Instituto Tecnológico Superior de Misantla

Maestría en sistemas computacionales

DIGITALIZADOR 3D PARA DIAGNÓSTICO Y TRATAMIENTO DE PIE PLANO

TESIS

PARA OBTENER EL GRADO DE MAESTRO EN SISTEMAS COMPUTACIONALES.

PRESENTA:

ING. FÉLIX DÍAZ SANTOS

ASESOR:
DR. RAJESH ROSAN BISWAL
CO-ASESOR:
DR. EDDY SANCHEZ DE LA CRUZ

MISANTLA, VERACRUZ, AGOSTO 2018

INSTITUTO TECNOLÓGICO SUPERIOR DE MISANTLA
DIVISIÓN DE ESTUDIOS PROFESIONALES
AUTORIZACIÓN DE IMPRESIÓN DE TRABAJO DE TITULACIÓN MAESTRÍA

FECHA: 23 de Agosto de 2018.

ASUNTO: **AUTORIZACIÓN DE IMPRESIÓN DE TESIS.**

A QUIEN CORRESPONDA:

Por medio de la presente se hace constar que el (la) C:

FELIX DÍAZ SANTOS

estudiante de la maestría en SISTEMAS COMPUTACIONALES con No. de Control 162T0054 ha cumplido satisfactoriamente con lo estipulado por el **Lineamiento de Posgrado para la obtención del grado de Maestría** mediante **Tesis.**

Por tal motivo se **Autoriza** la impresión del **Tema** titulado:

DIGITALIZADOR 3D DE BAJO COSTO PARA ESCANEO DE PIE PLANO USANDO CÁMARA Y LÁSER

Dándose un plazo no mayor de un mes de la expedición de la presente a la solicitud del examen para la obtención del grado de maestría.

ATENTAMENTE

DIVISIÓN DE ESTUDIOS PROFESIONALES

DR. Rajesh Roshan Biswal
Presidente

M.S.C. Eddy Sánchez de la Cruz
Secretario

M.S.C. Galdino Martínez Flores
Vocal

Archivo.

Agradecimientos

Agradezco primeramente a Dios, por haberme dado la fortaleza, la sabiduría y los recursos para poder terminar esta etapa de mi vida, agradezco a mi familia, pilares de orgullo y ejemplo, por haberme apoyado durante todo este proceso, le doy las gracias a mi director de tesis el Dr. Rajesh Roshan Biswal por su tiempo, dedicación y apoyo.

Agradezco a las instituciones que permitieron la realización del presente trabajo, al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico brindado, a todos mis profesores que me brindaron sus conocimientos en las diferentes materias que a lo largo de la maestría, a nuestros compañeros, gracias por su apoyo y paciencia.

Por ultimo gracias a todas las personas con las que tuve el privilegio de compartir y que me brindaron su apoyo incondicional.

Resumen

En el presente trabajo se propone aplicar un digitalizador 3d para escanear la superficie de los pies para analizar y determinar si es un pie plano, en caso de ser detectado un pie plano se propone generar una plantilla con las geometrías necesarias y medidas específicas particulares de cada pie. El método empleado para realizar la digitalización 3D, es el método de triangulación láser, en el cual se hace uso de una cámara web para capturar las imágenes, un láser lineal para proyectar contornos, mismos que de acuerdo a su deformación permiten determinar la profundidad. Para generar la tercera dimensión se hace uso de un eje móvil, el cual tiene la función de desplazar la línea láser para generar un barrido de toda la superficie. El sistema se desarrolló en Java y da como resultado un archivo en extensión STL con una superficie definida por triángulos que representa la plantilla generada para la corrección del pie plano. El archivo STL es compatible con software de impresión 3D.

Abstract

In the present work it is proposed to apply a 3d digitizer to scan the surface of the feet to analyze and determine if it is a flat foot, if a flat foot is detected, it is proposed to generate a template with the necessary geometries and specific measurements specific to each foot. The method used to perform 3D scanning is the laser triangulation method, in which a webcam is used to capture the images, a linear laser to project contours, which according to their deformation allows to determine the depth. To generate the third dimension, a moving axis is used, which has the function of moving the laser line to generate a sweep of the entire surface. The system was developed in JAVA and results in a file in STL extension with a surface defined by triangles that represents the template generated for the correction of the flat foot. The STL file is compatible with 3D printing software.

Índice general

Índice de figuras

Índice de tablas

Índice de algoritmos

Capítulo 1

Generalidades

1.1. Introducción

El pie plano [1] es definido como una caída de la planta del pie, también conocida como bóveda plantar y se da cuando se el individuo presenta menos arco del habitual, es decir, el eje del pie se encuentra hacia dentro. Un dato curioso es que al nacer todos tenemos el pie plano, aunque es un pie plano flexible que con el paso del tiempo y el crecimiento del niño, se corrige. Sin embargo también suele darse el caso en que un adulto con un pie normal, padezca un pie plano por causa de una falla tibial posterior [2]. Esto implica que el músculo que mantiene al arco deja de funcionar adecuadamente provocando que la planta del pie caiga cada vez más hacia dentro. Para diagnosticar esta patología se emplean sistemas de análisis de la huella con sensores de presión, sistemas de visión artificial a través de diferentes métodos de digitalizado; láser o luz estructurada, de esta manera se observa en una computadora cómo es la huella y con eso podemos determinar el diagnóstico [3]. Es importante identificar la naturaleza del pie plano, el cual puede ser flexible, semi flexible o rígido. Para identificar el tipo de patologia es común realizar diferentes test biomecánicos. En los pacientes menores de 4 años, es muy importante realizar un buen diagnóstico dado que puede ser un pie que puede corregirse con ejercicios de fortalecimiento. Después de los 4 años, es probable que solo deba hacer plantillas personalizadas para modificar la disposición de los ejes del pie, ya que estos músculos comienzan a trabajar de forma inesperada. En algunos casos particulares es necesario considerar la necesidad de realizar una cirugía. Cuando se trata de adultos, lo ideal es contar con una plantilla personalizada de buena calidad la cual minimice los resultados de un pie plano. Un claro síntoma de contar con pie plano es el cansancio, ya que conlleva a un desgaste energético mayor al de un pie normal, que resulta en mucho esfuerzo para caminar y más cuando uno ya está cansado ya que la sensación que da es de arrastrar los pies. También suele presentarse síntomas de pesadez en las piernas, debido a problemas de retorno de la circulación y la presión muscular dentro de la pantorrilla y la región interna de la pierna. Otro punto de vista excepcionalmente importante es que, al caer el pie hacia adentro, expulsa la rodilla de su centro de trabajo. Un pie cuenta con 26 huesos [4] y, a pesar de que está nivelado, tiene una pequeña capacidad de ajuste, una rodilla puede ser

considerada como una bisagra, y en caso de no encontrarse alineada se frotará, y en caso de que se frote, se desgastará. Como resultado la rodilla se vuelve interna y es excepcionalmente simple de relacionar un pie plano con un valgo de rodilla (genu).

La solución propuesta a esta problemática, es la construcción de un digitalizador 3d para el escaneo de la planta del pie, para determinar de manera activa la profundidad del arco planar. Con las métricas obtenidas del pie, generar una plantilla impresa en 3d personalizada para cada paciente. Al final de la investigación se observo que para imprimir las plantillas solo es necesario conocer las dimensiones del pie y la profundidad del arco plana, sin necesidad de digitalizar todo el pie.

1.2. Planteamiento del problema

"En México, entre 15 y 20 por ciento de la población padece pie plano y de no ser tratado con oportunidad, puede ocasionar desgaste articular de tobillo, rodilla, cadera y columna", indicó la doctora María del Carmen García Ruiz, ortopedista y traumatóloga adscrita al Servicio de Ortopedia del Hospital General de México Eduardo Liceaga (HGM). El 60 por ciento de las personas que acuden al servicio de ortopedia del hospital, presentan este problema. Se trata de una condición que se caracteriza por la ausencia o deformidad del arco de la planta del pie, el cual da estabilidad a la marcha, distribuye las presiones o cargas, sirve de resorte de músculos y ligamentos, facilita la adaptación a las irregularidades del terreno y contribuye en los movimientos de impulsión o amortiguamiento. A pesar de que se puede identificar a cualquier edad, es conveniente que sea diagnosticado a partir de los dos años, ya que antes y desde el nacimiento los infantes cuentan con un cojinete graso en esta parte del pie que puede dar la apariencia de pie plano.

«En febrero del 2015 el Departamento de Investigación de la Universidad Autónoma de Tamaulipas en conjunto con el Hospital General Regional No. 6 Dr. Ignacio García Téllez, Instituto Mexicano del Seguro Social, Ciudad Madero, Tamaulipas, México realizaron un estudio analítico, transversal»[5] con 1,128 infantes de 9 a 11 años de edad, de los cuales el 48.8 % correspondieron al sexo masculino y el 51.2 % al femenino. Realizaron mediciones antropométricas (peso, talla, perímetro de cintura y cadera). Se calculó el índice de masa corporal (IMC) y consideró como obesidad cuando el IMC fuera mayor del percentil 95. Encontrando que la prevalencia de sobrepeso-obesidad fue del 49.1 % y de pie plano fue del 12.1 % (Hombres: 8.1 %, Mujeres: 4 %; p = 0.28). La asociación entre obesidad y pie plano fue significativa (p mayor a 0.001) y con un riesgo 2.5 veces mayor en los niños con sobrepeso-obesidad en comparación con los de peso normal.

El estudio llego a la conclusión que existe una estrecha relación entre la obesidad y el pie plano, por lo que se indica implementar medidas de prevención secundaria en la población. Los problemas ortopédicos asociados con el pie plano son mas frecuentes en niños. Los médicos familiares y pediatras son altamente consultados por este tipo de patologías. Por tal motivo es importante tener una clara diferenciación entre un pie normal y un pie plano

que necesita tratamiento de manera anticipada.

El empleo de la tecnología láser escáner 3D se ha ido incorporando a diferentes campos a lo largo de los años. Esta tecnología se está viendo involucrada en áreas de trabajo tan diferentes como puedan ser, el control dimensional de componentes industriales o la generación de modelos digitales del terreno, pasando por aplicaciones como la documentación de patrimonio cultural o la generación de entornos virtuales en vídeo juegos.

A esta constante evolución se están incorporando nuevos equipos como es el caso de las impresoras 3D, cada vez más accesibles y precisas, que ofrecen grandes posibilidades en la arquitectura, el diseño industrial, el entretenimiento o el campo médico, al permitir convertir modelos 2D en prototipos reales. La implementación de un escáner 3D que permita detectar el arco del pie del paciente, obteniendo altura y profundidad dando pauta a ofrecer una plantilla personalizada, la cual atienda las necesidades del paciente. Es necesario mencionar la importancia de los programas de modelado tridimensional que son cada vez más potentes. Los cuales permiten el tratamiento completo de los datos extraídos de los equipos escáner 3D.

1.2.1. Justificación

Hoy en día resulta complicado desarrollar de manera tangible nueva tecnología, tenemos en contra miles de inventos y descubrimientos, así como el recurso económico y tecnológico. De manera teórica podemos plantear cientos de teorías, proyectos o productos. Comprobar físicamente alguna de estas teorías podría tomarnos años por las limitantes tecnológicas no de la época, sino de las tecnologías al alcance. En el área de la tecnología nos preocupa obtener siempre lo último, nuevos procesadores, nuevos smartphones, internet más rápido, etc... si bien es cierto que la tecnología avanza a pasos agigantados lo que hoy es nuevo mañana es obsoleto. ¿Pero cuál es el significado de obsoleto? ¿Lo obsoleto ya no sirve o no es conveniente usarlo? ¿Sí propongo una solución con tecnología obsoleta el producto no vale la pena?. Por definición de diccionario obsoleto es "Que no se usa en la actualidad, que ha quedado claramente anticuado." Lo obsoleto no implica que no funcione en la época actual, un ejemplo de esto es el puerto serial RS232 de las computadoras el cual ya no se encuentra presente en la mayoría de las LapTop, sin embargo muchos dispositivos continúan trabajando con el protocolo RS232 a través de un COM Virtual por USB.

Proponer una solución con tecnología obsoleta para competir con tecnología actual no tiene sentido, es como comparar el lenguaje de programación en ensamblador con Python, naturalmente Python se lleva las palmas, sin embargo que pasa si tenemos un microcontrolador de 8 bits este simplemente no será capaz de correr ningún lenguaje de alto nivel, en el mejor de los casos C o C++. ¿Si las computadoras actuales son de 64bits un microcontrolador de 8 bits es obsoleto? en el tema computacional efectivamente es obsoleto, pero en el tema aplicación aún tiene mucho mercado, un refrigerador tiene un microcontrolador de 8 bits, un ventilador, el control remoto del televisor, la llave de nuestro auto, la lavadora, el microondas

entre otros aparatos. ¿Cuál es el factor común? Simplemente que no está compitiendo con la tecnología de la cual proviene, si no que funciona como soporte para otras áreas y tecnologías.

El presente trabajo de investigación propone aplicar tecnologías computacionales en la rama de la salud. En específico se propone desarrollar un digitalizador 3D basado en el método de triangulación láser aplicado para el diagnóstico y tratamiento del pie plano. En realidad no se está inventando el hilo negro, actualmente existen tecnologías más rápidas, confiables y precisas que la triangulación láser, sin embargo esta tecnología es suficiente para medir las profundidades del arco del pie. Actual mente existen equipos llamados podoscopios [6] digitales los cuales realizan la misma función. Estos equipos son instrumentos médicos utilizados principalmente en España, la intensión de este trabajo es ofrecer una alternativa con tecnología Mexicana para el diagnóstico y tratamiento del pie plano.

1.3. Objetivos

1.3.1. Objetivo general

Desarrollar un digitalizador 3d capaz de diagnosticar el pie plano así como el tratamiento adecuado en caso de requerirse.

1.3.2. Objetivos específicos

- Desarrollar un dispositivo que escanee la longitud del pie mediante una cámara web y un láser.
- Generar una nube de puntos que permita la reconstrucción digital de la superficie del pie.
- Analizar y determinar si cuenta o no con pie plano.
- Generar una propuesta de tratamiento (plantilla) en base al diagnóstico.

1.4. Hipótesis

¿El uso de un escáner 3d para digitalizar la planta del pie permite un diagnóstico completo para el pie plano?

1.4.1. Hipótesis alterna

El análisis en tres dimensiones aporta mayor información para diagnosticar el pie plano.

1.4.2. Hipótesis nula

La información adicional que aporta un análisis en tres dimensiones no es significativa para diagnosticar el pie plano.

1.5. Motivación

El desarrollo de este trabajo responde al conocimiento adquirido a lo largo de la vida académica, con formación en electrónica, automatización, control y programación.

Entre los antecedentes de investigación se encuentran los siguientes proyectos:

1. Desarrollo de un escáner 3D por medio de un medidor de distancias ultrasónico con un barrido vertical y radial del objeto. Proyecto de taller de investigación.

2. Actualización del escáner 3D ultrasónico a visión por computadora a través de una cámara web para objetos cilíndricos utilizando el método de contornos y tomando como referencia una longitud conocida. Proyecto para control de calidad en piezas maquinadas por un torno CNC en el laboratorio de manufactura integrada por computadora en el Instituto Tecnológico de Veracruz.

3. Modificación de escáner 3D para objetos cilíndricos a escáner por triangulación laser. Proyecto para la Universidad Politécnica de Tlaxcala

Basado en la experiencia previa y en una enfermedad propia surge la idea de aplicar esta tecnología para dar soporte a un área no tecnología, en este caso el área de la salud, particularmente a la ortopedia.

1.6. Propuesta de solución

Los podoscopios 2D son fundamentales en las consultas donde se requiere un podoscopio y queremos dar una imagen de innovación. Los podoscopios 2D son una evolución de los podoscopios tradicionales de espejos y lámparas. Los podoscopios 2D son instrumentos médicos que modernizan y agilizan el diagnóstico del pie plano, ofrecen realizar algunas mediciones en tiempo real. Sin embargo son instrumentos que no permiten un análisis detallado de profundidad, por lo que a pesar de agilizar el diagnóstico del pie plano el tratamiento sigue siendo un trabajo artesanal. El éxito de la fabricación de la plantilla dependerá en su totalidad de la experiencia del técnico ortopedista.

Por tal motivo se propone desarrollar un podoscopio 3D, el cual permita además el análisis 2D tradicional un análisis 3D el cual permite determinar las profundidades del arco del interno del pie. Al contar con esta información, es posible generar una plantilla en 3D la cual puede ser impresa fácilmente por una impresora 3D. Este modelo de trabajo proporciona la

flexibilidad de modelar y corregir las plantillas antes de su fabricación, optimizando tiempos y costos.

1.7. Alcances y limitaciones

1.7.1. Alcances

En el presente trabajo se abarca el diseño y construcción de un digitalizador 3D enfocado al diagnóstico del pie plano. El digitalizador 3D para pies también conocido como podoscopio 3D es un instrumento médico que permite diseñar una plantilla personalizada para cada paciente. El digitalizador 3D en conjunto son un software desarrollado en Java pretende generar una nube de puntos que representen la superficie del pie, para posteriormente reproducir la superficie del pie a través de triángulos mediante el uso del formato STL.

Al contar con la superficie del pie el software analizara las características de la misma para determinar si cuenta con el pie plano, en caso de ser clasificado como un pie plano el software procede a la generación del modelo 3D de la plantilla para su posterior impresión a través de una impresora 3D.

1.7.2. Limitaciones

Dentro de las limitantes nos encontramos varios factores a considerar, a continuación se mencionan los más impactantes.

Limitantes Electrónicas.

Desde el punto de vista electrónico nos encontramos con la integración de los componentes electrónicos, en este caso la cámara y el circuito de control del sistema de barrido del láser. En el desarrollo de la tesis se hace uso de una cámara web USB y de un Arduino para el control del barrido laser, lo ideal sería tener una tarjeta electrónica que manipulara el barrido láser y la adquisición de la imagen por parte de la cámara. Esto permitiría contar con un solo cable USB en lugar de dos.

Limitantes Mecánicas.

Para la construcción de la estructura se utilizó perfil de Item de 30x30mm y ángulo de 1.5 pulgadas de aluminio y cristal de 6mm, por la facilidad que ofrecen estos materiales para ser trabajados. Un producto final se esperaría una estructura de acero inoxidable para la higiene y un cristal templado el cual permita y garantice cargar el peso de una persona adulta promedio (80Kg). Para las correderas se utilizó varilla de 8mm de acero inoxidable siendo recomendable usar varillas de acero inoxidables endurecidas al cromo para mayor precisión.

Software

Para el tema del software el presente trabajo se centra en la generación de la plantilla 3D mediante un formato STL para la definición de la superficie por triángulos. No se da soporte para el modelado, edición y fabricación 3D, sin embargo se mencionan alternativas de software sugerido para dar soporte a estos puntos.

1.8. Estructura de la tesis

El presente documento se encuentra organizado en cinco capítulos, mismos que a continuación se describen:

Capítulo 1, contiene el marco metodológico que rige la investigación, definiendo el objetivo general y los objetivos específicos, así como también el planteamiento del problema que origina el presente trabajo, se detalla además los alcances y limitaciones del presente trabajo.

Capítulo 2, presenta el marco teórico y contiene todo el conocimiento, trabajos, documentos y/o conocimientos, que fundamenta los trabajos de investigación y que sirven como base para iniciar el proceso del presente trabajo.

Capítulo 3, describe cada uno de los experimentos realizados dentro de la investigación, detallando el trabajo realizado, el pre procesamiento de las imágenes.

Capítulo 4, se presentan el análisis de los resultados obtenidos en la investigación.

Capítulo 5, contiene las conclusiones a las que se llegó con la investigación y propone algunas mejoras así como lineas de investigación futuras para la investigación.

Capítulo 2

Marco teórico

2.1. Algoritmos de procesamiento

Todo sistema de visión artificial [7] (SVA) tiene como propósito la "comprensión" de las características de una imagen para facilitar y automatizar tareas que de otro modo tendrían que ser realizadas por un humano.

Para entender el concepto de visión artificial se puede utilizar una comparación con el sistema de visión humano (SVH). El SVH interpreta las imágenes basándose en reconocimiento por experiencia, por su parte el SVA utiliza instrucciones programadas por un humano para extraer características físicas y convertirlas en una cantidad medible a la cual se le puedan aplicar métodos matemáticos y lógicos para interpretarlos siguiendo pasos sucesivos, a este proceso se le conoce como algoritmo.

2.2. Calibración de la cámara

La calibración es el proceso que analiza como una cámara proyecta un objeto real en el plano de la cámara para así poder extraer información y algunas propiedades de la imagen conocida, para determinar los parámetros libres de la cámara recurriendo a modelos matemáticos para resolver las ecuaciones e incógnitas generadas en el proceso. Así mismo se extrae información que se quiere eliminar sobre la distorsión debido a la lente y a los defectos de fabricación.

Típicamente se usa un patrón de tablero de ajedrez en un plano para generar esta correspondencia de puntos del objeto real en 3D a puntos en 2D de la imagen capturada (ver Figura:2.1). Como los puntos pertenecen a una conFiguración conocida, se establecen las variables que ayudan a dar solución a los parámetros mencionados anteriormente con ayuda de otras ecuaciones que serán explicadas más adelante.

