestra en la figura 63.

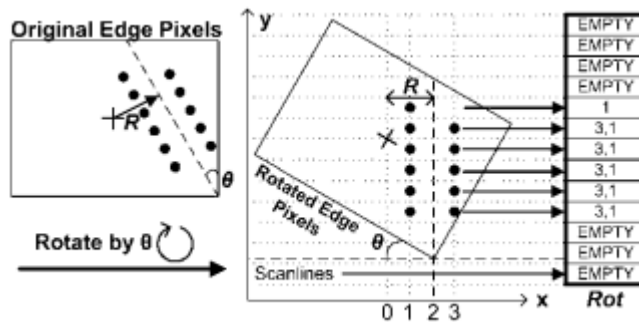


Figura 63: Rotacion de pixeles del borde con respecto al centro de la imagen con un angulo θ , donde las coordenadas horizontales de los pixeles que han sido girados son insertados en la matriz Rot.

Fuente: Li et al. (2005), Bilateral Symmetry Detection for Real Time Robotics Applications.

Después de que se obtuvo la matriz Rot con los pixeles correspondiente y su votación, el algoritmo calcula los picos más altos de la matriz, cuyos valores sean próximos a 0, lo que permite la detección de la línea de simetría; dicho algoritmo se muestra a continuación.



Algorithm 1: Fast Symmetry Detection

Input: I – Source Image
Output: sym – Symmetry Line Parameters (R, θ)
Parameters:
 D_{min} – Minimum distance threshold
 D_{max} – Maximum distance threshold
 H – Hough Accumulator
 $\theta_{lower}, \theta_{upper}$ – Detection Angle Range
 N_{lines} – Number of symmetry lines returned

```

edgePixels  $\leftarrow$  (x,y) locations of edge pixels in  $I$ 
 $H[ ][ ] \leftarrow 0$ 
for  $\theta_{index} \leftarrow \theta_{lower}$  to  $\theta_{upper}$  do
     $\theta \leftarrow \theta_{index}$  in radians
    Rot  $\leftarrow$  Rotate edgePixels by angle  $\theta$ . See Figure 2
    for each row in Rot do
        for each possible pair  $(x_1, x_2)$  in current row do
             $dx \leftarrow |x_2 - x_1|$ 
            if  $dx < D_{min}$  OR  $dx > D_{max}$  then
                continue to next pair
             $x_0 \leftarrow (x_2 + x_1)/2$ 
            Increment  $H[x_0][\theta_{index}]$  by 1
for  $i \leftarrow 1$  to  $N_{lines}$  do
     $sym[i] \leftarrow \max(R_{index}, \theta_{index}) \in H$ 
    Bins around  $sym[i]$  in  $H \leftarrow 0$ 

```

Figura 64: Algoritmo de Deteccion de Simetria

Fuente: Li et al.(2005), Bilateral Symmetry Detection for Real Time Robotics Applications.

5.2.19. Aplicación del método de simetría en las imágenes tomograficas de la palta Hass

1. Método de los momentos estadísticos

Debido a que el plugin Moment Calculator obtuvo mejores resultados, se procedió a aplicarlo a las imágenes de la palta Hass.



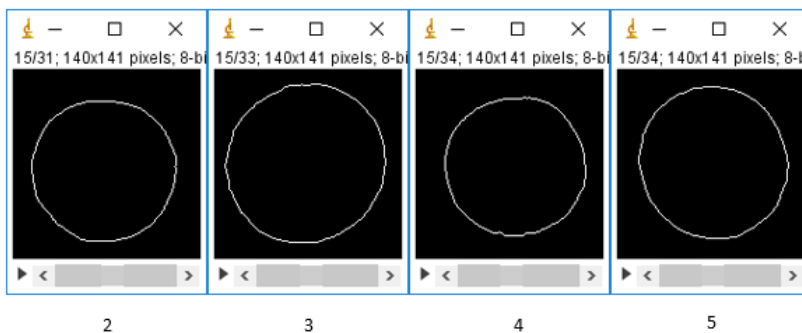


Figura 65: Imágenes de las paltas Hass

Fuente: Elaboración propia

Dando como resultado la siguiente tabla:

Tabla 22 Tabla de resultados de simetría de cada palta evaluada

Evalú.	X	Y
Palta 2	0.0109208	0.0375793
Palta 3	0.01606897	0.025
Palta 4	-0.00761591	-0.0006999
Palta 5	0.02095431	0.0102
Palta 8	-0.00052961	-0.01808553
Palta 12	-0.02103448	-0.00317241
Palta 13	0.00830614	-0.03905821
Palta 14	0.03493103	-0.01113793
Palta 15	0.00843478	-0.01052775
Palta 17	-0.00986957	-0.01304348
Palta 18	0.00328	0.0070091
Palta 19	0.0327931	0.01275862
Promedio	0.0080533	-0.00026485

Fuente: Elaboración propia

En la tabla nro. 21 se puede apreciar que, de las 12 paltas evaluadas, todas son simétricas en ambos ejes; por lo que se puede afirmar que, las paltas son simétricas.

Estos resultados fueron calculados con un margen de error de 0.083%.



2. Método de puntos equidistantes

Para la aplicación de este método, se utilizó imágenes segmentadas de dos canales, es decir, imágenes con valores entre 0 y 255 de la palta del tipo Hass.

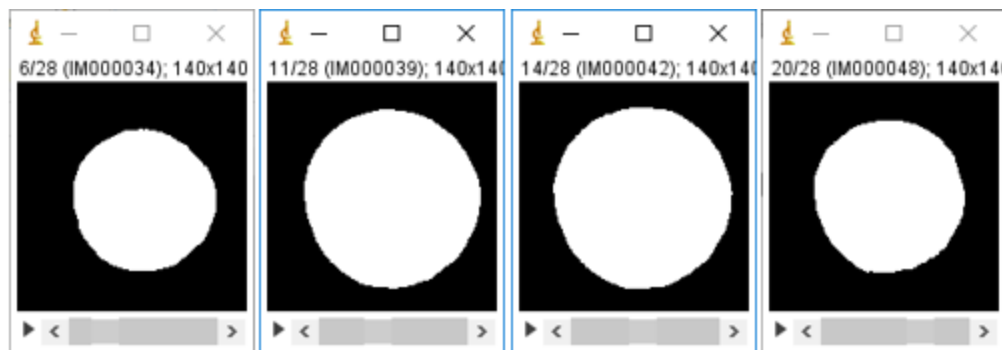


Figura 66: Imagen de la palta Hass segmentada

Fuente: Elaboración propia

Tras la aplicación del método, se obtuvo el siguiente resultado:

Tabla 23 Tabla de resultados de simetría en cada palta evaluada

Nro Palta	Prom.Sim XY	Prom.Sim DD
Palta 2	0.97	0.949
Palta 3	0.979	0.958
Palta 4	0.978	0.957
Palta 5	0.955	0.974
Palta 8	0.943	0.987
Palta 12	0.976	0.956
Palta 13	0.968	0.95
Palta 14	0.961	0.97
Palta 15	0.978	0.985
Palta 17	0.949	0.967
Palta 18	0.976	0.956
Palta 19	0.935	0.945
Promedio	0.964	0.962833333

Fuente: Elaboración propia



En la tabla **nro. 23** se puede apreciar que, de las 12 paltas evaluadas, todas presentaron un valor de simetría muy cercano a 1 tanto en el eje X e Y; por lo que se puede afirmar que, las paltas son simétricas al emplear este segundo metodo.

Estos resultados fueron calculados con un margen de error de 0.8%.

3. Detección de simetría rapida

Para la aplicación de este método, se procedio a separar el stack de imágenes segmentadas en imágenes individuales, esto con motivo de que el algoritmo no procesa varias imágenes al mismo tiempo.

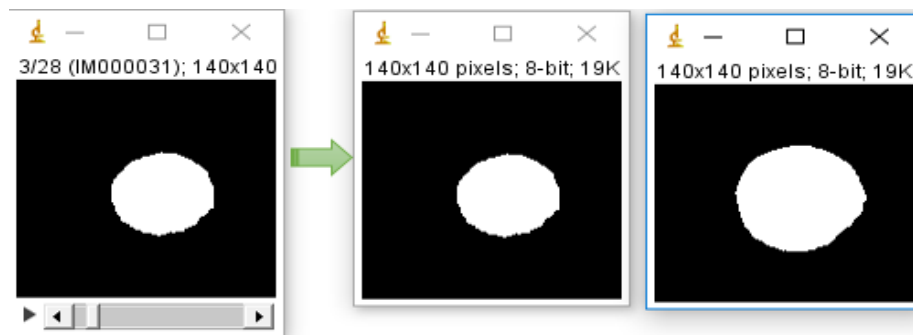


Figura 67: Separacion de stack de imágenes a imágenes individuales

Fuente: Elaboracion propia

Una vez que se obtuvo las imágenes, se procedió a aplicar el algoritmo.

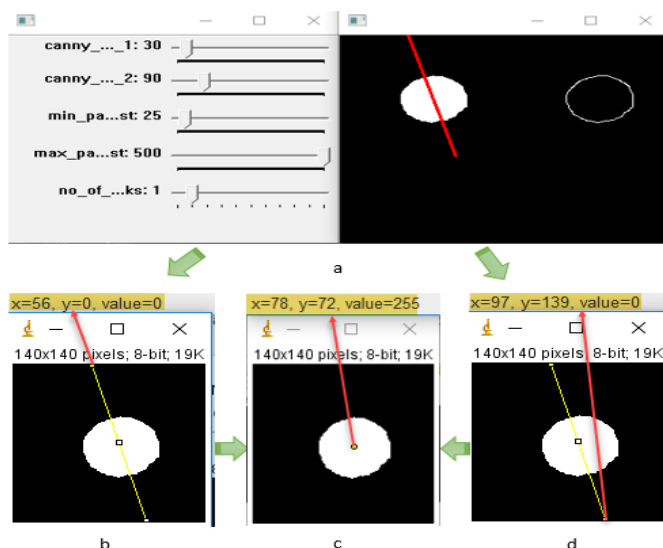


Figura 68: Resultado después de aplicar el algoritmo: a) Representa la línea de simetría arrojado por algoritmo, b) y d) Representan los puntos arrojados por el algoritmo, c) Representa el punto central de la imagen.

Fuente: Elaboración propia

Como se puede observar en la **figura 68**, la línea que dibuja el algoritmo divide en dos partes iguales al objeto de la imagen; dicha línea de simetría pasa por el punto central de la imagen, el cual coincide con el punto central arrojado por el algoritmo de puntos equidistante.

Lo que permite concluir que, al aplicar este método la palta es simétrica.

