Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación
Laboratorio de Computación Gráfica

Escáner 3D de Bajo Costo empleando WebCams

Trabajo Especial de Grado
Presentado ante la ilustre
Universidad Central de Venezuela

Por la Bachiller:
Astrid Celeste Narváez González
C.I.: V-17.402.982

Para optar al título de
Licenciada en Computación

Tutor:
Prof. Esmitt Ramírez

Caracas, Octubre 2010

Acta

Quienes suscriben, miembros del Jurado designado por el Consejo de la Escuela de Computación, para examinar el Trabajo Especial de Grado presentado por la bachiller Astrid Celeste Narváez González, con el título **"Escáner 3D de Bajo Costo empleando WebCams",** a los fines de optar al Título de Licenciada en Computación, dejan constancia de lo siguiente:

Leído como fue, dicho trabajo por cada uno de los miembros del Jurado, se fijó el día 19 de Octubre del 2010 a la 2:00 p.m., para que su autor lo defendiera en forma pública, lo que hizo en el Centro de Computación Gráfica de la Escuela de Computación, mediante la presentación oral de su contenido, luego de lo cual, respondió a las preguntas formuladas por el jurado y público en general. Finalizada la Defensa Pública del Trabajo Especial de Grado, el Jurado decidió aprobarlo.

En fe de lo cual se levanta la presente Acta, en Caracas a los 19 días del mes de Octubre del 2010, dejándose también constancia de que actuó como Coordinador del Jurado el Profesor Tutor Esmitt Ramírez.

_____

Profesor Esmitt Ramírez (Tutor)

_____ _____

Profesor Ernesto Coto Profesor Robinson Rivas

Nota Preliminar

Agradecimientos

A lo largo del camino que he recorrido han sido varias las personas que han jugado un papel importante en mi vida, personas que me han acompañado en las buenas y en las malas, que me han hecho sonreír cuando quizá quería llorar y que me han brindado lo mejor de sí en cada momento.

Primero que todo debo agradecer a Dios por darme la vida, por permitirme vivir todos y cada uno de los momentos que he vivido, por guiar mis pasos y poner en el camino a todas las personas que me han acompañado.

A ti mamá, gracias por estar siempre presente, por estar a mi lado, por cuidarme y por dar todo de ti para sacarnos adelante a mi hermano y a mí. A ti manito, que aunque estés lejos es como si estuvieras aquí conmigo. Gracias por estar presente en todo momento, por alegrarte de mis triunfos y acompañarme y levantarme en mis derrotas, gracias por ser tú mi hermano, es un orgullo para mí que lo seas.

A ti papá, si algún día llegas a leer esto. Gracias, porque fue contigo con quien di mis primeros pasos en una computadora y quien me enseñó tres principios básicos que me permitieron llegar a donde estoy: constancia, paciencia y perseverancia.

Gracias a ti Engel que siempre has estado a mi lado, acompañándome cuando participaba en un maratón, cuando participaba en algún concurso, cuando me tocaba trasnocharme para hacer algún proyecto tú siempre estuviste ahí, dándome ánimos y siempre confiando en mí y en lo que era capaz de lograr, incluso cuando yo misma no lo hacía.

Diana, amiga y hermana, aunque estás lejos desde hace ya varios años, gracias. Gracias por leer mis correos todos los días. Por siempre tener esas palabras que sólo una persona que te conoce, como me conoces tú, pueden impulsarte a seguir adelante, pueden borrar una molestia o tristeza y convertirla en una sonrisa, por estar siempre que necesitaba a alguien a mi lado, aunque hubiese muchos kilómetros de distancia entre nosotras.

Mi cu, tú también fuiste importante en este camino. Siempre pendiente de mí, cuidándome y dándome lo mejor de ti, con el buen ánimo y las buenas energías que te caracterizan. Gracias mi Ann.

A usted profe Esmitt, que compartió conmigo la última etapa de este recorrido. Gracias por impulsarme a ser mejor cada día, por intentar que no me estresara y por ser positivo y entusiasta en todo momento, eso logró que no saliera corriendo en varias oportunidades.

A mi familia, amigos y profesores que me acompañaron en este recorrido, gracias, porque cada uno de ustedes ha influido en mí de alguna manera. Estas páginas quedan cortas para poder nombrar y reconocer a todos los que han estado presentes en mi vida y me han hecho ser quien soy, me han llenado y fortalecido tanto personal como profesionalmente y me han hecho crecer cada día para ser una mejor persona.

Mi alma mater, la Universidad Central de Venezuela, has servido de inspiración para muchos y has llenado de grandeza este país, gracias.

Índice General

Introducción

En el mundo actual cada vez cobra mayor importancia contar con información más precisa y parecida a la realidad posible, con el objetivo de generar mayor información a partir de ella, obtener modelos para estudio, con fines recreativos y culturales, entre otros.

Una de las maneras de obtener este tipo de información es a través de un escáner 3D, que permite captar objetos de la realidad e introducirlos en un computador. Gracias a ello, hoy en día es posible realizar estudios médicos muy precisos para el diagnóstico de enfermedades o para la reconstrucción de miembros del cuerpo y prótesis dentales.

Al mismo tiempo, en el área del entretenimiento, los escáneres 3D permiten la creación de películas y videojuegos mucho más reales e interactivos. En el campo industrial, permite escanear y modelar piezas mecánicas. También, es posible recrear museos virtuales en donde las obras de artes puedan ser observadas desde todos sus ángulos. Además, una vez que ha sido escaneado un objeto su estructura y características pueden ser modificadas a través de aplicaciones que manejen información 3D para luego obtener una nueva versión del mismo.

En el presente trabajo se muestra el diseño e implementación de un prototipo de escáner 3D de bajo costo que puede ser accesible para usuarios comunes. Para ello, se capturan varias imágenes estéreo de un objeto a través de dos cámaras web conectadas a un computador, y se utilizan diversos algoritmos de geometría computacional.

El documento está organizado en los siguientes capítulos: El Capítulo 1 presenta los problemas que han surgido alrededor de este tema y los motivos por los cuáles se desarrolla esta investigación. Además se definen los objetivos a alcanzar. El Capítulo 2 se presenta una investigación detallada del proceso que involucra la reconstrucción 3D de un objeto, detallando conceptos, bases teóricas y métodos involucrados. Igualmente, se explican los detalles de algunos escáneres 3D de bajo costo y las áreas donde pueden ser utilizados en el Capítulo 3. En el Capítulo 4 se describe el diseño de la solución propuesta, mostrando la estructura de la aplicación y los recursos utilizados. Además se detalla la implementación de la misma, considerando los algoritmos y técnicas empleadas.

Los resultados experimentales obtenidos y el análisis de los mismos se encuentran en el Capítulo 5. Por último se dan a conocer las conclusiones finales y se proponen algunos trabajos a desarrollar en un futuro.

Capítulo 1 Planteamiento del Problema

La reconstrucción de geometría 3D es un proceso a través del cual, objetos reales son reproducidos en un computador, manteniendo sus características físicas (dimensiones, volumen y forma). Existen diversas técnicas de reconstrucción y métodos de mallado 3D, cuyo objetivo principal es realizar la conexión del conjunto de puntos representativos del objeto en forma de elementos de superficie, ya sean triángulos, cuadrados o cualquier otra forma geométrica.

Para realizar una reconstrucción 3D de la geometría de un objeto, se ha de utilizar un dispositivo de captura de datos conocido como escáner 3D. Dependiendo del tipo de escáner seleccionado, se requiere conocer a priori las características del objeto, ya que, al utilizar los escáneres de contacto con la superficie a escanear, por ejemplo, no es posible hacer una toma de un rostro ó de objetos que se puedan deformar fácilmente. Entonces, en este caso, sería ideal utilizar un escáner que no tenga contacto con el objeto.

Al utilizar un escáner sin contacto, se debe seleccionar adecuadamente cuál se va a emplear de acuerdo a varios factores: tamaño, permeabilidad y uniformidad de la textura que cubre al objeto, ambiente controlado de escaneo, entre otros. En años recientes han sido desarrollados diversos dispositivos, con características y requerimientos diferentes. El escáner conocido como Milkscanner [1], por ejemplo, requiere sumergir el objeto en un líquido, pudiendo ocasionar daños a la estructura original del mismo. Otro problema es generado por el escáner de triangulación, el cual requiere la utilización de un láser que no puede ser empleado en superficies sensibles a la acción de éste (e.g. ojos de un rostro).

Este trabajo tiene como finalidad realizar un escáner 3D sin contacto y de bajo costo, que no ocasione daño alguno a la superficie del objeto, utilizando para ello un par de cámaras web, y que permita reproducir el mallado 3D del mismo.

Objetivo General

Construir un escáner sin contacto de bajo costo empleando un par de WebCams.

Objetivos Específicos

- Utilizar un par de WebCams comerciales de bajo costo con una resolución de, al menos, 2.0 megapíxeles para realizar capturas estereóscopicas del objeto en cuestión.
- Aplicar algoritmos de mejoramiento de las imágenes sobre las tomas realizadas.
- Realizar la segmentación de las imágenes capturadas para obtener el objeto a estudiar.
- Emplear el algoritmo de triangulación de Delaunay [2], para la construcción de un objeto escaneado.
- Crear un ambiente semi-controlado para la adquisición de las imágenes.
- Obtener un archivo en formato .OFF con la solución del proceso de reconstrucción 3D.

Adicionalmente, se construirá un prototipo que proveerá funcionalidades que le permitan al usuario interactuar con la aplicación, donde estarán implementados los algoritmos necesarios para realizar un escaneo 3D de un objeto real.

El prototipo utilizará dos cámaras web, conectadas a un computador, que realizarán las capturas del objeto en la escena tomando como referencia el modelo estereoscópico biológico. Se tomarán las imágenes bajo un ambiente controlado sobre una base que realiza rotaciones sobre su propio eje, según una medida angular preestablecida, de tal manera que el objeto alcance una rotación completa de 360º. Luego, se ejecutará el proceso de calibración de las cámaras y el procesamiento de las imágenes adquiridas utilizando un conjunto de algoritmos que permitirán obtener la reconstrucción 3D del objeto escaneado.

Capítulo 2 Visión Estereoscópica en la Reconstrucción 3D

A lo largo del proceso de reconstrucción 3D de un objeto se deben tomar en cuenta diversos elementos, tales como: la webcam como dispositivo de captura de datos utilizado, los métodos empleados en el procesamiento de las imágenes y aquellos utilizados para la representación tridimensional del objeto en cuestión, entre otros. A continuación estudiaremos estos conceptos para el desarrollo de dicho trabajo.

2.1. Imagen

En su sentido más amplio, se denomina imagen a la representación figurativa de una cosa. Es la representación de una realidad captada a través de los sentidos. Las imágenes son captadas por nuestra vista y permanecen allí, pudiendo luego plasmarse, por ejemplo, sobre un lienzo, un papel, etc.

Pueden también ser captadas por una lente óptica o reflejadas en un espejo. Son entonces, copias de las realidades más o menos fidedignas, ya que no es lo mismo una foto que un dibujo, que captan sus características esenciales, y que pueden dejar de lado algunas características de menor importancia.

Además de las imágenes físicas visibles, también existen físicas invisibles, con las mismas características de las visibles pero que quedan fuera del rango al que los seres humanos son sensibles, tales como las imágenes infrarrojas o ultravioletas, que para ser captadas requieren el uso de filtros especiales. También hay que acotar que, las imágenes físicas visibles, pueden ser permanentes como un póster o un documento impreso, o transitorias como las generadas en un monitor.

En términos informáticos, una imagen es un caso particular de señal, una función que especifica una determinada distribución de intensidades lumínicas [3]. Más concretamente, una imagen se entiende como la serie de valores atribuidos a una función que asigna a todos los puntos de un dominio un valor de intensidad determinado.

Por ejemplo, en una imagen 2D monocromática este valor vendría dado por una función simple de dos variables $f(x,y)$ en donde x,y denotan las coordenadas espaciales referidas a un ámbito espacial determinado, y f un valor correspondiente en cada punto que es proporcional a la intensidad de iluminación de ese punto o nivel de gris, en el caso de una imagen acromática.

Por otro lado, en el caso de imágenes cromáticas, este valor vendría dado por tres funciones simples de dos variables $f_r(x,y)$, $f_g(x,y)$ y $f_b(x,y)$, que expresan la intensidad de iluminación de un punto x,y, en el mismo ámbito, para los tres componentes cromáticos primarios: rojo, verde y azul.