Figura 2.1: Patrón en diferentes posiciones y orientaciones

Modelo de la cámara "PINHOLE"

El modelo de la cámara "Pinhole" describe este tipo de correspondencia de 3D a 2D de manera matemática, realizando una relación de triángulos semejantes. Tomando como referencia el modelo más sencillo de una cámara, el cual describe la correspondencia de los puntos 3D a 2D que hace el proceso de calibración de manera matemática; llamado "modelo Pinhole". El cual se muestra en la Figura: 2.2

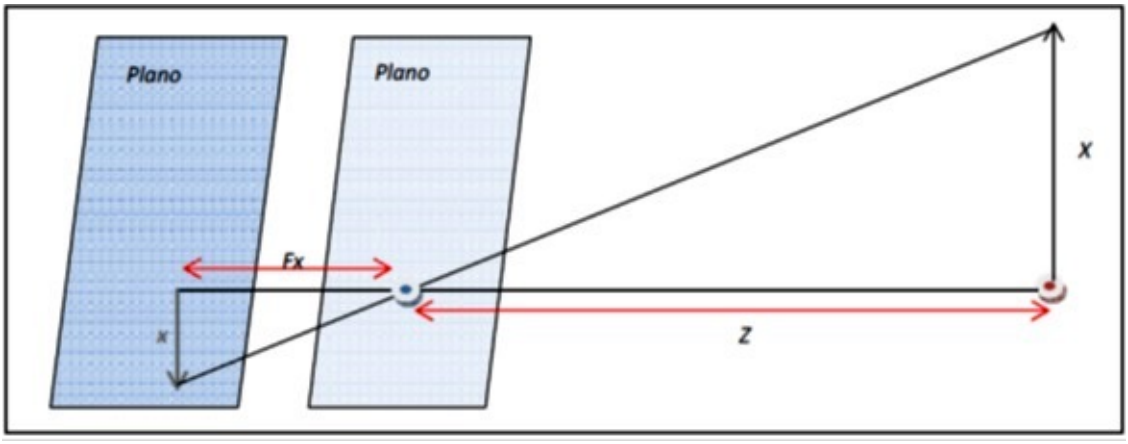


Figura 2.2: Modelo de cámara Pinhole

Utilizando semejanza entre triángulos se tiene:

$$\frac{f_x}{-X} = \frac{z}{X} \tag{2.1}$$

Dónde:

fx: Es la longitud focal de la cámara, Z : Es la distancia de la cámara al objeto, X: Es la longitud del objeto, x: Es la imagen del objeto en el plano de la imagen. Ahora, de acuerdo con Bradsky y Kaehler [8], reorganizando los parámetros de la escena del modelo "pinhole" (y haciendo que el objeto real aparezca en al lado derecho), la geometría del sistema se vuelve más fácil de analizar, porque el punto en la intersección del plano de la imagen y el eje óptico es ahora el punto principal de nuestro sistema como lo muestra la Figura 2.3.

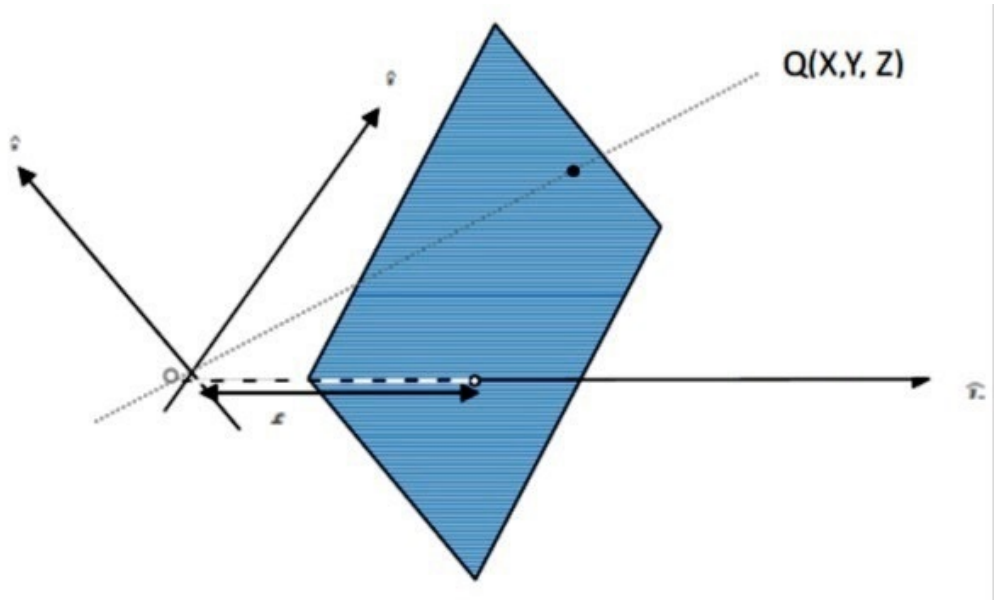


Figura 2.3: Modelo de cámara Reorganizado.

Luego del modelo anterior se deriva que un punto en el mundo físico Q con coordenadas X,Y,Z, es proyectado en la pantalla de acuerdo a las siguientes ecuaciones:

$$x_{screen} = f_x \left(\frac{X}{z}\right) + c_x \tag{2.2}$$

$$y_{screen} = f_y \left(\frac{X}{z}\right) + c_y \tag{2.3}$$

Dónde:

xscreen: Es la coordenada del punto x en la pantalla, fx : Distancia focal de la cámara en X, cx : Desplazamiento en X del centro de proyección de la pantalla, yscreen: Es la coordenada del punto y en la pantalla, fy: Distancia focal de la cámara en Y, cy: Desplazamiento en Y del centro de proyección de la pantalla. En este modelo se introducen dos longitudes focales diferentes al modelo "Pinhole", porque la medida de los pixeles en una cámara no son iguales para todos los dispositivos, estos dependen de varias características que la conforman, pero principalmente depende de la resolución de las imágenes captadas, es decir depende del

número de puntos por pulgada [9] que el sensor de la cámara es capaz de capturar del objeto escaneado.

En cámara típica de bajo costo los pixeles son rectangulares en vez de cuadrados mientras que en cámaras con mayor resolución los pixeles tenderán a ser más cuadrados y la relación de pixeles por medida tomada será mucho más acertada. Ahora bien, con las ecuaciones 2.1, 2.2 y 2.3 se definen los parámetros que la optica y geometría interna de la cámara, llamados parámetros intrínsecos. Vale la pena mencionar que estos parámetros son constantes mientras no se varíen las posiciones relativas entre la óptica y el sensor imagen. Por lo tanto la proyección de los puntos en el mundo físico Q con coordenadas X, Y, Z en el plano focal q con coordenadas x, y, z se resumen en una matriz q de 3x3 llamada M, de la siguiente manera:

$$q = MQ \tag{2.4}$$

Dónde:

$$M = \begin{bmatrix} Fx & 0 & Cx \\ 0 & Fy & Cy \\ 0 & 0 & 1 \end{bmatrix}, q = \begin{bmatrix} x \\ y \\ z \end{bmatrix} yQ = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{2.5}$$

Hasta aquí se definieron las parámetros intrínsecos de la cámara, los cuales se ocupan de la geometría interna de la cámara, ahora como se mencionó anteriormente la calibración de una cámara también desea eliminar las distorsiones de la cámara dadas debido a la lente las cuales se dan principalmente por razones de fabricación.

Distorsión Radial

Surge como resultado de la forma de la lente, lo cual hace que se distorsione la ubicación de pixeles cerca de los bordes de la imagen, generando un fenómeno de "abultamiento" o de efecto «ojo de pez» como se muestra en la Figura 2.4

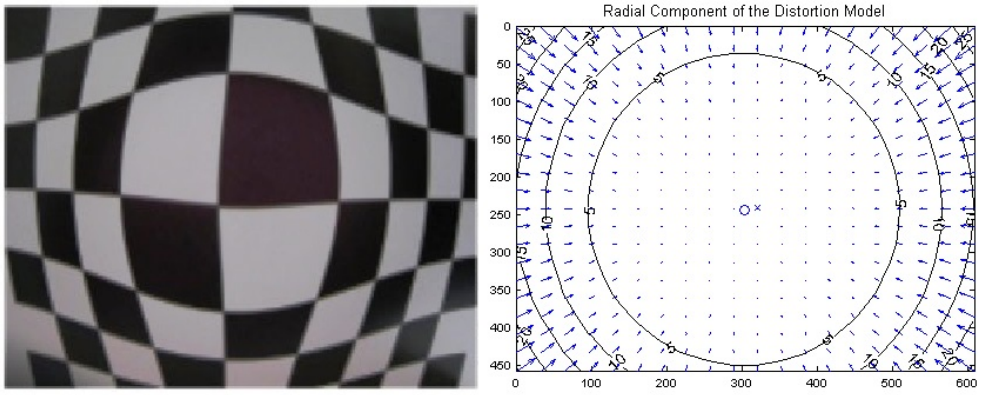


Figura 2.4: Distorsión Radial - Efecto ojo de pez

Los rayos más lejos del centro de la lente se doblan más que los del centro de la imagen, es decir que la distorsión es 0 en el centro óptico de la cámara y aumenta a medida que se avanza hacia la periferia. Esta distorsión es pequeña y puede ser caracterizada por los primeros términos de la expansión dela serie de Taylor alrededor de r=0. En donde el primer término se llama convencionalmente k1 y la segundo k2 para cámaras de baja resolución mientras que para cámaras altamente distorsionadas tales como lentes de ojo de pez se utiliza un tercer término, k3. Por lo tanto la ubicación radial de un punto en la imagen será re - escaldo de acuerdo a las ecuaciones:

$$x_{corrected} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \tag{2.6}$$

$$y_{corrected} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \tag{2.7}$$

Donde x,y es la ubicación original de los puntos distorsionados y (x corrected , y corrected) es la nueva ubicación como el resultado de la corrección.

Distorsión tangencial

Surge debido a los defectos de fabricación de la lente como a su mismo proceso de ensamblaje sobre la cámara, al no quedar totalmente paralelo a al plano imagen. Utilizando la ecuación anterior se deben agregar dos parámetros para caracterizarla:

$$x_{corrected} = x + (2p_1 y + p_2(r^2 + 2x^2)) \tag{2.8}$$

$$y_{corrected} = x + (p_1(r^2 + 2x^2) + 2p_2 x) \tag{2.9}$$

Finalmente para terminar el proceso de calibración se determinar el vector de traslación y la matriz de rotación para cada imagen que la cámara toma de un objeto particular como lo muestra la Figura 2.5 [8].

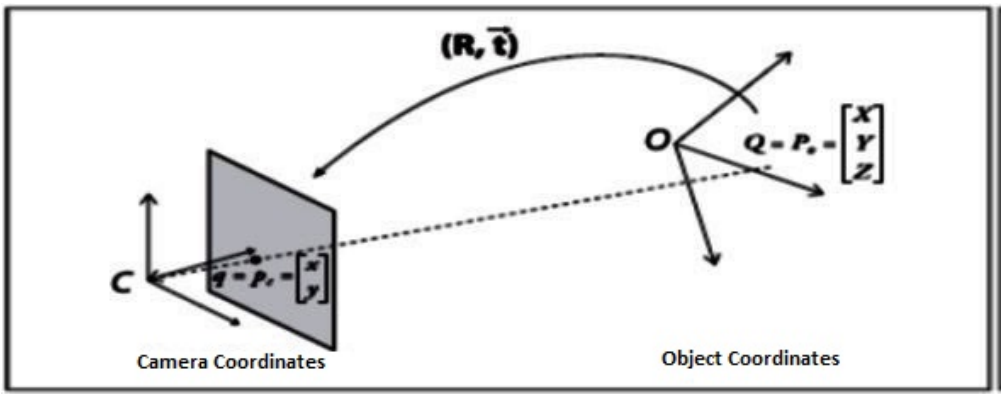


Figura 2.5: Coordenadas De La Cámara y Coordenadas Del Objeto para matriz de rotación y traslación

Lo cual se hace hallando un punto del mundo real (Po) en la cámara (Pc), de tal manera que dicha relación está dada por:

$$Pc = R(Pc - T) \tag{2.10}$$

En conclusión para la calibración se tienen 4 parámetros intrínsecos (fx; fy; cx; cy) y 5 parámetros de distorsión: que se dividen en 3 radiales (k1; k2; k3) y 2 tangenciales (p1; p2), estos últimos son agrupados en el vector de distorsión que contiene k1; k2; p1; p2 y k3, el vector de translacion (T) y la matriz de rotacion (R).

2.2.1. OpenCV

OpenCV (Open Source Computer Vision) es una librería de funciones de programación para visión por computador en tiempo real. La librera tiene más de 500 algoritmos optimizados. Se usan en todo el mundo, tiene más de dos millones de descargas y más de cuarenta mil personas en el grupo de usuarios. Es usado en campos tan variados como el arte interactivo, inspección de minas y robótica avanzada. Las metas iniciales para OpenCV eran, entre otras, mejorar la calidad de investigación en visión avanzada al brindar códigos libres y optimizados para la infraestructura de visión básica de tal manera que se permita el desarrollo de más y mejores herramientas con base en las ya existentes.

OpenCV es una colección de funciones en C y algunas clases de C++ que implementan varios algoritmos populares de procesamiento de imágenes y visión por computador. En términos generales, OpenCV brinda una amplia plataforma a un API de alto nivel que incluye cerca de 300 funciones en C y unas pocas clases en C++. Además constantemente se mejoran los vínculos Python para OpenCV; este no tiene dependencias de librerías externas[10].

Este lenguaje se estructura en cinco componentes principales, cuatro de los cuales se muestran en la Figura 2.6 [8]

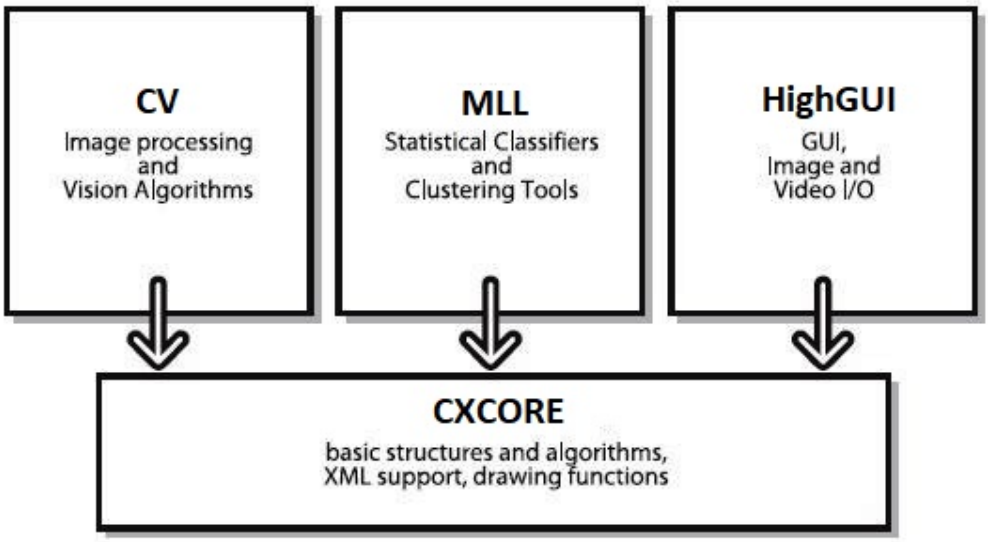


Figura 2.6: Estructura OpenCv

Donde CV es el componente que contiene el procesamiento de imagen básico y los algoritmos de visión computarizados de más alto nivel. MLL es la máquina de aprendizaje de librerías, la cual incluye varios clasificadores estadísticos y herramientas de clustering. El HighGUI contiene rutinas I/O y funciones para almacenar y cargar videos e imágenes. CXCore contiene las estructuras de datos básicos y el contenido.

Otro componente es el CvAux que contiene áreas extintas (reconocimiento facial integrado) y algoritmos experimentales (segmentación de fondo y primer plano) Algunos de estos algoritmos ofrecidos por esta gran librería son utilizados en el presente proyecto para el procesamiento de imágenes.

2.2.2. Arduino

Arduino [11] Se trata de un marco de trabajo electrónico de código abierto basado en una placa con un microcontrolador AVR. Su filosofía está orientada a un programa y equipo adaptable, simple de utilizar, Arduino se ha diseñado para ajustarse a los requisitos de todo tipo de empresas abiertas, desde principiantes hasta especialistas en tecnología mecánica o equipos electrónicos (Arduino, 2015). Antes que nada, se trata de un microcontrolador, es decir, un sistema embebido en un chip, con su CPU, memoria de programa, memoria de información y circuitos para controlar periféricos. El microcontrolador necesita para su operación de algunos circuitos auxiliares, tales como:

- Entrada de alimentación
- Fuente de reloj

- Circuito de RESET
- Medio de comunicación USB
- Puertos de entrada y salida, etc

La plataforma arduino tambien cuenta con un entorno de desarrollo, que permite interactuar de manera muy sencilla. De esta manera, se puede caracterizar como un dispositivo básico para contribuir a la creación de modelos, situaciones u objetos intuitivos para empresas multidisciplinarias.

2.3. Herramientas Computacionales

2.3.1. Hardware

A continuación se describe el hardware empleado en la presente investigación:

- LapTop Dell 5000 procesador i7, 16GB RAM, 120 GB SSD
- WebCam Logitech [12]
- Arduino Uno [11]
- Motor Nema 17 [13]
- Puntero láser
- Driver Pololu A4988 [14]

2.3.2. Software

A continuación se definen las herramientas de software utilizados para la realización del presente trabajo:

- Java 8
- OpenCV
- NetBeans IDE 8

2.4. Estado del arte

En la búsqueda de información 3D de una escena se ha convertido en uno de los principales objetivos de la visión artificial manejado por la multitud de empresas que buscan obtener beneficios de aplicar la ingeniera inversa.

En búsqueda de alcanzar este objetivo de la forma más óptima posible, como ya hemos mencionado en la introducción, se han desarrollado métodos:

- Pasivos: Parten del emparejamiento de puntos correspondientes en al menos dos imágenes.
- Activos: Dentro de estos se encuentra el de luz estructurada, se basan en la generación de características sintéticas por un emisor de luz sobre una escena carente de puntos de interés.

2.4.1. Análisis de equipos láser

Para lo cual la implicación de los equipos láser escáner 3D y las impresoras 3D al mundo de las prótesis, han permitido un gran paso en el equilibrio entre funcionalidad, comodidad y apariencia. No se trata únicamente de crear una prótesis ortopédica, sino de nuevas tecnologías que empiecen a ofrecer una ayuda inestimable a la hora de crear cualquier extensión artificial que el cuerpo requiera. El objetivo que se marcaron científicos desde hace más de cuarenta años no es una tarea sencilla. Han sido muchas las tecnologías que en búsqueda de unos resultados de calidad, han ido viendo la luz en los últimos años. Una de las primeras tecnologías resultante, nació de la necesidad por captar la escena de los mecanismos de visión artificial [7], la cual es la reconstrucción tridimensional de escenas con un par de cámaras estereoscópicas [15].

2.4.2. Clasificación de equipos láser escáner 3D

La tarea de capturar nuestro entorno que percibimos de forma visual de forma 3D es posible. Con lo cual no sería inverosímil intentar imitar a nuestro sistema de visión (dos receptores ópticos como los ojos) utilizando dos cámaras que realicen la función de nuestros ojos. Aplicando posteriormente un algoritmo que reconstruya la escena. Este proceso tiene su grado de dificultad pero es posible realizar con éxito siguiendo las etapas y técnicas apropiadas. Aunque un punto a considerar es que el ser humano puede captar su entorno en tres dimensiones gracias también a otros factores psicológicos, dentro de esta la propia experiencia.

"Esto provoca que no podamos sacar el provecho que en 1840 el descubridor de la estereoscopia Sir Charles Wheatstone habría imaginado. Aun así es el método más utilizado dentro de la tecnología de visión artificial. Ya que su carga de hardware es muy inferior a la

mayoría de dispositivos de captación 3D.

Un resumen del proceso de obtención de la imagen tridimensional mediante este mecanismo seria el siguiente:

- Calibración. Obtención de los parámetros intrínsecos y de distorsión de cada cámara en particular.
- Correspondencia. Identificación de la proyección correspondiente en la imagen contraria.
- Reconstrucción. Cálculo de la coordenada espacial a partir de la disparidad en las proyecciones.

Algunas aplicaciones e implementaciones de los digitalizadores o scanner 3D se presentan a continuación:

- Odontología, a través de las prótesis dentales.

Ofreciendo implantes dentales fabricados a medida, evitando el empleo de modelos estándar que pueden producir molestias o incluso dolores. La posibilidad de obtener modelos 3D de la mandíbula y dentadura, permite el desarrollo y posterior impresión 3D [16] de cualquier modelo de implante dental.

- Prótesis oculares y faciales.