CAPITULO VI: Conclusiones y Recomendaciones

6.1. Conclusiones

- a) Después de consultar y analizar varias fuentes como es el MINAGRI, Sierra Exportadora, AREX; se determinó que, las características de la palta Hass son: su peso esta entre los 140 a 350 gramos, presenta forma oval con piel rugosa de color verde.
- b) Después de haber construido varios prototipos para la toma de imágenes digitales, se concluyó que, el mejor prototipo presenta las siguientes características: Base de cartón prensado de tamaño 40cm X 40cm con un espesor de 1cm, un plato giratorio de 15cm de diámetro con un espesor de 0.5cm y un punzón en el centro de 2cm de altura, dos parantes de 22cm de alto para el fondo, un brazo deslizante con una altura de 22cm con punzón de 2cm, una riel de 22cm de altura para la distancia y otra riel c de 22cm con 10 niveles separados por 1cm.
- c) Se determinó que, se debería contar con imágenes digitales de 12 paltas rotadas a 10°; es decir construir base de datos con 216 imágenes digitales y con 12 tomografías de paltas Hass cada una con 91 cortes coronal.
- d) El recorte manual en las imágenes tomográficas permitió la identificación de la palta Hass y los fotogramas donde aparece la misma.
- e) El método de Otsu segmentó la pulpa de la palta Hass con manchas en su interior lo que conllevó a aplicar la técnica de relleno de regiones para hacer el limpiado de las manchas, lo que permitió la detección correcta de los bordes con el nuevo método del Ing.Manuel Forero Vargas.
- f) La técnica de PCA permitió hallar los valores y vectores propios de la matriz de co varianza, dando paso al cálculo de los ángulos directores de las componentes X, Y, Z; estos ángulos permitieron la alineación de las componentes X, Y, Z; de manera que las componentes X, Z forman un ángulo aproximado a 90° y la componente Y forma un ángulo aproximado de 180° o 0°



- g) Para la validación del método de simetría se utilizó el código descrito en el libro de Numerical Recipes The art of Scientific Computing Y el plugin Moment Calculator; estos algoritmos fueron aplicados a 11 imágenes sintéticas simétricas y asimétricas dando como resultado que, el plugin Moment Caluclator obtuvo mejores resultados que el código descrito en el libro de Numerical Recipes The art of Scientific Computing en la identificación de imágenes sintéticas simétricas y asimétricas.
- h) El método de los momentos estadístico estimó un valor promedio de 0.0080533 para el eje X y -0.00026485 para el eje Y; lo que representa una buena aproximación a 0.0; el método de puntos equidistante mostro un valor promedio de 0.96 en ambos ejes lo que denota la correcta estimación de la simetría de la palta Hass al aproximarse al valor 1; lo que permite concluir que, la palta Hass es simétrica en conjunto tras la aplicación de ambos métodos.

6.2. Recomendaciones

- a) Implementar le proceso de simetría en otros lenguajes como Matlab o Python para ver los resultados que se obtienen y compararlos con los obtenidos en este trabajo de investigación
- b) Se recomienda comparar los resultados obtenidos con el método planteado para la estimación de la simetría de la palta Hass con otras técnicas existente como la transformada de Haugh, la transformada general simetría entre otros.
- c) Aplicar otro método de detección de borde en la imagen como puede ser el método de la gradiente.



Referencias

- AGRODATAPERU. (2017). Aguacate Paltas Perú Exportación 2017 Julio - Agrodataperu. Retrieved October 22, 2017, from <https://www.agrodataperu.com/2017/08/aguacate-paltas-peru-exportacion-2017-julio.html>
- Aguilar, G. (1995). Procesamiento Digital De Imágenes Utilizando Filtros Morfológicos, 347.
- Burger, W., & Burge, M. J. (2009). *Principles of Digital Image Processing. Principles of Digital Image Processing*. London: Springer. <https://doi.org/10.1007/978-1-84800-195-4>
- Canul-arceo, L., López-martínez, J., & Narváez-díaz, L. (2015). Algoritmo rápido de la transformada de Hough para detección de líneas rectas en una imagen Fast algorithm of the Hough transform for straight line detection in an image, 7, 8–13.
- Chavez Burbano, P. (2015). Implementación de interfaz gráfica para comparación visual de métodos de segmentación y procesamiento de video ..., (July). Retrieved from https://www.researchgate.net/profile/Patricia_Chavez-Burbano/publication/277829411_Implementacion_de_interfaz_grafica_para_comparacion_visual_de_metodos_de_segmentacion_y_procesamiento_de_video_usando_matlab/links/559d432408aeb45d1715b75b/Implementacion-d
- Dolgis, I., Ortega, R., Miguel, A., & Benítez, I. (2015). Técnicas de Segmentación de Imágenes Médicas Técnicas de Segmentación de Imágenes Médicas, (December 2008).
- García Santillán, E. (2016). Aplicación de técnicas de procesado de imagen para la segmentación de núcleos en muestras histológicas humanas, 109. Retrieved from https://riunet.upv.es/bitstream/handle/10251/70432/47446589G_TFG_14733577572447932831980581352993.pdf?sequence=2
- González González, R. A. (2010). Algoritmo basado en Wavelets aplicado a la detección de incendios forestales. Mexico. Retrieved from http://catarina.udlap.mx/u_dl_a/tales/documentos/mel/gonzalez_g_ra/
- Gorgas García, J., Cardiel López, N., & Zamorano Calvo, J. (2011). *Estadística básica para estudiantes de ciencia*.
- Heredia Favieri, N. M. (2015). Reconocimiento de Objetos en Imágenes RGB-D. Argentina: Tesis para Licenciatura en Ciencias de la Computación. Retrieved from <https://www.dc.uba.ar/inv/tesis/licenciatura/2015/heredia.pdf%7B%25%7D5Cnhttps://www.dc.uba.ar/>



- Hernandez Cano, S. E. (2012). *Algebra lineal*. Mexico.
- HERNÁNDEZ CASTORENA, G. A. (2015). ANÁLISIS PARA EL PROCESAMIENTO DE NUBES DE PUNTOS A PARTIR DE PERFILES BIDIMENSIONALES TESIS. Retrieved from <http://eprints.uanl.mx/12984/1/1080237982.pdf>
- Hofer, M. (2008). *Manual Práctico TC: Introducción a la TC* (5ta ed.). Argentina: EDITORIAL MEDICA PANAMERICANA S.A.
- Jimy, M. S., & Cortés, A. (2011). TÉCNICAS ALTERNATIVAS PARA LA CONVERSIÓN DE IMÁGENES A COLOR A ESCALA DE GRISES EN EL TRATAMIENTO DIGITAL DE IMÁGENES Color images Conversion Alternatives Techniques to grayscale in Digital Image Processing, (47), 207–212.
- Jolliffe, I. T. (2002). Principal Component Analysis, Second Edition. *Encyclopedia of Statistics in Behavioral Science*, 30(3), 487. <https://doi.org/10.2307/1270093>
- Julie y Billy. (2007). *Probabilidad y estadística para ingeniería y ciencias*.
- La Serna Palomino, N., & Román Concha, U. (2009). Técnicas de Segmentación en Procesamiento Digital de Imágenes. *Revista de Ingeniería de Sistemas E Informática*, 6(2), 9–16. [https://doi.org/10.1016/S0186-1042\(14\)70160-3](https://doi.org/10.1016/S0186-1042(14)70160-3)
- Li, W. H., & Zhang, A. M. (2005). Detección bilateral Simetría por Tiempo real Las aplicaciones de robótica, 1–26.
- Li, W. H., Zhang, A. M., & Kleeman, L. (2005). Bilateral Symmetry Detection for Real Time Robotics Applications, 1–26.
- Liu, Y. (2008). Computational Symmetry in Computer Vision and Computer Graphics. *Foundations and Trends® in Computer Graphics and Vision*, 5(1–2), 1–195. <https://doi.org/10.1561/06000000008>
- MINAGRI-DGPA. (2015). *La Palta “Producto Estrella de Exportación.”*
- Nariño, U. A. (2016). Técnicas de umbralización para el procesamiento digital de imágenes de GEM- Foils Thresholding techniques for digital image processing of GEM-Foils, 21(4), 352–359.
- Passariello, G., & Mora, F. (1995). *Las imágenes medicas* (Universida). Venezuela.
- Peña, D. (2002). Análisis de datos multivariantes. *Book*, (December), 515. <https://doi.org/8448136101>
- Plaza Miguel, A. (2002). Proposición, Validación y Prueba de una Metodología Morfológica para el Análisis de Datos Hiperespectrales que Integra Información Espacial y Espectral, 260.



- Preess, W. H., Teukolsky, S. A., Vetterking, W. T., & P., F. B. (2007). *Numerical Recipes The art of Scientific Computing*. (C. U. Press, Ed.) (Third). Estados Unidos.
- Rastrepo V., A. (1998). Procesamiento de Imágenes medicas.
- Rojas Z, Teddy V.; Sanz F., Wilmer; Arteaga, F. (2008). Sistema de visión por computadora para la detección de objetos esféricos a través de la transformada de Hough Vision system by computer for the detection of spherical objects using the circular Hough transform.
- Ruíz Muñoz, D. (2004). *Manual de Estadística*. Retrieved from <http://www.eumed.net/cursecon/libreria/drm/drm-estad.pdf>
- Santo, R. do E. (2012). Principal Component Analysis applied to digital image compression, *10*(55 11), 135–139. <https://doi.org/10.1590/S1679-45082012000200004>
- Trujillo, J. P., Rivera, J. H., & Serna, W. (2010). Descripción Del Estándar Dicom Para Un Acceso Confiable a La Información De Las Imágenes Médicas . *Scientia et Technica*, *45*(0122–1701), 289–294. <https://doi.org/10.22517/23447214.347>
- Vallejo Ayala, P., & Zambrano, J. (2010). *Física Vectorial 1* (Rodin). Ecuador.
- Vieyra, M. V. (2013). Análisis de componentes principales en imágenes de teledetección .



Anexo I

Peso de la pulpa, pepa y cascara de la palta Hass

N° de palta	peso normal	peso de pulpa	peso de pepa	peso de cascara
P1	168.1777	94.5624	34.2122	38.6302
P2	177	120.278	24.8445	30.147
P3	183.2259	124.0521	26.4469	31.5519
P4	167.8004	108.18	28.1736	30.4166
P5	148.7894	95.8453	27.3969	23.125
P6	200.202	113.4808	43.4747	42.1069
P7	167.7318	104.8531	28.1331	34.2192
P8	220.5748	157.605	36.346	26.5352
P9	181.6351	104.0217	37.1722	40.3002
P10	164.779	88.5512	28.7577	45.8135
P11	162.57	105.8345	21.3224	34.4206
P12	151.7122	103.8577	27.192	19.062
P13	214.6933	130.4311	36.0022	47.6624
P14	196.7346	133.0722	31.36	30.9558
P15	164.5684	100.324	26.7697	36.8218
P16	181.4846	118.1109	23.6563	39.1601
P17	155.8017	83.6592	26.7006	44.637
P18	160.1475	97.1102	25.4946	37.3208
P19	171	109.376	28.1728	32.2856
Promedio	175.7162526	110.1687053	29.55938947	35.00904211
Desv.Estandar	19.54828891	17.12154746	5.370191714	7.38324601
LimMaximo	195.2645415	127.2902527	34.92958119	42.39228812
LimMinimo	156.1679637	93.0471578	24.18919776	27.62579609

Calculando la muestra	
Población	19
Nivel. Confianza	95%
error	5%
Muestra	19



Anexo II

Calculo de la media y los valores de la varianza de los ejes X, Y, Z del ejemplo del punto 5.2.10

Para agilizar estos cálculos, se empleó la herramienta Microsoft Excel 2016

X	Y	Z	$x-\bar{X}$	$y-\bar{y}$	$z-\bar{z}$	$(xi-\mu)^2$	$(yi-\mu)^2$	$(zi-\mu)^2$
1	1	1	-0.6	-0.5	-1	0.36	0.25	1
1	2	1	-0.6	0.5	-1	0.36	0.25	1
2	1	1	0.4	-0.5	-1	0.16	0.25	1
2	2	1	0.4	0.5	-1	0.16	0.25	1
1	1	2	-0.6	-0.5	0	0.36	0.25	0
2	1	2	0.4	-0.5	0	0.16	0.25	0
2	2	2	0.4	0.5	0	0.16	0.25	0
1	1	3	-0.6	-0.5	1	0.36	0.25	1
2	2	3	0.4	0.5	1	0.16	0.25	1
2	2	4	0.4	0.5	2	0.16	0.25	4