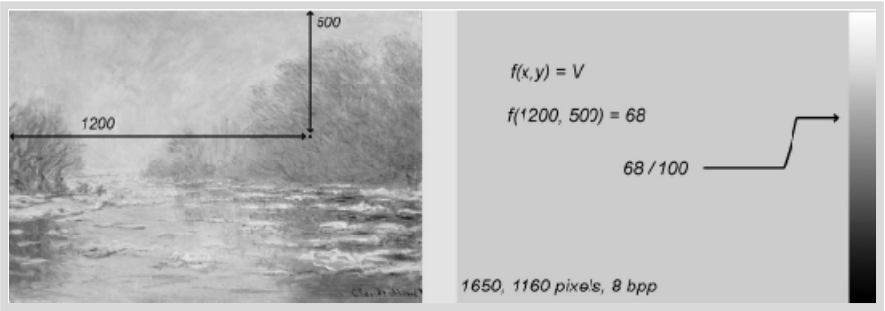


Figura 1 Imagen como función simple de dos variables

La Figura 1 ilustra lo mencionado anteriormente, se tiene un tamaño de 1650 puntos en sentido horizontal y 1160 en sentido vertical para la imagen. El punto marcado tiene las coordenadas $(1200,500)$. La función asociada, asigna a este punto el valor de 68 sobre un rango de 100 valores posibles $f(x,y) = 68$, lo que refleja la intensidad de iluminación en ese punto en específico.

Una imagen digital 2D consta, entonces, de celdas o puntos llamados píxeles, ordenados en líneas y columnas [4]. Entre los aspectos a considerar en una imagen digital se tienen:

2.1.1. Profundidad de color: Es el número de bits que definen cada píxel y determina el máximo número de colores que puede tener. Por ejemplo, en escala de grises cada punto de la imagen se almacena en 1 Byte (1 Byte = 8 bits), donde su valor numérico representa su tono, que puede oscilar entre blanco (255) y negro (0). Por otro lado, una imagen a color se genera con sus componentes RGB por síntesis aditiva. Así pues, la imagen a color se compone de tres imágenes. Cada una es una imagen en

escala de grises, pero como va asociada a cada uno de los colores primarios, al mostrarla el computador la colorea adecuadamente. La suma de las tres, por síntesis aditiva, compone la imagen final. Como se muestra en la Figura 2a la calidad de la imagen es mucho mejor que la de la Figura 2b pues esta última utiliza una menor cantidad de colores.

a) 256 colores b) 16 colores

Figura 2 Diferencia visual al emplear diversos valores para la profundidad de color

2.1.2. Tamaño de archivo: Cantidad de información que contiene, puede ser medida en bits, bytes o algún múltiplo. Una imagen en color en las mismas condiciones que una en escala de grises no tiene el mismo tamaño, ya que, si es en color RGB (de 24 bits de profundidad) se compone de tres canales, por tanto tiene el triple de información que la misma en escala de grises, 3 bytes en vez de 1 por píxel.

2.1.3. Resolución: Es la medida de cantidad de píxeles por unidad de longitud, comúnmente píxeles por pulgadas. La resolución es la relación entre las dimensiones digitales (las medidas en píxeles) y las físicas (las que tiene una vez impresa). Se suele abreviar como ppp (puntos por pulgada) o dpi (*dot per inch*). Como la resolución mide el número de píxeles por longitud, se deduce que a mayor resolución, mayor número de puntos en el mismo espacio y, por ello, mayor definición.

2.2. Visión Estereoscópica

La visión estereoscópica constituye uno de los procedimientos para la obtención de la forma de los objetos en una escena. En este caso, la forma se determina a través de la distancia de los objetos en relación con un sistema de referencia.

La visión estereoscópica artificial utiliza como referencia el modelo estereoscópico biológico [5]. En estos sistemas el desplazamiento relativo de los ojos permite obtener la profundidad de los objetos o tercera dimensión mediante un simple proceso de triangulación a partir de las dos imágenes generadas por el mismo objeto de la escena 3D en cada ojo. Esto es posible porque el hecho de que los ojos estén desplazados entre sí, hace que las imágenes de los objetos en ellos se muestren desplazadas según la distancia de los objetos a los ojos.

En la Figura 3 a) se muestra un sistema estereoscópico biológico observando dos objetos (estrella y triángulo) a distintos niveles de profundidad, en las correspondientes retinas de los dos ojos se obtienen las respectivas imágenes. Si se solapan ambos ojos se obtiene la imagen de la Figura 3 b), en la que se observa que la separación relativa entre las imágenes de los dos triángulos es menor que la separación relativa entre las imágenes de las estrellas. Esto se debe al hecho de que la estrella en la escena 3D se encuentra más próxima a los ojos que el triángulo.

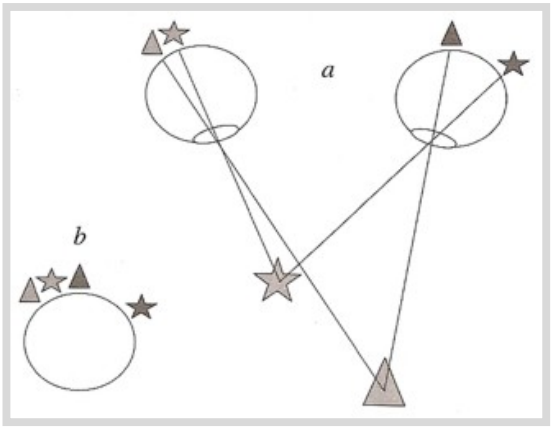


Figura 3 Sistema de Visión Estereoscópico Biológico, con dos objetos en la escena 3D (estrella y triángulo)

En visión estereoscópica artificial se utilizan dos o más cámaras separadas entre sí una cierta distancia relativa con las que se obtienen las correspondientes imágenes. Generalmente se utilizan dos como en el modelo biológico. El procedimiento consiste en captar dos o más imágenes de una misma escena. Cada imagen es capturada desde una posición de las cámaras ligeramente diferente a las anteriores, presentándose las imágenes ligeramente separadas entre sí, siendo éste el fundamento básico de la visión estereoscópica. Este hecho va a permitir la obtención de la distancia a la que se encuentra un determinado objeto.

La obtención de las imágenes de la escena ligeramente desplazadas se puede obtener por alguno de los procedimientos siguientes:

- Alineando dos o más cámaras de forma que se sitúen ligeramente desplazadas en el espacio
- Desplazando una única cámara una cierta distancia y captando las imágenes en las diferentes posiciones de desplazamiento.

Un sistema convencional está caracterizado por un par de cámaras con sus ejes ópticos mutuamente paralelos y separados por una distancia horizontal que se denomina línea base, en la Figura 4 es el parámetro *b*. Las cámaras tienen sus ejes ópticos perpendiculares a la línea base y sus líneas de exploración o epipolares paralelas a la línea base. Las líneas epipolares son líneas uniendo las imágenes izquierda y derecha de un mismo punto en la escena.

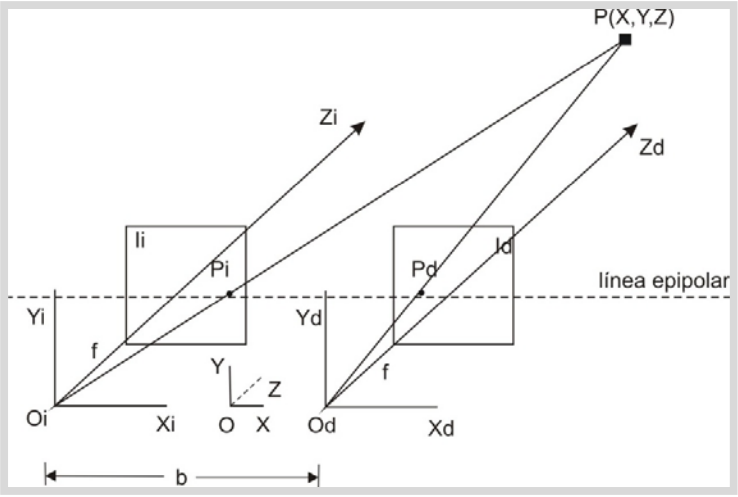


Figura 4 Geometría de un par de cámaras en estéreo con ejes ópticos paralelos

Este sistema de ejes ópticos paralelos se caracteriza porque el desplazamiento entre los centros ópticos de las dos cámaras es horizontal, lo que implica que las imágenes de un punto determinado de la escena captado por ambas cámaras sólo difiere en la componente horizontal.

Cuando se utiliza esta geometría, la profundidad a la que se encuentra un objeto, resulta inversamente proporcional a la disparidad *d* de la imagen y para una profundidad dada, a mayor *b* mayor disparidad, lo que sugiere que la línea base puede incrementarse para mejorar la exactitud de la profundidad medida, pero ello lleva consigo el hecho de que ahora ambas imágenes tienen menos características comunes, es decir menos bordes o regiones procedentes del objeto de la escena, debido a las desapariciones y oclusiones de las imágenes del objeto.

El proceso de reconstrucción 3D mediante un par estereoscópico de cámaras, comprende las siguientes fases:

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

2.3. Webcams

Son dispositivos de captura digital de video de baja resolución que pueden tomar imágenes y transmitirlas a través de Internet, ya sea a una página web o a otra u otras computadoras de forma privada. Por lo general, pueden transmitir imágenes en vivo pero también pueden capturar imágenes o pequeños videos (dependiendo del software de la cámara web) que pueden ser grabados y transmitidos por Internet. Este dispositivo se clasifica como de entrada, ya que, por medio de él podemos transmitir imágenes hacia el computador. Un modelo de este tipo de dispositivo se presenta en la Figura 5.

Figura 5 Webcam Notebook Desktop 480K DAG-402

La instalación básica de una cámara web consiste en una cámara digital conectada a una computadora, normalmente a través del puerto USB. Esta cámara es como el resto de las cámaras digitales y, lo que realmente le da el nombre de cámara web, es el software que la acompaña.

Las cámaras web normalmente están formadas por una lente, un sensor de imagen y la circuitería necesaria para manejarlos. Existen distintos tipos de lentes, siendo las lentes plásticas las más comunes. Los sensores de imagen pueden ser CCD (*Charge Coupled Device*) o CMOS (*Complementary Metal Oxide Semiconductor*). Las cámaras web para usuarios medios suelen ofrecer una resolución VGA (640x480 píxeles) con una tasa de unos 30 fotogramas ó cuadros por segundo, si bien en la actualidad están ofreciendo resoluciones medias de 1 a 2 megapíxeles.

La circuitería electrónica es la encargada de leer la imagen del sensor y transmitirla al computador. Algunas cámaras usan un sensor CMOS integrado con la circuitería en un único chip de silicio para ahorrar espacio y costes.

2.3.1. Tipos de Sensores

Actualmente existen dos tecnologías para la fabricación de sensores destinados a las cámaras digitales [6]: los CCD (*Charge Coupled Device* o Dispositivo de Cargas Acopladas), que fueron los primeros en aparecer en el mercado, y los CMOS (*Complementary Metal Oxide Semiconductor* o Semiconductor de Óxido de Metal Complementario). Ambos están fabricados con materiales semiconductores, concretamente de Metal-Óxido (MOS) y están estructurados en forma de una matriz, con filas y columnas.

Su función es la de acumular una carga eléctrica en cada una de las celdas de esta matriz. La carga eléctrica almacenada dependerá, en todo momento, de la cantidad de luz que incida sobre ella. A mayor intensidad luminosa, mayor será la carga que ésta adquiera.

Aunque tienen la misma finalidad, existen diferencias notables entre ambas tecnologías:

- El sensor puede visualizarse como una matriz con miles (o millones) de celdas solares en miniatura, donde cada una de las celdas convierte la luz de una pequeña porción de la imagen (un píxel) en electrones. Lo siguiente es realizar la lectura del valor correspondiente a cada una de las celdas. En un sensor CCD, la información de cada una de las celdas es enviada a través del chip hacia una de las esquinas del arreglo, y ahí un convertidor análogo a digital traduce el valor de cada una de las celdas. Lo que mantiene una estructura muy simple, pero que tiene como inconveniente la necesidad de un chip adicional que se encargue del tratamiento de la información proporcionada por el sensor, lo que se traduce en un gasto mayor y equipos más grandes. La estructura de este sensor puede observarse en la Figura 6.