Tradicionalmente un oftalmólogo podía crear un molde de la cara y posteriormente una prótesis ocular empleando caucho, para finalmente aplicar el color de la piel y pestañas. En este proceso la habilidad artística del modelador podía influenciar notoriamente el resultado final.

Gracias a avances como los publicados por la Academia Americana de Oftalmología (AAO) en 2014, se está comenzando a realizar pruebas como la construcción de máscaras formadas a partir de escaneados 3D y su posterior impresión 3D. La exactitud geométrica y la posibilidad de inyectar pigmentos en el material para emparejar el tono de piel a la hora de imprimir el modelo, ofrecen un avance destacable respecto a los métodos tradicionales.

Todas las novedades aportadas a estos campos poseen aspectos en común, además de la eficiencia de los modelos, el tiempo que conlleva crearlos y su coste se ven drásticamente reducidos. Casos más complejos son la creación de implantes espinales llevada a cabo en Pekín desde hace unos años. Con la ayuda de escáneres láser 3D se realiza un estudio de cada paciente y posteriormente se imprime un modelo 3D en titanio del implante. Aún más novedoso, es el proyecto encabezado por el Departamento de Defensa estadounidense, a través del Instituto de las Fuerzas Armadas de Medicina Regenerativa (Armed Forces Institute of Regenerative Medicine, AFIRM). Con la colaboración de unas 30 instituciones más

(universidades, hospitales y otros socios), llevan desde 2008 estudiando la reproducción de tejidos humanos y su posterior impresión 3D, haciendo posible por ejemplo reconstrucciones parciales y/o totales de órganos humanos.

2.4.3. Fundamentos de medida láser 3D con posibles aplicaciones ortopédicas

Por lo descrito en párrafos anteriores por el nombrado ahorro temporal y por el también mencionado al comienzo del proyecto, ahorro económico, las tecnologías más recomendables son las activas sin contacto, por ello hemos trabajado con un escáner de luz estructurada. La luz estructurada no es la única opción que encontramos dentro de esta designación de técnicas activas, por ello haremos una breve introducción a las técnicas más habituales que podemos englobar en este grupo, y que se utilizan con habitualidad en escáneres de captación 3D. Como por ejemplo [17]:

Tiempo de vuelo

El escáner de tiempo de vuelo 3D [18] emplea un láser para graduar la separación del dispositivo en cada punto del objeto. La manera de medir la distancia consiste en cronometrar el tiempo que tarda un pulso de luz emitido por el escáner en recorrer la distancia al objeto y volver. Como la velocidad de la luz es conocida (C), para obtener la distancia (D) al punto resolveremos la ecuación D=(C*T)/2, donde T es el tiempo cronometrado. Este tipo de escáner mide un punto de su campo de visión cada vez, siendo necesario mover el medidor para escanear puntos diferentes. El movimiento puede hacerse moviendo el telémetro o usando un sistema giratorio de espejos. El sistema giratorio de espejos es más eficaz pues son más ligeros y se pueden mover más rápido y con mayor precisión. Estos escáneres pueden capturar del orden de 10000 a 100000 puntos por segundo.

Triangulación

El escáner 3d de triangulación [19] es un escáner activo que usa la luz láser para examinar el objeto. En este caso el brillo del láser en el objeto se examina mediante una cámara fotográfica para determinar su posición. Dependiendo de lo lejano esté el punto del objeto en que brilla el láser, incidirá en diversos sitios del campo visual de la cámara. Esta tecnología se llama de triangulación porque el punto donde brilla el láser, el emisor láser y la cámara forma un triángulo como puede observar en la Figura 2.7. De este triángulo conocemos el lado que une la cámara con el emisor láser, el ángulo de la esquina del emisor láser también es conocido, y el ángulo de la esquina de la cámara se puede determinar examinando la localización del punto en el campo visual de la cámara. Así con estos tres valores se obtiene la forma y tamaño del triángulo formado y se determina la posición tridimensional de cada punto del objeto.

En la mayoría de los casos, en lugar de analizar un solo punto, se analiza un segmento, con lo que se acelera el proceso de captura.

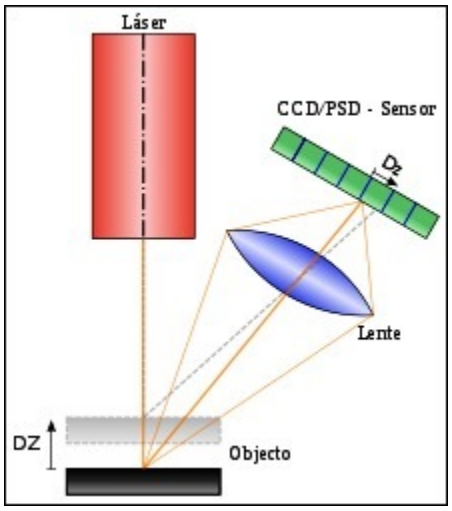


Figura 2.7: Triangulación

Respecto a los escáneres de tiempo de vuelo, los escáneres de triangulación cuentan con mayor precisión (del orden de 10 micrómetros), pero tienen un campo de acción de unos cuantos metros; mientras que los de tiempo de vuelo pueden operar en radios de acción de hasta kilómetros con precisiones del orden de milímetros.

Holografía conoscópica

Es un método de interferometría [20] que se basa en pasar un haz reflejado en una superficie a través de un traslucido birrefringente, esto puede ser un cristal con dos puntos de refracción, uno establecido y otro subordinado en el punto de frecuencia, el resultado son dos haces paralelos que están hechos para colisionar con un punto focal , esta colisión es monitoreada por un sensor CCD, la recurrencia de estas colisiones indica la posición del objeto sobre la cual se irradia el láser, como se muestra en la Figura 2.8

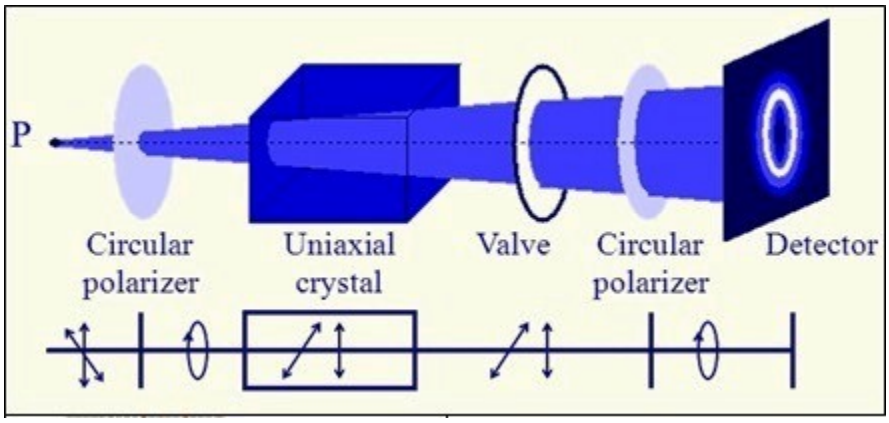


Figura 2.8: Holografía Conoscópica

Este procedimiento permite la estimación de lagunas en su colineal, llegando a correcciones mucho mejor que una micra. La ventaja de este método es que utiliza luz no coherente, lo que implica que la fuente de luz no tiene que ser láser, ya que era una condición en la que es monocromática.

Luz Estructurada

Esta tecnología emplea un patrón de luz especifico el cual puede ser una cuadricula, líneas verticales, líneas horizontales o una nube de puntos; este patrón se proyecta al objeto para analizar la deformación del patrón y en base a esa deformación obtener el modelo. El reflejo se captura con una cámara fotográfica y posteriormente mediante unos algoritmos se determina la posición de cada punto en el espacio 3d. El patrón de luz suele consistir en un conjunto de líneas paralelas generadas bien por interferencia Figura 2.9. Proyección y captación luz estructurada [21] aplicada al área de la gestión de calidad y la conservación del patrimonio histórico-artístico láser o por proyección.

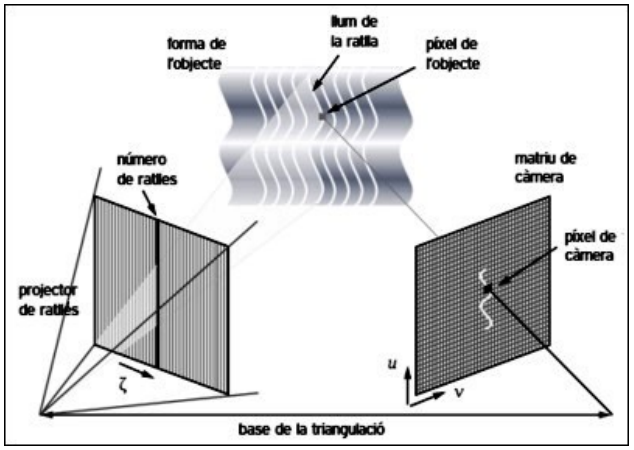


Figura 2.9: Proyección y captación de luz estructurada

En algunos casos, dos cámaras fotográficas a los lados del emisor de luz proporcionan mejores resultados. Mediante el análisis de la deformación de las líneas se obtienen los puntos 3d. La anchura de una línea es una función de la inclinación de la superficie del objeto en que se refleja. La frecuencia y la fase de la línea también aportan información, que se pueden analizar mediante la transformada de Fourier.

Como con el resto de tecnologías ópticas, este tipo de escáneres tienen problemas con las superficies transparentes y reflexivas puesto que la luz no infiere en ellas el mismo reflejo que en las opacas. Una manera de solucionar este problema es aplicando una capa fina de laca opaca a las superficies problemáticas. Una de las ventajas de los escáneres 3d de luz estructurada es la velocidad de digitalización, ya que en lugar de escanear punto por punto, son capaces de escanear toda un área en el campo de visión de la cámara, lo que reduce drásticamente la deformación reconstrucción 3d producida por el movimiento tanto del objeto como del escáner.

Capítulo 3

METODOLOGIA

3.1. Pre-procesado

El pre-procesado de imagenes [22] se encarga de aplicar todas las operaciones que se consideren necesarias sobre la imagen de entrada, con el objetivo de mejorar los resultados del sistema. Por lo tanto permite solucionar posibles problemas con la imagen de entrada lo que mejora la capacidad de adaptación del programa a variaciones en los datos con los que se trabaja. En este punto solo comentaremos algunos de los pre-procesados existentes.

3.1.1. Normalización de las imágenes

Disponemos de una serie de imágenes de tamaño MxN pixeles. Con este algoritmo se pretende ajustar las dimensiones de todas las imágenes de manera que adopten el mismo tamaño, obteniendo así uniformidad en sus dimensiones. Este pre procesado permitirá aplicar otros pre-procesados, como por ejemplo calcular la diferencia entre dos imágenes, así como facilitar el cálculo de las distancias entre imágenes.

3.1.2. Corrección de la iluminación

Este es uno de los grandes problemas a los que nos podemos enfrentar en todos los sistemas de visión por computadora. Un cambio en el nivel de iluminación de las imágenes, ya sea local o global, afectará enormemente al proceso de clasificación de las imágenes. Se pueden corregir estos problemas mediante algoritmos que compensen el cambio en la iluminación en cada una de las imágenes. Un sistema que se podría utilizar con el fin de corregir la iluminación seria comprobar las imágenes una a una y variar el nivel de iluminación en las que fuese necesario para conseguir más uniformidad.

3.2. Triangulación láser

Como primer investigación, se realizó el cálculo de distancia a través del método de triangulación. El método consiste en alinear una cámara web y un puntero láser, utilizando Java como plataforma de programación, se procesa la imagen obtenida por la cámara web y se aplica el método de triangulación para calcular a que distancia se encuentra el objeto. Para calcular la distancia entre la cámara y el objeto podemos disponer la cámara y el láser como se muestra en la Figura 3.1.

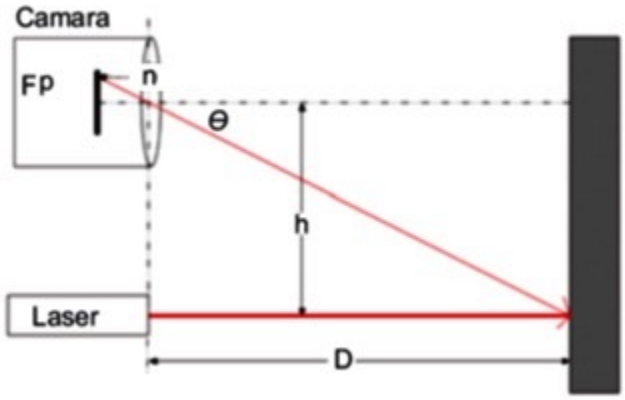


Figura 3.1: Alineación de camara y laser

Dónde: n= número de pixeles del centro al plano focal h= separación entre el láser y el centro de la cámara θ = ángulo formado entre el centro del plano focal de la cámara y el punto de incidencia del láser. D= distancia entre el objeto y la cámara.

Si conocemos h y conocemos θ por trigonometría podemos calcular la distancia:

$$D = \frac{h}{tan\theta} \tag{3.1}$$

Este es un problema muy fácil de resolver, de manera teórica. Pero en realidad cunado se construye un modelo con este diagrama, lograr un perfecto paralelismo entre el láser y la cámara es una tarea difícil de realizar de manera manual. Un modelo más realista es el presentado en la Figura 3.2

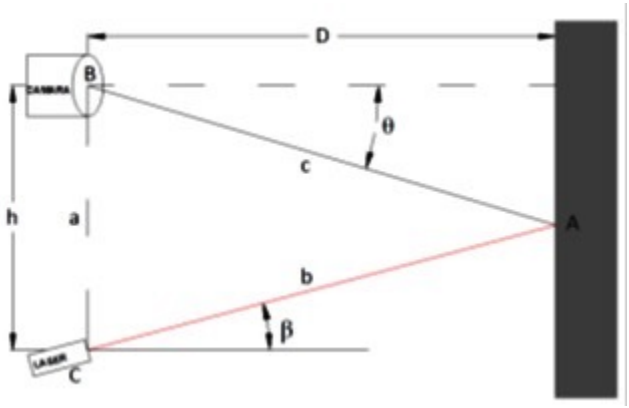


Figura 3.2: Modeolo real en 2D

El ángulo β es un ángulo no deseado que se encuentra de manera intrínseca en el ensamble de la cámara con el láser. β es una constante que se puede medir y determinar con relativa facilidad. Esta variable adicional cambia el modelo de comportamiento de la función para determinar la distancia. Matemáticamente es posible establecer más de una solución para este problema.

Analizando las variables observamos que conocemos las siguientes: h, θ, β.

Una propuesta para encontrar D es conocer el segmento c y multiplicarlo por el coseno de θ, para lograrlo primero debemos calcular c. Por ley de senos podemos establecer la siguiente relación:

$$\frac{senA}{a} = \frac{senC}{c} \tag{3.2}$$

Despejando c obtenemos la siguiente ecuación:

$$c = \frac{a * senC}{senA} \tag{3.3}$$

El ángulo C podemos determinarlo usando el ángulo β de la siguiente manera:

$$c = 90° - \beta \tag{3.4}$$

De igual modo podemos encontrar el ángulo B:

$$B = 90° - \theta \tag{3.5}$$

Recordando que la suma de los ángulos internos de un triángulo siempre es igual a 180° podemos encontrar el ángulo A:

$$A = 180° - B - C \tag{3.6}$$

Reemplazando las ecuaciones 3.4 y 3.5 en 3.6 obtenemos:

$$A = 180° - (90° - \theta) - (90° - \beta) = \theta + \beta \tag{3.7}$$

Tomando en cuenta que h es igual al segmento a, reemplazamos las ecuaciones 3.7 y 3.4 en 3.3 para obtener:

$$c = \frac{h * sen(90° - \beta)}{sen(\theta + \beta)} \tag{3.8}$$

A partir de c podemos calcular fácilmente D si conocemos θ

$$D = c * cos\theta \tag{3.9}$$

Tomando la ecuación 3.7 y reemplazando en 3.9 obtenemos la ecuación completa para encontrar D:

$$D = \frac{h * cos\theta * sen(90° - \beta)}{sen(\theta + \beta)} \tag{3.10}$$

Ahora tenemos que determinar el ángulo θ:

$$\theta = gp * n + d \tag{3.11}$$

$$gp = \frac{\alpha}{ph} \tag{3.12}$$

Dónde: gp = grados por pixel n = número de pixeles del centro al plano focal d = desplazamiento en grados α = ángulo de visión horizontal de la cámara ph = pixeles horizontales de la cámara

La primera etapa consiste en medir la distancia en pixeles de un objeto conocido. Para esto el software solicita al usuario realizar un click sobre el inicio del objeto conocido y otro sobre el fin del objeto conocido. Al dar dos click cuyas coordenadas son almacenadas. Luego se determina la distancia en píxeles simplemente dividiendo la medida de referencia entre el número de pixeles entre los dos puntos almacenados. Finalmente se almacena dicho valor en una variable.

Para comprobar el resultado de las ecuaciones deducidas se desarrolló un programa en Java donde se programaron las ecuaciones para monitoreo de los resultados en tiempo real. El programa muestra una línea vertical color azul que indica la mitad de la cámara, una cruz color rojo que ubica la posición del punto láser en la imagen. En la Figura 3.3 se muestra la pantalla principal del programa desarrollado. El programa permite calcular la distancia de acuerdo a los coeficientes indicados en la sección de variables.

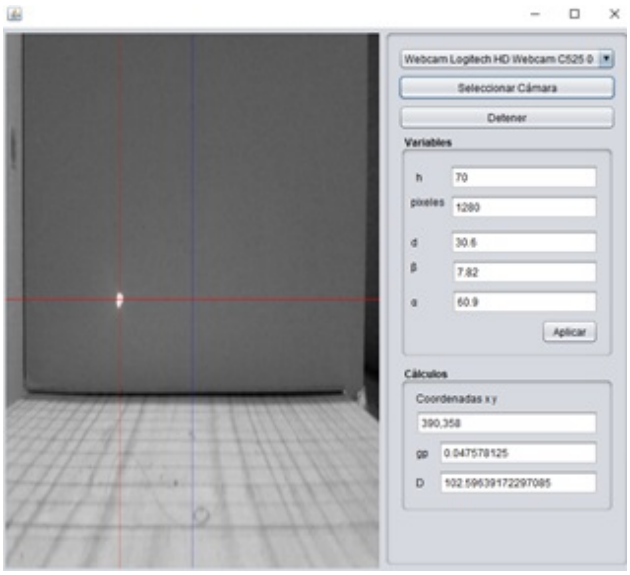


Figura 3.3: Ventana principal del programa de calculo de distancia por triangulación

En la Tabla 3.1 se presentan los datos obtenidos en el experimento. La primera columna muestra la posición en pixeles, θ representa el angulo del láser, la siguiente columna indica la distancia calculada en milímetros y finalmente el error relativo del cálculo.

n	Distancia real	θ	Distancia calculada	Error
69	107	27.31711	107.0573	0.05 %
123	117	24.74789	116.9981	0.00 %
175	127	22.27383	127.9859	0.77 %
220	137	20.13281	138.9064	1.37 %
259	147	18.27727	149.6952	1.80 %
285	157	17.04023	157.7154	0.45 %
319	167	15.42258	169.4059	1.42 %
342	177	14.32828	178.2253	0.69 %
365	187	13.23398	187.9132	0.49 %
388	197	12.13969	198.6127	0.81 %

Tabla 3.1: Resultados de las mediciones

La Figura 3.4 muestra el comportamiento del ángulo θ contra la distancia estimada. El angulo θ se representa en el eje de las abscisas, la distancia estimada se encuentra en el eje de las ordenadas. Como se puede observar a mayor distancia la resolución de θ se reduce, lo que implica menor detalle en el calculo de la distancia.

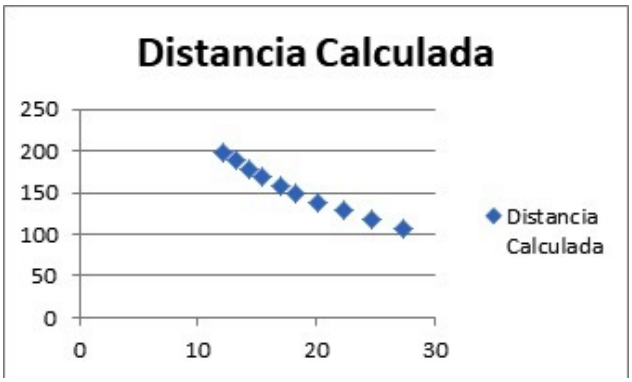


Figura 3.4: Gráfica de comportamiento

3.2.1. Función de calibración

Una vez encontrado el modelo matemático del cálculo de distancia a través del método de triangulación, podemos analizar su comportamiento. Esto con la intención de encontrar una función de calibración más simple. El principal reto al que nos enfrentamos en un sistema de visión es la calibración. Como se demostró anteriormente, para calcular la distancia de un objeto utilizando la triangulación láser, es necesario conocer el ángulo β y la distancia h.