	X	Y	Z
Media	1.6	1.5	2
Varianza	0.24	0.25	1



Anexo III

Tiempo empleado por cada algoritmo en cada stack de imágenes de cada palta

Acción	Pca(s)	Ángulos directores(s)	Detección de borde(s)
Palta 2	0.042	0.001	1.674
Palta 3	0.016	0.001	1.229
Palta 4	0.027	0.002	1.741
Palta 5	0.016	0.002	1.056
Palta 8	0.031	0.001	1.087
Palta 12	0.036	0.002	1.66
Palta 13	0.038	0.001	1.752
Palta 14	0.011	0.001	1.237
Palta 15	0.033	0.002	0.883
Palta 17	0.031	0.001	0.997
Palta 18	0.04	0.001	1.186
Palta 19	0.04	0.001	1.201



Anexo IV

Cuadro de rotaciones para el alineamiento del stack de la imagen

Palta Nro. Rotación	2	3	8	13	17	4	5	12	14	15	18	19
1	-1.14	1.29	1.57	-2.15	-0.72	1.53	1.43	1.52	1.75	0.73	1.52	0.87
2	23.35	23.98	-82.3	22.35	-55.13	27.86	-67.25	26.52	-62.78	-57.06	26.56	-44.4
3	21.48	20.42	28.14	15.47	5.77	14.72	-32.37	21.11	-30.8	-67.83	21.06	-77.26
4	20.34	14.85	-32.97	22.02	10.96	7.8	10.44	15.86	12.05	2.31	14.48	24.72
5	14.83	15.67	19.18	23.17	22.13	13.65	12.97	14.53	15.4	4.86	17.07	-49.94
6	15.23	14.92	25.78	23.3	-44.93	12.58	9.47	13.7	8.35	10.74	16.17	
7	9.35	11.22	28.36	10.99		12.92	13.66	10.71	3.98	21.94	9.58	
8	8.81	5.73	27.39	5.57		10.44	12.28	5.04		-43.02	8.11	
9		4.21	18.02	5.21		7.75	11.83	3.68			5.44	
10		4.04	8.88	2.71		7.81	23.69				6.13	
11		1.46	3.9	3.31		2.06	23.46				3.45	
12		2.24	4.02			3.67	12.43				3.55	
13			1.87			2.42	7.64					
14			1.11			2.8	5.33					
15							1.65					
16							1.44					

Nota: El primer registro de color amarillo hace referencia al valor del ángulo que se ha utilizado para rotar el eje z, todas las demás filas pertenecen al valor del ángulo utilizado para hacer la rotación en el eje X

valores finales para los ángulos de Alineación de los ejes x, y, z

	Palta 2	Palta 3	Palta 8	Palta 13	Palta 17	Palta 4	Palta 5	Palta 12	Palta 14	Palta 15	Palta 18	Palta 19
x	90.63	90.76	90.25	90.35	90.34	90.43	90.71	90.47	90.669	90	90.81	90.61
y	179.35	179.16	179.67	0.4	179.62	179.55	0.71	179.52	0.92	0.68	0.82	0.61
z	89.9	89.67	89.79	90.24	89.85	89.87	90.04	89.99	89.38	89.9	90.16	89.96



Anexo V

Imagen de referencia para hallar el borde una imagen

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0										Orientación de la comprobación de pixeles vecinos que no son borde Continuación de la comprobación de pixeles vecino que si son bordes						
1								No es borde	No es borde							
2		No es borde	No es borde	$D=(1+2)\text{mod}8$ $D=3$	$D=0$	$D=(0+2)\text{mod}8$ $D=2$	No es borde	No es borde	No es borde							
3		No es borde	$D=(2+2)\text{mod}8$ $D=4$	$D=(1+2)\text{mod}8$ $D=3$		$D=(6+2)\text{mod}8$ $D=0$	$D=(7+2)\text{mod}8$ $D=1$	No es borde	No es borde	No es borde	No es borde	No es borde	No es borde	No es borde	No es borde	No es borde
4	No es borde	No es borde	$D=(1+2)\text{mod}8$ $D=3$					$D=(7+2)\text{mod}8$ $D=1$	$D=(6+2)\text{mod}8$ $D=0$	No es borde	No es borde	$D=(1+2)\text{mod}8$ $D=3$	$D=(0+2)\text{mod}8$ $D=2$	$D=(0+2)\text{mod}8$ $D=2$	No es borde	No es borde
5	No es borde	$D=(3+2)\text{mod}8$ $D=5$						$D=(6+2)\text{mod}8$ $D=0$	$D=(3+2)\text{mod}8$ $D=5$	No es borde	$D=(1+2)\text{mod}8$ $D=3$	$D=(4+2)\text{mod}8$ $D=6$	$D=(6+2)\text{mod}8$ $D=0$	No es borde	No es borde	No es borde
6	No es borde	No es borde	$D=(4+2)\text{mod}8$ $D=6$	$D=(4+2)\text{mod}8$ $D=6$	$D=(3+2)\text{mod}8$ $D=5$		$D=(5+2)\text{mod}8$ $D=7$	No es borde	$D=(7+2)\text{mod}8$ $D=1$	$D=(3+2)\text{mod}8$ $D=5$	$D=(5+2)\text{mod}8$ $D=7$	No es borde	No es borde	No es borde	No es borde	No es borde
7		No es borde	No es borde	No es borde	No es borde	No es borde	$D=(5+2)\text{mod}8$ $D=7$	No es borde	No es borde	No es borde	$D=(5+2)\text{mod}8$ $D=7$	No es borde	No es borde	No es borde	No es borde	No es borde
8						No es borde	No es borde	No es borde	No es borde	No es borde	No es borde	No es borde	No es borde	No es borde	No es borde	No es borde
9																
10																
11																



Anexo VI

Integración de ImageJ en el IDE Netbeans

a. Descarga del código fuente de ImageJ

ImageJ es un programa de código abierto dedicado al procesamiento de imágenes, por lo que el código fuente está disponible; por ello se descargó la versión ij151 de imageJ desde su página oficial

b. Creación de un nuevo proyecto de tipo Java Free-From Project

Desde el IDE Netbeans 8.2, se creó un proyecto de tipo Java Free-Form Project debido a que es una plantilla para ejecutar proyectos de forma libre permitiendo la gestión del código fuente para ejecutar comandos Ant; se trabajó con la versión de IDE netbeans 8.2.

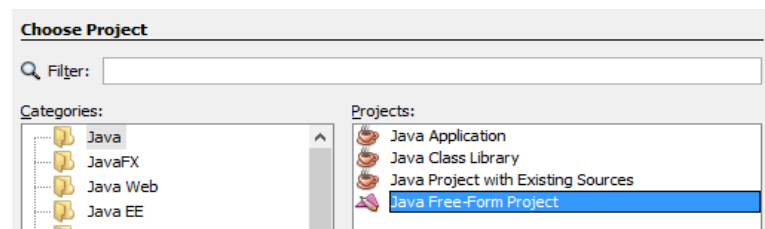


Figura 69: Selección de tipo de proyecto a crear.

Fuente: Elaboración propia

c. Localizando la carpeta source y el Archivo build xml de ImageJ

Después de descargar y descomprimir el código fuente de imageJ, se ubica la carpeta **source**, la cual contiene el código fuente de ImageJ y dentro de ella se encuentra el archivo build.xml y desde netbeans seleccionamos la carpeta source y el archivo build xml



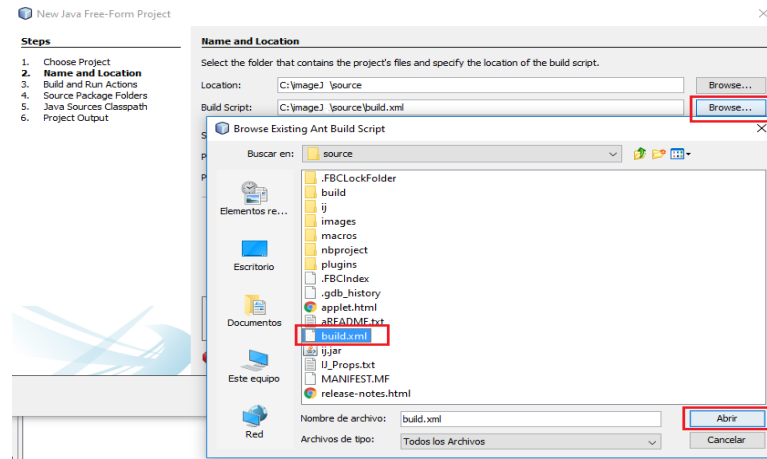


Figura 70: Ubicación del archivo fuente XML.

Fuente: Elaboración propia

d. Agregando la carpeta ij, plugin y seleccionando el JDK para la configuración del proyecto

La carpeta ij contiene la interfaz gráfica de ImageJ y todos los controles necesarios para el procesamiento de imágenes y la carpeta plugin contiene los plugins que creamos; ambas carpetas se encuentran dentro de la carpeta de source.



En el caso del JDK utilizo la versión 1.8.

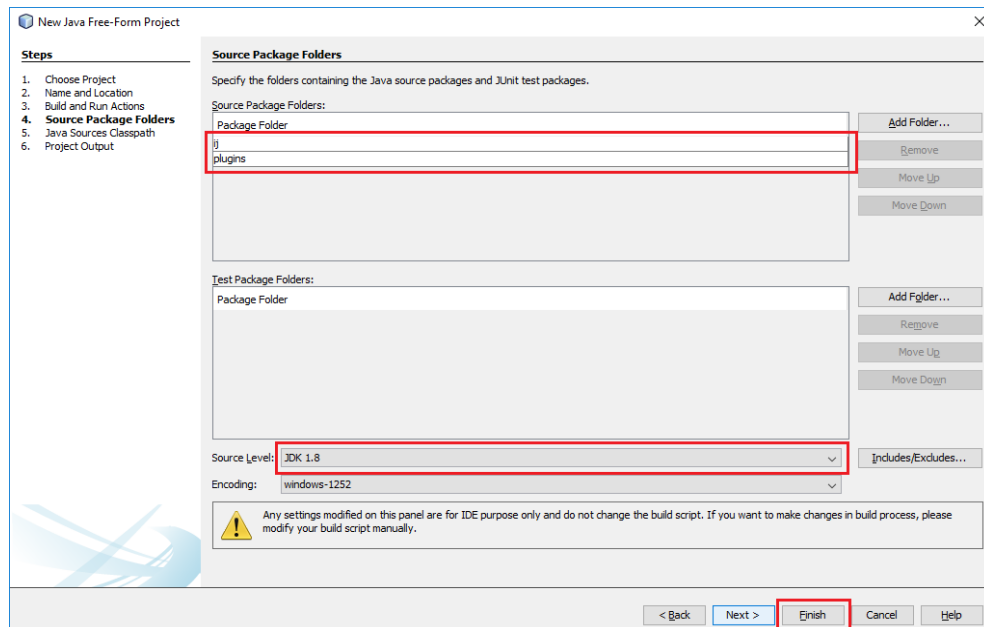
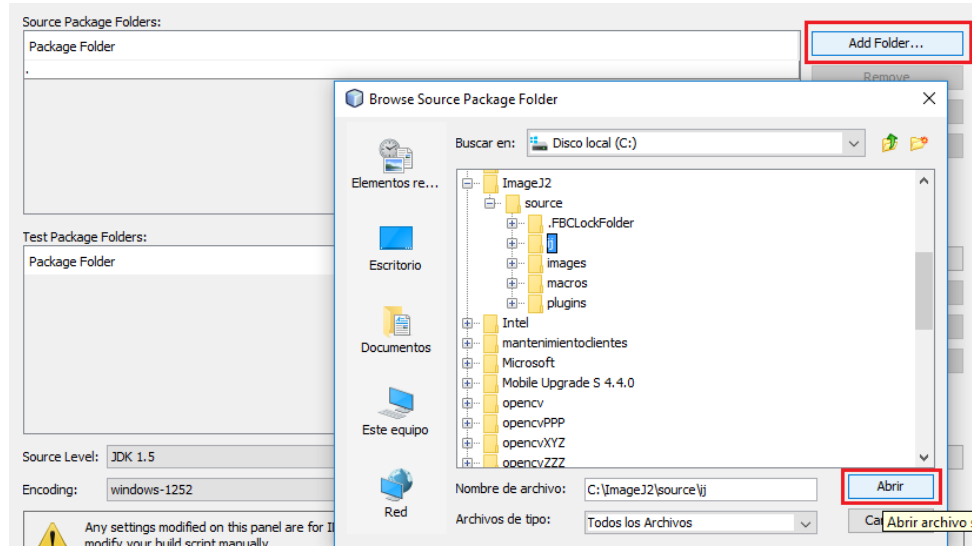


Figura 71: Selección de archivos para los plugins e interface grafica de ImageJ.