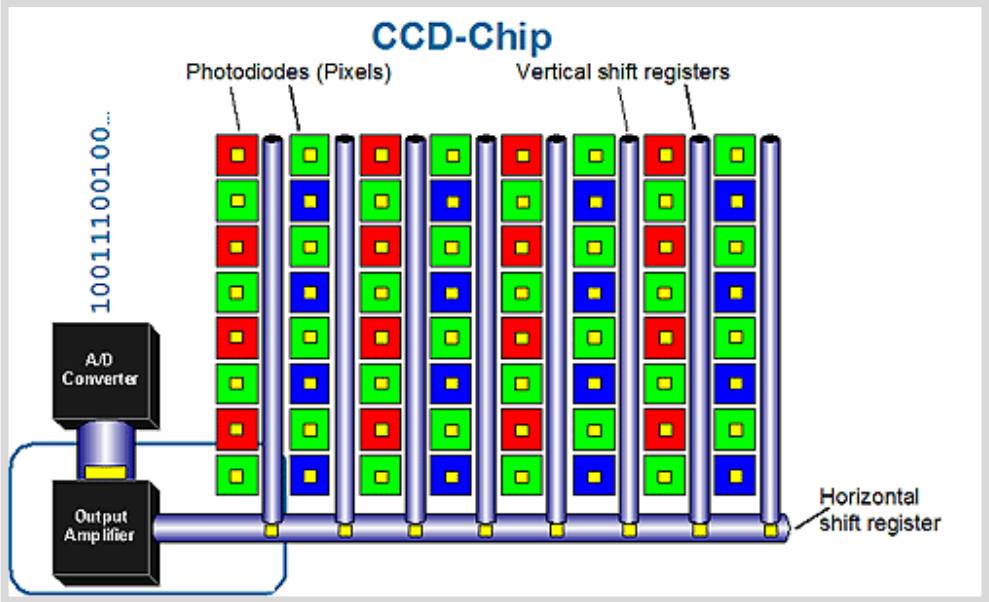


Figura 6 Estructura de un sensor CCD (*Charge Coupled Device*)

Por otro lado, en un sensor CMOS cada celda es independiente. La diferencia principal es que aquí la digitalización de los píxeles se realiza internamente en unos transistores que lleva cada celda, por lo que todo el trabajo se lleva a cabo dentro del sensor y no es necesario un chip externo encargado de esta función. Con esto se consigue reducir costes y emplear equipos más pequeños. La estructura interna del sensor CMOS puede observarse en la Figura 7.

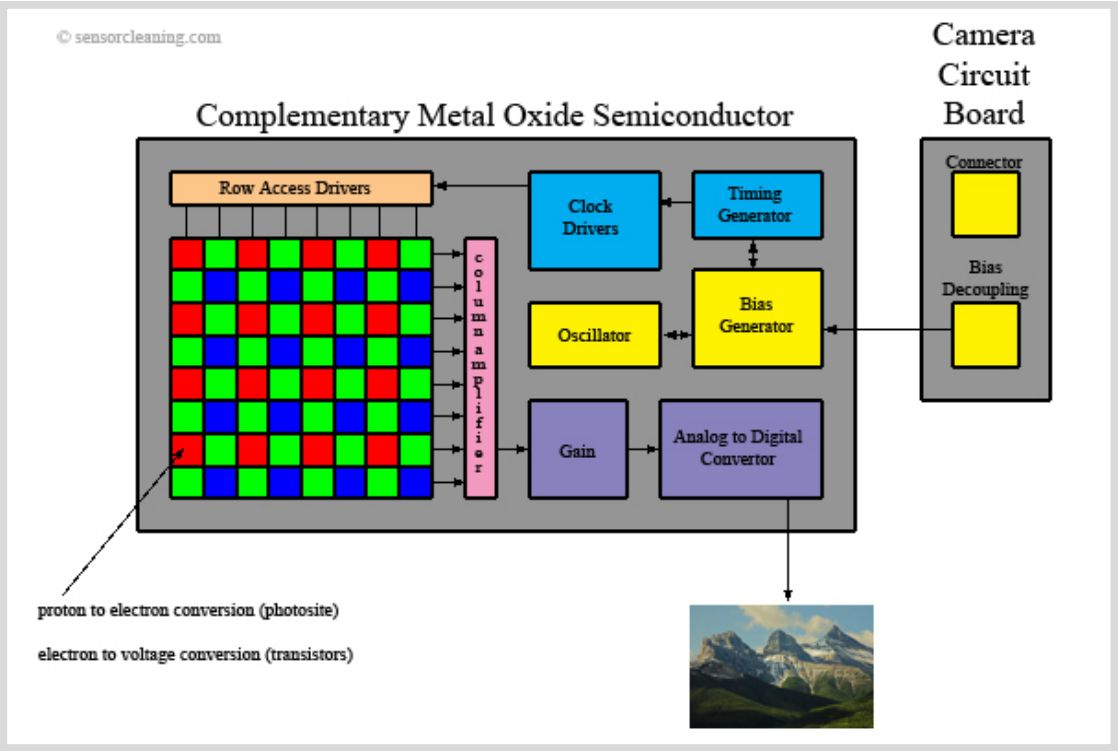


Figura 7 Estructura de un sensor CMOS (*Complementary Metal Oxide Semiconductor*)

- Es necesario que el sensor tenga una responsividad elevada, es decir, que sea más sensible a la luz, para que en condiciones pobres de iluminación se comporte mucho mejor. En este caso, los sensores CMOS resultan superiores a los de tecnología CCD, debido a que integran elementos amplificadores en cada celda.
- En cuanto al ruido, los sensores CCD aventajan a los de tecnología CMOS. Esto es debido a que el procesado de la señal se lleva a cabo en un chip externo, el cual puede optimizarse mejor para realizar esta función. En cambio, en el CMOS, al realizarse todo el proceso de la señal dentro del mismo sensor, produce más ruidos en la lectura.

- Al comparar la velocidad con la que se captura la imagen, se observa que los CMOS son bastante superiores a los CCD, debido a que muchas funciones, como la propia conversión analógico-digital son realizadas en el propio sensor.
- Por último, otro aspecto en el que los sensores CMOS son superiores a los CCD es en el *blooming*. Este fenómeno se produce cuando un píxel se satura por la luz que incide sobre él y a continuación empieza a saturar a los que están a su alrededor, creando efectos y patrones no deseados. Esto se da principalmente en sensores CCD, que necesitan de algunos trucos de construcción para evitarlos. En cambio, gracias a su construcción, los sensores CMOS no sufren de este defecto.

Como en tantos otros casos, no hay una tecnología superior de por sí, sino situaciones en las que cada tecnología es más adecuada

2.3.2. Funcionamiento

El funcionamiento de una webcam es simple, una cámara de vídeo toma imágenes, las pasa a un computador que las traduce a lenguaje binario y las envía a través de internet. Más específicamente:

- Una cámara toma imágenes que envía regularmente a un computador, algunas se actualizan cada pocos segundos y otras cada varias horas.
- El computador mediante hardware/software adecuado traduce las imágenes a formato binario, generalmente archivos JPEG, dada su buena relación calidad/tamaño.
- Si lo que se pretende es utilizar esas imágenes para construir un video de calidad sin saltos de imagen, se necesitará que la cámara web alcance una tasa de unos 15 a 30 cuadros por segundo.
- En los videos destinados a ser subidos en Internet o ser enviados a dispositivos móviles, es mejor una cadencia de 14 fotogramas por segundo. De esta manera se consigue ahorrar espacio y aun así seguirá teniendo calidad, aunque podrían aparecer ligeros saltos en el movimiento.

2.3.3. Calibración

El proceso de calibración de las cámaras puede dividirse en dos tareas a resolver: calibración interna y calibración externa [7]. La primera, consiste en determinar la relación entre coordenadas imagen y las direcciones de vista en el espacio de medidas de un sistema de coordenadas de cámara. Los parámetros intrínsecos caracterizan las propiedades inherentes al sistema óptico de cámara, como pueden ser longitud focal efectiva, radio de distorsión de las lentes, etc. Por consiguiente, se permite relacionar los puntos de la escena expresados en el sistema de coordenadas tridimensional relativo a la cámara, con los puntos correspondientes en la imagen expresados en un sistema de coordenadas del plano imagen (en píxeles).

Por otro lado, la calibración externa consiste en determinar la posición y orientación de la cámara respecto a un sistema de referencia preestablecido. Los parámetros extrínsecos caracterizan la calibración externa e indican, por tanto, la posición y orientación de la cámara con respecto a un sistema de coordenadas global o del mundo, permitiendo relacionar las coordenadas del mundo P con el sistema de coordenadas tridimensional asociado a la cámara P', tal como se observa en la Figura 8.

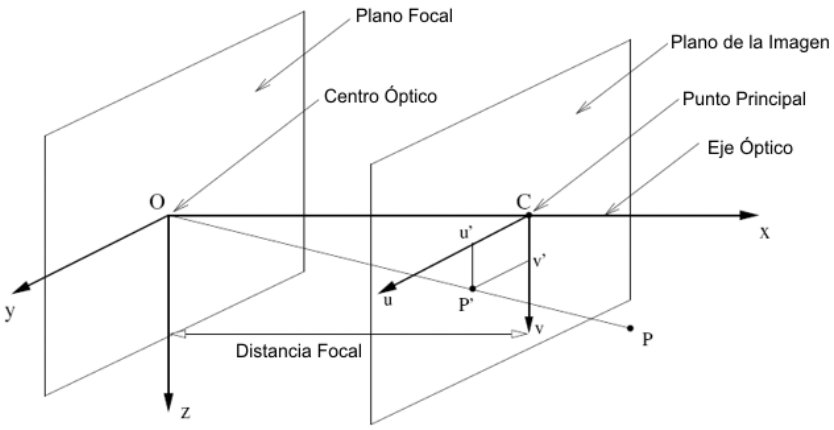


Figura 8 Sistema de Coordenadas de la Cámara y de la Imagen en el modelo de proyección

En la actualidad existen varios métodos para la calibración de una cámara. Estos métodos pueden clasificarse según diferentes criterios [8], algunos de ellos son:

- Según el método de resolución se pueden clasificar en lineales frente a no lineales o iterativos. Los métodos lineales utilizan métodos de resolución de sistemas de ecuaciones basados en mínimos cuadrados. Estos métodos obtienen una matriz de transformación que relaciona los puntos 3D en el mundo con sus proyecciones 2D en la imagen. En este caso no se calculan los parámetros que modelan la distorsión de la cámara por lo que los resultados obtenidos son bastante aproximados, sin embargo son fáciles de implementar y muy rápidos para su ejecución. Si, por el contrario, se requiere un modelo de la cámara más complejo en el que se incluyan las distorsiones que produce la cámara, es necesario minimizar índices no lineales de forma iterativa. El índice a minimizar suele incluir la distancia entre los puntos medidos en la imagen y los puntos proyectados obtenidos con el modelo de la cámara. La ventaja de estos métodos iterativos es que cualquier modelo puede ser calculado y además la exactitud del mismo aumenta con el número de iteraciones hasta que converge. Sin embargo, son mucho más lentos y necesitan partir de una buena aproximación de los parámetros para garantizar esta convergencia.
- Otra clasificación de los métodos de calibración puede realizarse basándose en el resultado de la misma. Una calibración explicita obtiene directamente los parámetros del modelo de la cámara, mientras que en una implícita se obtienen matrices de transformación que contienen el conjunto de todos los parámetros. Aunque no se conoce el valor exacto de alguno de los parámetros, los resultados pueden ser utilizados para realizar medidas 3D y la generación de coordenadas en la imagen. Los métodos implícitos no son aptos para modelar la cámara ya que los parámetros que se obtienen no corresponden con los reales de la cámara.
- Finalmente, atendiendo a las características de la plantilla que se utiliza para la calibración, existen métodos que utilizan plantillas 3D, 2D, 1D o no utilizan plantilla. Los métodos que utilizan plantillas de referencia basan la calibración de la cámara en establecer una relación entre las coordenadas conocidas de los puntos en la plantilla y las coordenadas de estos puntos en la imagen. En el caso de plantillas 3D con una sola imagen de la misma es posible realizar la calibración. En este caso la plantilla consiste en dos o tres planos ortogonales entre ellos, como se observa en la Figura 9.

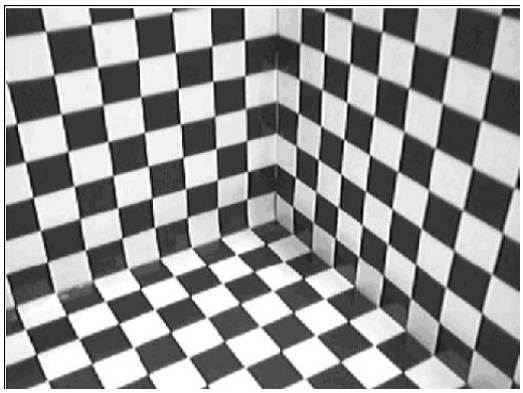


Figura 9 Plantilla 3D para la calibración de la cámara

Como se asumen los puntos en planos se evitan errores de medidas de las coordenadas de puntos en la plantilla ya que se asume la misma para todos los puntos del mismo plano. Por el contrario este tipo de calibración requiere de una costosa elaboración de la plantilla a emplearse.