Numéricamente hablando resulta una ecuación muy simple, sin embargo, calcular este par de constantes de manera efectiva resulta complicado, por lo que se exploran alternativas más rápidas y efectivas. Primero analizamos el comportamiento del modelo matemático, para lograrlo basta con tabular y graficar el resultado. Para la tabulación se utilizaron las constantes del punto anterior, la Tabla 3.2 muestra la tabulación de la ecuación 3.10.

n	θ	Distancia calculada
69	27.3171	107.0573
123	24.7479	116.9981
177	22.1787	128.4411
231	19.6095	141.8127
285	17.0402	157.7154
339	14.4710	177.0284
393	11.9018	201.089
447	9.3326	232.0348
501	6.7634	273.5092
555	4.1941	332.2712

Tabla 3.2: Tabulación de ecuación 3.10

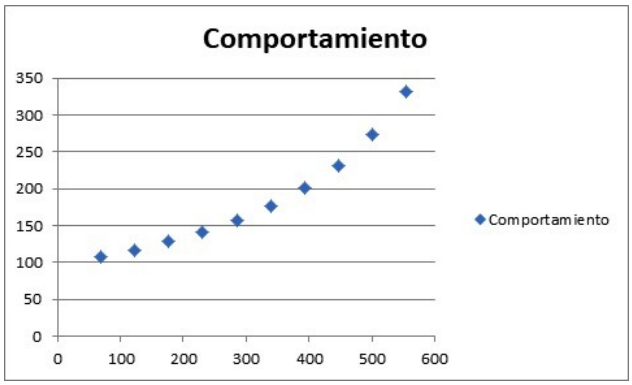


Figura 3.5: Grafica de comportamiento de distancia en funcion de pixeles

Como se puede observar en la Figura 3.5, es una función trascendental, por lo que realizar una regresión no será efectivo. Por otro lado estamos observando el comportamiento general, si acotamos el rango de medición podemos obtener un comportamiento en un entorno controlado. Para los fines de este trabajo las distancias a medir se encuentran en un rango de 0 a 10 milímetros. Procedemos a tabular con valores aproximados de 0 a 10mm. (Tabla 3.3)

n	θ	Distancia calculada
69	27.3171094	107.057283
75	27.0316406	108.098075
81	26.7461719	109.153869
87	26.4607031	110.225044
93	26.1752344	111.311993
99	25.8897656	112.415122
105	25.6042969	113.534852
111	25.3188281	114.671617
117	25.0333594	115.825866
123	24.7478906	116.998064

Tabla 3.3: Tabulación de ecuación 3.10 en un rango de 10mm

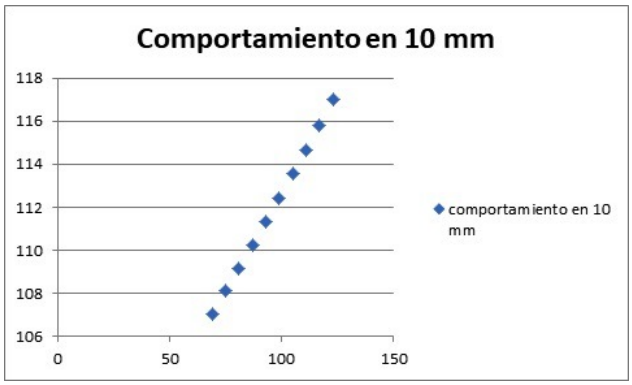


Figura 3.6: Grafica de comportamiento a 10mm

Como se puede observar en la Figura 3.6, la función presenta un comportamiento lineal. Para comprobarlo procedemos a realizar tres regresiones, una lineal Figura 3.7, una cuadrática Figura 3.8 y una exponencial Figura 3.9.

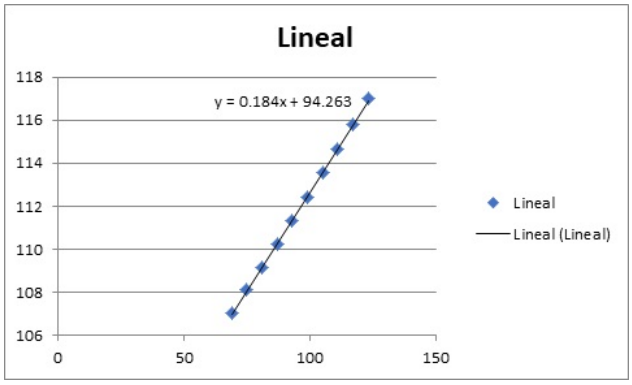


Figura 3.7: Regresión Lineal

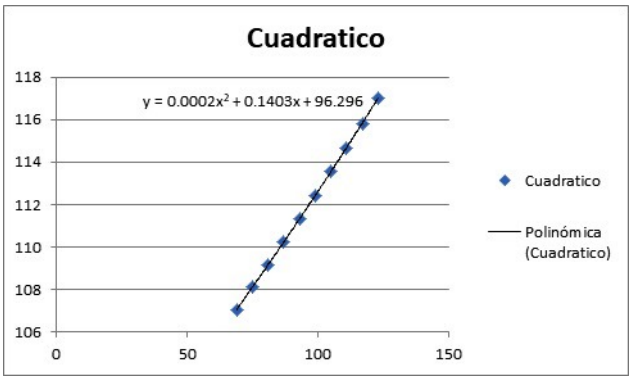


Figura 3.8: Regresión Cuadratica

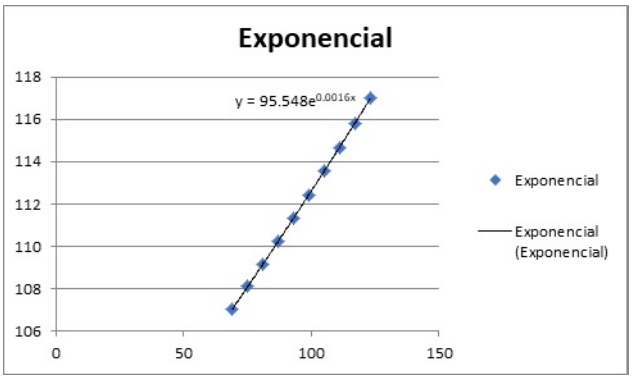


Figura 3.9: Regresión Exponencial

Visualmente no se aprecia diferencia en las regresiones. Para encontrar la regresión de mejor performance se realiza una Tabla comparativa con los porcentajes de error.

n	Distancia calculada	Lineal	Cuadratica	Exponencial	Error Lineal	Error Cuadratico	Error exponencial
69	107.057283	106.959	106.9289	106.7008089	-0.0918 %	-0.1199 %	-0.3330 %
75	108.098075	108.063	107.9435	107.7300692	-0.0324 %	-0.1430 %	-0.3404 %
81	109.153869	109.167	108.9725	108.769258	0.0120 %	-0.1662 %	-0.3524 %
87	110.225044	110.271	110.0159	109.818471	0.0417 %	-0.1897 %	-0.3689 %
93	111.311993	111.375	111.0737	110.877805	0.0566 %	-0.2141 %	-0.3901 %
99	112.415122	112.479	112.1459	111.9473576	0.0568 %	-0.2395 %	-0.4161 %
105	113.534852	113.583	113.2325	113.0272273	0.0424 %	-0.2663 %	-0.4471 %
111	114.671617	114.687	114.3335	114.1175137	0.0134 %	-0.2949 %	-0.4832 %
117	115.825866	115.791	115.4489	115.2183172	-0.0301 %	-0.3255 %	-0.5245 %
123	116.998064	116.895	116.5787	116.3297393	-0.0881 %	-0.3584 %	-0.5712 %

Tabla 3.4: Tabla comparativa de regresiones

Como se puede observar en la Tabla 3.4 el error relativo es menor para la regresión lineal. Si tabulamos y graficamos los errores absolutos obtenemos los resultados de la Tabla 3.5.

n	Distancia calculada	Lineal	Cuadratico	Exponencial	Error lineal	Error cuadratico	Error exponencial
69	107.057283	106.959	106.9289	106.7008	0.10	0.13	0.3565
75	108.098075	108.063	107.9435	107.7301	0.04	0.15	0.3680
81	109.153869	109.167	108.9725	108.7693	-0.01	0.18	0.3846
87	110.225044	110.271	110.0159	109.8185	-0.05	0.21	0.4066
93	111.311993	111.375	111.0737	110.8778	-0.06	0.24	0.4342
99	112.415122	112.479	112.1459	111.9474	-0.06	0.27	0.4678
105	113.534852	113.583	113.2325	113.0272	-0.05	0.30	0.5076
111	114.671617	114.687	114.3335	114.1175	-0.02	0.34	0.5541
117	115.825866	115.791	115.4489	115.2183	0.03	0.38	0.6075
123	116.998064	116.895	116.5787	116.3297	0.10	0.42	0.6683

Tabla 3.5: Tabla comparativa de regresiones con errores absolutos en mm

Podemos notar como el error absoluto es menor a un milímetro para cualquiera de las regresiones. Para la regresión lineal el error es de una décima de milímetro en el peor de los casos, la Figura 3.10 compara el comportamiento de las diferentes regresiones analizadas. Considerando que el modelo lineal puede ser calculado con solo dos puntos, este es el modelo que seleccionaremos. Hay que considerar que el modelo lineal puede ser adoptado debido a que solo vamos a operar en una región.

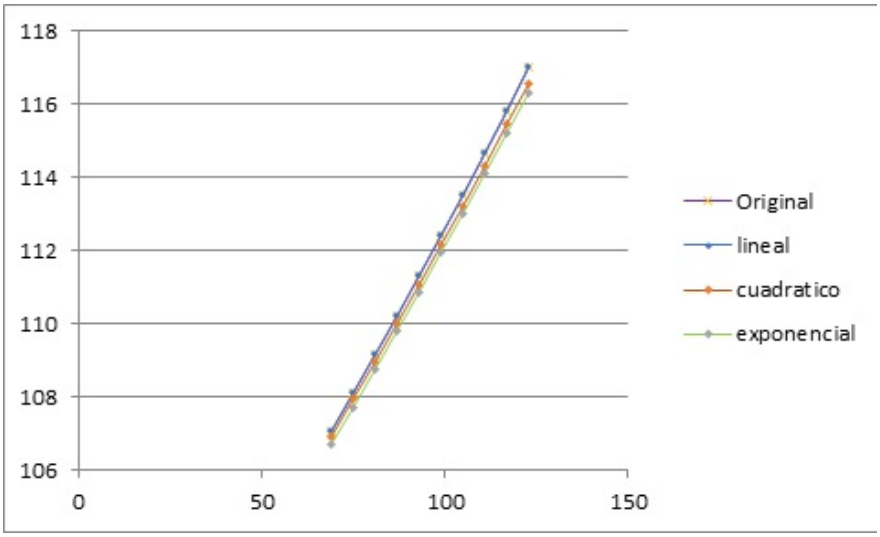


Figura 3.10: Grafica comparativa

3.2.2. Generar una línea láser

Para generar una línea láser se empleó un puntero lineal, el cual se muestra a en la Figura 3.11.

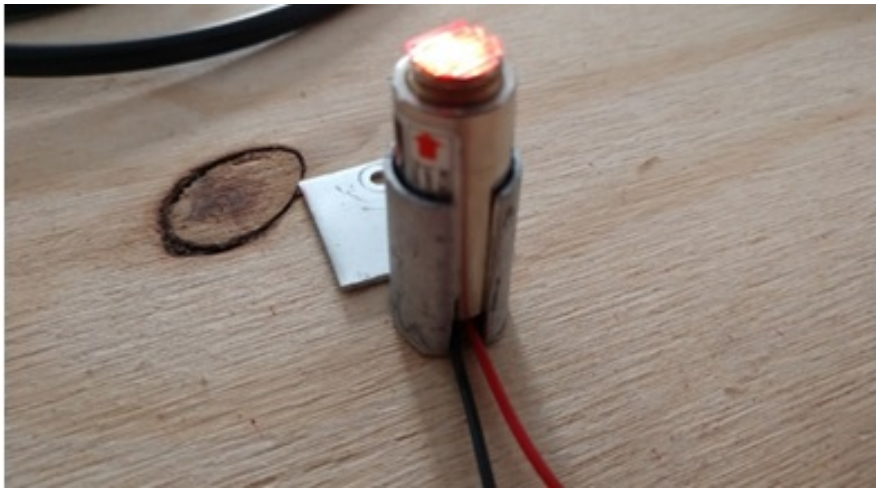


Figura 3.11: Puntero Láser Original

El difusor del puntero el cual es el encargado de generar el patrón de línea, fue reemplazado por pequeños tramos de fibra óptica, los cuales permitieron ampliar el ángulo de proyección de la línea láser, la Figura 3.12a) muestra el difusor original en forma de triangulos y la Figura 3.12b) el difusor cilíndrico propuesto construido a partir de fibra optica. Como se puede observar el difusor permite un mayor angulo de proyección, el cual permite colocar el puntero láser a una menor distancia de la plataforma de digitalización

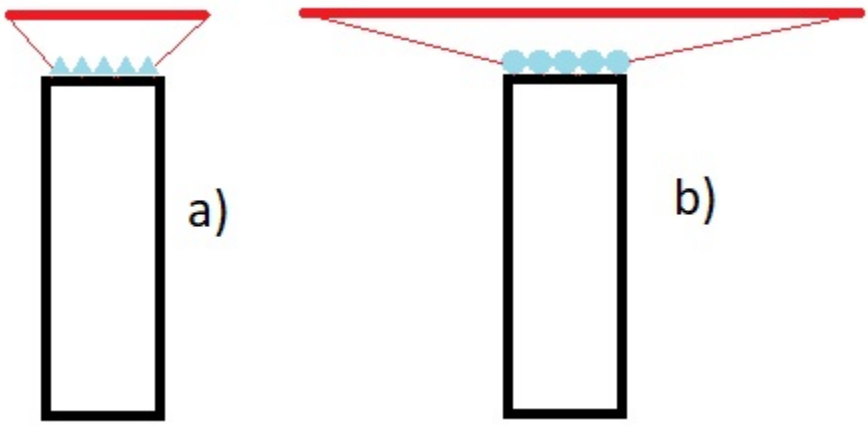


Figura 3.12: a) difusor original, b) difusor modificado

La Figura 3.13 muestra la construcción del difusor a partir de la fibra óptica, mientras que la Figura 3.14 muestra el ensamble completo del difusor.

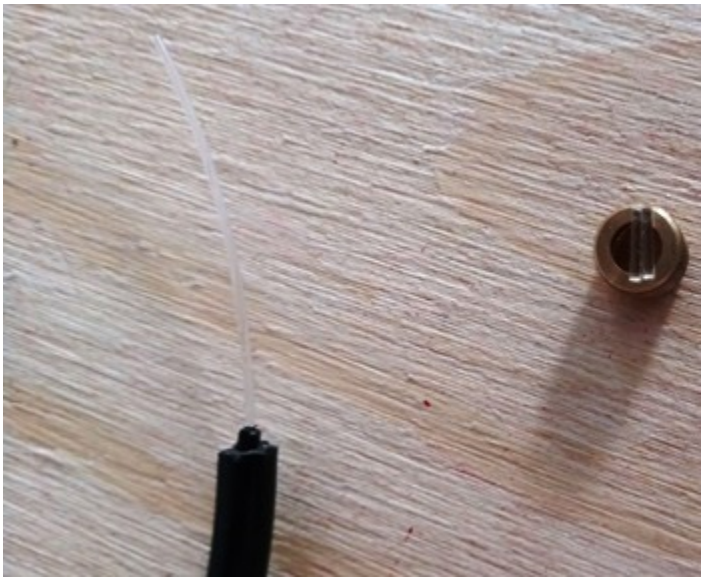


Figura 3.13: Construcción del difusor con fibra óptica

Figura 3.14: Difusor completo

3.3. Conexiones eléctricas del prototipo

Para el control del prototipo se utilizaron los siguientes materiales:

1 Motor a pasos nem17 de 200 pasos por revolución. El motor a pasos será el encargado de desplazar el eje móvil para escaneo de la superficie del pie.

1 Driver Pololu A4988. Se trata de un controlador de micropasos para motor a pasos, este permite aumentar la resolución de los pasos del motor, multiplicando por 16 el total de pasos por revolución. También tiene la función de acoplar los niveles de voltaje y corriente que requiere el motor para trabajar.

1 Arduino Uno. Tiene la función de servir como control del motor a pasos y del láser, permite comunicar el programa de digitalización con el prototipo de digitalizador.

1 Puntero láser. Genera un haz láser en forma de línea, el cual permitirá determinar la profundidad de acuerdo a la deformación de la línea.

1 Interruptor de final de carrera. Funciona como referencia para encontrar el origen del eje móvil. Los componentes se conectan de la siguiente manera:

En la Figura 3.15 se muestra el esquema de conexiones eléctricas utilizado para el prototipo del digitalizador.

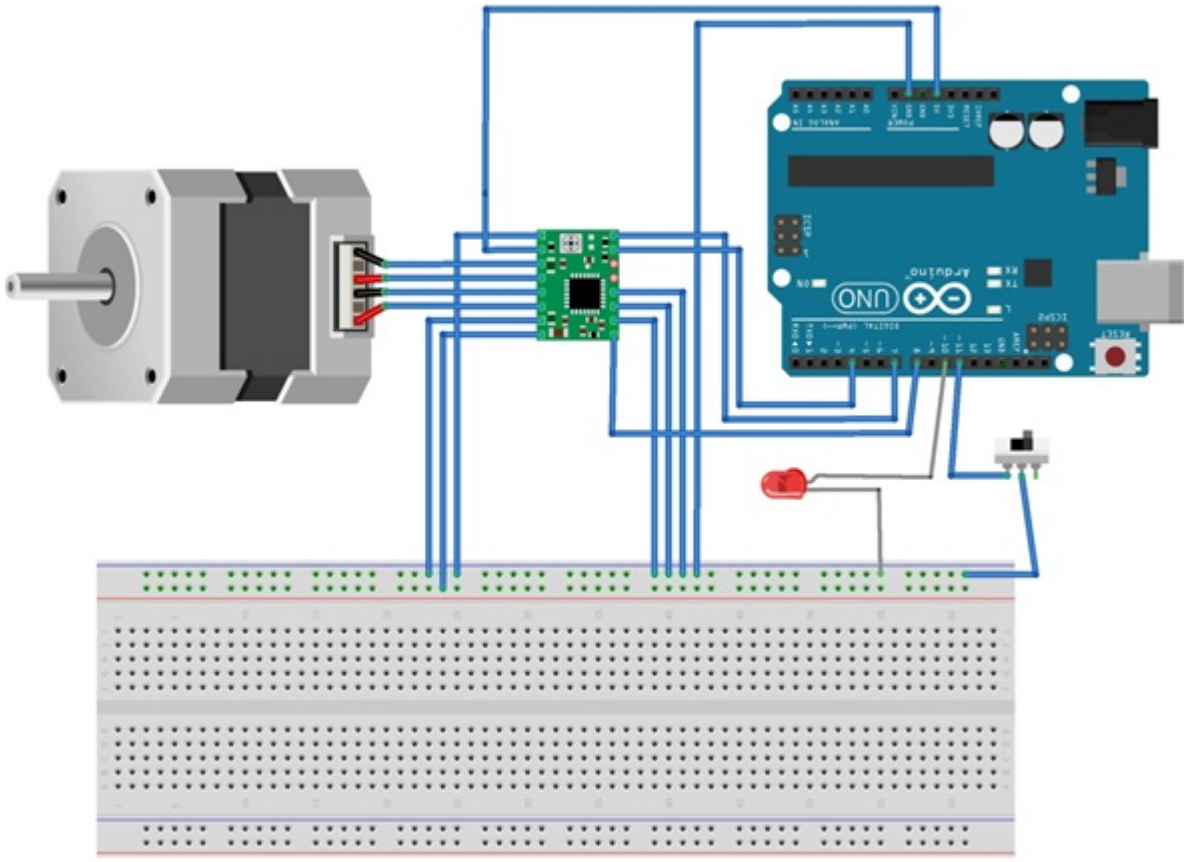


Figura 3.15: Esquema eléctrico

3.4. Lógica del programa para Arduino

La Figura 3.16 muestra el diagrama de flujo del programa que ejecuta la tarjeta Arduino Uno. El Arduino se encarga de controlar el eje móvil, el puntero láser y la retro iluminación led, en el anexo 5.1 se detalla el programa escrito en C para la tarjeta Arduino.