Fuente: Elaboración propia

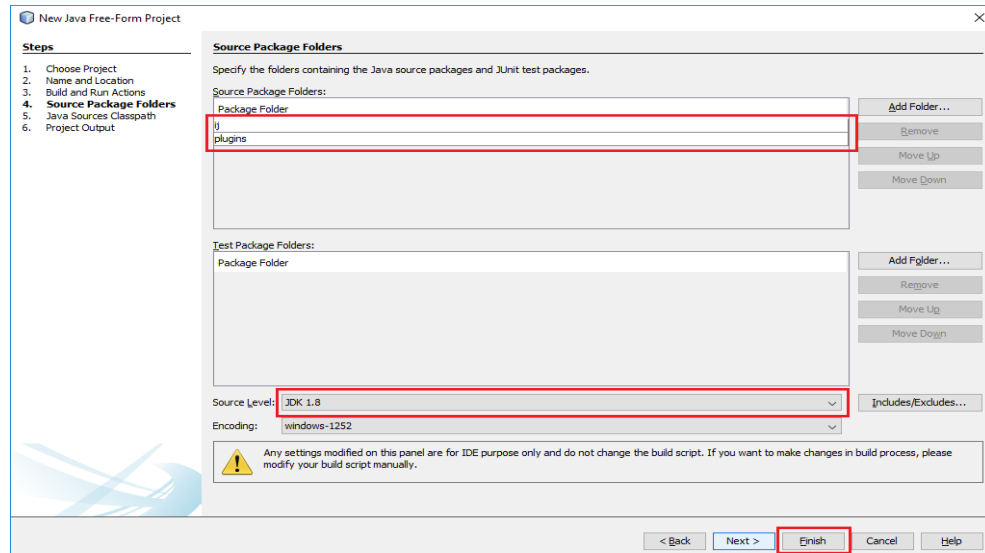


Figura 72: Selección de JDK para el proyecto.

Fuente: Elaboración propia

e. Compilando ImageJ con Netbeans

Al momento de finalizar la configuración del proyecto, por defecto se nos abrirá en el IDE de netbeans el archivo build xml, en el cual modificaremos algunos parámetros para la correcta compilación del proyecto:

- 1) Cambiar el valor de source 1.5 por la versión del jdk que tenemos instalado; en este caso el 1.8
- 2) Borrar la propiedad target
- 3) Borrar las líneas 22 y 23.



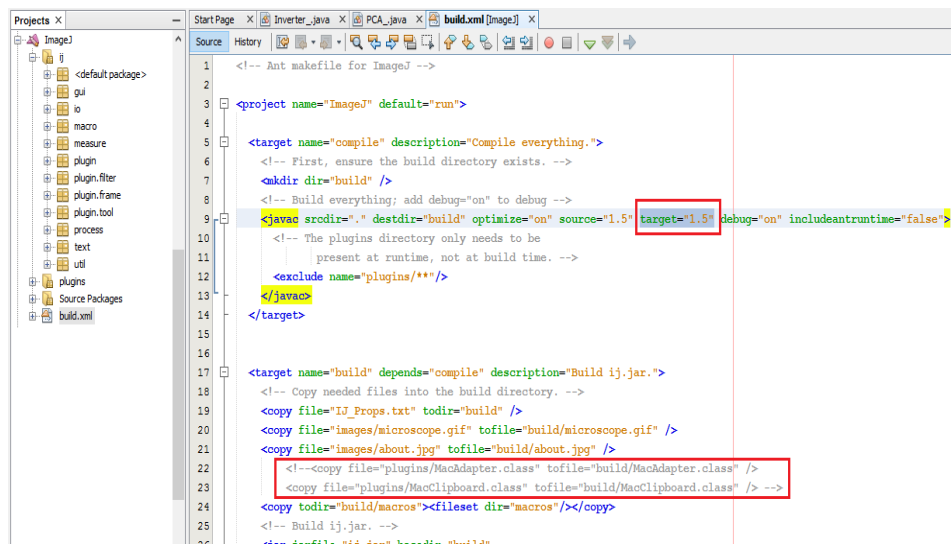


Figura 73: Configuración de archivo XML.

Fuente: Elaboración propia

Una vez hecho los cambios, se guarda las modificaciones realizadas y se procede a hacer la construcción del proyecto y para ello hacemos clic en **Run/Build Project(ImageJ)**

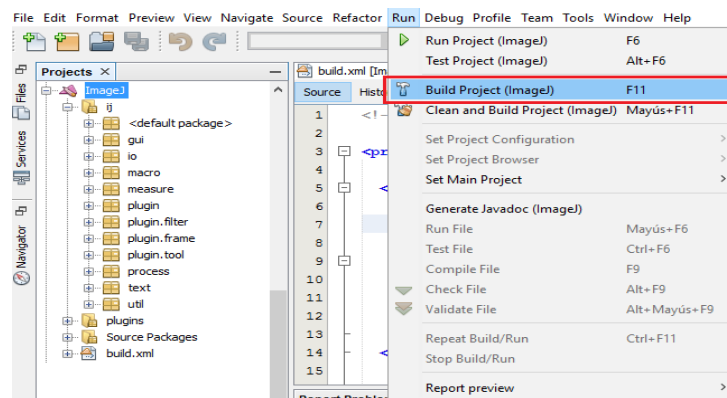


Figura 74: Compilación del proyecto.

Fuente: Elaboración propia

Al ejecutar el paso anterior se nos mostrara un mensaje de compilación en la consola de netbeans mostrando algunas alertas de color rojo informando de algunas opciones o configuraciones que ya no se utilizan



f. Configurando la ejecución del proyecto

En esta parte se ingresa a las propiedades del proyecto donde agregaremos los jar que necesita ImageJ para su ejecución, para ello hacemos click sobre la opción file y seleccionamos del menú la opción Project properties(ImageJ) y en la ventana que se nos muestra seleccionaremos la opción Java Source Classpath.

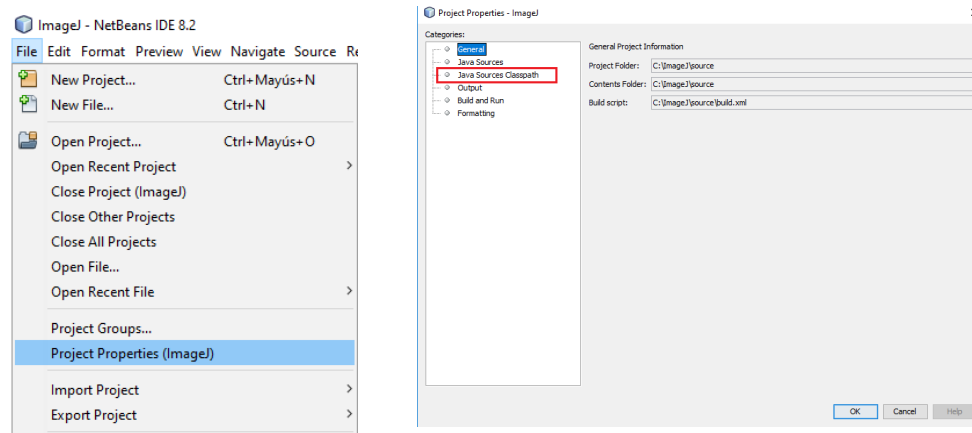


Figura 75: Selección de Classpath del proyecto.

Fuente: Elaboración propia

Una vez seleccionado opción Java Source Classpath, agregaremos el jar para la clase ij y para los plugins; el jar se encuentra en la carpeta source de ImageJ

Una vez agregado el plugin hacemos click en finalizar.



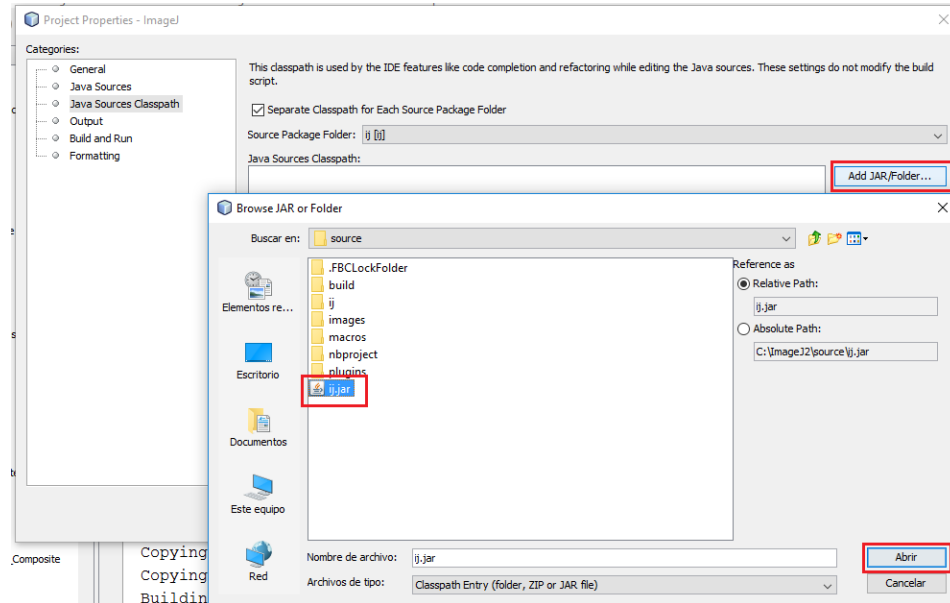


Figura 76: Agregando Jar para los plugin.

Fuente: Elaboración propia

g. Ejecutar el proyecto

Para este paso seleccionaremos el proyecto y haremos click sobre la opción Run de la barra de menú de netbeans y seguidamente run Project; si todo está conforme se nos mostrará la interfaz gráfica de imageJ.