En el caso de utilizar plantillas 2D es necesario tomar varias imágenes de la misma desde varias posiciones o cambiar la posición y orientación de la plantilla. No es necesario conocer las posiciones desde donde se toman las imágenes. Este método resulta más versátil ya que la elaboración de la plantilla se puede realizar fácilmente. En la Figura 10 se observa un ejemplo de este tipo de plantilla.

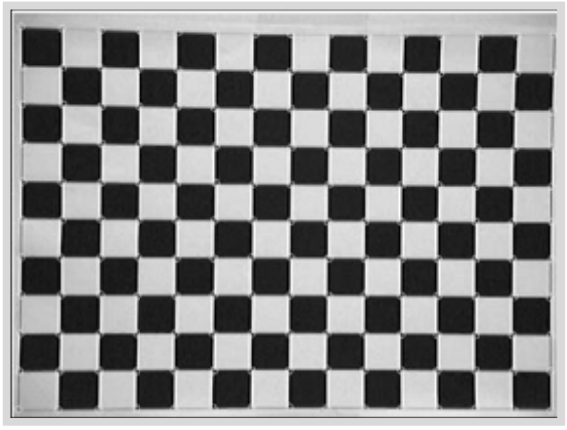


Figura 10 Plantilla 2D para la calibración de la cámara

Los métodos de calibración basados en plantillas 1D son muy útiles en el caso de calibrar sistemas con varias cámaras. En el caso de utilizar métodos basados en

plantillas 3D ó 2D, dado que es necesario que todas ellas vean varios puntos de la plantilla de calibración a la vez, resulta complicado establecer una posición para la misma a no ser que la plantilla sea transparente. Es por este motivo que el método de calibración basado en una plantilla 1D resulta atractivo a la hora de calibrar un sistema con varias cámaras.

Basándose en el estado actual de los procesos de calibración descritos hasta ahora, resulta complicado elegir un método eficiente que permita calibrar la cámara en cualquier situación. El método de Tsai [9], representa un proceso clásico de calibración basado en las medidas de las coordenadas de los puntos de una plantilla 3D respecto a un punto de referencia fijo. Este método ha sido ampliamente utilizado en el siglo pasado.

Por otro lado, el método de Zhang [10] representa una nueva era en el proceso de calibrado de la cámara. Este método utiliza las coordenadas de los puntos situados en una plantilla plana 2D tomando diferentes imágenes de la misma desde diferentes posiciones y orientaciones. De esta forma se combinan las ventajas de los métodos de calibración basados en las medidas de las coordenadas de la plantilla con las ventajas de la auto calibración en la cual no es necesario utilizar plantilla. Este modo de calibración resulta muy flexible desde el punto de vista de que tanto la cámara como la plantilla pueden ser movidas libremente y además se pueden tomar tantas imágenes como se quieran sin tener que volver a realizar medidas en la plantilla.

El método de Tsai obtiene una estimación precisa de los parámetros de la cámara si los datos de entrada están poco contaminados con ruido. Teniendo en cuenta que son necesarios al menos cien puntos en la plantilla y que las coordenadas han de estar referidas a un origen de coordenadas fijo, es imprescindible un adecuado diseño de la plantilla de calibración además de una medida exacta de las coordenadas de los puntos. A pesar de todo, la posibilidades de cometer errores en las medidas son altas. Por el contrario, el método de calibración de Zhang basado en plantilla 2D no requiere tal especial diseño de la plantilla, ni tampoco una medición tan exacta de los puntos de la misma. Además, la sensibilidad del algoritmo de calibración frente a errores en las medidas puede ser mejorada

aumentando el número de puntos en la plantilla, simplemente imprimiendo un tablero de ajedrez con mayor cantidad de cuadros.

2.4. Procesamiento Digital de Imágenes

Es el conjunto de técnicas que permite modificar una imagen digital con el objetivo de mejorarla o extraer información de ella. Por tanto, depende del problema específico a resolver el que se emplee una u otra técnica. Entre éstas, están los métodos en el dominio espacial [11], que se basan en manipulaciones directas sobre los píxeles de la imagen.

2.4.1. De Procesamiento Puntual

Las operaciones puntuales se basan en el valor de cada pixel, transformando los píxeles de la imagen al aplicar la misma función sobre cada uno de ellos [3]. Ejemplos de estas operaciones puntuales son las modificaciones de contraste y brillo en las que, al aumentar el brillo en una imagen monocromática, se suma una cantidad determinada a cada píxel. Dentro de este grupo, la operación conocida como umbral permite resaltar puntos determinantes en la configuración de la estructura de un objeto.

2.4.1.1. Umbral

La operación conocida como umbral simple consiste en utilizar el valor de intensidad cero para todos los píxeles cuyo nivel de gris se encuentra por debajo de un cierto valor (llamado el umbral) y el valor máximo de intensidad para todos los píxeles con un valor mayor. De esta manera, el resultado de la aplicación de este umbral es una imagen binaria que contiene píxeles negros y blancos; es por eso que a veces se utiliza el término binarización. El umbral hace posible destacar formas u objetos en una imagen, no obstante, la dificultad consiste en la selección del umbral a utilizar.

Se muestra una imagen en escala de grises en la Figura 11a y el resultado de una operación de aplicación de umbrales con valores de 125 en la Figura 11b y 200 en la Figura 11c.

a) Imagen original

b) Umbral 125

c) Umbral 200

Figura 11 Procesamiento de una imagen empleando la operación de umbral simple

2.4.2. De Procesamiento por grupo de píxeles

Es la operación que se aplica a una imagen para resaltar o atenuar detalles espaciales, modificando el valor de cada píxel de acuerdo con los valores de los píxeles que le rodean; se trata de transformar los niveles de gris originales de tal forma que se parezcan o diferencien más de los correspondientes a los píxeles cercanos.

Estas imágenes se caracterizan por un parámetro denominado frecuencia espacial que, puede definirse, como el número de cambios que ocurren en el valor del píxel (o brillo) por unidad de distancia para alguna región particular de la imagen. Si sobre un área dada de la imagen ocurren pocos cambios de brillo se considera como un área de baja frecuencia (e.g. grandes extensiones agrícolas, cuerpos de agua extensos, etc.). Si, por el contrario, los cambios de brillo son numerosos y notorios tendremos un área de alta frecuencia (e.g. calles o caminos en zonas urbanas, parcelas agrícolas pequeñas, etc.). Una imagen puede filtrarse para acentuar o eliminar una banda de frecuencias espaciales, tales como las altas o bajas frecuencias, o para resaltar

solamente las transiciones abruptas en la imagen como los bordes de objetos. Para ello se deben seguir los siguientes pasos:

- Se define una ventana de convolución que contiene un arreglo de coeficientes o factores de ponderación. Estos arreglos se definen como operadores o *kernels* (máscara), cuyo tamaño es normalmente de un número impar de píxeles (3x3, 5x5, 7x7, etc.)
- Dicho *kernel* se mueve a través de la imagen original, y el valor del píxel central del *kernel* en la imagen de salida se obtiene multiplicando cada coeficiente del *kernel* por el correspondiente valor del píxel en la imagen original y sumando el resultado de todos los productos resultantes [12]. La operación se repite para cada píxel de la imagen original. Un ejemplo de ello se puede observar en la Figura 12.

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

a) Kernel

25	25	27
24	25	26
22	21	23

b) Imagen Original

	24	

c) Imagen Resultante

Figura 12 Filtrado de una imagen

El algoritmo de Canny es usado para detectar todos los bordes existentes en una imagen. Este algoritmo está considerado como uno de los mejores métodos de detección de contornos mediante el empleo de máscaras de convolución y basado en la primera derivada.

Algoritmo de Canny: Uno de los métodos relacionados con la detección de bordes es el uso de la primera derivada, que es usada porque toma el valor de cero en todas las regiones donde no varía la intensidad y tiene un valor constante en toda la transición de intensidad [13]. Por tanto un cambio de intensidad se manifiesta como un cambio brusco en la primera derivada, característica que es usada para detectar un borde. El algoritmo de Canny consiste en tres grandes pasos:

- Obtención del gradiente

  En este paso se calcula la magnitud y orientación del vector gradiente en cada píxel. Para la obtención del gradiente, lo primero que se realiza es la aplicación de un filtro de Gauss a la imagen original con el objetivo de suavizar la imagen y tratar de

eliminar el posible ruido existente. Sin embargo, se debe de tener cuidado de no realizar un suavizado excesivo, pues se podrían perder detalles de la imagen y provocar un pésimo resultado final.

El suavizado se obtiene promediando los valores de intensidad de los píxeles en el entorno de vecindad con una máscara de convolución de media cero y desviación estándar σ. En la Figura 1Figura 13 se muestran dos ejemplos de máscaras que se pueden utilizar para realizar el filtrado de Gauss. Una vez que se suaviza la imagen, para cada píxel se obtiene la magnitud y módulo (orientación) del gradiente, obteniendo así dos imágenes.

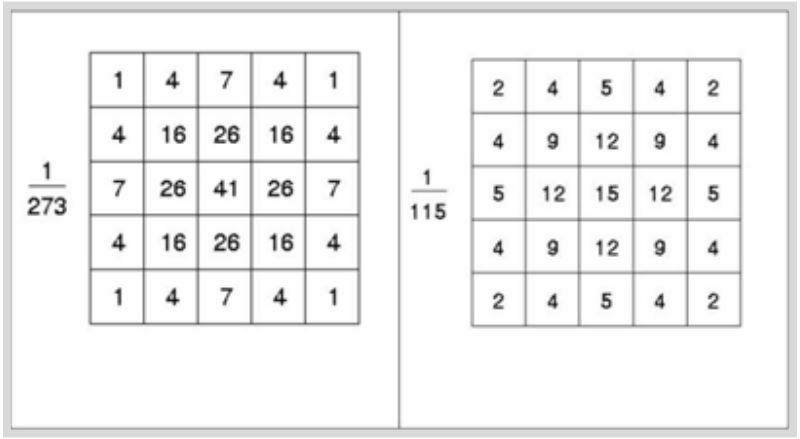


Figura 13 Máscaras (kernels) recomendadas para obtener el filtrado de Gauss

- Supresión no máxima

  En este paso se logra el adelgazamiento del ancho de los bordes, obtenidos con el gradiente, hasta lograr bordes de un píxel de ancho. Las dos imágenes generadas en el paso anterior sirven de entrada para generar una imagen con los bordes adelgazados. El procedimiento es el siguiente: se consideran cuatro direcciones identificadas por las orientaciones de 0º, 45º, 90º y 135º con respecto al eje horizontal. Para cada píxel se encuentra la dirección que mejor se aproxime a la dirección del ángulo de gradiente.

  Posteriormente se observa si el valor de la magnitud de gradiente es más pequeño que al menos uno de sus dos vecinos en la dirección del ángulo obtenida en el paso anterior. De ser así se asigna el valor 0 a dicho píxel, en caso contrario se asigna el

valor que tenga la magnitud del gradiente. La salida de este paso es una imagen con los bordes adelgazados.

Un dato importante a añadir sería la introducción de ruido durante este procedimiento, el cual podría ocasionar una confusión en la asignación de valores de magnitud del vector gradiente de cada uno de los píxeles. De ser así, los píxeles con valor de magnitud 0 podrían pasar a tener un valor diferente y a considerarse como borde.

- Histéresis de umbral

  Se aplica una función de histéresis basada en dos umbrales; con este proceso se pretende reducir la posibilidad de aparición de contornos falsos. Se consideran dos umbrales (el primero más pequeño que el segundo), de manera que para cada píxel de la imagen (con valor diferente de 0) se observa si la magnitud del vector gradiente es superior al segundo umbral; de ser así, se asignará un 1 a dicho píxel. De lo contrario, se tendrá que comparar si ese valor de magnitud es superior al primer umbral. En el caso de que sea superior, también se asignará un 1 a ese píxel, pero si se trata de un valor inferior, se le asignará un 0 (es decir, por debajo del primer umbral no se detectará ningún píxel como borde). Como resultado, se obtiene una imagen binaria (salida de la función) que en adelante se representará como una matriz.

  En la Figura 14 se puede observar el resultado de aplicar este tipo de filtro.

Figura 14 Algoritmo de Canny

2.4.3. Extracción de Esquinas

Es un buen método para la obtención de puntos importantes en las imágenes. Se fundamenta en la búsqueda de esquinas, que son puntos característicos muy poco susceptibles a cambios de rotación y escala. Una esquina o *corner* se caracteriza por ser una región de la imagen con cambios de intensidad en diferentes direcciones. Éste será el principio básico de búsqueda de puntos de Harris.