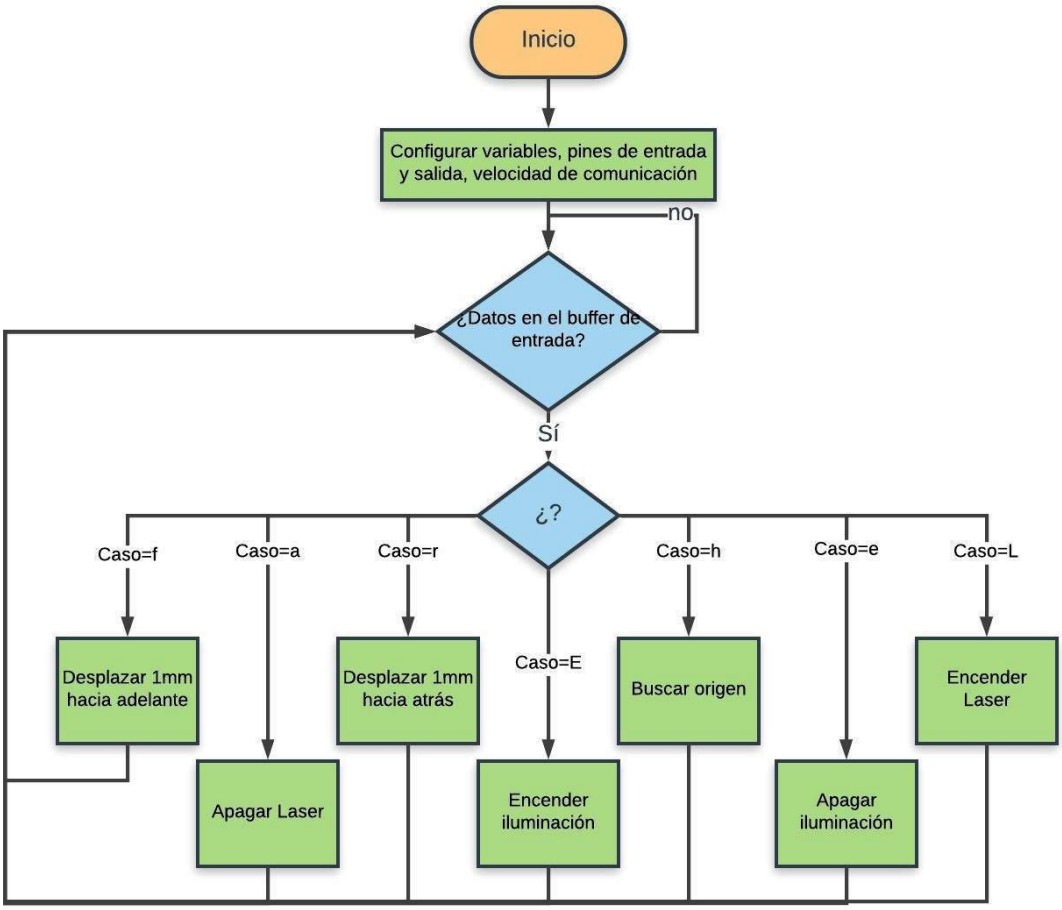


Figura 3.16: Diagrama de Flujo del programa en Arduino

3.5. Determinar pasos por milímetro para el motor a pasos para el eje móvil

Para la construcción del prototipo se empleó un motor a pasos bifásico, tipo NEMA17 de 200 pasos por revolución. Para controlarlo se utilizó un controlador de micropasos modelo A4988 de 16 micropasos. El cálculo de la resolución del motor se realiza de la siguiente manera:

Resolución= pasos x micropasos= 200 x 16=3200 pasos por revolución.

Dado que el eje móvil tendrá una resolución de 1 milímetro, es necesario calcular cuántos impulsos se requieren para desplazar el eje un milímetro.

El eje móvil se encuentra conectado a una polea dentada de 20 dientes con un diámetro de 12.73 mm, tal como se muestra en la Figura 3.17

Figura 3.17: Diámetro de polea

Considerando el diámetro de la polea, podemos calcular el desplazamiento lineal del eje con respecto a una revolución del motor, dicho cálculo basta con encontrar la circunferencia de la polea:

$$Circunferencia = \pi x D = 3{,}1416x12{,}73mm \approx 40mm \tag{3.13}$$

Una vez conocido el desplazamiento lineal por revolución, podemos calcular cuántos micro pasos (ppr) requiere el motor para desplazarse un milímetro.

$$imm = \frac{3200ppr}{40mm} = 80ppr \tag{3.14}$$

Con los datos calculados se procede a desarrollar el programa en Arduino. (Ver Anexo 5.1)

3.6. Ensamble del prototipo de digitalizador 3D para pies

Para construir el prototipo de digitalizador 3D se utilizó perfil de aluminio extruido tipo bosh de 30x30mm, ángulo de aluminio de 1.5 pulgadas, varilla lisa de 8 mm como corredera, el prototipo se ensamblo como se muestra en la Figura 3.18.

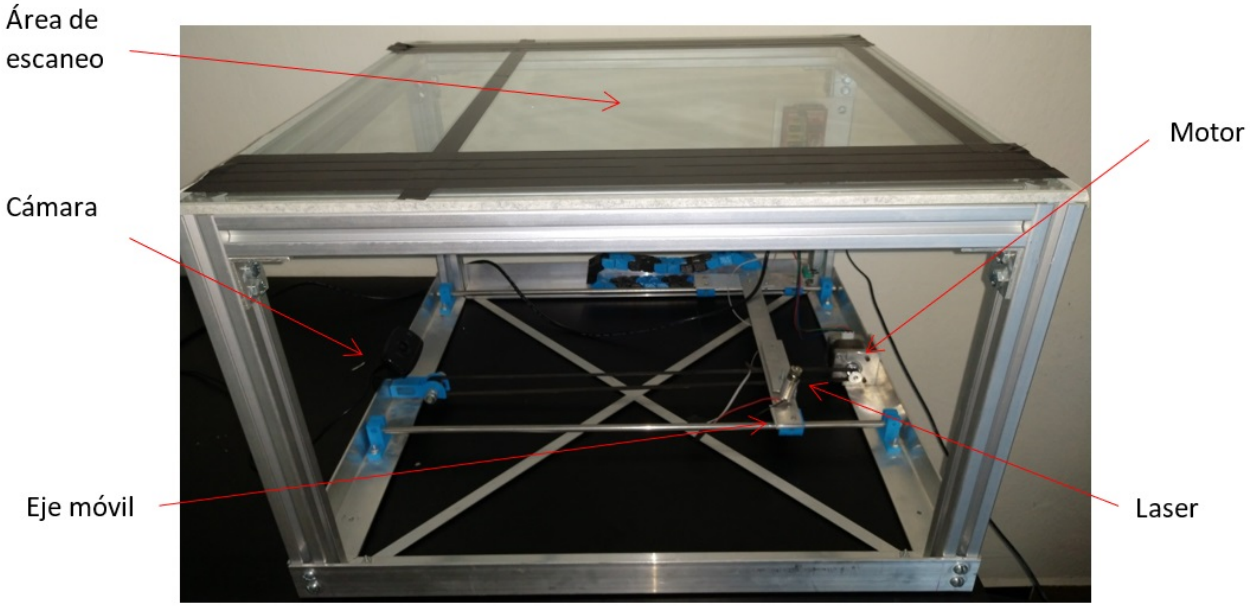


Figura 3.18: Ensamble de prototipo

3.7. Calibración del Digitalizador

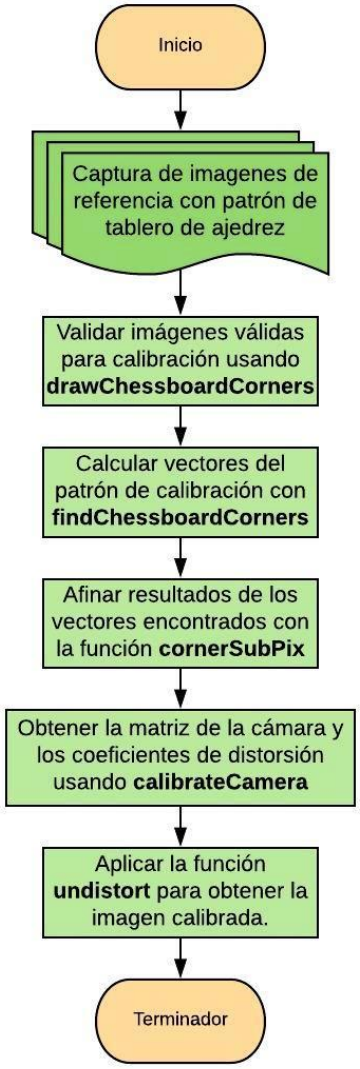
La calibración del digitalizador se realiza en cuatro etapas:

1. Calibración de la cámara
2. Corrección de perspectiva
3. Procesamiento de imagen
4. Polinomio de colocación para cálculo de profundidad

3.7.1. Calibración de la cámara

Consiste en corregir errores en la fabricación y construcción de la cámara, corrige los defectos radiales y tangenciales descritos anteriormente. Para la calibración se hace uso de la función findChessboardCorners de la librería de opencv para Java, la cual se encuentra

contenida en el paquete Calib3d. El procedimiento para hacer uso de dicha función se describe a continuación: (Ver Anexo 5.2)



3.7.2. Corrección de Perspectiva

Cuando se obtiene una foto estamos viendo una representación bidimensional de una escena tridimensional. La perspectiva permite convertir una escena de tres dimensiones en una de dos. Se tienen identificados cuatro tipos de perspectiva, en nuestro caso nos centraremos en la perspectiva Lineal, este tipo de perspectiva las líneas paralelas parecen converger en un punto en el horizonte llamado Punto de fuga. La sensación de profundidad es absolutamente engañosa; en cualquier caso, esta sensación es un método de composición realmente crítico.

La forma en la que se encuentra colocada la cámara con respecto al área de digitalización permite observar el desface del haz láser, sin embargo la imagen obtenida presenta una perspectiva lineal la cual debe ser compenzada antes de realizar cálculos sobre la imagen. La Figura 3.19 muestra la imagen captada por la cámara.

Figura 3.19: Imagen original sin sorreccion de perspectiva

Para convertir la imagen con perspectiva lineal a una imagen plana sin perspectiva, se emplea la transformación de perspectiva, indicando 4 puntos de referencia y haciendo uso del medodo getPerspectiveTransform() de OpenCV para obtener la matriz y warpPerspective() para aplicar la transformación. (Ver anexo 5.3)

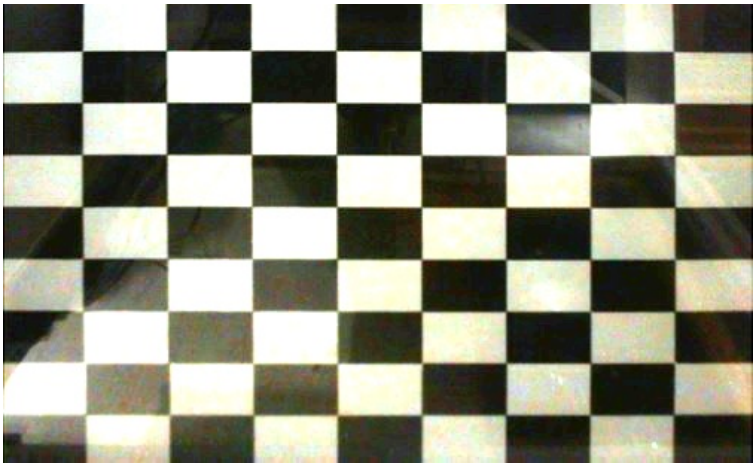


Figura 3.20: Imagen con perspectiva lineal corregida

Como se puede observar en la Figura 3.20 las lineas del patron de ajedrez se aprecian totalmente paralelas al marco de la imagen.

3.7.3. Pre-procesamiento de imagen

El procesamiento de la imagen se realizó en cinco pasos:

- Realizar filtro gausiano
- Filtrar color del haz laser
- Aplicar mascara a imagen original
- Binarizar imagen
- Generar linea de un pixel de ancho

Realizar filtro gausiano

El primer paso consiste en realizar un filtro gaussiano [23] a la imagen, con el objetivo de eliminar los saltos bruscos de color de los pixels, en la Figura 3.21 se muesta la sección de arriba la imagen original si el filtro gaussiano, en la sección de abajo se puede observa el resultado de aplicar el filtro gaussiano.

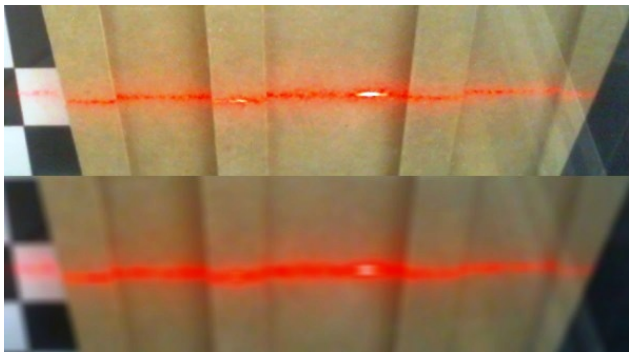


Figura 3.21: Comparación entre imagen normal e imagen con filtro gaussiano

Filtrar color del haz laser

Para lograr un filtrado correcto de color, es necesario seleccionar un espacio de color [24] que facilite las operacioes de filtrado.Un espacio de color se define como un sistema de interpretación de color. Depende del modelo de color en combinación con los dispositivos físicos que permiten las representaciones de color. Un espacio de color puede ser arbitrario, con colores particulares asignados según el sistema y estructura matemática.

Un modelo de color se describe como un modelo matemático abstracto que establece la forma en la que los colores pueden representarse, comúnmente se representan como la combinación lineal de tres o cuatro valores o componentes de color. De manera nativa la cámara utilizada representa el color en formato RGB, para lo cual hace uso de tres byte, uno para cada color, la Figura 3.22 muestra la representación del modelo RGB.

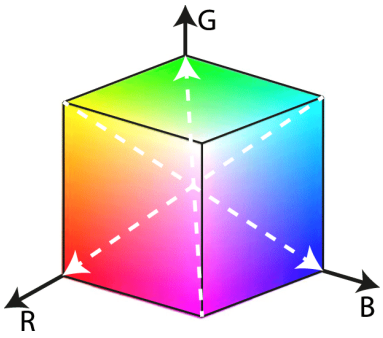


Figura 3.22: Modelo RGB

Para realizar el filtrado del haz láser se procede a transformar el espacio de color RGB a un espacio HSV. HSV consta de los componentes: H-matiz (hue), S-saturación, V-valor de intensidad. Se trata de una transformación no lineal del espacio RGB, la Figura 3.23 muestra la representación del espacio HSV.

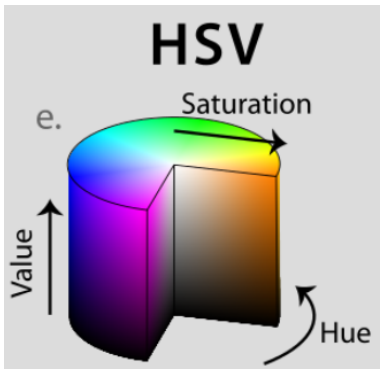


Figura 3.23: Modelo HSV

El espacio HSV permite identificar un color particular usando un solo valor, el tono , en lugar de tres valores. En el caso de OpenCV H tiene valores de 0 a 180, S y V de 0 a 255. El color rojo , en OpenCV , tiene los valores de matiz aproximadamente en el rango de 0 a 10 y de 160 a 180. La Figura 3.24 muestra la imagen original y la imagen filtrada.

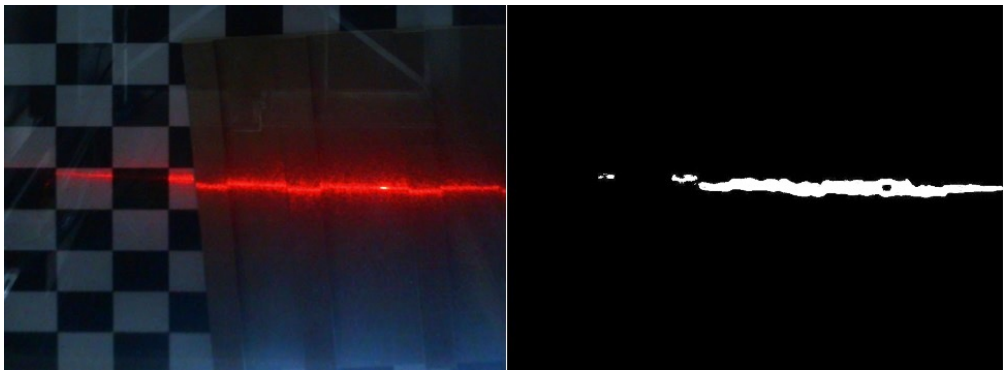


Figura 3.24: Imagen antes de filtrar - Imagen filtrada

Como se puede observar en la Figura 3.24 el filtro produce algunos vacios, por lo cual se aplica la operacion de cerrado de imagen. La Figura 3.25 muestra el resultado despues de cerrar la imagen.

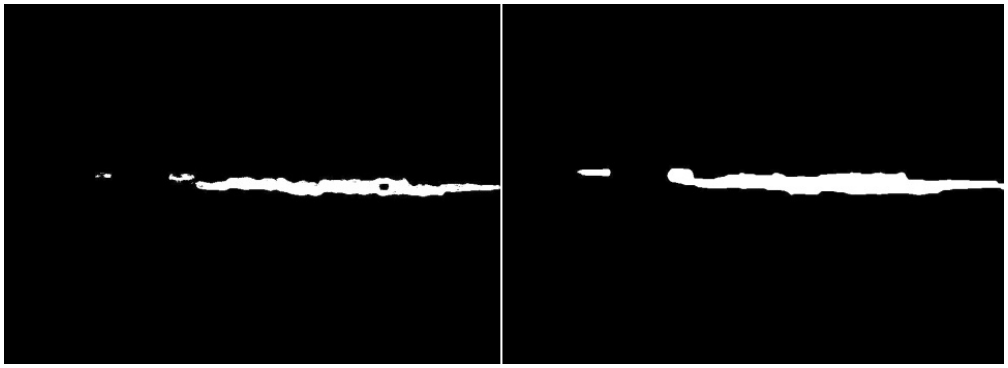


Figura 3.25: Imagen sin cerrado-Imagen con operacion de cerrado

Aplicar mascara a imagen original

Una ves obtenida la mascara con la región donde se encuentra el haz láser, se procede a aplicar la mascara a la imagen original para eliminar el ruido de la imagen. La Figura 3.26 muestra como se aplica la mascara a la imagen original.

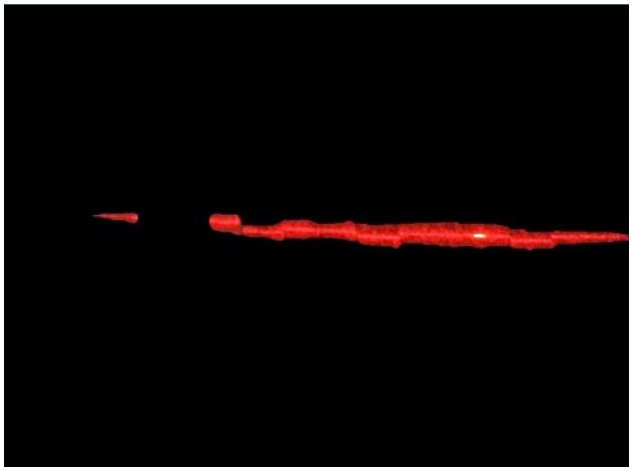


Figura 3.26: Imagen original con mascara

Binarizar imagen

Antes de binarizar la imagen [25] con la mascara, procedemos a convertir la imagen RGB a una imagen en escala de grises, una ves en escala de grises se realiza una equilización de la imagen para estandarizar la el brillo de la imagen, una vez ecualizada la imagen se aplica la binarización ajustando los umbrales para solo considerar los pixeles mas brillantes. La Figura 3.27 muestra la imagen en escala de grises y el resultado de la binarización.

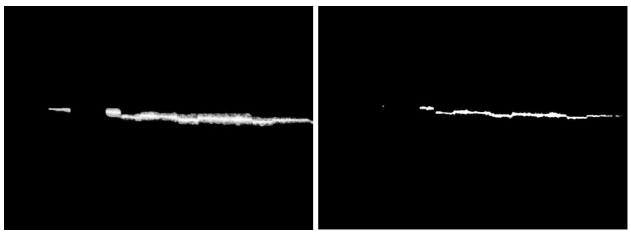


Figura 3.27: Escala de grises - Binarizado

Generar linea de un pixel de ancho

Como se puede observar en la imagen 3.27 la representación de la linea ocupa mas de un pixel de ancho, antes de realizar cálculos con la imagen filtrada es necesario generar una linea de un solo pixel de ancho. Se sabe la mayor concentración de luz pasa por el centro del área detectada, por lo que se procede a desarrollar un algoritmo de euristica propia capas de encontrar el cento de la linea laser. El método se describe en el anexo 5.4, la Figura 3.28 muestra el resultado de aplicar el algoritmo onePixel() a la imagen.

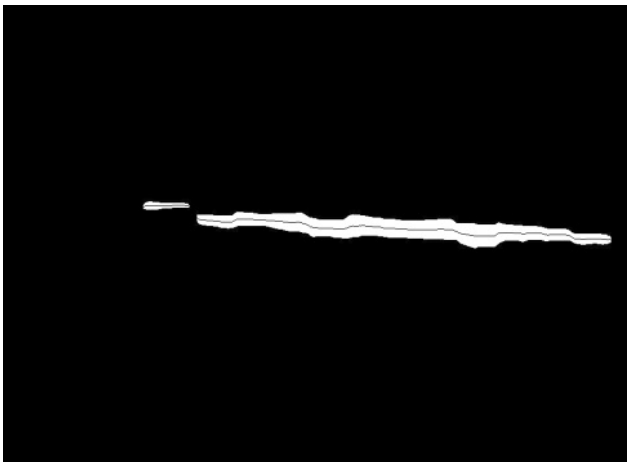


Figura 3.28: Deteccion de linea laser

3.7.4. Función para cálculo de profundidad

Como ya se describió en la sección anterior el comportamiento de la triangulación en distancias menores a 10 milímetros tiene un comportamiento lineal, por lo que solo es necesario conocer dos puntos para calcular la relación de pixeles contra distancia, la Figura 3.29 muestra la distancia entre pixeles para 3mm.