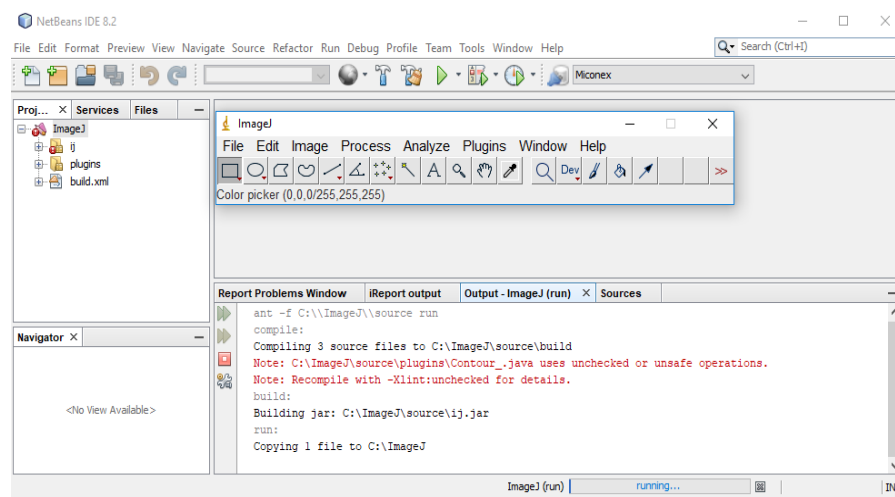


Figura 77: Ejecución del programa.

Fuente: Elaboración propia



Anexo VII

Creando y compilando plugin en ImageJ

a) Creación de la clase para el plugin

Las clases que se crean para los plugin son clases normales de JAVA, las cuales deben implementar la clase Plugin de ImageJ; para ello hacemos click derecho sobre la carpeta plugin, luego seleccionamos la opción New java class y le asignamos un nombre y finalmente click en finish.

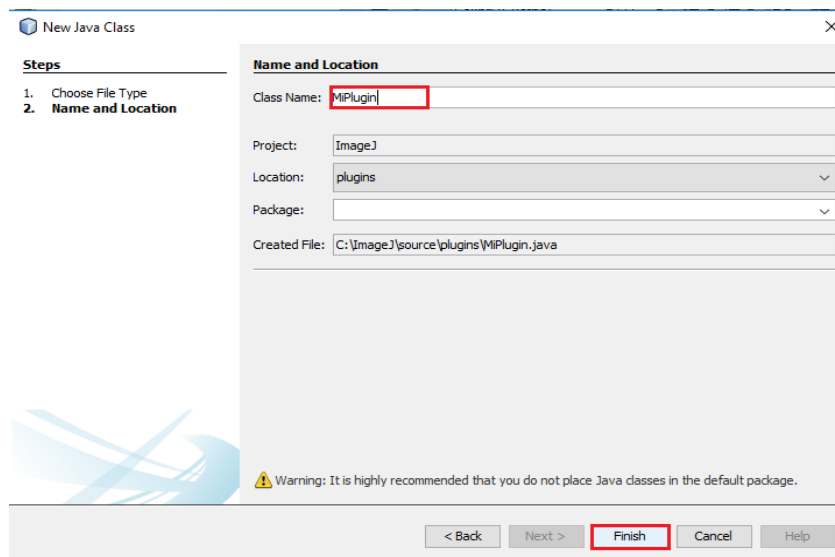


Figura 78: Creación de clase a crear.

Fuente: Elaboración propia



b) Escribir el código del plugin dentro de la ventana del editor de netbeans

```
import ij.*;
import ij.process.*;
import ij.gui.*;
import java.awt.*;
import ij.plugin.*;
import ij.plugin.frame.*;

Public class Mi_Plugin implements Plugin {

    Public void run (String arg) {

        ImagePlus imp = IJ.getImage ();

        IJ.run (imp, "Invert", "");

        IJ.wait (1000);

        IJ.run (imp, "Invert", "");

    }
}
```

Figura 79: Código a implementar para nuevo plugin.

Fuente: Elaboración propia

c) Compilando el código escrito en el editor de netbeans

Para compilar el plugin, hacemos click derecho sobre la clase creada y seleccionamos la opción Compile File, se nos mostrara na ventana donde haremos click en generate; al hacer esto se genera un archivo xml donde medicaremos la línea 9 para establecer la carpeta de destino del plugin y la versión del jdk de java en el parámetro source.



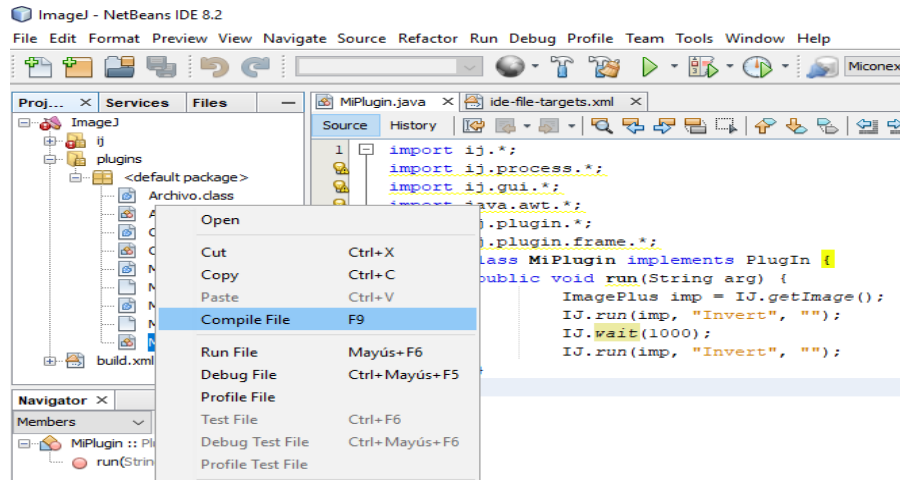


Figura 80: Compilación de plugin.

Fuente: Elaboración propia

d) Eliminar la línea 8 para que no cree una nueva carpeta

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <project basedir=".." name="ImageJ-IDE">
3 <!-- TODO: edit the following target according to your needs -->
4 <!-- (more info: http://www.netbeans.org/kb/articles/freeform-config.html#compilesingle)
5 <target name="compile-selected-files-in-plugins">
6 <fail unless="files">Must set property 'files'</fail>
7 <!-- TODO decide on and define some value for ${build.classes.dir} -->
8 <!-- <mkdir dir="${build.classes.dir}"/> -->
9 <javac destdir="plugins" includes="${files}" source="1.8" srcdir="plugins">
10 <classpath path="ij.jar"/>
11 </javac>
12 </target>
13 </project>
    
```

Figura 81: Modificación de línea para no crear una nueva carpeta al compilar el plugin.

Fuente: Elaboración propia



e) Ejecutar el proyecto

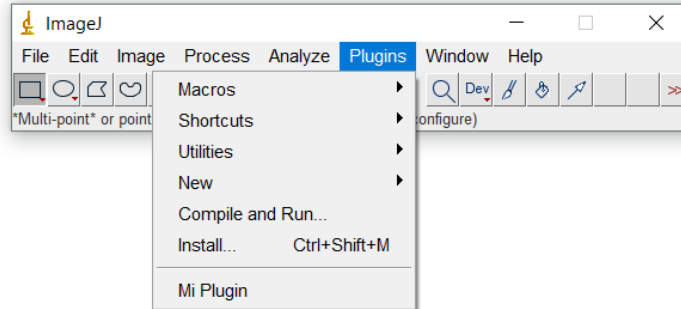


Figura 82: Verificación de plugin creado en ImageJ.

Fuente: Elaboración propia



Anexo VIII

Implementación de la técnica PCA y cálculo de los ángulos directores

```
import ij.*;
import ij.process.*;
import ij.plugin.filter.*;
import java.text.SimpleDateFormat;
import java.util.Date;

public class PCA_binaryImage implements PlugInFilter {

    private ImagePlus sImp, rImp; //ImagePlus es la representación de una imagen en
        ImageJ, que se basa en el ImageProcessor
    private ImageStack sStack, rStack; //Un ImageStack es una matriz ampliable de
        imágenes

    @Override
    public int setup(String args, ImagePlus imp) {
        if (imp.getStackSize() == 1) {
            return DONE;
        }
        //Se debe comprobar que la imagen sea binaria - pendiente
        if (args.equals("about")) {
            showAbout();
        }
        this.sImp = imp; //pasa la imagen al atributo de la clase imagen de origen.
        return DOES_ALL + NO_UNDO + NO_CHANGES; // Todas las imágenes, No
            deshacer la acción, No acepta cambios.
    }

    /**
     * Método para aplicar PCA a una pila de imágenes.
     *
     * @param ip una referencia de tipo ImageProcessor a objetos procesadores de
     * imagen para ciertos tipos de imágenes.
     */
    public String getHoraActual() {
        Date hoy = new Date();
        SimpleDateFormat formateador = new SimpleDateFormat("hh:mm:ss:SSS");
        return formateador.format(hoy);
    }
}
```



```

@Override
public void run(ImageProcessor ip) {
    // System.out.println("hora inicio PCA:"+getHoraActual());
    //ImageProcessor es una clase abstracta padre de los procesadores de imagen
    // para ciertos tipos de imágenes.
    //un ImageProcessor proporciona métodos para trabajar realmente en los datos de
    // imagen.
    sStack = slmp.getStack(); //Lee el stak de origen y las pasa la variable
    rStack = slmp.createEmptyStack();

    //Calcula PCA
    PCA(sStack, rStack);

    //Genera una imagen de destino con la pila resultante de la aplicación del PCA
    //rImp = new ImagePlus("PCA " + " (" + slmp.getTitle() + ")", rStack);
    //Muestra en una ventana la imagen resultante.
    //rImp.show();
}

```

```

public void PCA(ImageStack sStack, ImageStack rStack) {
    double[][] mCov = calcularMatrizCovarianza(sStack);
    double[] valPropios = new double[3];
    double[][] mVectoresPropios = valoresPropios(valPropios, mCov);
    System.out.println("Matriz de convarianza");
    printMatrix(mCov);
    System.out.println("valores propios");
    printVector(valPropios);
    System.out.println("Matriz de vectores propios");
    printMatrix(mVectoresPropios);
    // System.out.println("hora FIN PCA :"+getHoraActual());
    System.out.println("Angulos");
    // System.out.println("hora inicio Angulos:"+getHoraActual());
    double[] v1 = extraeVector(mVectoresPropios, 0);
    printVector(angulosVector(v1));
    double[] v2 = extraeVector(mVectoresPropios, 1);
    printVector(angulosVector(v2));
    double[] v3 = extraeVector(mVectoresPropios, 2);
    printVector(angulosVector(v3));
    // System.out.println("hora FIN Angulos:"+getHoraActual());
}

```

```

private double[] extraeVector(double[][] matrix, int col) {

```



```

double[] res = new double[matrix.length];
for (int i = 0; i < matrix.length; i++) {
    res[i] = matrix[col][i];
}
return res;
}

public double[][] calcularMatrizCovarianza(ImageStack sStack) {

    // Calcular los promedio de las coordenadas en X, Y y Z de los píxeles de la región
    // de interés
    int nSlides = sStack.getSize();//Cantidad de imagenes del stack
    int width = sStack.getWidth(); //Ancho de la imagen del stack
    int height = sStack.getHeight(); //Alto de la imagen del stack
    int size = width * height; //Tamaño del vector de pixeles de cada imagen (ancho *
        // alto)
    double sumX = 0, sumY = 0, sumZ = 0;
    int contPoints = 0; //Contador de pixeles de la región de interes de cada imagen