Detector de Harris: Es ampliamente utilizado debido a su elevada invarianza ante escala, rotación, cambios de iluminación y ruido en la imagen [14]. Este detector está basado en una matriz $C(x, y)$ que se calcula sobre una subventana pxp para cada punto de interés en la posición (x, y).

$$C(x,y) = \begin{pmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{pmatrix}$$

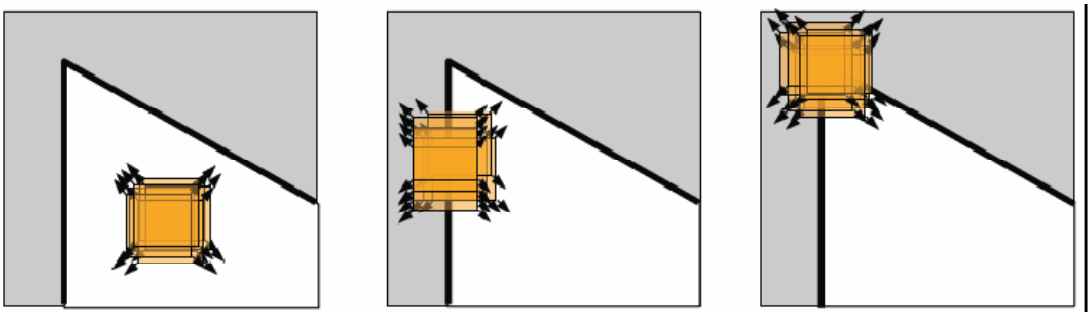
donde Ix, Iy son los gradientes de la imagen en dirección horizontal y vertical respectivamente. Sean λ_1 y λ_2 los valores propios de la matriz $C(x, y)$, la función de autocorrelación R tendrá un pico si ambos valores propios son altos. Esto significa que desplazamientos en cualquier dirección producirán un incremento significativo, indicando que se trata de una esquina.

$$R = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)$$

Algunos autores han obtenido buenos resultados experimentales con $\mathrm{k} = 0.04$. Se pueden obtener tres tipos de regiones como resultado [15]:

- *Flat* o plana: No ocurren cambios en ninguna dirección. En este caso, ambos valores son pequeños, lo que indica que la función de autocorrelación es plana, por tanto la zona de la imagen tiene una intensidad aproximadamente constante, como se observa en la Figura 15 a).
- *Edge* o borde: En la Figura 15 b) se observa este tipo de región, que se caracteriza porque no hay cambios en la dirección del propio *eje*. En este caso, se presentan cambios pequeños en la dirección del eje, pero altos en dirección perpendicular al borde.

- *Corner* o esquina: Hay cambios significativos en todas direcciones, como se observa en la Figura 15 c). Los dos valores son elevados y se observarán picos bruscos en la función.



a) *Flat* o Plana b) *Edge* o borde c) *Corner* o Esquina

Figura 15 Tipos de Regiones Detectadas

Con esta técnica se obtiene la nube de puntos característicos de la imagen que determinan la forma del objeto en cuestión.

2.5. Correspondencia de puntos

El problema de la correspondencia consiste en encontrar, dadas dos o más imágenes de una misma escena tridimensional tomadas desde diferentes puntos de vista, un conjunto de puntos en una de las dos imágenes donde cada uno de ellos pueda ser identificado con el mismo punto en la otra imagen.

Dicha situación suele producirse cuando son utilizadas dos imágenes de la misma escena, esto es lo que se llama el problema de correspondencia estéreo. Este concepto puede ser generalizado al problema de correspondencia de tres vistas o, en general, el problema de la correspondencia de N vistas.

La detección, medición e interpretación del movimiento son necesarias para analizar la información relativa al desplazamiento del objeto. En general, se considera que existe una relación entre la escena 3D y la secuencia de imágenes 2D asociada, esta relación se encuentra en el flujo óptico de la imagen.

El flujo óptico, entonces, puede definirse como el estimado del campo de movimiento, el cual puede ser obtenido a partir de las variaciones de primer orden en los patrones de brillo de la imagen, siendo el campo de movimiento la información de dirección y magnitud de movimiento en todos los puntos de una región determinada.

La porción visible de la proyección del Campo de Movimiento sobre un plano imagen se denomina Flujo de Imagen [16]. Las técnicas de flujo óptico estiman el flujo de Imagen operando sobre los cambios de intensidad en una secuencia de imágenes.

La estimación del movimiento a través del cambio de intensidades presenta defectos intrínsecos (tanto en sistemas de visión biológica como artificial), que en determinados casos provocarán que la estimación se aparte de los vectores de movimiento reales de los objetos. En sistemas de visión biológica, estas limitaciones dan lugar a ciertos tipos de lo que se conoce como fenómenos de ilusión óptica. Los problemas típicos se producen cuando:

- Un objeto en movimiento devuelve un patrón de brillo constante. En este caso, no hay cambios en el nivel de intensidad y se obtiene un flujo óptico nulo aunque exista movimiento, por ejemplo, una esfera lisa rotando iluminada por una fuente puntual estática.
- Un objeto estático tiene un patrón de brillo variable. En este caso, obtendremos un flujo óptico no nulo aunque no existe movimiento, tal es el caso de la misma esfera en reposo, iluminada por una fuente puntual en movimiento.

El análisis de escenas se puede complicar enormemente si se tiene en consideración aspectos como oclusiones, transparencias u objetos no rígidos. Normalmente, se considera un espacio reducido del problema donde, idealmente se deben cumplir los siguientes requisitos:

- Debe existir iluminación uniforme.
- La imagen debe reflejar la luz de forma no anómala.
- La imagen debe moverse en un plano paralelo al plano imagen.

Uno de los algoritmos más empleados para la estimación del flujo óptico, es el algoritmo de Lucas-Kanade Piramidal [17] propuesto por Bruce D. Lucas y Takeo Kanade y desarrollado por Jean-Yves Bouguet.

Algoritmo de Lucas-Kanade Piramidal: Este algoritmo está pensado para calcular el flujo óptico de una secuencia de vídeo, es decir, para realizar seguimiento de objetos [18]. Actualmente está considerada como una de las técnicas clásicas en el procesamiento de imágenes, y ha sido aplicado y adaptado a muchos problemas. Está diseñado para funcionar con secuencias de vídeo, por lo que solo funciona para imágenes con una distancia muy reducida entre los puntos correspondientes. A cambio de esta limitación, lo que se obtiene son tiempos de ejecución muy buenos.

El algoritmo realiza el cálculo del flujo óptico sobre una representación piramidal de la imagen. Esta pirámide se corresponde a una serie de imágenes resultado de una reducción de tamaño de la imagen inicial, según factores que van en múltiplos de dos. Esto se muestra en la Figura 16, donde I_0 será la imagen inicial y las imágenes I_1, I_2 e I_3 serán las siguientes imágenes de la pirámide.

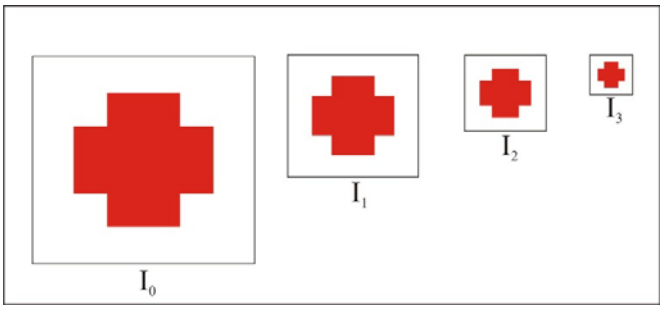


Figura 16 Representación piramidal de una imagen en el Algoritmo de Lucas-Kanade Piramidal

El seguimiento de los puntos se realiza a lo largo de los diferentes niveles de profundidad de las pirámides. De esta forma el algoritmo es más robusto y su convergencia más rápida siempre que se escojan adecuadamente los parámetros del algoritmo (niveles de la pirámide, condiciones para las iteraciones, entre otros). Se trata de un algoritmo iterativo de cálculo de flujo óptico entre imágenes, que pueden formar parte de una secuencia de video o bien pueden ser imágenes independientes con alguna región en común.

El seguimiento piramidal se basa en calcular el vector de flujo óptico entre dos imágenes para la región de interés, partiendo de una región con poca información, pasando posteriormente a refinar el cálculo realizado ampliando la información involucrada

subiendo el nivel de la pirámide. De esta forma, una vez se tiene calculado el vector de movimiento en un nivel, se calcula dicho vector para el nivel superior, tomando en cuenta información sobre regiones que no aparecían en el nivel inferior. Así, partiendo de la región alrededor del punto origen, el nivel final de la pirámide constituye la imagen completa.

Una vez obtenidos los puntos correspondientes, son utilizados para obtener la información de profundidad, ya que ésta será inversamente proporcional a la distancia entre dichos puntos.

2.6. Reconstrucción 3D

La reconstrucción tridimensional es un proceso que consiste en analizar un conjunto de imágenes para encontrar la posición relativa de los puntos correspondientes y, basándose en un conjunto de parámetros de la configuración de las cámaras, determinar la posición del punto en el espacio tridimensional mediante un proceso denominado triangulación.

Un método para realizar la reconstrucción de las coordenadas $\mathrm{X}, \mathrm{Y}, \mathrm{Z}$ de un punto en el espacio se realiza obteniendo dos imágenes distintas del punto w, como se mencionó anteriormente. La distancia entre el centro de las dos lentes se llama línea base *b*, el objetivo de este procedimiento es encontrar las coordenadas $\mathrm{X}, \mathrm{Y}, \mathrm{Z}$ del punto w, teniendo sus respectivas proyecciones en las dos imágenes, o sea, los puntos $\mathrm{x1}, \mathrm{y1}; \mathrm{x2}, \mathrm{y2}$. En la Figura 17 se puede observar un ejemplo de la geometría de dos cámaras en estéreo. En ella el sistema de coordenadas de referencia está en O, siendo la longitud focal efectiva de cada cámara f, y la línea base b, como se dijo anteriormente.

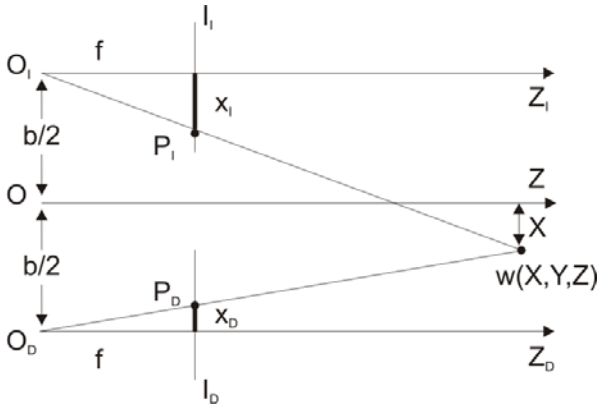


Figura 17 Geometría de dos cámaras en estéreo con ejes ópticos paralelos desde una perspectiva superior

Se supone que las dos cámaras son idénticas y que los ejes de coordenadas de ambas están alineados, lo único que los diferencia es el punto de origen, por lo que la coordenada Z del punto w será la misma para ambas cámaras. Los puntos $P_I(x_I, y_I, z_I)$ y $P_D(x_D, y_D, z_D)$ son las proyecciones del punto w de la escena 3D.

Si tenemos las matrices de proyección P y $P^{'}$, obtenidas a partir del proceso de calibración, y un conjunto de puntos en correspondencia, entonces es posible obtener los puntos 3D originales. Uno de los problemas con los que nos encontramos es que los rayos que unen las correspondencias con los centros de las cámaras generalmente no intersectan en un punto, sino que determinan rectas convergentes que no se llegan a tocar. Esto es así debido a que la precisión de las imágenes es finita y a que las correspondencias no son exactas.

El método más simple para reconstruir un punto es la triangulación [19]. Este método consiste en determinar los dos rayos que unen cada correspondencia con el foco de la cámara y después encontrar el punto cuya distancia a dichas rectas sea mínima. Las proyecciones $\mathrm{x} = \mathrm{PX}$ y $x^{'} = P^{'}X$ se pueden combinar de tal manera que se obtenga un sistema de la forma $AX = 0$. Para eliminar el factor de escala se utiliza el producto escalar de la forma $\mathrm{x} \times \mathrm{PX} = 0$ y se obtiene lo siguiente

$$x(p^{3T}X) - (p^{1T}X) = 0$$
$$y(p^{3T}X) - (p^{2T}X) = 0$$
$$x(p^{2T}X) - y(p^{1T}X) = 0$$

Estas ecuaciones son lineales en los componentes de X y una de ellas depende linealmente de las otras dos. Al introducir la información del otro punto, entonces se obtiene la matriz

$$A = \begin{pmatrix} xp^{3T} - p^{1T} \\ yp^{3T} - p^{2T} \\ x'p'^{3T} - p'^{1T} \\ y'p'^{3T} - p'^{2T} \end{pmatrix}$$

donde p^{nT} y p'^{nT} $n = 1,2,3$ son los vectores fila de las matrices P y P'. Para resolver el sistema se puede utilizar la descomposición SVD (*Singular Value Descomposition)*.