Figura 3.29: Diferencia de pixeles para 3mm

Dado que la linea láser sera desplazada, lo mas conveniente para el calculo de profundidad es una relación entre pixeles y milimetros, para encontrar ese relación basta con aplicar la formula 3.15. El resultado para este caso sera de 2 pixeles por milímetro.

$$pxmm = \frac{px}{mm} \tag{3.15}$$

Capítulo 4

Análisis de resultados

4.1. Análisis en dos dimensiones

A continuación, se presenta un análisis simple que permite calcular distancias entre dos puntos de una imagen. Para entregar valores en unidades conocidas (milímetros) es necesario incluir un patrón de referencia que permita determinar la relación entre pixeles y milímetros. Considerando que la cámara se encuentra previamente calibrada, es posible lograr resultados aproximados.

Una vez realizado el proceso de calibración y el pre-procesado de la imagen, procedemos a analizar las imágenes obtenidas. En principio el resultado es un análisis en dos dimensiones. Primeramente, se calcula la equivalencia de pixeles por milímetros, para ello basta con aplicar la formula 3.15. En este caso la imagen 4.1 tiene un patrón de referencia de 10 cm del punto **a** al punto **b**, una referencia de 9.4 cm del punto **c** al punto **d**, consta de 335 pixeles horizontales entre **c,d** y 200 pixeles verticales entre **a,b**, por lo que la relación horizontal queda en 3.56 pxmm y 2 pxmm para la relación vertical. Hay que considerar que no podemos tener fracciones de pixel, sin embargo mantenemos la fracción para el cálculo con el fin de reducir los errores por truncamiento.

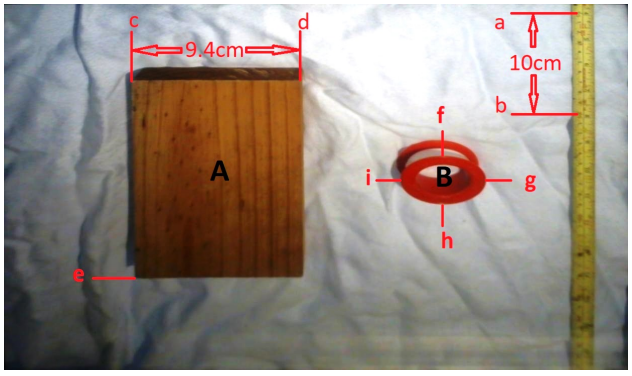


Figura 4.1: Relación de pixeles

Para validar las relaciones se procede a medir la distancia en pixeles de los objetos **A** y **B**.

4.1.1. Medición del objeto A

El objeto de prueba **A** tiene dimensiones conocidas en los puntos **c,d** de 9.4cm y 19.5cm en los puntos **c,e**. Los puntos **c,d** fueron empleados como referencia para el cálculo horizontal, por lo que no tiene sentido compararlos horizontalmente. La distancia entre los puntos **c,e** es de 392 pixeles, para calcular la distancia en base a los pixeles por milímetro podemos despejar los **mm** de la ecuación 3.15 quedando de la siguiente manera:

$$mm = \frac{px}{pxmm} \tag{4.1}$$

Reemplazando los términos en la ecuación 4.1 se obtiene lo siguiente:

$$mm = \frac{392}{2} = 196$$

El largo real del objeto A es de 195 mm, contra los 196 mm calculados, se puede observar claramente que el error absoluto es de 1 mm, lo cual representa un error relativo de 0.5 %.

4.1.2. Medición del objeto B

El objeto de prueba **B** presenta características mas complejas al tratarse de una geometría circular. La distancia entre el punto **f** y el punto **h** es de 96 pixeles, aplicando la ecuación 4.1:

$$mm = \frac{96}{2} = 48$$

La distancia entre el punto **i** y el punto **g** es de 170 pixeles, aplicando la ecuación 4.1:

$$mm = \frac{170}{3{,}56} = 47{,}75$$

Como se puede observar tenemos una diferencia en los cálculos de 0.25 milímetros, esta diferencia debe ser considerada de acuerdo a nuestras necesidades de resolución, para nuestro caso esperamos una resolución de un milímetro, por lo que podemos redondear los cálculos, en tal caso considerando el redondeo y una resolución de un milímetro obtendríamos una distancia calculada de 48 milímetros horizontales y 48 mm verticales, con lo que podríamos considerar que la relación de pixeles es correcta.

4.1.3. Análisis de pie en dos dimensiones

El presente trabajo se centra en diagnosticar si un paciente cuenta con pie plano, para ello nos apoyamos de la Figura 4.2, la cual muestra la relación geométrica de un pie, de acuerdo con los segmentos **B** y **C** se establesen las siguientes relaciones:

$$C \approx 3B \quad \text{Pie Normal} \tag{4.2}$$

$$C < 3B \quad \text{Pie Plano} \tag{4.3}$$

$$C > 3B \quad \text{Pie Cavo} \tag{4.4}$$

Donde:
L representa la longitud del pie, sin considerar los dedos.
C se encuentra en el talon anterior y representa la sección mas ancha del pie.
B se encuentra ubicado al centro longitudinal del pie.

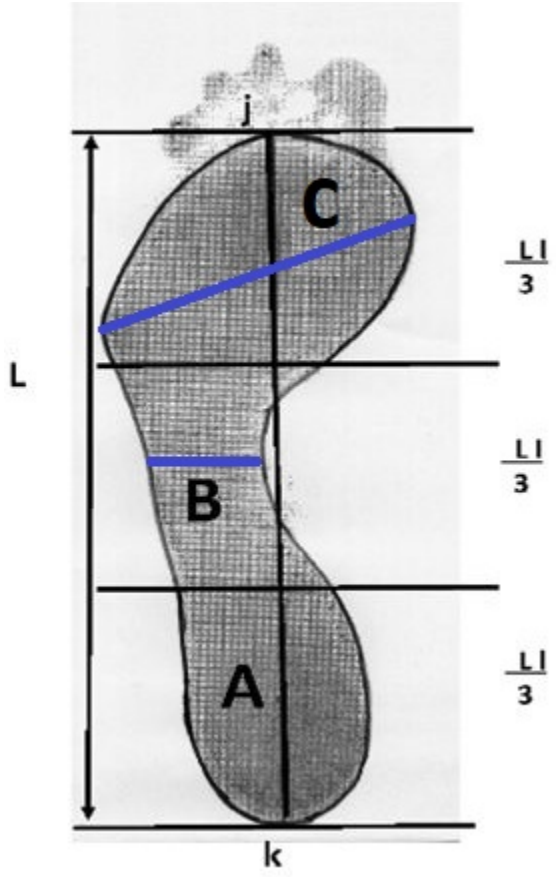


Figura 4.2: Relacion geométrica del pie

La Figura 4.3 muestra un análisis en dos dimensiones para determinar si el pie escaneado es plano. Para diagnosticar el pie es necesario demostrar que el segmento **C** es aproximadamente igual a 3 veces el segmento **B**, si esto se cumple entonces consideramos que es un pie normal, si C es menor que B entonces se trata de un pie plano. Las coordenadas de los puntos **a,b,c,d** se enumeran a continuacion:
a(32,83)
b(188,102)
c(97,144)
d(168,144)

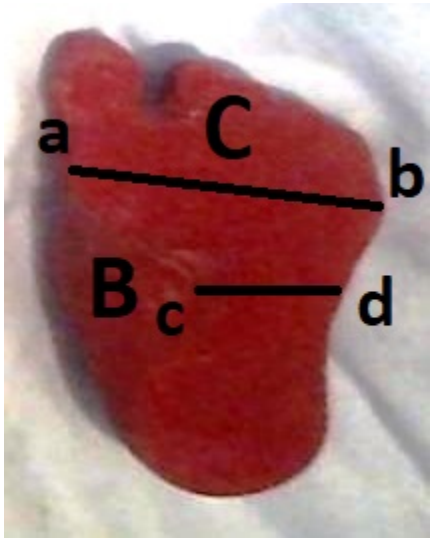


Figura 4.3: Pie de muestra escaneado

Las coordenadas se expresan en pixeles, hay que recordar que se tienen resoluciones diferentes para los pixeles horizontales y para los verticales, por lo que es necesario convertir las coordenadas de pixeles a milímetros. La conversión se realiza aplicando la ecuación 4.1, quedando las coordenadas en milímetros de la siguiente manera:
a(9,42)
b(53,51)
c(27,72)
d(47,72)

Para calcular las longitudes del segmento **C** y del segmento **B** se aplica la formula 4.5 para medir distancia entre dos puntos.

$$D = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2} \tag{4.5}$$

Para el segmento **C** se obtiene una longitud de 72mm y para el segmento **B** una longitud

de 45 mm, comparando **C** contra **3B** se obtiene lo siguiente:

$$72mm < 135mm$$

Como se puede observar el resultado pertenece a la relación 4.3 indicando que se trata de un pie plano.

4.1.4. Repetibilidad en dos dimensiones

En todo sistema de medición hay qué analizar la variabilidad en el sistema causada por el dispositivo de medición, este análisis es conocido como repetibilidad, la cual se define como: «la variación que se observa cuando el mismo operador mide la misma parte muchas veces, usando el mismo sistema de medición, bajo las mismas condiciones» [26]. Otro análisis altamente realizado a equipos de medición es la reproducibilidad la cual es definida como «qué tanto de la variabilidad en el sistema de medición es causada por las diferencias entre los operadores» [26], por la naturaleza de la reproducibilidad no sera tomada en cuenta para este estudio. La repetibilidad se expresa en porcentaje. Para el análisis en dos dimensiones basta una sola captura de la cámara, lo que implica que las variables de medición no cambian no importa cuantas veces se realice la medición. Esta situación implica que la repetibilidad del sistema de medición sera de 100 % en todo momento para dos dimensiones.

4.2. Análisis en tres dimensiones

El proceso de digitalización 3d se realiza a través de un barrido vertical, la secuencia de digitalización se muestra en la Figura 4.4, para esta digitalización se utilizo un paso de 10mm

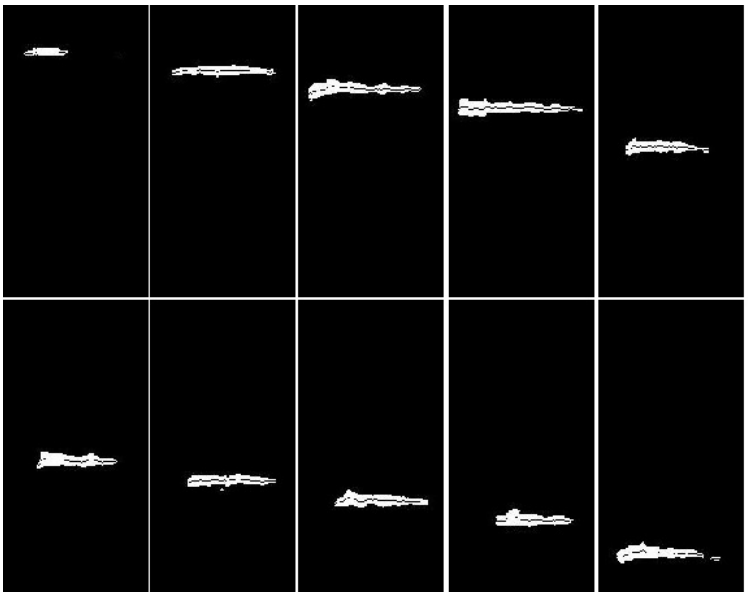


Figura 4.4: Secuencia a de digitalización a 10mm de resolución

Hay que considerar que el eje móvil agrega una variable al calculo de profundidad. En cada iteración se conoce la posición del eje móvil, el cual será mapeado al eje Y, las coordenadas X del cuadro de imagen son mapeadas al eje X pasando por la relación de pixeles por milímetros, el eje Z se calcula mediante las coordenadas Y del cuadro de imagen aplicando la relación de pixeles por milímetro para Z. El resultado de la digitalización se muestra en la Figura 4.5. Para tomar una referencia con respecto al eje móvil se modifico el algoritmo onePixel() 5.4 quedando como se muestra en el anexo 5.5. En general la adaptación del código solo incluye guardar las coordenadas de la linea detectada para una referencia posterior.

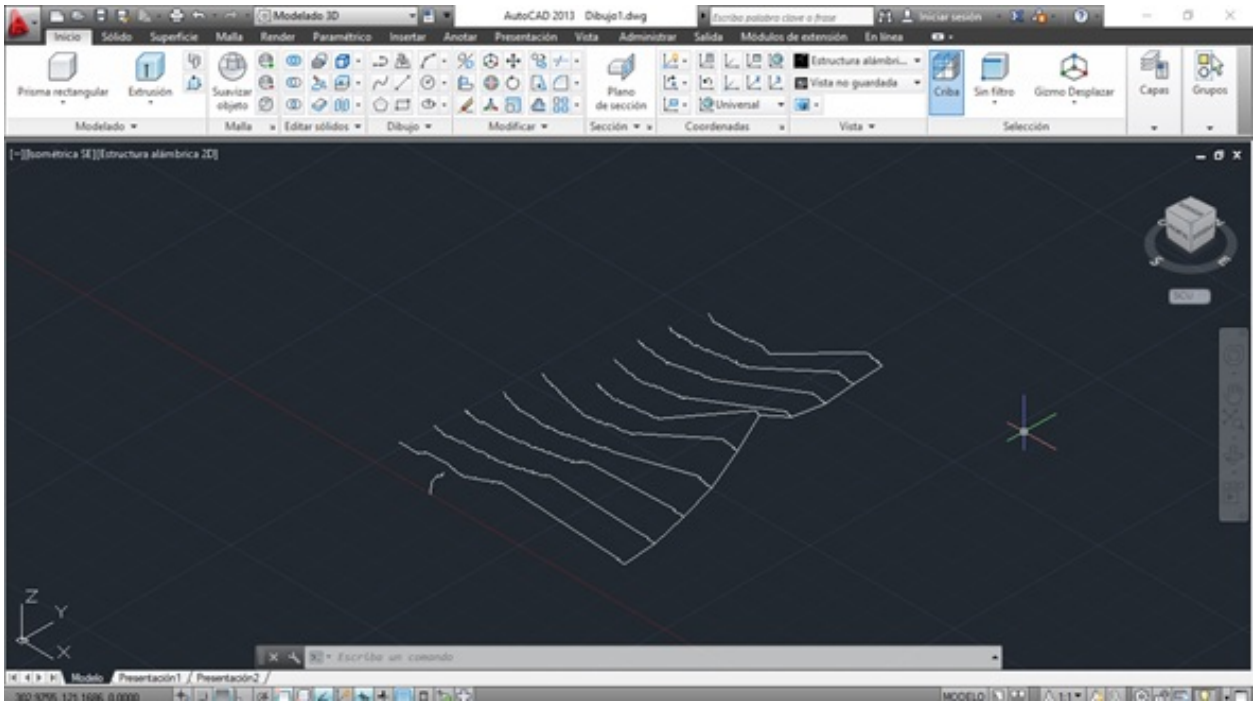


Figura 4.5: Malla de alambre a 10mm de resolución

El proceso de digitalización comienza con la instrucción home, la cual desplaza el eje móvil a la posición cero y se invoca enviando el comando «h», posterior al envío del comando se espera un intervalo de 10 segundos máximo para recibir respuesta de comando ejecutado. El siguiente paso es encender el láser con el comando «L», inmediatamente se inicializa un archivo SCR [27] en el cual se escriben los puntos que describen a la linea detectada en forma de polilinea. Un ciclo for es el encargado de realizar la digitalización cuadro por cuadro. En cuanto se termina el ciclo se cierra el archivo SCR y el láser se apaga mediante el comando «l», para finalizar el eje móvil de manda a su inicio nuevamente con el comando «h». El método que ejecuta el proceso se detalla en el anexo 5.6

A partir del SCR se pueden realizar mediciones a través de AutoCAD, en ente caso nos interesa conocer la profundidad del arco. La Figura 4.6 muestra el resultado de la medición. Como se observa el resultado de la medición resulto ser de 3mm.

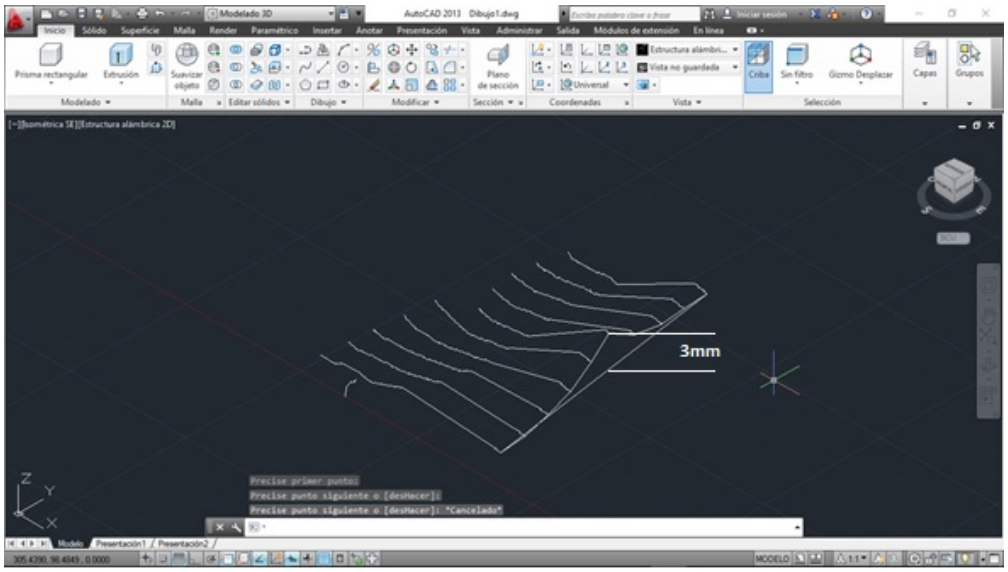


Figura 4.6: Medición de la profundidad del arco del pie

4.2.1. Repetibilidad en tres dimensiones

Como ya se mencionó en la sección anterior de repetibilidad en dos dimensiones al realizar las mediciones con una imagen estática el resultado siempre es el mismo, sin embargo para la repetibilidad en tres dimensiones no sucede lo mismo. Los cálculos para medir profundidad se realizan mediante un video en tiempo real, lo que implica que la cámara muestre pequeñas diferencias entre cuadro de imagen y cuadro de imagen aunque se trate del mismo medio de captura; esto se debe a que la cámara cuenta con algoritmos progresivos para mejorar la imagen en función de la luminiscencia y enfoque. Para determinar la repetibilidad en tres dimensiones se realizo un análisis de R&R utilizando Excel tomando como muestra video en tiempo real de la misma escena. El análisis se realizo en tres partes con 10 intentos y un operador. La Tabla 4.1 muestra los datos para el análisis de R&R.

Operador	Parte	Intento	Medición
1	1	1	495
1	1	2	494
1	1	3	495
1	1	4	495
1	1	5	495
1	1	6	495
1	1	7	495
1	1	8	495
1	1	9	495
1	1	10	495
1	2	1	495
1	2	2	495
1	2	3	495
1	2	4	495
1	2	5	495
1	2	6	496
1	2	7	496
1	2	8	495
1	2	9	496
1	2	10	495
1	3	1	495
1	3	2	495
1	3	3	496
1	3	4	496
1	3	5	495
1	3	6	495
1	3	7	495
1	3	8	495
1	3	9	495
1	3	10	495

Tabla 4.1: Resultados de las mediciones

Como primer resultado se obtienen los componentes de la varianza, la talba 4.2 muestra los porcentajes de contribución y la variación del estudio. Un dato a tener en cuenta es la reproducibilidad, la cual aparece en cero debido a que el estudio se realizó con un solo operador. La Figura 4.7 muestra la gráfica de los componentes de la varianza.