    //----Recorre el stack de imagenes
    for (int z = 0; z < nSlides; z++) {
        //Extrae los pixeles de cada imagen y lo almacena en un Vector
        byte[] pixeles = (byte[]) sStack.getProcessor(z + 1).getPixels();
        //Recorre el vector de píxeles
        for (int i = 0; i < size; i++) {
            //Evalua si el pixel pertenece a la región de interés.
            //Siendo que la imagen es binarizada los píxeles de la región de interés tienen
            //valor=255 (Blanco)
            //y los píxeles del fondo tienen valor=0 (negro)
            if (pixeles[i] != 0) {
                sumX += (i % width); //(i%width) calcula el valor de la coordenada X del
                // pixel
                sumY += (i / width); //(i/width) calcula el valor de la coordenada Y del pixel
                sumZ += z; //El numero de Slides(imagenes del stack) representa el valor
                // de la coordenada Z del pixel
                contPoints++; //Cuenta la cantidad de píxeles que correspondian a la región
                // de interes
            }
        }
    }
    sumX = sumX / contPoints; //Promedio de las coordenadas X de todos los píxeles
    // de la región de interes
    sumY = sumY / contPoints; //Promedio de las coordenadas Y de todos los píxeles de
    // la región de interes
}

```



```

sumZ = sumZ / contPoints; //Promedio de las coodenadas Z de todos los píxeles de
    la región de interes

//Calcular la matriz de covarianza
double corXX = 0, corYY = 0, corZZ = 0, corXY = 0, corXZ = 0, corYZ = 0;
double[][] mCovar = new double[3][3];

for (int z = 0; z < nSlides; z++) {
    byte[] pixeles = (byte[]) sStack.getProcessor(z + 1).getPixels();
    for (int i = 0; i < size; i++) {
        if (pixeles[i] != 0) {
            double x = i % width;//Coordenada X
            double y = i / width;//coordenadaY

            corXX += (x - sumX) * (x - sumX);
            corYY += (y - sumY) * (y - sumY);
            corZZ += (z - sumZ) * (z - sumZ);
            corXY += (x - sumX) * (y - sumY);
            corXZ += (x - sumX) * (z - sumZ);
            corYZ += (y - sumY) * (z - sumZ);
        }
    }
}

//Diaganonal
mCovar[0][0] = corXX / (contPoints - 1);
mCovar[1][1] = corYY / (contPoints - 1);
mCovar[2][2] = corZZ / (contPoints - 1);
//directa
mCovar[1][0] = corXY / (contPoints - 1);
mCovar[2][0] = corXZ / (contPoints - 1);
mCovar[2][1] = corYZ / (contPoints - 1);
//Inversa
mCovar[0][1] = corXY / (contPoints - 1);
mCovar[0][2] = corXZ / (contPoints - 1);
mCovar[1][2] = corYZ / (contPoints - 1);

return mCovar;
}

public double[][] valoresPropios(double[] valores, double[][] mCov) {
    int n = valores.length;
    final double CERO = 1e-8;
    double maximo, tolerancia, sumsq;

```




```
double x, y, z, c, s;  
int contador = 0;  
int i, j, k, l;  
double[][] a = copyMatrix(mCov);//matriz copia
```

```
double[][] p = new double[n][n];  
double[][] q = new double[n][n];
```

```
//matriz unidad  
for (i = 0; i < n; i++) {  
    q[i][i] = 1.0;  
}  
do {  
    k = 0;  
    l = 1;  
    maximo = Math.abs(a[k][l]);  
    for (i = 0; i < n - 1; i++) {  
        for (j = i + 1; j < n; j++) {  
            if (Math.abs(a[i][j]) > maximo) {  
                k = i;  
                l = j;  
                maximo = Math.abs(a[i][j]);  
            }  
        }  
    }  
    sumsq = 0.0;  
    for (i = 0; i < n; i++) {  
        sumsq += a[i][i] * a[i][i];  
    }  
  
    tolerancia = 0.0001 * Math.sqrt(sumsq) / n;  
    if (maximo < tolerancia) {  
        break;  
    }  
    //calcula la matriz ortogonal de p  
    //inicialmente es la matriz unidad  
    for (i = 0; i < n; i++) {  
        for (j = 0; j < n; j++) {  
            p[i][j] = 0.0;  
        }  
    }  
    for (i = 0; i < n; i++) {  
        p[i][i] = 1.0;  
    }  
}
```



```

y = a[k][k] - a[l][l];
if (Math.abs(y) < CERO) {
    c = s = Math.sin(Math.PI / 4);
} else {
    x = 2 * a[k][l];
    z = Math.sqrt(x * x + y * y);
    c = Math.sqrt((z + y) / (2 * z));
    s = signo(x / y) * Math.sqrt((z - y) / (2 * z));
}
p[k][k] = c;
p[l][l] = c;
p[k][l] = s;
p[l][k] = -s;
a = producto(p, producto(a, traspuesta(p)));
q = producto(q, traspuesta(p));
contador++;
} while (contador < 10);

if (contador == 10) {
    System.out.println("No se han podido calcular los valores propios");
}
//valores propios
//double[] valores=new double[n];
for (i = 0; i < n; i++) {
    valores[i] = (double) Math.round(a[i][i] * 1000) / 1000;
}
//vectores propios
return q;
}

//matriz traspuesta
private double[][] traspuesta(double[][] a) {
    int n = a.length; //dimensión
    double[][] d = new double[n][n];
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            d[i][j] = a[j][i];
        }
    }
    return d;
}

//producto de dos matrices
private double[][] producto(double[][] a, double[][] b) {

```

```
int n = a.length;
double[][] resultado = new double[n][n];
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        for (int k = 0; k < n; k++) {
            resultado[i][j] += a[i][k] * b[k][j];
        }
    }
}
return resultado;
}

private double[][] copyMatrix(double[][] source) {
    double[][] copy = new double[source.length][source.length];
    for (int i = 0; i < copy.length; i++) {
        System.arraycopy(source[i], 0, copy[i], 0, copy.length);
    }
    return copy;
}

private int signo(double x) {
    return (x > 0 ? 1 : -1);
}

public void printMatrix(double[][] matrix) {
    for (int i = 0; i < matrix.length; i++) {
        for (int j = 0; j < matrix.length; j++) {
            System.out.print(matrix[i][j] + " | ");
        }
        System.out.println("\n");
    }
}

public void printVector(double[] vector) {
    for (int i = 0; i < vector.length; i++) {
        System.out.print(vector[i] + " | ");
    }
    System.out.println("\n");
}

public double[] angulosVector(double[] vector) {
    double[] respuesta = new double[3]; //Angulos para X, Y y Z
    //Calcular el módulo del vector
```



```
double moduloVector = 0;
for (int i = 0; i < vector.length; i++) {
    moduloVector += vector[i] * vector[i];
}
moduloVector = Math.sqrt(moduloVector);
//Calcular los angulos del vector
for (int i = 0; i < vector.length; i++) {
    double val = vector[i] / moduloVector;
    double anguloRadianes = Math.acos(val); //Angulo en radianes
    respuesta[i] = Math.toDegrees(anguloRadianes); //Conversión del angulo a
    grados
}

return respuesta;
}

public void showAbout() {
    IJ.showMessage("PCA_", "Análisis estadístico multivariante de una serie de
    imágenes\n");
}
}
```



Anexo IX

Código del plugin para hallar contorno de una imagen

```

/**
 * Contour_Intersection_points_2D.java Created on 24 April 2012, 17:13 by Manuel
 * Guillermo Forero-Vargas e-mail: mgforero@yahoo.es
 *
 * Function: This plug-in gets the first contour found in every image.
 *
 * Note: This plugin only works with the first contour found in the image.
 *
 * This plugin do not create a new stack.
 *
 * * Función: Este plugin obtiene el primer contorno que se encuentra en cada imagen.
 * *
 * Nota: Este complemento sólo funciona con el primer contorno que se encuentra en la
 *      imagen.
 * *
 * Este complemento no crea una pila nueva.
 * *
 * Copyright (c) 2012 by Manuel Guillermo Forero-Vargas e-mail:
 * mgforero@yahoo.es
 *
 * This plugin is free software; you can redistribute it and/or modify it under
 * the terms of the GNU General Public License version 2 as published by the
 * Free Software Foundation.
 *
 * This program is distributed in the hope that it will be useful, but WITHOUT
 * ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
 * FITNESS
 * FOR A PARTICULAR PURPOSE. See the GNU General Public License for more
 * details.
 *
 * You should have received a copy of the GNU General Public License along with
 * this plugin; if not, write to the Free Software Foundation, Inc., 675 Mass
 * Ave, Cambridge, MA 02139, USA.
 */
import ij.*;
import ij.gui.GenericDialog;
import ij.plugin.filter.*;
import ij.process.*;
import java.io.File;

```



```
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;

public class Contour_ implements PlugInFilter {

    ImagePlus imp;
    int height, width, contador = 0;
    double angulo;
    Archivo archivo;
    List<int[]> coordenadas = new ArrayList();
    int z;

    public String getHoraActual() {
        Date hoy = new Date();
        SimpleDateFormat formateador = new SimpleDateFormat("hh:mm:ss:SSS");
        return formateador.format(hoy);
    }

    @Override
    public int setup(String arg, ImagePlus imp) {
        if (arg.equals("about")) {
            showAbout();
            return DONE;
        }
        this.imp = imp;
        z = 0;
        contador = imp.getImageStackSize();
        System.out.println("--->" + contador);
        if (imp == null) {
            IJ.noImage();
            return DONE;//no se ejecuta el metodo (4096)
        }
        //-----
        //Verification of binary image
        ImageStatistics stats = imp.getStatistics();//obtiene el histograma
        if (stats.histogram[0] + stats.histogram[255] != stats.pixelCount) { //se verifica que la
            imagen este en binraia
            IJ.error("8-bit binary image (0 and 255) required.");
            return DONE;
        }

        height = imp.getHeight();
```



```

width = imp.getWidth();

//ventana de dialogo
GenericDialog gd = new GenericDialog("Angulo de toma de muestra");
gd.addNumericField("Angulo: ", 5, 0);
gd.showDialog();
if (gd.wasCanceled()) {
    return DONE;
}
angulo = (int) gd.getNextNumber();
//
return IJ.setupDialog(imp, DOES_8G + DOES_8C);
}

@Override
public void run(ImageProcessor ip) {
    // System.out.println("hora inicio:"+getHoraActual());
    int initX = -1, initY = -1, posX = 0, posY = 0, dir = 6, x = 0, y;
    int cuadrante = 1;

    // Número de visitas vecinas alrededor de un píxel. Utilizado
    // para detectar cuando la región consta de un solo píxel
    int visitedNeighbours;//Number of visited neighbors around a pixel. Used
    //to detect when the region consists of just one pixel
    int[] temp = new int[2];

    int[] centerPixel = getCenterPixel(imp.getStack());
    System.out.println("x=" + centerPixel[0] + " Y=" + centerPixel[1]);
    //-----
    // Se realiza una copia de la imagen original y se encuentra el punto
    //inicial del contorno
    byte[][] pixels = new byte[width][height];
    //Se barrerá la imagen de derecha a izquierda con valor fijo en el eje Y de acuerdo
    a la
    //coordenada y del pixel central
    z++; //Contador de slides
    for (x = (height - 1); x >= 0; x--) { //Recorre la imagen en el eje X de derecha a
        izquierda
        y = centerPixel[1]; //coordenada Y del pixel central, para recorrer derecho en el eje
        Y
        if (ip.getPixel(x, y) != 0)//Si hay al menos un pixel blanco en la
        {//imagen encuentra el contorno
            posX = initX = x;
            posY = initY = y;

```