2.7. Descomposición por Valores Singulares (SVD)

La descomposición por valores singulares (*Singular Value Descomposition, SVD)* es una técnica muy utilizada en el campo de la visión artificial para descomponer matrices [20]. Una matriz A puede descomponerse de la siguiente forma:

$$A = UDV^T$$

Donde U y V son matrices ortogonales, y D es una matriz diagonal cuyos valores $(d_1, \dots, d_n)$ son los autovalores de A.

Es importante destacar que la descomposición por autovalores solamente existe para matrices cuadradas. Sin embargo, también se puede realizar esta descomposición en matrices no cuadradas.

La Descomposición por Valores Singulares puede definirse para cualquier matriz $A \in \Re^{mxn}$, donde existe una descomposición del tipo

$$A = UDV^T$$

tal que:

- U es una matriz $m\,x\,n$ con columnas ortogonales. Sus columnas son los valores singulares izquierdos de A.
- D es una matriz diagonal $n\,x\,n$ con todos sus valores positivos. Tiene todos sus elementos igual a 0, a excepción de los elementos $min\,(m,n)$ de su diagonal principal. Esos valores son reales y están ordenados de mayor a menor, y reciben el nombre de valores singulares de A.
- V^T es una matriz ortogonal $n\,x\,n$. Los valores de la diagonal de V se llaman valores singulares derechos de A.

Esta descomposición se puede representar como:

$$[A] = [U]\,[D]\,[V^T]$$

Dicho de otra manera, consiste en aplicar a la matriz A una transformación lineal y ortogonal, de manera de obtener una matriz $A \in C^{mxn}$ que preserva la norma cuadrada de A, pero utilizando pocas columnas.

La técnica SVD permite resolver fácilmente sistemas de ecuaciones lineales sobredeterminados por mínimo error cuadrático, tanto homogéneos como no homogéneos.

En el caso de un sistema homogéneo $Ax = 0$, las filas de V correspondientes a los menores valores singulares son vectores unitarios que minimizan $||Ax||^2$. La solución ideal sería el espacio nulo, pero debido a errores numéricos u otras causas, los valores singulares que deberían ser cero no lo son exactamente. Aún así, las filas de V correspondientes a los menores valores singulares son las mejores aproximaciones a la solución en el sentido de mínimo error cuadrático.

Calcular la SVD consiste en encontrar los valores propios y los vectores propios de AA^T y A^TA. Los vectores propios de A^TA forman las columnas de V y los vectores propios de AA^T las columnas de U. Además, los valores singulares en S son las raíces cuadradas de los valores propios de AA^T o A^TA.

Los vectores propios, autovectores o eigenvectores de un operador lineal son los vectores no nulos que, cuando son transformados por el operador, dan lugar a un múltiplo escalar de sí mismos, con lo que no cambian su dirección. Este escalar λ recibe el nombre valor propio, autovalor, valor característico o eigenvalor [21]. A continuación se presenta un ejemplo para comprender mejor cómo resolver la SVD.

$$A = \begin{bmatrix} 1 & 3 & -2 \\ 3 & 1 & 2 \\ -1 & 1 & 1 \end{bmatrix}$$

La matriz $A^T A$ es

$$\begin{bmatrix} 1 & 3 & -1 \\ 3 & 1 & 1 \\ -2 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 3 & -2 \\ 3 & 1 & 2 \\ -1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 11 & 5 & 3 \\ 5 & 11 & -3 \\ 3 & -3 & 9 \end{bmatrix}$$

En este caso, la ecuación característica es:

$$\det(A^T A - \lambda I) = \lambda^3 + 31\lambda^2 - 276\lambda + 576 = 0$$

Entonces, los valores propios de $A^T A$ son $\lambda_1 = 16$, $\lambda_2 = 12$ y $\lambda_3 = 3$. Por lo tanto, los valores singulares de la matriz A son $\sigma_1 = 4$, $\sigma_2 = 2\sqrt{3}$ y $\sigma_3 = \sqrt{3}$.

Luego, al calcular los respectivos valores propios unitarios de $A^T A$, se obtiene:

$$\vec{v}_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \qquad \vec{v}_2 = \frac{1}{\sqrt{6}} \begin{bmatrix} 1 \\ -1 \\ 2 \end{bmatrix} \quad \text{y} \quad \vec{v}_3 = \frac{1}{\sqrt{3}} \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix}$$

Por otra parte, la matriz AA^T es

$$AA^T = \begin{bmatrix} 14 & 2 & 0 \\ 2 & 14 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

Y sus respectivos vectores propios unitarios son

$$\vec{u}_1 = \begin{bmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \\ 0 \end{bmatrix}, \qquad \vec{u}_2 = \frac{1}{\sqrt{6}} \begin{bmatrix} \frac{-\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \\ 0 \end{bmatrix} \quad \text{y} \quad \vec{u}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Finalmente, si $U = [\overrightarrow{u_1} \quad \overrightarrow{u_2} \quad \overrightarrow{u_3}]$, $V = [\overrightarrow{v_1} \quad \overrightarrow{v_2} \quad \overrightarrow{v_3}]$ y S = $\begin{bmatrix} 4 & 0 & 0 \\ 0 & 2\sqrt{3} & 0 \\ 0 & 0 & \sqrt{3} \end{bmatrix}$. Entonces

$$A = \begin{bmatrix} \frac{1}{2}\sqrt{2} & -\frac{1}{2}\sqrt{2} & 0 \\ \frac{1}{2}\sqrt{2} & \frac{1}{2}\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 4 & 0 & 0 \\ 0 & 2\sqrt{3} & 0 \\ 0 & 0 & \sqrt{3} \end{bmatrix} \begin{bmatrix} \frac{1}{2}\sqrt{2} & \frac{1}{2}\sqrt{2} & 0 \\ \frac{1}{6}\sqrt{6} & -\frac{1}{6}\sqrt{6} & \frac{1}{3}\sqrt{6} \\ -\frac{1}{3}\sqrt{3} & \frac{1}{3}\sqrt{3} & \frac{1}{3}\sqrt{3} \end{bmatrix} = \begin{bmatrix} 1 & 3 & -2 \\ 3 & 1 & 2 \\ -1 & 1 & 1 \end{bmatrix}$$

2.8. Registro

Puesto que en el proceso de adquisición de imágenes generalmente solo se realizan tomas desde una sola dirección, se hace necesario realizar un proceso de alineación de las muestras que inicialmente están referenciadas a la posición de la cámara, mediante transformaciones geométricas con el fin de poder conformar un modelo 3D representativo del objeto real [22]. Este proceso se conoce como registro de imágenes de rango.

Se tiene una superficie M de la cual se realizan N capturas F_i de diferentes posiciones donde cada una corresponde a una imagen de rango, se deben encontrar N transformaciones geométricas T_i que acoplen las imágenes de rango F_i para que se puedan expresar en un mismo sistema de coordenadas $S(x, y, z)$ y así poder construir un modelo digital del objeto.

El algoritmo ICP (*Iterative Closest Point*) [23] presenta una forma estructurada e iterativa de encontrar la transformación geométrica óptima para alinear una muestra de una superficie P, con otra superficie X que debe contener a P, tal como se observa en la Figura 18.

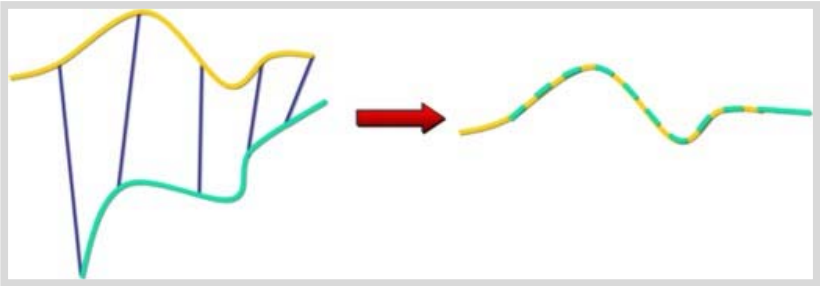


Figura 18 Alineación de Superficies por el Algoritmo ICP

Si una imagen de rango P está siendo registrada a un conjunto de datos X mediante una matriz de rotación R y a un vector de traslación T, entonces el algoritmo ICP intenta minimizar la función objetivo

$$\Sigma^2 = \sum_{i=1}^{N} ||x_i - (Rp_i + T)||^2$$

Donde $P = \{pi\}$ es el conjunto de puntos de la imagen que está siendo registrada, $X = \{xi\}$ es el conjunto de puntos de la imagen de rango de referencia a la cual P está siendo registrada, y cada punto p_i corresponde a x_i para cualquier valor de i. N es el número de puntos en el conjunto de datos P, R es la matriz de rotación 3×3 y T es el vector de traslación del registro.

Los centroides de ambos conjuntos deben ser calculados como sigue

$$x_c = \frac{1}{H} \sum_{h=1}^{H} x_h \quad \text{y} \quad p_c = \frac{1}{H} \sum_{h=1}^{H} p_h$$

Luego, se restan de los respectivos puntos de control

$$p_h^n = p_h - p_c \quad \text{y} \quad x_h^n = x_h - x_c$$

resultando la ecuación de minimización como sigue

$$\Sigma^2 = \sum_{h=1}^{H} ||x_h^n - Rp_h^n||^2$$

Esta ecuación puede ser resuelta mediante el método de descomposición SVD. Con lo que R puede ser calculada como

$$R = VU^T$$

y, una vez obtenida R se puede calcular T como sigue

$$T = x_c - Rp_c$$

con x_c y p_c los centroides de ambos conjuntos de puntos.

Después de aplicar el proceso de registro a un conjunto de imágenes de rango, se obtiene una nube de puntos en el espacio que contiene la información geométrica del objeto.

2.9. Mallas Geométricas

Una malla es un conjunto de celdas contiguas que permite representar en forma discreta el dominio de un problema a resolver numéricamente [24]. La confiabilidad de la solución numérica obtenida depende de la calidad de la malla generada. Los criterios de calidad a usar dependen del método numérico escogido y del tipo de problema a resolver.

Las mallas geométricas se pueden clasificar en estructuradas y no estructuradas. Las mallas estructuradas se caracterizan por estar compuestas de celdas de un tamaño similar y del mismo tipo, por ejemplo, triángulos o rectángulos en dos dimensiones, tetraedros o hexaedros en tres dimensiones. Tienen una dimensión definida, siendo ésta la cantidad de nodos en cada eje de la malla. Estas mallas son fáciles de generar pero no permiten modelar eficientemente problemas geométricamente complejos. En la Figura 19 se puede observar un ejemplo de este tipo de malla.

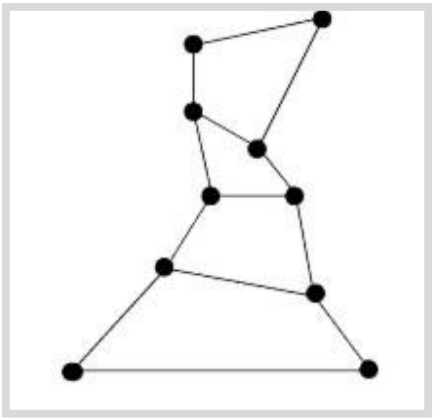


Figura 19 Malla geométrica estructurada

Por otro lado, las mallas no estructuradas permiten el uso de celdas de distinto tipo y/o de diferente tamaño y no tienen una dimensión definida. Estas mallas requieren de algoritmos y estructuras de datos más complejos que las anteriores, pero permiten modelar geometrías complejas y optimizar el número de celdas usadas de acuerdo a la necesidad de la aplicación. La Figura 20 muestra un ejemplo de este tipo de mallas.

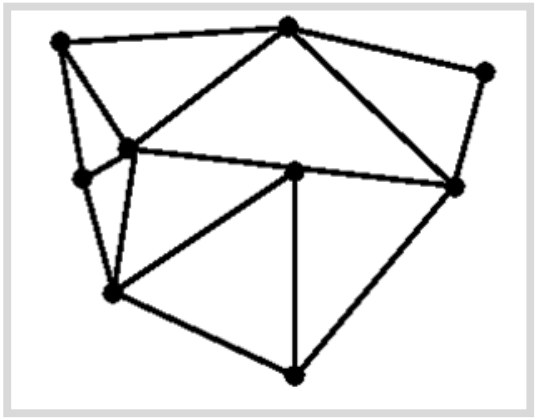


Figura 20 Malla geométrica no estructurada

La triangulación computacional de un conjunto de puntos del plano constituye actualmente una técnica básica en aplicaciones de ingeniería moderna, tales como en modelación de terrenos, reconstrucción y reconocimiento de objetos y en el análisis de problemas físicos. Un algoritmo muy utilizado para la generación de mallas no estructuradas es el algoritmo de Triangulación de Delaunay. A continuación se dará una explicación más detallada del mismo.