Fuente	Componentes de la varianza	Contribución
Gage R&R	0.170	86.63 %
Repetibilidad	0.170	86.63 %
Reproducibilidad	0.000	0.00 %
Operadores	0.000	0.00 %
Operadores * Partes	0.000	0.00 %
Entre las partes	0.026	13.37 %
Total	0.197	100.00 %

Tabla 4.2: Contribución de las fuentes para la varianza

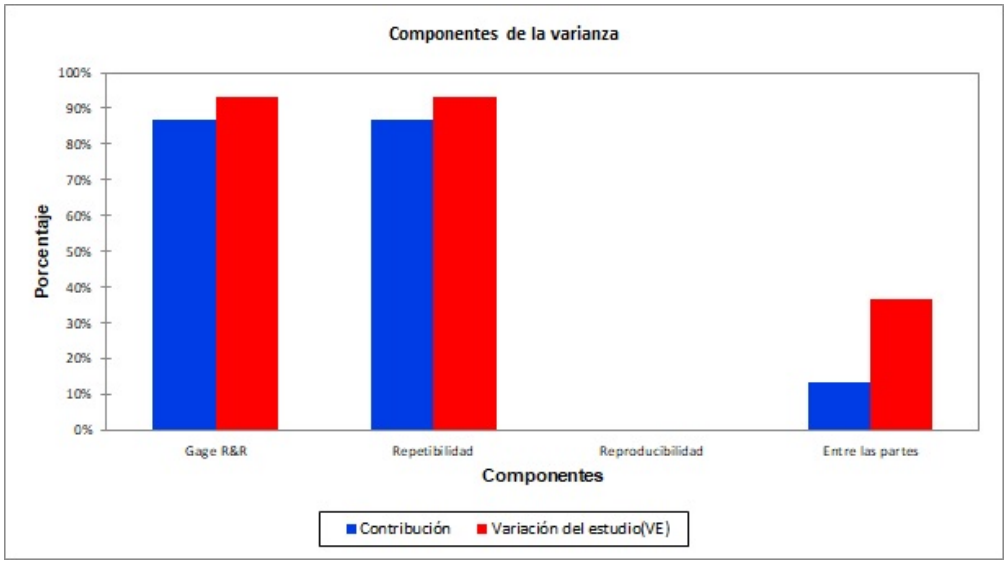


Figura 4.7: Gráfica de componentes de la varianza

La Tabla 4.3 muestra las fuentes consideradas para la varianza, el análisis de Gage R&R [28] se compone de la repetibilidad y la reproducibilidad, donde la reproducibilidad sera cero debido a que el estudio solo se realiza con un operador.

Fuente	Gage R&R	Repetibilidad	Reproducibilidad	Entre las partes
Contribución	86.63 %	86.63 %	0.00 %	13.37 %
Variación del estudio(VE)	93.07 %	93.07 %	0.00 %	36.57 %

Tabla 4.3: Componentes de la varianza

A continuación la Tabla 4.4 muestra los detalles de las observaciones para la elaboración de la gráfica barra x. Donde CL representa el limete de control, LCL representa el limite de control inferior y UCL indica el limite de control superior.

Fase — Parte	Tamaño del subgrupo	Media	Mínimo	Máximo	CL	LCL	UCL	C Límite inf.	C Límite sup.	B Límite inf.	B Límite sup.
1 — 1	10	494.900	494.000	495.000	495.133	494.517	495.750	494.928	495.339	494.722	495.544
1 — 2	10	495.300	495.000	496.000	495.133	494.517	495.750	494.928	495.339	494.722	495.544
1 — 3	10	495.200	495.000	496.000	495.133	494.517	495.750	494.928	495.339	494.722	495.544

Tabla 4.4: Detalles para las observaciones

Tomando como base los datos de la Tabla 4.4 se realiza la gráfica 4.8 en donde se muestra el comportamiento de las mediciones sobre los límites de control.

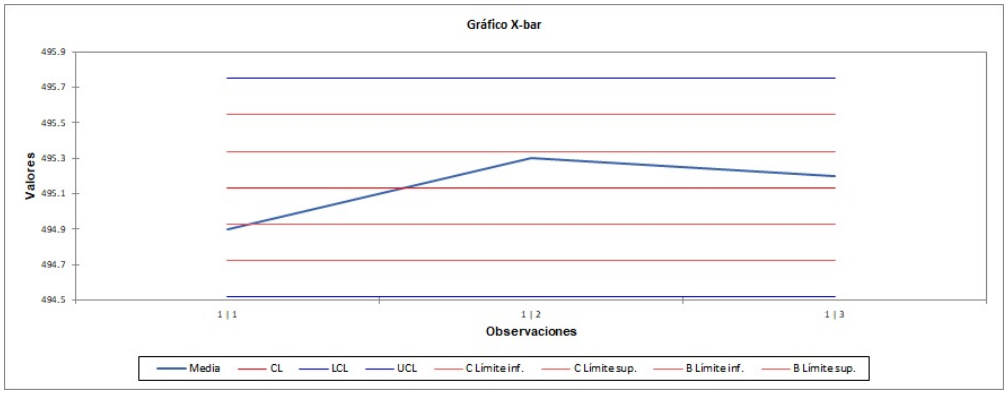


Figura 4.8: Gráfico X-bar

En resumen, tomando el mínimo (494) y el máximo (496) valor reportado tenemos una diferencia de 2 pixeles máximo que aplicando la ecuación 3.15 nos representa un milímetro de error.

4.2.2. Plantilla 3D

Para generar la plantilla se tomo como base un modelo de pie en 3D, la edición se llevo a cabo con la plataforma online tinkercad, la imagen 4.9 muestra el resultado de la plantilla que será usada como patrón. Tinkercad permite exportar el modelo 3D a un formato STL el cual puede ser parametrizado fácilmente a través de un software de manufactura asistida por computadora (CAM) [29].

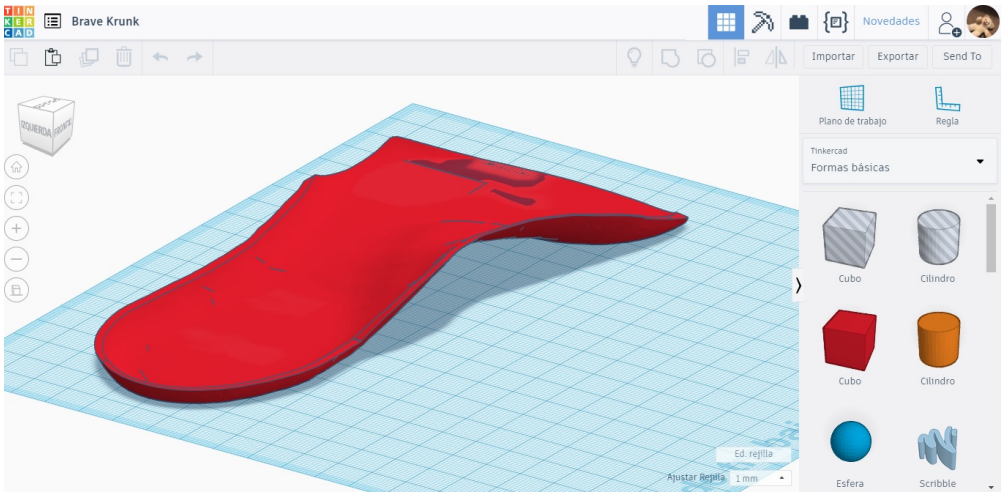


Figura 4.9: Plantilla 3D

Tomando como base las medidas obtenidas del digitalizador se procede a escalar el modelo en STL utilizando el software CAM Cura Ultimaker [30]. Utilizando Cura Ultimaker en el menú escalar se ingresan las dimensiones obtenidas con el digitalizador y se procede a ajustar la plantilla, la altura del eje Z se recomienda de 15 milímetros pero deberá ser supervisada por un experto en el área. La Figura 4.10 muestra el Cura Ultimaker escalando la plantilla.

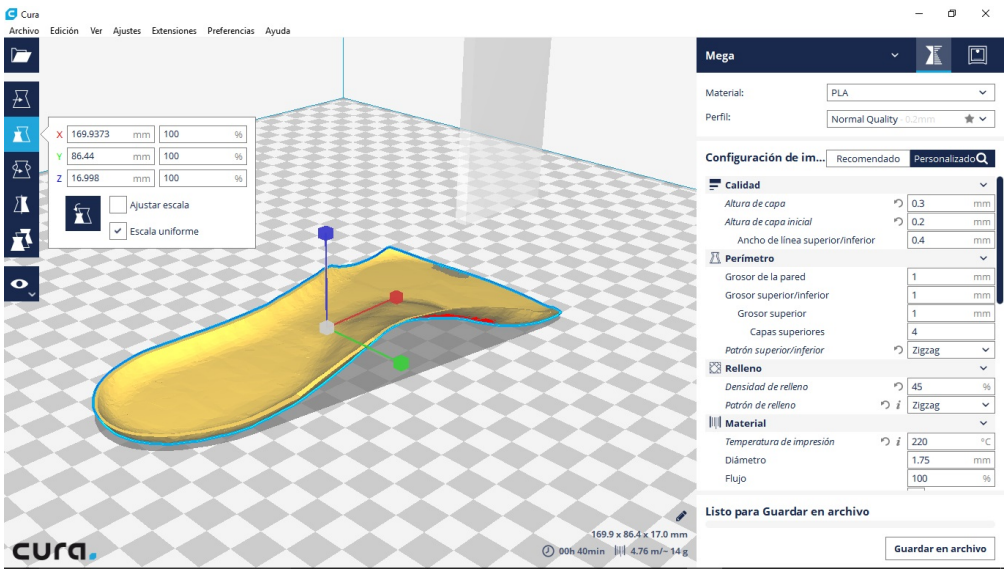


Figura 4.10: Cura Ultimaker

El anexo 5.7 muestra un pequeño fragmento de código G [31] generado por el software CAM Cura Ultimaker. La Figura 4.11 muestra la impresión de la plantilla, para imprimirla se utilizo una impresora 3d de 50 x 50 x 50 milímetros con una altura de capa de 0.3 milímetros y dilató 80 minutos la impresión.

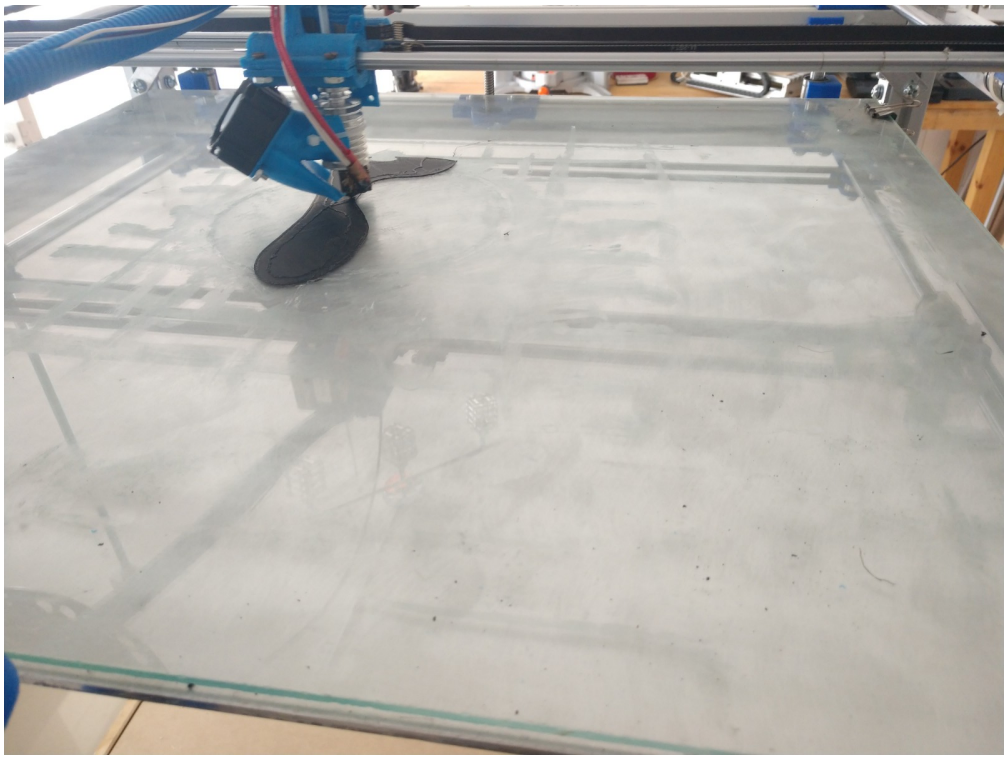


Figura 4.11: Impresión de plantilla

Capítulo 5

Conclusiones y Trabajos futuros

De acuerdo a los resultados obtenidos se llegó a las siguientes conclusiones:

El podoscopio clásico se trata de un sencillo y conveniente instrumento clínico para diagnóstico de pies, que permite visualizar y estudiar las huellas de la planta y los distintos ejes del pie. Permite asimismo, realizar el estudio del pie, tanto normal como patológico, con una elevada rapidez y comodidad.

Sin embargo no permite conocer la profundidad del arco, sin mencionar que las mediciones deben hacerse manualmente y las plantilla de manera artesanal, la Figura 5.1 muestra un par de plantillas fabricadas manualmente.

Figura 5.1: Plantillas de manufactura manual

La fabricación de plantillas ortopédicas de manera manual requieren de habilidades técnicas especializadas por parte de los técnicos ortopedistas, ademas de considerar que el tiempo

de fabricación y la precisión de la plantilla dependerá de la experiencia . Para reducir los costos de fabricación de las plantillas se han desarrollado modelos estándar que pueden adquirirse inmediatamente en las tiendas de ortopedia, estas plantillas se clasifican de acuerdo a las medidas estándar para calzado y son consideradas como universales. La principal desventaja de estas plantillas radica en que al ser un producto estándar no se adapta 100 % a las dimensiones de todos los pies. La Figura 5.2 muestra una plantilla estándar con gel para fijar su movimiento.

Figura 5.2: Plantillas estandar de fijación por gel

Las plantillas generadas con el digitalizador 3D tienen la ventaja de ser plantillas personalizadas y fabricadas a la medida, lo que al usuario le puede reducir la incomodidad de usar una plantilla que no se amolde completamente a su pie. Como desventaja con respecto a las plantillas de molde estándar se encuentra el tiempo de fabricación necesario para producir un par de plantillas. Un ejemplo del tiempo requerido para la impresión de una plantilla con dimensiones de 180mm x 75mm x 20mm fue de 80 minutos, la Figura 5.3 muestra una plantilla impresa. La Tabla 5.1 muestra una comparativa entre los tipos de plantillas.

Figura 5.3: Plantillas impresa fijación por talón

Tipo de plantilla	Método de Fabricación	Método de fijación	Tiempo de Entrega	Ergonomía (comodidad)
Personalizada Manual	Artesanal	Zapato	3 a 5 dias	Alta
Estándar	Molde	Gel — Talón	Inmediato	Mediana
Personalizada	Impresión 3D	Talón	2 a 3 horas	Alta

Tabla 5.1: Detalles para las observaciones

Por otro lado se observó que para la parametrización de la plantilla es suficiente realizar el análisis del pie en dos dimensiones, con lo que se obtienen las medidas para la fabricación de la plantilla. El uso del digitalizador 3D basta con solo medir la profundidad del arco planar para determinar si tiene o no tiene pie plano, incluso solo la profundidad del arco puede ser considerada como factor determinante para diagnosticar si tiene o no tiene pie plano. Esto quiere decir que no tiene caso digitalizar completamente el pie, solo enfocar las mediciones en la sección B de la Figura 4.2.

Para responder a la hipótesis planteada en este estudio «El análisis en tres dimensiones aporta mayor información para diagnosticar el pie plano» se concluye que la información obtenida por un análisis en tres dimensiones solo acelera el tiempo de diagnostico y garantiza un diagnostico correcto al realizar la medición directa sobre la variable de interés (profundidad del arco) ya que el método tradicional determina el diagnostico en base a una relación (medida indirecta) para estimar si cumple o no cumple con la geometría del pie, lo cual se presta a la presencia de falsos positivos por la medición de una perspectiva diferente. Sin embargo la fabricación de las plantillas impresas en 3D se puede lograr desde un diagnostico tradicional o un diagnostico en tres dimensiones, lo que permite considerar la plantilla modelo desarrollada en este estudio como un producto significativo para los especialistas ortopédicos.

5.1. Trabajos futuros

La primera línea de continuación de esta tesis es la integración de los componentes electrónicos en una sola tarjeta de control. Durante el desarrollo de esta tesis los esfuerzos se han enfocado en demostrar la funcionalidad del instrumento desarrollado, por lo que la optimización de los componentes ha sido un tema secundario. Se considera apropiada la incorporación de componentes especializados para la captura de imágenes, así como para el control de todo el instrumento como tal. Se propone el desarrollo de un sistema integral que permita la interacción completa de los módulos que no se contemplan en este trabajo, tales como la visualización y edición de la plantilla 3D. Realizar el estudio de los materiales óptimos para la impresión de las plantillas, tomando como base la calidad del material y la plantilla, el tiempo de vida esperado por la plantilla, la resistencia al uso cotidiano y el peso máximo de las personas candidatas a este tipo de plantillas. Otro tema a considerar es la construcción de una impresora 3D especializada para la fabricación de plantillas, considerando los materiales más convenientes para la fabricación de la misma.

Final mente se propone la incorporación de un especialista en el área de ortopedia el cual pueda retroalimentar el diseño del instrumento a partir de su experiencia profesional.

Bibliografía

[1] RH Hernández Guerra. Prevalencia del pie plano en niños y niñas en las edades de 9 a 12 años. 2006.

[2] Stephen F Conti. Posterior tibial tendon problems in athletes. *The Orthopedic clinics of North America*, 25(1):109–121, 1994.

[3] José Gorgues. [fichas de ortopedia] ortesis de silicona. *Offarm: Farmacia y Sociedad*, 25(5):138–142, 2006.

[4] A Viladot. Anatomía funcional y biomecánica del tobillo y el pie. *Revista Española de Reumatología*, 30(9):469–477, 2003.

[5] Héctor Iván Saldívar-Cerón, Alberto Garmendia Ramírez, Marco Antonio Rocha Acevedo, and Pedro Pérez-Rodríguez. Obesidad infantil: factor de riesgo para desarrollar pie plano. *Boletín Médico del Hospital Infantil de México*, 72(1):55 – 60, 2015.

[6] A Zárate Barchello, Pereira López, J Ibarrola Zárate, A Kikuchi, and L Sanabria. Flat feet prevalence in school children of asunción and great asunción during 2008. *Anales de la Facultad de Ciencias Médicas (Asunción)*, 42(2):13–18, 2009.

[7] CALIDAD Y SER. Visión artificial. 2006.

[8] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library*. .º'Reilly Media, Inc.", 2008.

[9] Tom Ang. *La fotografía digital: guía para la creación y manipulación de imágenes*. Editorial Blume, SA, 2002.

[10] Willow Garage Intel. Open source computer vision library. *URL http://opencv. willowgarage. com. Retrieved May*, page 22, 2012.

[11] Arduino Uno Arduino and UNO Genuino. Url: https://www. arduino. cc/en/main. *ArduinoBoardUno (visited on May 23, 2016)*, 2014.

[12] Logitech hd webcam c525. https://www.logitech.com/assets/46921/7/hd-webcam-c525-quickstart-guide.pdf.

[13] Nema 17 step motor. https://www.mouser.com/datasheet/2/30/5017-009-845352.pdf.

[14] Driver pololu a4988. https://www.pololu.com/file/0J450/a4988-DMOS-microstepping-driver-with-translator.pdf.

[15] Santiago Martín, Javier Suárez, Ramón Rubio, and Ramón Gallego. Aplicación de los sistemas de visión estereoscópica en las enseñanzas técnicas. *Escuela de Ingenieros Técnicos Industriales de Gijón. Universidad de Oviedo*, 2004.

[16] Hod Lipson and Melba Kurman. *La revolución de la impresión 3D: el presente y el futuro de una máquina que puede crear (casi) cualquier cosa*. Anaya multimedia (GA), 2015.

[17] APLICACIÓN EN DOMÓTICA. Problema de ingeniería. 2008.

[18] Vicente Bayarri Cayón, Elena Castillo López, et al. Los nuevos algoritmos de procesado y tendencias en geometría computacional para la explotación y valoración de datos 3d. 2014.

[19] E Cuesta, A García, B Álvarez, G Valiño, and JC Rico. Análisis de la influencia del acabado superficial en la calidad del digitalizado con sensores láser por triangulación. In *Proc. of the 2 nd Manuf. Eng. Soc. Intern. Conf.(CISIF-MESIC 07)*.

[20] Palmer MDR Salvador. Un analisis de la aplicacion de las tecnicas de interferometria con luz laser al estudio de las deformaciones en ortodoncia. 1990.

[21] Pablo Gil, Elías José Manchón Lopez, Fernando Torres, Jorge Pomares, and Francisco Gabriel Ortiz Zamora. Reconstrucción tridimensional de objetos con técnicas de visión y luz estructurada. 2002.

[22] Nora La Serna Palomino and Ulises Norberto Román Concha. Técnicas de segmentación en procesamiento digital de imágenes. *Revista de investigación de Sistemas e Informática*, 6(2):9–16, 2009.

[23] B Aldalur and M Santamaría. Realce de imágenes: filtrado espacial. *Revista de teledetección*, 17:31–42, 2002.

[24] C Perez, MA Vicente, C Fernandez, O Reinoso, and A Gil. Aplicacion de los diferentes espacios de color para deteccion y seguimiento de caras. *Proceedings of XXIV Jornados de Automatica*, 2003.

[25] Rafael Magro. Binarización de imágenes digitales y su algoritmia como herramienta aplicada a la ilustración entomológica. *Boletín de la Sociedad Entomológica Aragonesa*, 53:443–464, 2013.