```
double angulo360 = 0.0;
double anguloAnterior = 0.0;

do {

    pixels[posX][posY] = (byte) 255;
    dir = (dir + 2) % 8;//suma 2 a su direccion y saca su modulo 8

    //Calculando el ángulo de la coordenada del pixel del borde respecto al pixel
    del centro
    double anguloPixel = angulo(centerPixel[0], centerPixel[1], posX, posY);

    //Transformando el angulo en grados sexagesimales de 0 a 360
    if (cuadrante == 1) {
        angulo360 = 90 - anguloPixel;
        if (angulo360 == 90) {
            cuadrante = 2;
        }
    } else if (cuadrante == 2) {
        angulo360 = anguloPixel + 90;
        if (angulo360 == 180) {
            cuadrante = 3;
        }
    } else if (cuadrante == 3) {
        angulo360 = (anguloPixel - 90) + 180;
        if (angulo360 == 270) {
            cuadrante = 4;
        }
    } else {
        angulo360 = (180 - anguloPixel) + 270;
    }

    //-----
    //Captura las coordenadas de los pixeles del borde que están a intervalos
    de angulo
    //(valor ingresado por la ventana)
    int[] fila = new int[3];
    if (angulo360 > anguloAnterior + angulo) {
        fila[0] = z - 1;
        fila[1] = posX;
        fila[2] = posY;
        coordenadas.add(fila);
        anguloAnterior = angulo360;
    }
}
```




```

    }
    //-----

    System.out.println(" xc:" + centerPixel[0] + " yc:" + centerPixel[1] + " posX:"
+ posX + " posY:" + posY + " Angulo:" + (angulo360));
    temp = direction(posX, posY, dir);
    visitedNeighbours = 0;
    //si los valores de estos son menores a 0 es porque se salen del rango de la
imagen
    //y se disminuye su direccion en 1
    while (temp[0] < 0 || ip.getPixel(temp[0], temp[1]) == 0) {
        dir = (dir + 7) % 8;
        temp = direction(posX, posY, dir);
        visitedNeighbours++;
        if (visitedNeighbours > 7)//Only 7 new directions can be visited, in addition
to the initial direction
        {
            break;
        }
    }
    posX = temp[0];
    posY = temp[1];
} while ((posX != initX || posY != initY) && visitedNeighbours <= 7);
//-----
for (y = 0; y < height; y++) {
    for (x = 0; x < width; x++) {
        ip.putPixelValue(x, y, pixels[x][y] & 0xff);
    }
}
//Guardar los datos de los puntos de acuerdo al angulo en un archivo
//Convierte un arrayList a matriz
int[][] coordenadas2 = (int[][]) coordenadas.toArray(new int[][]{});
//Envío la matriz al objeto
archivo = new Archivo(coordenadas2, imp.getTitle());
//Graba el archivo
archivo.escribir();

imp.show();
imp.updateAndDraw();
//-----
//y = height;//To break the second for loop
break;
}
}

```



```
//System.out.println("hora ffin:"+getHoraActual());
    if (contador == z) {
        momento_estadistico m = new momento_estadistico();
        m.calcularMedia(imp.getTitle());
    }
}
```

```
private int[] direction(int posX, int posY, int direccion) {
    int[] retorno = new int[2];
```

```
    switch (direccion) {
        case 0:
            retorno[0] = posX + 1;
            retorno[1] = posY;
            break;
        case 1:
            retorno[0] = posX + 1;
            retorno[1] = posY - 1;
            break;
        case 2:
            retorno[0] = posX;
            retorno[1] = posY - 1;
            break;
        case 3:
            retorno[0] = posX - 1;
            retorno[1] = posY - 1;
            break;
        case 4:
            retorno[0] = posX - 1;
            retorno[1] = posY;
            break;
        case 5:
            retorno[0] = posX - 1;
            retorno[1] = posY + 1;
            break;
        case 6:
            retorno[0] = posX;
            retorno[1] = posY + 1;
            break;
        case 7:
            retorno[0] = posX + 1;
            retorno[1] = posY + 1;
            break;
```

```

    }
    if (retorno[0] < 0 || retorno[1] < 0 || retorno[0] >= width || retorno[1] >= height) {
        retorno[0] = -1;
        retorno[1] = -1;
    }
    return retorno;
}

void showAbout() {
    IJ.showMessage("About Contour...",
        "This plug-in filter finds the first contour in every image.");
}

private int[] getCenterPixel(ImageStack sStack) {
    // Calcular los promedio de las coordenadas en X, Y y Z de los píxeles de la región
    // de interés
    int nSlides = sStack.getSize();//Cantidad de imagenes del stack
    int width = sStack.getWidth(); //Ancho de la imagen del stack
    int height = sStack.getHeight(); //Alto de la imagen del stack
    int size = width * height; //Tamaño del vector de pixeles de cada imagen (ancho *
        // alto)
    double sumX = 0, sumY = 0, sumZ = 0;
    int contPoints = 0; //Contador de pixeles de la región de interes de cada imagen
    int[] pixelCentral = new int[2];
    //---Recorre el stack de imagenes
    for (int z = 0; z < nSlides; z++) {
        //Extrae los pixeles de cada imagen y lo almacena en un Vector
        byte[] pixeles = (byte[]) sStack.getProcessor(z + 1).getPixels();
        //Recorre el vector de píxeles
        for (int i = 0; i < size; i++) {
            //Evalua si el pixel pertenece a la región de interés.
            //Siendo que la imagen es binarizada los píxeles de la región de interés tienen
            //valor=255 (Blanco)
            //y los píxeles del fondo tienen valor=0 (negro)
            if (pixeles[i] != 0) {
                sumX += (i % width); //(i%width) calcula el valor de la coordenada X del
                // pixel
                sumY += (i / width); //(i/width) calcula el valor de la coordenada Y del pixel
                //sumZ+=z; //El numero de Slides(imagenes del stack) representa el valor
                // de la coordenada Z del pixel
                contPoints++; //Cuenta la cantidad de píxeles que correspondian a la región
                // de interes
            }
        }
    }
}

```



```
}  
sumX = sumX / contPoints; //Promedio de las coodenadas X de todos los píxeles de  
    la región de interes  
sumY = sumY / contPoints; //Promedio de las coodenadas Y de todos los píxeles de  
    la región de interes  
//sumZ=sumZ/contPoints; //Promedio de las coodenadas Z de todos los píxeles de  
    la región de interes  
pixelCentral[0] = (int) sumX;  
pixelCentral[1] = (int) sumY;  
return pixelCentral;  
}  
  
private double angulo(int xc, int yc, int xp, int yp) {  
    double hipo = Math.hypot(xp - xc, yp - yc);  
    double anguloRadian = Math.acos((yp - yc) / hipo);  
    //System.out.println("Radianes:"+anguloRadian);  
    return Math.toDegrees(anguloRadian);  
}  
  
}
```

ANEXO X

Código Para Hallar Los 3 Momentos Estadísticos

```
import ij.*;
import ij.plugin.filter.*;
import ij.process.*;
import ij.gui.*;
import ij.measure.*;
import ij.text.*;
import java.awt.*;
import java.util.Date;
```

```
/* Moment_Calculator plugin
```

```
*   @author      Francois Richard
*   @author      University of Ottawa - Earth Sciences
*   @author      richard@science.uottawa.ca

*   @date        04-JUN-2001

*   @history     08-Feb-2006    Added irregular ROI support based on code
from Masking_example.java
```

This plug-in computes spatial moments up to the 4th order for the selected (rectangular) ROI, along with some parameters derived from them. Results are displayed in the main ImageJ window, in addition to those set for ImageJ's Measure command 'Analyze|Set Measurements'. To export to a text file, use 'File|Save As|Measurements' or right-click on the table and select 'Save As'

Note that spatial moments are a very simple and powerful way to describe the spatial distribution of values, provided they have a sufficiently strong central tendency, that is, a tendency to cluster around some particular value. This implies that "background" pixel values are small (e.g. zones where the quantity of interest, such as concentration, is zero). Conversely, zones of high concentration (density, etc.) should also have a high pixel values. This can lead to meaningless results, for example, in the case of uncalibrated images, where (white) background pixels are equal to 255 (for an 8-bit greyscale image).

**** Interpretation of spatial moments ****



- * order 0 = TOTAL MASS [units: concentration, density, etc.]
- * order 1 = location of CENTRE OF MASS in x and y from 0,0 [units: L]
- * order 2 = VARIANCE (spreading) around centroid in x and y [units: L²]
- * order 3 = coeff. of SKEWNESS (symmetry) in x and y [units: n/a]
 - > =0 : SYMMETRIC distribution
 - > <0 : Distribution asymmetric to the LEFT
(tail extends left of centre of mass)
 - > >0 : Distribution asymmetric to the RIGHT
(tail extends right of centre of mass)
- * order 4 = KURTOSIS (flatness) in x and y [units: n/a]
 - > =0 : Gaussian (NORMAL) distribution
 - > <0 : Distribution FLATTER than normal
 - > >0 : Distribution MORE PEAKED than normal
 - > <-1.2: BIMODAL (or multimodal) distribution

** Parameters derived from 2nd moments ** (from Awcock (1995) "Applied Image Processing")

- * ELONGATION (ECCENTRICITY) = Ratio of longest to shortest distance vectors from the object's centroid to its boundaries
- * ORIENTATION = For elongated objects, describes the orientation (in degrees) of the "long" direction with respect to horizontal (x axis)

*/

```
public class Moment_Calculator implements PlugInFilter, Measurements {
    ImagePlus imp;
    boolean done;
    static boolean firstTime = true;
    static boolean show_imageName = true;
    static boolean show_m00 = true;
    static boolean show_xC = true;
    static boolean show_yC = true;
    static boolean show_xxVar = true;
    static boolean show_yyVar = true;
    static boolean show_xyVar = true;
    static boolean show_xSkew = true;
    static boolean show_ySkew = true;
    static boolean show_xKurt = true;
    static boolean show_yKurt = true;
    static boolean show_orientation = true;
    static boolean show_eccentricity = true;
    static double dCutoff = 0.0; // default cutoff (minimum) value for calcs
```



```
        // (only values >= dCutoff are used)
        // (use "0" to include all positive pixel values)
static double dFactor = 1.0; // default factor
        // (multiplies pixel values prior to calculations)

public int setup(String arg, ImagePlus imp) {
    if (IJ.versionLessThan("1.23k")) // needs the new PluginFilter interface
        return DONE;
    if (arg.equals("about")) {showAbout(); return DONE;}
    this.imp = imp;
    IJ.register(Moment_Calculator.class);
    return DOES_ALL+DOES_STACKS+NO_CHANGES;
} // end of 'setup()' method

public void run(ImageProcessor ip) {
    if (done)
        return;

    if (firstTime || Analyzer.getResultsTable().getCounter()==0) {
        if (Analyzer.resetCounter()) {
            setMoments(); // similar to Analyze|Set Measurements
            firstTime = false;
        } else {
            return; // user canceled save changes dialog
        }
    }

    int measurements = Analyzer.getMeasurements(); // defined in Set Measurements
    dialog
    Analyzer.setMeasurements(measurements);
    Analyzer a = new Analyzer();
    Calibration cal = imp.getCalibration();
    //ImageStatistics stats = imp.getStatistics(measurements);
    ImageStatistics stats = ImageStatistics.getStatistics(ip,measurements,cal);

    // Declare & initialize variables

    double zero = 0.0;
    double m00 = zero;
    double m10 = zero, m01 = zero;
    double m20 = zero, m02 = zero, m11 = zero;
    double m30 = zero, m03 = zero, m21 = zero, m12 = zero;
    double m40 = zero, m04 = zero, m31 = zero, m13 = zero;
    double xC=zero, yC=zero;
```