2.9.1. Algoritmo de Delaunay

Dado un conjunto de puntos $\{P_i\}$, se define el polígono de Voronoi V_j como

$$V_j = \{p: \|p - P_j\| \leq \|p - P_i\|, \forall\, i \neq j\}$$

en otras palabras, V_j es la región del espacio cuyos puntos se encuentran más cerca de P_j, que de cualquier otro punto del conjunto [25]. El conjunto de todos los polígonos de Voronoi es también conocido como diagrama de Dirichlet. De la definición se obtiene que un lado del polígono de Voronoi debe estar a igual distancia de dos puntos del conjunto $\{P_i\}$, siendo un segmento de la recta bisectriz de dichos puntos. Si se unen todos los pares de puntos cuyas bisectrices tienen algún segmento que es frontera de un polígono de Voronoi, el resultado es una triangulación del convexo que contiene a todos los puntos. Esta triangulación, figura dual del diagrama de Dirichlet, es conocida como la triangulación Delaunay del conjunto de puntos $\{P_i\}$. En la Figura 21 se muestra un conjunto de puntos en el plano, junto con su diagrama de Dirichlet y triangulación Delaunay asociados.

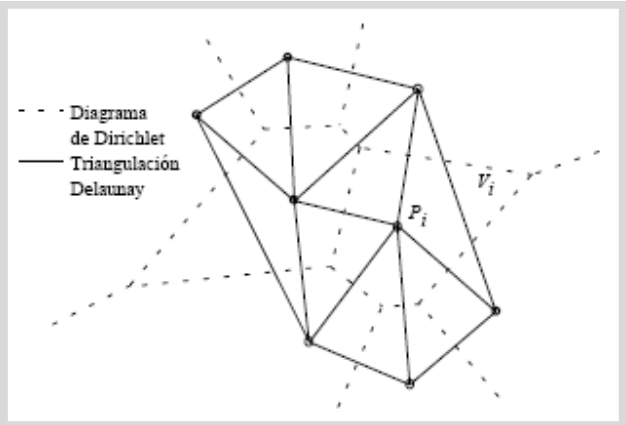


Figura 21 Conjunto de puntos en el plano, diagrama de Dirichlet y Triangulación de Delaunay

En este contexto, el algoritmo de Delaunay consiste en la construcción de una triangulación lo más cercana posible a la equilátera para el conjunto de datos y se basa en el criterio llamado Prueba del Círculo [26].

La Prueba del Círculo consiste en que dado un conjunto de puntos del plano, tres puntos definen un triángulo de Delaunay si el círculo que circunscribe a este triángulo no contiene ningún otro punto del conjunto de datos, siendo los vértices del diagrama de Dirichlet, por tanto, los centros de los círculos circunscriptos a cada triángulo Delaunay asociado. Además, una triangulación es una triangulación de Delaunay si cada triángulo satisface el criterio anterior. En la Figura 22, en el primer ejemplo, el triángulo T1 satisface la prueba del círculo, ya que el punto P se encuentra fuera del mismo. Por otro lado, el triángulo T2 del segundo ejemplo, no satisface la prueba, ya que el punto P está en el interior del círculo que circunscribe a este triángulo.

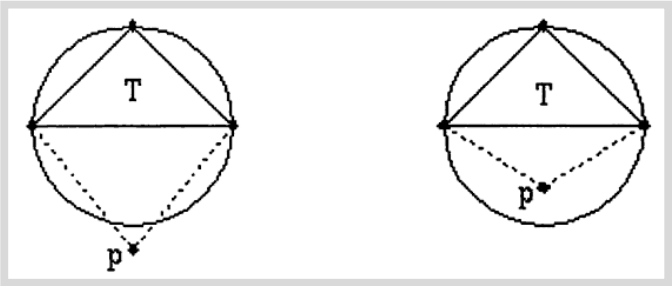


Figura 22 Prueba del Círculo en Triangulación de Delaunay

La triangulación de Delaunay maximiza los ángulos interiores de los triángulos de la triangulación. Eso es muy práctico porque al usar la triangulación como modelo tridimensional los errores de redondeo son mínimos. Por eso, en general se usan triangulaciones de Delaunay en aplicaciones gráficas. En la Figura 23, se puede observar la representación de una triangulación de Delaunay con los círculos circunscritos utilizados.

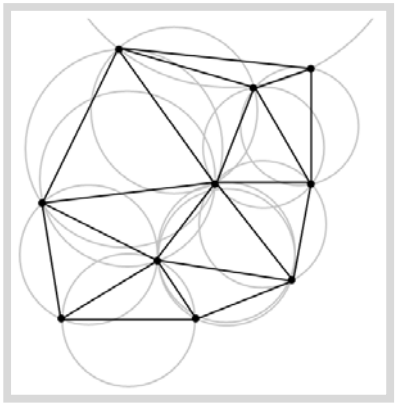


Figura 23 Triangulación de Delaunay

Capítulo 3 Trabajos Previos

En la actualidad es posible obtener una representación digital de un objeto 3D utilizando un dispositivo (escáner) que se encargue de obtener toda la información y coordenadas necesarias. Hasta el momento, se han realizado importantes avances en la construcción de dispositivos de este tipo con un bajo costo de adquisición, lo que permite acercar estas herramientas al usuario común.

Estos escáneres se pueden dividir en dos grandes grupos: Escáner de Contacto y Escáner sin Contacto. Los primeros, miden el objeto mediante contacto físico y permiten obtener las coordenadas de los puntos gracias al desplazamiento de una punta sobre la superficie a digitalizar. Son muy precisos y poseen una elevada resolución, pero son muy lentos comparados con otros tipos de escáneres y, además, han de requerir mucho trabajo manual, pues el dispositivo debe recorrer toda la superficie del objeto.

Para emplear estos sistemas de contacto se necesita que las piezas tengan la rigidez suficiente para que no se deformen por el contacto de la punta y, debido a la geometría de las puntas, es imposible digitalizar algunas ranuras, huecos y ángulos interiores.

Por otro lado, los Escáner sin Contacto miden las dimensiones del objeto sin ningún tipo de contacto con el objeto a medir. Su principal ventaja es que tienen una velocidad de adquisición de datos muy superior a las de los digitalizadores con contacto.

Entre los Escáner sin Contacto se pueden distinguir dos categorías: Los escáneres de visión pasiva, que son aquellos que no emiten ningún tipo de luz, por el contrario, detectan la radiación ambiental reflejada por el objeto. Estos escáneres suelen ser de bajo costo porque en la mayoría de los casos no necesitan de un hardware especial.

Un tipo de escáner de visión pasiva, son los escáneres estereoscópicos [27], los cuales utilizan dos cámaras separadas horizontalmente que miran a la misma escena y, al analizar la diferencia entre ambas cámaras, es posible determinar la distancia de cada punto de la imagen.

En la segunda categoría se encuentran los escáneres de visión activa, que utilizan una fuente de luz específica para determinar las coordenadas tridimensionales de los puntos de medida. Los sistemas ópticos se fundamentan en el cálculo de la profundidad.

Existen tres tipos de escáner de visión activa: Por telemetría, están basados en el tiempo de viaje de la luz y utilizan un láser para medir al objeto. Estos dispositivos se fundamentan en un telémetro láser. Luego, se encuentran los escáneres de visión activa por Triangulación, que también utilizan un láser para medir el entorno y, adicionalmente, incorporan una cámara para localizar la posición del punto del láser reflejado en la superficie analizada. Se denomina triangulación porque el emisor láser, que incide en el objeto y en la cámara, forman un triángulo tal como se observa en la Figura 24.

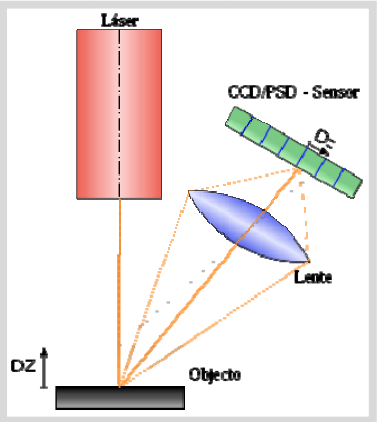


Figura 24 Principio de un Escáner de Triangulación

Por último, se encuentran los escáneres de Luz Estructurada, que proyectan un patrón sobre un objeto y miden la deformación, con una cámara de técnica similar a la de la triangulación. El cálculo de la profundidad consiste en resolver la intersección plano-recta de la proyección.

Algunos de los escáneres de bajo costo desarrollados para obtener un modelo digital de un objeto 3D se mencionan a continuación.

3.1. Lego NXT 3D

Es un escáner de contacto desarrollado por Philippe Hurbain (co-autor del libro *Extreme NXT: Extending the Lego Mindstorms NXT to the Next Level*). El dispositivo es un robot capaz de escanear pequeños objetos 3D para incluirlos en programas de CAD [28], en la Figura 25 puede observarse la estructura del mismo.

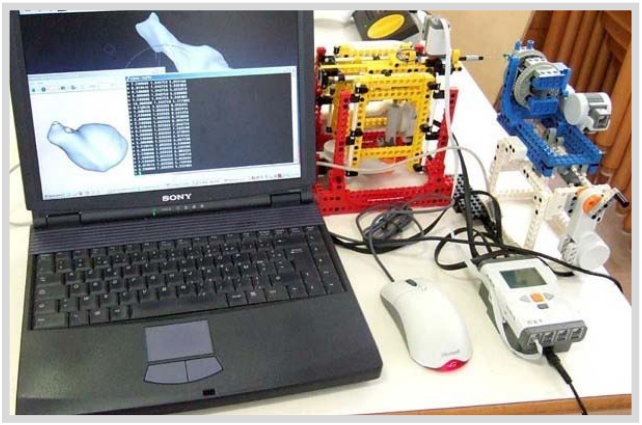
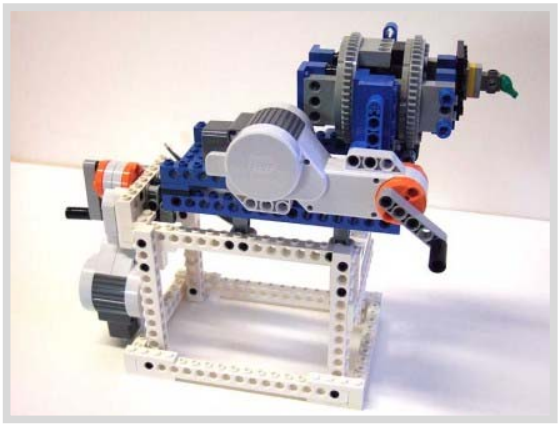


Figura 25 Escaner Lego NXT 3D

Consta de dos plataformas: Una para el objeto que se va a explorar (Figura 26a) y otra que se encarga de la exploración (Figura 26b), la cual utiliza como sonda una aguja obtenida de una máquina de coser.



a) Plataforma del Objeto a Explorar

b) Plataforma de Exploración

Figura 26 Plataformas del Lego NXT 3D

La plataforma que realiza la exploración mueve la aguja tanto de izquierda a derecha como de arriba a abajo. Cuando la aguja toca el objeto, el punto de localización de este contacto se registra. Por su parte, la plataforma que contiene al objeto es capaz de moverlo y de rotarlo. El escaneo de una pieza pequeña puede llevar varias horas, por lo que debe estar permanentemente conectado a la corriente.

3.2. Milkscanner

Es un escáner sin Contacto de Visión Pasiva. Debe su nombre a que para la realización de los escaneos requiere de un líquido lo suficientemente opaco, como por ejemplo leche o tinta, por lo que el mayor inconveniente de este escáner es que el objeto a escanear se mojará durante el proceso [1]. Este sistema se basa en el hecho de que al capturar la silueta de un objeto en diferentes etapas de sumersión se generan rodajas que, al ser correctamente interpretadas, permiten crear la información 3D del objeto.

Para realizar el escaneo se debe colocar una webcam sobre el recipiente que contiene al objeto. Se comienza a añadir el líquido, de tal forma que cada tres cucharadas de éste se toma una foto. De estas imágenes se restará la parte del líquido, con lo que el resto será el objeto en cuestión. Con ello, se podrá crear un mapa de desplazamiento que, posteriormente, puede ser importado por algún programa de creación 3D. En la Figura 27 se pueden observar dos modelos de este tipo de escáner.