[26] Soporte de minitab. https://support.minitab.com/es-mx/minitab/18/help-and-how-to/quality-and-process-improvement/measurement-system-analysis/supporting-topics/gage-r-r-analyses/what-is-a-gage-r-r-study/.

[27] Autocad architecture 2011. http://images.autodesk.com/adsk/files/autocad-aca-user-guide-spanish.pdf.

[28] Introduction to gage r&r studies. http://www.swtest.org/swtw-library/1998proc/PDF/T1Hank.PDF.

[29] Kunwoo Lee. *Principles of cad/cam/cae systems*. Addison-Wesley Longman Publishing Co., Inc., 1999.

[30] Cura ultimaker. https://ultimaker.com/download/73560/180104-Ultimaker-CuraConnect-Manuals-

[31] Felipe Díaz del Castillo Rodríguez et al. Programación automática de máquinas cnc. 2016.

Anexos

```
1  #define EN 8           // activar driver A4988
2  #define Z_DIR 7        // pin de direccion del motor
3  #define Z_STP 4        // pin de pasos
4  #define limite 11      // interruptor de final de carrera
5  #define laser 10       // encendido del laser
6  #define led 12         // encendido de la iluminacion led
7  #define ppmm 80        // pasos por milimetro
8  #define aceIni 380     // aceleracion inicial
9  #define aceFin 180     // aceleracion final
10 #define aceInc 2       // incremento de la aceleracion
11 #define aceInt 10      // intervalo de aceleracion
12 #define maximo 270     // desplazamiento maximo del eje en mm
13 #define timeoutAce 50 // timeout para la aceleracion
14
15 //variables globales
16 int posActual;
17 int aceActual;
18 int intervalo=0;
19 long tiempoActual=0;
20 boolean dirActual;
21 boolean hecho=false;
22 long delta;
23 int avance=0;
24 int retroceso=0;
25 //desplazar un milimetro, dir=true avance, dir=false retrocede
26 void un_mm (boolean dir){
27   digitalWrite (Z_DIR, dir);
28   if(dir){
29     if(posActual>=maximo)return;
30     posActual++;
31   }else {
32     if(posActual==0)return;
33     posActual--;
34   }
35   if((delta>timeoutAce)||(dir!=dirActual)){
36     aceActual=aceIni;
37   }else{
38     if(intervalo%aceInt==0){
39       aceActual-=aceInc;
40       if(aceActual<=aceFin){
41         aceActual=aceFin;
42       }
43     }
44   }
45   for (int i = 0; i <ppmm; i++,intervalo++) {
46     digitalWrite (Z_STP, HIGH);
47     delayMicroseconds (aceActual);
48     digitalWrite (Z_STP, LOW);
49     delayMicroseconds (aceActual);
```

```
50      if(intervalo%aceInt==0){
51        aceActual-=aceInc;
52        if(aceActual<=aceFin){
53          aceActual=aceFin;
54        }
55      }
56    }
57    dirActual=dir;
58    hecho=true;
59    tiempoActual=millis();
60  }
61  //busca origen del eje
62  void home(){
63    digitalWrite (Z_DIR, false);
64    aceActual=aceIni;
65    while(true){
66      digitalWrite (Z_STP, HIGH);
67      delayMicroseconds (aceActual);
68      digitalWrite (Z_STP, LOW);
69      delayMicroseconds (aceActual);
70      if(intervalo%aceInt==0){
71        aceActual-=aceInc;
72        if(aceActual<=aceFin){
73          aceActual=aceFin;
74        }
75      }
76      if(!digitalRead(limite)){
77        break;
78      }
79      intervalo++;
80    }
81    posActual=0;
82    un_mm(true);
83  }
84  //inicializa pines y velocidad de comunicacion
85  void setup () {
86    Serial.begin(9600);
87    pinMode (Z_DIR, OUTPUT);
88    pinMode (Z_STP, OUTPUT);
89    pinMode (EN, OUTPUT);
90    pinMode (laser, OUTPUT);
91    pinMode (led, OUTPUT);
92    pinMode(limite,INPUT_PULLUP);
93    digitalWrite (EN, LOW);
94  }
95  //ciclo principal
96  void loop () {
97    if(Serial.available()){
98      switch(Serial.read()){
99        case 'h':
100       case 'H':
```

```
101           home();
102           break;
103         case 'f':
104           avance++;
105           break;
106         case 'r':
107           retroceso++;
108           break;
109         case 'l':
110           digitalWrite (laser, LOW);
111           Serial.println("ok");
112           break;
113         case 'L':
114           digitalWrite (laser, HIGH);
115           Serial.println("ok");
116           break;
117         case 'e':
118           digitalWrite (led, LOW);
119           Serial.println("ok");
120           break;
121         case 'E':
122           digitalWrite (led, HIGH);
123           Serial.println("ok");
124           break;
125       }
126     }else{
127       if(avance>0){
128         un_mm(true);
129         avance--;
130       }else if(retroceso>0){
131         un_mm(false);
132         retroceso--;
133       } else if(hecho){
134         Serial.println("ok");
135         hecho=false;
136       }
137     }
138     delta=millis()-tiempoActual;
139 }
```

Índice de algoritmos 5.1: Codigo de control para Arduino

```
1 package com.bigzener.gui;
2 import java.io.File;
3 import java.util.ArrayList;
4 import org.opencv.calib3d.Calib3d;
5 import org.opencv.core.CvType;
6 import org.opencv.core.Mat;
7 import org.opencv.core.MatOfPoint2f;
8 import org.opencv.core.MatOfPoint3f;
9 import org.opencv.core.Point3;
```

```
10 import org.opencv.core.Rect;
11 import org.opencv.core.Size;
12 import org.opencv.core.TermCriteria;
13 import org.opencv.highgui.Highgui;
14 import org.opencv.imgproc.Imgproc;
15 /**
16  *
17  * @author BIGZENER
18  */
19 public class CalibChessBoard {
20     private static final int FLAGS_CORNER = Calib3d.CALIB_CB_ADAPTIVE_THRESH
21             | Calib3d.CALIB_CB_FAST_CHECK
22             | Calib3d.CALIB_CB_NORMALIZE_IMAGE;
23     private static final int FLAGS_CALIB = Calib3d.CALIB_ZERO_TANGENT_DIST
24             | Calib3d.CALIB_FIX_PRINCIPAL_POINT
25             | Calib3d.CALIB_FIX_K4
26             | Calib3d.CALIB_FIX_K5;
27     private TermCriteria criteria = new TermCriteria(TermCriteria.EPS
28             + TermCriteria.MAX_ITER, 40, 0.001);
29     private final Size winSize = new Size(5, 5), zoneSize = new Size(-1, -1);
30     private final Size patternSize;
31     private ArrayList <Mat>objectPoints;
32     private ArrayList <Mat>imagePoints = new ArrayList();
33     private ArrayList <Mat>vImg;
34     private final Mat cameraMatrix = Mat.eye(3, 3, CvType.CV_64F);
35     private final Mat distCoeffs = Mat.zeros(8, 1, CvType.CV_64F);
36     private final ArrayList <Mat>rvecs = new ArrayList();
37     private final ArrayList <Mat>tvecs = new ArrayList();
38     private Mat map1 = new Mat();
39     private Mat map2 = new Mat();
40     private final double squareSize;
41
42     /**paso 1 (solo una vez)
43      * constructor de la clase para calibar la camara
44      * @param patternSize tama\~{n}o del patron de ajedrez
45      * @param squareSize tama\~{n}o del lado del cuadrito del patron en
    milimetros
46      */
47     public CalibChessBoard(Size patternSize,double squareSize) {
48         this.patternSize = patternSize;
49         this.squareSize=squareSize;
50     }
51     /**paso 2 (solo una vez)
52      * establece los vectores de las imagenes de referencia
53      * @param path ruta donde se almacenan las imagenes para calibrar
54      */
55     public void setImages(String path) {
56         vImg = new ArrayList();
57         objectPoints = new ArrayList();
58         imagePoints = new ArrayList();
59         MatOfPoint3f corners3f = getCorner3f();
```

```
60          for (File f : new File(path).listFiles()) {
61              Mat mat = Highgui.imread(f.getPath(),Highgui.CV_LOAD_IMAGE_COLOR);
62              if (mat == null || mat.channels() != 3)continue;
63              Mat gray = new Mat();
64              Imgproc.cvtColor(mat, gray, Imgproc.COLOR_BGR2GRAY);
65              MatOfPoint2f corners = new MatOfPoint2f();
66              if (!getCorners(gray, corners))continue;
67              objectPoints.add(corners3f);
68              imagePoints.add(corners);
69              vImg.add(mat);
70          }
71      }
72      /**paso 3 (solo una vez)
73       * calcula la matris de distorcion de la camara
74       */
75      public void calibrate() {
76          Size imageSize=vImg.get(0).size();
77          map1 = new Mat();
78          map2 = new Mat();
79          Mat r = new Mat();
80          Rect rect = new Rect();
81
82          Imgproc.initUndistortRectifyMap(cameraMatrix, distCoeffs, r,
83      Calib3d.getOptimalNewCameraMatrix(cameraMatrix, distCoeffs, imageSize, 1.0,
        imageSize, rect, false),
84          imageSize, CvType.CV_16SC2, map1, map2);
85      }
86      /** paso 4
87       * retrona una imagen corregida aplicando la matriz de distorcion
88       * @param img imagen original
89       * @return imagen corregida
90       */
91      public Mat undistort(Mat img){
92          Mat undistored = new Mat();
93          Imgproc.undistort(img, undistored, cameraMatrix, distCoeffs);
94          return undistored;
95      }
96      /**
97       * valida si la imagen contiene el patron de tablero de ajedrez
98       * aplica cornerSubPix para mejorar el resultado
99       * @param gray imagen en escala de grises
100      * @param corners matriz de esquinas
101      * @return
102      */
103     private boolean getCorners(Mat gray, MatOfPoint2f corners) {
104         if (!Calib3d.findChessboardCorners(gray, patternSize,corners,
        FLAGS_CORNER))
105             return false;
106         Imgproc.cornerSubPix(gray, corners, winSize, zoneSize,criteria);
107         return true;
108     }
```

```
109     /**
110      * mapeo de vertices de los cuadros encontrados con su distancia por lado
111      * @return
112      */
113     private MatOfPoint3f getCorner3f() {
114         MatOfPoint3f corners3f = new MatOfPoint3f();
115         Point3[] vp = new Point3[(int) (patternSize.height * patternSize.width
    )];
116         int cnt = 0;
117         for (int i = 0; i < patternSize.height; ++i)
118             for (int j = 0; j < patternSize.width; ++j, cnt++)
119                 vp[cnt] = new Point3(j * squareSize,i * squareSize, 0.0d);
120         corners3f.fromArray(vp);
121         return corners3f;
122     }
123 }
```

Índice de algoritmos 5.2: Clase en Java para calibración de la cámara

```
1  private Point vertices[]=new Point[4];
2  private Mat prespectiva(Mat scr){
3    double dst_w=Math.max(vertices[0].distance(vertices[1]),vertices[3].distance
       (vertices[2]));
4    double dst_h=Math.max(vertices[0].distance(vertices[3]),vertices[1].distance
       (vertices[2]));
5    List<org.opencv.core.Point> srcPts=new ArrayList<org.opencv.core.Point>();
6    List<org.opencv.core.Point> dstPts=new ArrayList<org.opencv.core.Point>();
7    for(Point p:vertices){
8      srcPts.add(new org.opencv.core.Point(p.x,p.y));
9    }
10   dstPts.add(new org.opencv.core.Point(0,0));
11   dstPts.add(new org.opencv.core.Point(dst_w-1,0));
12   dstPts.add(new org.opencv.core.Point(dst_w-1,dst_h-1));
13   dstPts.add(new org.opencv.core.Point(0,dst_h));
14   Mat srcMat=Converters.vector_Point2f_to_Mat(srcPts);
15   Mat dstMat=Converters.vector_Point2f_to_Mat(dstPts);
16
17   Mat prespectiveTransform=Imgproc.getPerspectiveTransform(srcMat, dstMat);
18
19   Mat dst=new Mat((int)dst_h,(int)dst_w,CvType.CV_32FC2);
20   Imgproc.warpPerspective(scr, dst, prespectiveTransform, new Size(dst_w,dst_h
       ));
21   return dst;
22 }
```

Índice de algoritmos 5.3: Método en Java para correccion de perspectiva

```
1  private Image onePixel(BufferedImage image){
2    int w=image.getWidth();
3    int h=image.getHeight();
4    Color c;
```

```
5     for(int i=0;i<w;i++){
6       int ini=0;
7       for(int j=0;j<h;j++){
8         c=new Color(image.getRGB(i, j));
9         if(c.getRed()>250&&ini==0){
10          ini=j;
11        }else if(c.getRed()<250&&ini!=0){
12          image.setRGB(i, ini+((j-ini)/2), Color.black.getRGB());
13          ini=0;
14          break;
15        }
16      }
17    }
18    return image;
19  }
```

Índice de algoritmos 5.4: Método en Java para detección de una sola linea

```
1  private int[] onePixel(BufferedImage image){
2    int w=image.getWidth();
3    int renglon[]=new int[w];
4    int h=image.getHeight();
5    Color c;
6    boolean vacio=true;
7    for(int i=0;i<w;i++){
8      if(i<cx0||i>cx1)continue;
9      int ini=0;
10     for(int j=h-1;j>=0;j--){
11       if(j<cy0||j>cy1)continue;
12       c=new Color(image.getRGB(i, j));
13       if(c.getRed()>250&&ini==0){
14         ini=j;
15       }else if(c.getRed()<250&&ini!=0){
16         image.setRGB(i, ini+((j-ini)/2), Color.black.getRGB());
17         renglon[i]=ini+((j-ini)/2);
18         ini=0;
19         vacio=false;
20         break;
21       }
22     }
23   }
24   if(vacio)return null;
25   return renglon;
26 }
```

Índice de algoritmos 5.5: Método en Java para detección de una sola linea con detalle de detección

```
1  void digitalizar(){
2    com.setOk(false);
3    com.send("h");
```

```
4     while (!com.isOk()) delay(250);
5     //ciclo de escaneo
6       com.send("L");
7       startSCR();
8       for(int i=0;i<PASOS;i++){
9         com.setOk(false);
10        com.send("fffffffff");
11        waitOk();
12        delay(1000);
13        pantalla1.repaint();
14        rengVer=p.getReng();
15        printSCR(rengVer,i*10,getMax(rengVer));
16      }
17      endSCR();
18      com.send("l");
19      com.setOk(false);
20      com.send("h");
21      while (!com.isOk()) delay(250);
22  }
```

Índice de algoritmos 5.6: Método en Java para digitalización

```
1  ;FLAVOR:RepRap
2  ;TIME:2413
3  ;Filament used: 4.76284m
4  ;Layer height: 0.3
5  ;Generated with Cura_SteamEngine 2.5.0
6  M104 S220
7  M109 S220
8  ; -- START GCODE --
9  G21                     ;set units to millimetres
10 G90                     ;set to absolute positioning
11 M106 S0                 ;set fan speed to zero (turned off)
12 G28 X0 Y0               ;move to the X/Y origin (Home)
13 G28 Z0                  ;move to the Z origin (Home)
14 G1 Z15.0 F1200          ;move Z to position 15.0 mm
15 G92 E0                  ;zero the extruded length
16 G1 E20 F200             ;extrude 20mm of feed stock
17 G92 E0                  ;zero the extruded length again
18 G1 F7200                ;set feedrate to 120 mm/s
19 ; -- end of START GCODE --
20 ;LAYER_COUNT:57
21 ;LAYER:0
22 M107
23 G0 F3000 X170.561 Y230.884 Z0.2
24 ;TYPE:SKIRT
25 G1 X170.89 Y230.656 E0.01331
26 G1 X176.875 Y226.987 E0.2468
27 G1 X177.386 Y226.727 E0.26587
28 G1 X185.367 Y223.439 E0.55297
29 G1 X185.827 Y223.285 E0.5691
```

```
G1 X190.099 Y222.168 E0.71596
G1 X190.405 Y222.102 E0.72638
G1 X194.166 Y221.461 E0.85327
G1 X194.879 Y221.412 E0.87704
G1 X198.341 Y221.516 E0.99224
G1 X201.433 Y221.512 E1.09508
G1 X201.615 Y221.516 E1.10114
G1 X210.266 Y221.942 E1.38922
G1 X210.379 Y221.95 E1.39298
G1 X212.557 Y222.126 E1.46566
G1 X212.809 Y222.155 E1.4741
G1 X216.703 Y222.748 E1.60511
G1 X217.159 Y222.848 E1.62063
G1 X220.647 Y223.854 E1.74137
G1 X225.965 Y225.134 E1.9233
G1 X231.318 Y226.104 E2.10424
G1 X236.745 Y226.775 E2.28612
G1 X242.175 Y227.136 E2.46712
G1 X247.635 Y227.189 E2.64873
G1 X253.074 Y226.934 E2.82983
G1 X258.506 Y226.371 E3.01147
G1 X263.899 Y225.501 E3.19316
G1 X266.511 Y224.966 E3.28184
G1 X271.846 Y223.633 E3.46474
G1 X277.051 Y222.015 E3.64603
G1 X282.16 Y220.104 E3.82745
G1 X287.152 Y217.907 E4.00885
G1 X292.006 Y215.436 E4.19001
G1 X296.719 Y212.689 E4.37145
G1 X301.264 Y209.685 E4.55265
G1 X305.716 Y206.365 E4.73737
G1 X305.941 Y206.21 E4.74646
G1 X307.478 Y205.236 E4.80698
G1 X307.691 Y205.111 E4.81519
G1 X309.272 Y204.255 E4.87499
G1 X309.562 Y204.115 E4.8857
G1 X313.115 Y202.585 E5.01436
G1 X313.604 Y202.415 E5.03158
G1 X314.82 Y202.088 E5.07346
G1 X315.067 Y202.03 E5.0819
G1 X316.984 Y201.657 E5.14686
G1 X317.152 Y201.629 E5.15252
G1 X319.755 Y201.249 E5.24002
G1 X320.1 Y201.215 E5.25155
G1 X324.302 Y201.01 E5.39147
G1 X324.838 Y201.024 E5.40931
G1 X325.366 Y201.117 E5.42714
G1 X325.874 Y201.288 E5.44496
G1 X326.352 Y201.533 E5.46283
G1 X326.787 Y201.845 E5.48063
G1 X327.172 Y202.219 E5.49849
```

```
G1 X327.497 Y202.646 E5.51633
G1 X327.765 Y203.139 E5.535
G1 X328.245 Y204.215 E5.57419
G1 X328.369 Y204.533 E5.58554
G1 X329.519 Y207.949 E5.70542
G1 X329.601 Y208.233 E5.71525
G1 X330.734 Y212.809 E5.87205
G1 X330.772 Y212.979 E5.87784
G1 X331.337 Y215.851 E5.97519
G1 X331.359 Y215.974 E5.97935
G1 X331.937 Y219.565 E6.10032
G1 X331.959 Y219.72 E6.10553
G1 X332.811 Y227.023 E6.35008
.
.
.
G1 F3600 E579.96842
G0 F7200 X266.55 Y217.4
G0 X271.241 Y215.683
G0 X274.848 Y214.146
G0 X274.843 Y212.076
G0 X256.974 Y217.751
G0 X257.094 Y219.011
G0 X257.015 Y219.026
G0 X257.354 Y219.17
;TYPE:SKIN
G1 F3600 E586.46842
G1 F1426.1 X257.31 Y219.126 E586.46997
;TIME_ELAPSED:2413.854481
G1 F3600 E579.96997
M107
; -- END GCODE --
M104 S0                     ;set extruder temperature to zero (turned off)
G91                         ;set to relative positioning
G1 Z10 E-20 F300            ;retract the filament a bit to release some of the
    pressure
G90                         ;set to absolute positioning
G28;
M84                         ;turn off steppers
M104 S0
;End of Gcode
;SETTING_3 {"global_quality": "[general]\\nversion = 2\\nname = empty\\
    ndefiniti
;SETTING_3 on = bq_hephestos\\n\\n[metadata]\\ntype = quality_changes\\
    nquality_
;SETTING_3 type = normal\\n\\n[values]\\nadhesion_type = skirt\\nbrim_width =
    3\
;SETTING_3 \ninfill_pattern = zigzag\\ninfill_sparse_density = 45\\
    nlayer_0_z_ov
;SETTING_3 erlap = 0.1\\nlayer_height = 0.3\\nlayer_height_0 = 0.2\\
    nmaterial_pr
```

```
126 ;SETTING_3 int_temperature = 220\\nretraction_speed = 60\\nskirt_line_count =
        2\
127 ;SETTING_3 \nspeed_layer_0 = 50.0\\nspeed_print = 60\\nspeed_topbottom =
        50.0\\n
128 ;SETTING_3 speed_travel_layer_0 = 50\\nspeed_wall = 50.0\\nsupport_enable =
        Fals
129 ;SETTING_3 e\\nsupport_infill_rate = 7\\nsupport_pattern = lines\\
        ntop_bottom_pa
130 ;SETTING_3 ttern = zigzag\\n\\n"}
```

Índice de algoritmos 5.7: Fragmento de código G para plantilla 3D