```

double xxVar = zero, yyVar = zero, xyVar = zero;
double xSkew = zero, ySkew = zero;
double xKurt = zero, yKurt = zero;
double orientation = zero, eccentricity = zero;
double currentPixel, xCoord, yCoord;

// Get image and ROI info
// Note: currently supports rectangular ROIs only

String imageName = imp.getTitle();
int width = ip.getWidth();
int height = ip.getHeight();
double pw = cal.pixelWidth;
double ph = cal.pixelHeight;
boolean isScaled = cal.scaled();
boolean isCalibrated = cal.calibrated();
String calUnits = cal.getValueUnit();
String units = cal.getUnits();
Roi roi = imp.getRoi();
Rectangle r = ip.getRoi();
byte[] mask = ip.getMaskArray();
int maskCounter = 0;
ip.setCalibrationTable(cal.getCTable());

// Compute moments of order 0 & 1

for (int y=r.y; y<(r.y+r.height); y++) {
    for (int x=r.x; x<(r.x+r.width); x++) {
        if (mask==null || mask[maskCounter++]!=0) {
            xCoord = (x+0.5)*pw; //this pixel's X calibrated coord. (e.g. cm)
            yCoord = (y+0.5)*ph; //this pixel's Y calibrated coord. (e.g. cm)
            currentPixel=ip.getPixelValue(x,y);
            currentPixel=currentPixel-dCutoff;
            if (currentPixel < 0) currentPixel = zero; //gets rid of negative pixel values
            currentPixel = dFactor*currentPixel;
/*0*/    m00+=currentPixel;
/*1*/    m10+=currentPixel*xCoord;
        m01+=currentPixel*yCoord;
        }
    }
}

// Compute coordinates of centre of mass

```



```

xC = m10/m00;
yC = m01/m00;

// Compute moments of orders 2, 3, 4

// Reset index on "mask"
maskCounter = 0;
for (int y=r.y; y<(r.y+r.height); y++) {
  for (int x=r.x; x<(r.x+r.width); x++) {
    if (mask==null || mask[maskCounter++]!=0) {
      xCoord = (x+0.5)*pw; //this pixel's X calibrated coord. (e.g. cm)
      yCoord = (y+0.5)*ph; //this pixel's Y calibrated coord. (e.g. cm)
      currentPixel=ip.getPixelValue(x,y);
      currentPixel=currentPixel-dCutoff;
      if (currentPixel < 0) currentPixel = zero; //gets rid of negative pixel values
      currentPixel = dFactor*currentPixel;
/*2*/   m20+=currentPixel*(xCoord-xC)*(xCoord-xC);
        m02+=currentPixel*(yCoord-yC)*(yCoord-yC);
        m11+=currentPixel*(xCoord-xC)*(yCoord-yC);

/*3*/   m30+=currentPixel*(xCoord-xC)*(xCoord-xC)*(xCoord-xC);
        m03+=currentPixel*(yCoord-yC)*(yCoord-yC)*(yCoord-yC);
        m21+=currentPixel*(xCoord-xC)*(xCoord-xC)*(yCoord-yC);
        m12+=currentPixel*(xCoord-xC)*(yCoord-yC)*(yCoord-yC);

/*4*/   m40+=currentPixel*(xCoord-xC)*(xCoord-xC)*(xCoord-xC)*(xCoord-xC);
        m04+=currentPixel*(yCoord-yC)*(yCoord-yC)*(yCoord-yC)*(yCoord-yC);
        m31+=currentPixel*(xCoord-xC)*(xCoord-xC)*(xCoord-xC)*(yCoord-yC);
        m13+=currentPixel*(xCoord-xC)*(yCoord-yC)*(yCoord-yC)*(yCoord-yC);
    }
  }
}

// Normalize 2nd moments & compute VARIANCE around centre of mass
xxVar = m20/m00;
yyVar = m02/m00;
xyVar = m11/m00;

// Normalize 3rd moments & compute SKEWNESS (symmetry) around centre of
mass
// source: Farrell et al, 1994, Water Resources Research, 30(11):3213-3223
xSkew = m30 / (m00 * Math.pow(xxVar,(3.0/2.0)));
ySkew = m03 / (m00 * Math.pow(yyVar,(3.0/2.0)));

```



```
// Normalize 4th moments & compute KURTOSIS (peakedness) around centre of
mass
// source: Farrell et al, 1994, Water Resources Research, 30(11):3213-3223
xKurt = m40 / (m00 * Math.pow(xxVar,2.0)) - 3.0;
yKurt = m04 / (m00 * Math.pow(yyVar,2.0)) - 3.0;

// Compute Orientation and Eccentricity
// source: Awcock, G.J., 1995, "Applied Image Processing", pp. 162-165
orientation = 0.5*Math.atan2((2.0*m11),(m20-m02));
orientation = orientation*180./Math.PI; //convert from radians to degrees
eccentricity = (Math.pow((m20-m02),2.0)+(4.0*m11*m11))/m00;

a.saveResults(stats, roi); // store in system results table
ResultsTable rt=Analyzer.getResultsTable(); // get the system results table

if (show_imageName) rt.addLabel("Image",imageName);
rt.addValue("Cutoff", dCutoff);
rt.addValue("Factor", dFactor);
if (show_m00) rt.addValue("Mass", m00);
if (show_xC) rt.addValue("xC", xC);
if (show_yC) rt.addValue("yC", yC);
if (show_xxVar) rt.addValue("xxVar", xxVar);
if (show_yyVar) rt.addValue("yyVar", yyVar);
if (show_xyVar) rt.addValue("xyVar", xyVar);
if (show_xSkew) rt.addValue("xSkew", xSkew);
if (show_ySkew) rt.addValue("ySkew", ySkew);
if (show_xKurt) rt.addValue("xKurt", xKurt);
if (show_yKurt) rt.addValue("yKurt", yKurt);
if (show_orientation) rt.addValue("Orient.", orientation);
if (show_eccentricity) rt.addValue("Elong.", eccentricity);

int counter = rt.getCounter();
if(counter==1) {
    updateHeadings(rt); // update the worksheet headings
    Date date = new Date(); //get today's date
    String comment = " [ "+date+" ]";
    IJ.write(comment);
    if(isScaled) {
        IJ.write(" Scaled image [ distance units = "+units+" ].");
    } else {
        IJ.write(" No spatial calibration [results in pixels].");
    }
}
if(isCalibrated) {
    IJ.write(" Calibrated image [ density units = "+calUnits+" ].");
}
```



```

    } else {
        IJ.write(" No density calibration [uncalibrated results].");
    }
}

IJ.write(rt.getRowAsString(counter-1));

} // end of 'run()' method

// Prompt user to select moments to display & set cutoff value
public void setMoments() {
    GenericDialog gd = new GenericDialog("Set Spatial Moments");
    gd.addCheckbox("Image_Name ", true);
    gd.addCheckbox("Total_Mass ", true);
    gd.addCheckbox("X_Centre of Mass ", true);
    gd.addCheckbox("Y_Centre of Mass ", true);
    gd.addCheckbox("X_Variance ", true);
    gd.addCheckbox("Y_Variance ", true);
    gd.addCheckbox("XY_Covariance ", true);
    gd.addCheckbox("X_Skewness ", true);
    gd.addCheckbox("Y_Skewness ", true);
    gd.addCheckbox("X_Kurtosis ", true);
    gd.addCheckbox("Y_Kurtosis ", true);
    gd.addCheckbox("Orientation ", true);
    gd.addCheckbox("Elongation ", true);
    gd.addNumericField("Cutoff Value: ", dCutoff, 4);
    gd.addNumericField("Scaling Factor: ", dFactor, 4);
    gd.addMessage(" Note: Pixel values will be converted prior to \n"+
        "      moment calculations using:\n\n"+
        "      pixelValue = Factor*(pixelValue-Cutoff) \n");
    gd.showDialog();
    if (gd.wasCanceled()) {
        IJ.showMessage("Moment Calculator", "Default values will be used.");
    }
    return;
}
show_imageName = gd.getNextBoolean();
show_m00 = gd.getNextBoolean();
show_xC = gd.getNextBoolean();
show_yC = gd.getNextBoolean();
show_xxVar = gd.getNextBoolean();
show_yyVar = gd.getNextBoolean();
show_xyVar = gd.getNextBoolean();
show_xSkew = gd.getNextBoolean();
show_ySkew = gd.getNextBoolean();

```



```

show_xKurt = gd.getNextBoolean();
show_yKurt = gd.getNextBoolean();
show_orientation = gd.getNextBoolean();
show_eccentricity = gd.getNextBoolean();
dCutoff = (double)gd.getNextNumber();
dFactor = (double)gd.getNextNumber();
}

public void updateHeadings(ResultsTable rt) { // Wayne Rasband
    TextPanel tp = IJ.getTextPanel();
    if (tp==null)
        return;
    String worksheetHeadings = tp.getColumnHeadings();

    String tableHeadings = rt.getColumnHeadings();
    if (!worksheetHeadings.equals(tableHeadings))
        IJ.setColumnHeadings(tableHeadings);
} // end of 'updateHeadings' method

// Create a message about this plugin in 'Help|About Plugins' submenu
// (must also modify IJ_Props.txt and add it to ij.jar for this to show up)

void showAbout() {
    IJ.showMessage("About Moment_Calculator...",
        " This plug-in computes spatial moments up to the 4th order\n" +
        " for the selected ROI.\n" +
        " Results are displayed in the main ImageJ window, in addition\n" +
        " to those set for ImageJ's Measure command 'Analyze|Set Measurements'\n" +
        " To export to a text file, use 'File|Save As|Measurements'.\n" +
        " or right-click on the table and select 'Save As'\n" +
        " \n" +
        " ** Interpretation of spatial moments ** \n" +
        " \n" +
        " * order 0 = TOTAL MASS [units: concentration, density, etc.]\n" +
    "\n"+
        " * order 1 = location of CENTRE OF MASS in x and y from 0,0 [units:
L]\n"+
        " * order 2 = VARIANCE (spreading) around centroid in x and y [units:
L^2]\n"+
        " * order 3 = coeff. of SKEWNESS (symmetry) in x and y [units: n/a]\n"+
        " --> =0 : SYMMETRIC distribution\n"+
        " --> <0 : Distribution asymmetric to the LEFT\n"+
        " (tail is to left of centre of mass)\n"+
        " --> >0 : Distribution asymmetric to the RIGHT\n"+

```



```

"                                (tail is to left of centre of mass)\n"+
" * order 4 = coeff. of KURTOSIS (flatness) in x and y [units: n/a]\n"+
"                                --> =0 : Gaussian (NORMAL) distribution\n"+
"                                --> <0 : Distribution FLATTER than normal\n"+
"                                --> >0 : Distribution MORE PEAKED than normal\n"+
"                                --> <-1.2: BIMODAL distribution\n"+
"\n"+
" ** Parameters derived from 2nd moments ** (from Awcock (1995) 'Applied
Image Processing')\n"+
"\n"+
" * ELONGATION = Ratio of longest to shortest distance vectors\n"+
" (ECCENTRICITY) from the object's centroid to its boundaries\n"+
"\n"+
" * ORIENTATION = For elongated objects, describes the orientation (in
degrees)\n"+
" of the 'long' direction with respect to horizontal (x axis)\n"+
"\n";
} // end of 'showAbout()' method

} // end of 'Moment_Calculator' class

```