Figura 27 Milkscanner

3.3. Proforma

Es un prototipo de software diseñado por unos estudiantes de la Universidad de Cambridge, Reino Unido, en el año 2009, que utiliza algoritmos probabilísticos para obtener una réplica digital tridimensional de un objeto utilizando únicamente una webcam. Pertenece a la categoría de los escáneres de Visión Pasiva.

El sistema consiste en ir tomando capturas del objeto y rotarlo para que el software vaya generando las mallas e indexando puntos [29]. Básicamente, este tipo de escaneo 3D transforma una nube de puntos 3D en un mallado a través del proceso de División de Tetraedros de Delaunay [2], que pueden ser cubiertos con las texturas apropiadas. Al terminar de escanear el objeto, Proforma puede hacer un seguimiento visual del mismo y representar el movimiento en tiempo real en la pantalla. En la Figura 28 se puede observar el proceso de reconstrucción del objeto al utilizar este software.

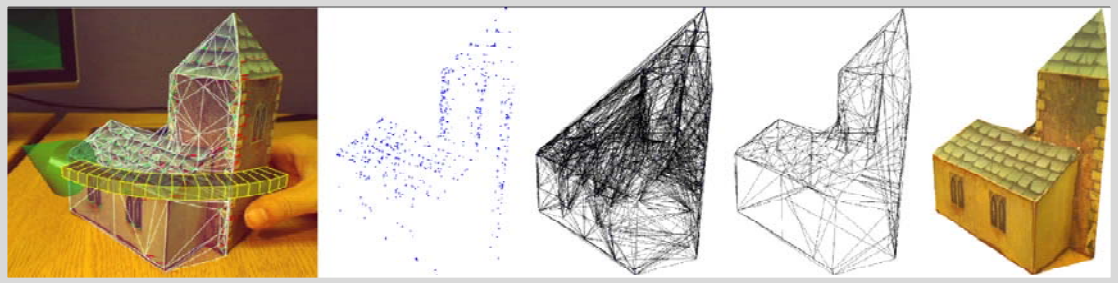


Figura 28 Etapas de Reconstrucción de un Objeto al utilizar Proforma

3.4. Escáner 3D SomPro

Otro escáner perteneciente a la categoría de Visión Pasiva es el 3D SomPro que, a partir de una cámara y usando este software, permite digitalizar figuras tridimensionales complejas por medio de una serie de fotografías tomadas desde diferentes ángulos (entre 20 y 30 fotografías). Es necesario utilizar una plantilla que se usará luego como punto de referencia para los cálculos del programa y un fondo unicolor en la escena [30]. En la Figura 29 se puede observar un ejemplo de ello.

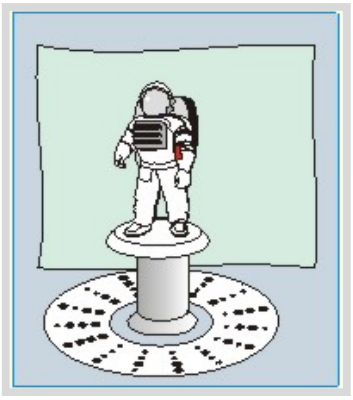


Figura 29 Plantilla de Referencia del 3D SomPro

El software, automáticamente, separa el objeto del fondo y crea una nube de puntos a partir de la plantilla que se nombró anteriormente, estos puntos se pueden usar entonces para extrapolar la forma del objeto. Provee herramientas que permiten editar la forma del objeto utilizando una silueta adicional o combinándola con un segundo escaneo del mismo. También, incorpora un editor que permite realizar modificaciones manualmente. En la Figura 30 se puede observar el resultado obtenido.

a) Foto tomada al objeto

b) Geometría del objeto escaneado

Figura 30 Proceso de Escaneo con el Software 3D SomPro

Si la información de color se reúne en cada uno de los puntos de entrada, entonces los colores y texturas en la superficie del objeto también se pueden determinar. En la Figura 31 se puede observar el resultado final.

Figura 31 Modelo 3D del Objeto Escaneado

3.5. David Laserscanner

Perteneciente a la categoría de escáneres por Triangulación, cuenta con un software que puede transformar una webcam común y un láser en un escáner láser en tres dimensiones [31]. Éste funciona sólo bajo un ambiente controlado, por ello es necesario emplear una plantilla de calibración para la calibración de la cámara (lo cual permite obtener ciertos parámetros internos de la cámara que serán necesarios a lo largo del proceso de reconstrucción) y como estructura de fondo durante el escaneo. La plantilla de calibración se puede observar en la Figura 32.

Figura 32 Córner de Calibración del David Laserscanner

Una vez que se ha calibrado la cámara, se recorre el objeto con el láser. La distancia entre la cámara y el plano láser (es decir, el ángulo de triangulación) debe ser lo más

amplio posible para una alta precisión y así obtener resultados óptimos. Se puede escanear tantas veces y durante tanto tiempo como se quiera, luego el software se encargará de analizar cada cuadro que toma la cámara para crear una imagen en tres dimensiones a partir de ellos. El detalle más importante para conseguir excelentes resultados es la calidad de las lentes de la webcam. En la Figura 33 se puede observar cómo se realiza el proceso de escaneo con este dispositivo.

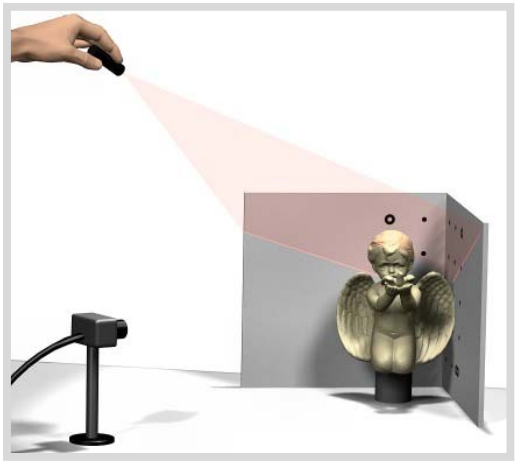


Figura 33 David Laserscanner

El resultado es una malla de triángulos de 360º, texturizados y suavizados, del modelo 3D, como se observa en la Figura 34.

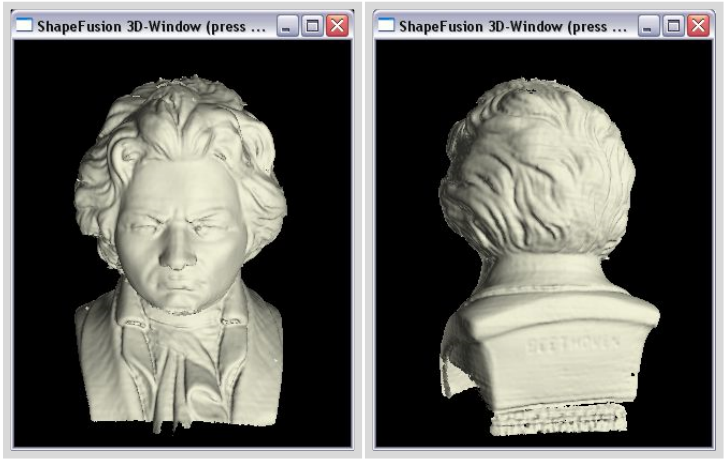


Figura 34 Modelo 3D realizado con David Laserscanner

Sin embargo, la calidad dependerá de las opciones elegidas durante la utilización de esta aplicación, tales como: Resolución, Suavizado y Eliminación. Por ello, es recomendable probar diferentes ajustes de estos parámetros.

Existe una versión gratuita y otra profesional de David Laserscanner, la diferencia está en que esta última puede generar archivos .OBJ con mayor resolución.

3.6. Escáner Láser de la Universidad de Alicante

También, dentro de los escáneres de Triangulación, se encuentra este escáner desarrollado por la Universidad de Alicante. Presenta un sistema de reconstrucción de objetos tridimensionales de medianas dimensiones y uso cotidiano [32]. El sistema de reconstrucción es de tipo educativo-docente y se caracteriza por su sencillez de construcción y por su bajo coste.

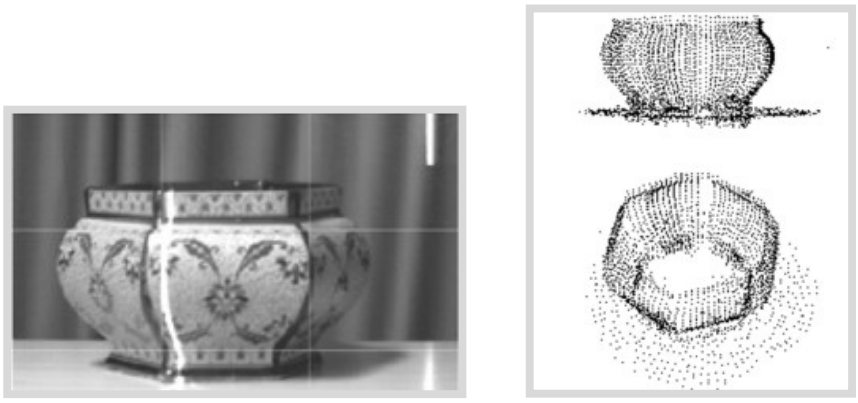
Está compuesto por una cámara CCD, por un láser óptico como fuente de luz para realizar el proceso de digitalización y por una mesa giratoria para realizar el barrido de la superficie del objeto que permita la obtención de la geometría de éste. En la Figura 35 se puede observar el Sistema Cámara – Láser empleado.

Figura 35 Sistema Cámara – Láser

Tanto los sistemas de adquisición, cámara y tarjeta de procesamiento, como el proyector láser, y la mesa giratoria están interconectados por un computador y un software desarrollado que se encarga de comunicar, gestionar y ejecutar las órdenes y comandos que hacen funcionar cada una de las partes de la célula de modo automático.

El primer paso es realizar la calibración de la cámara para obtener resultados precisos. Posteriormente, se inicia el proceso de digitalización, que consta de tres fases: La primera de ellas consiste en obtener el perfil del plano láser, la segunda la obtención de las coordenadas de los puntos situados sobre la superficie del objeto que forman parte del perfil del plano-láser y, en tercer lugar, estos puntos se ajustan mediante la utilización de un método de modelado.

Como salida se obtiene un archivo de texto que contiene las coordenadas de los puntos tridimensionales de cada uno de los perfiles obtenidos, en cada uno de los giros de la mesa. Ese archivo se utiliza como entrada del módulo de modelado y ajuste que se encarga de proporcionar una reconstrucción gráfica, ésta es realizada utilizando los procesos de Triangulación de Delaunay 3D y el algoritmo de Marching Cubes [33]. En la Figura 36 se puede observar el resultado de digitalizar un objeto con este dispositivo.



a) Vasija en Proceso Tridimensional

b) Reconstrucción tridimensional de la vasija

Figura 36 Digitalización 3D de una Vasija

Con este dispositivo se obtienen mejores resultados de reconstrucción dependiendo del grado de opacidad del objeto, suavidad de la superficie y color de ésta. Así, los mejores resultados se logran en objetos totalmente opacos, de colores claros y superficies suaves. Además, se pueden obtener niveles de precisión del orden del milímetro.

Capítulo 4 Diseño y Desarrollo de la Solución Propuesta

El presente capítulo tiene por objetivo señalar los dispositivos, herramientas y plataforma empleados en el desarrollo de la aplicación. Se mostrará la estructura de las clases utilizadas, así como librerías útiles en el desarrollo de la misma.

La implementación está basada en la utilización de diversos algoritmos del área de la Computación Gráfica y el procesamiento digital de imágenes, tales como:

- Filtros para la detección de bordes y esquinas.
- Algoritmo de triangulación para la reconstrucción 3D del objeto.
- Algoritmo ICP (*Iterative Closest Point*) para la alineación de las imágenes.
- Algoritmo de Triangulación de Delaunay para obtener el mallado del objeto reconstruido.

4.1. Plataforma de Hardware

Para ejecutar la aplicación se requiere un computador con memoria superior a 1GB, ya que la cantidad de estructuras de datos que se creen a lo largo del proceso dependen de la complejidad en las formas y texturas del objeto. Así mismo, se requieren dos cámaras web del mismo tipo y modelo, con una resolución igual o superior a 2MP, para poder obtener imágenes de buena calidad y que carezca de la propiedad de autoajuste del foco, porque con ella el proceso de calibración y toma de imágenes resulta erróneo.

La aplicación fue desarrollada en un computador con procesador AMD Athlon 64 X2 Dual Core de 2.60 Ghz, 2GB de memoria RAM y una tarjeta de video NVIDIA GeForce 7600 GT con 256 MB de memoria de video.

Por otro lado, para poder obtener el resultado deseado es necesario seguir una serie de pasos, que van desde la elección del dispositivo a utilizar, pasando por la adquisición de las imágenes, hasta llegar a la reconstrucción tridimensional del objeto en cuestión, como se observa en la Figura 37.

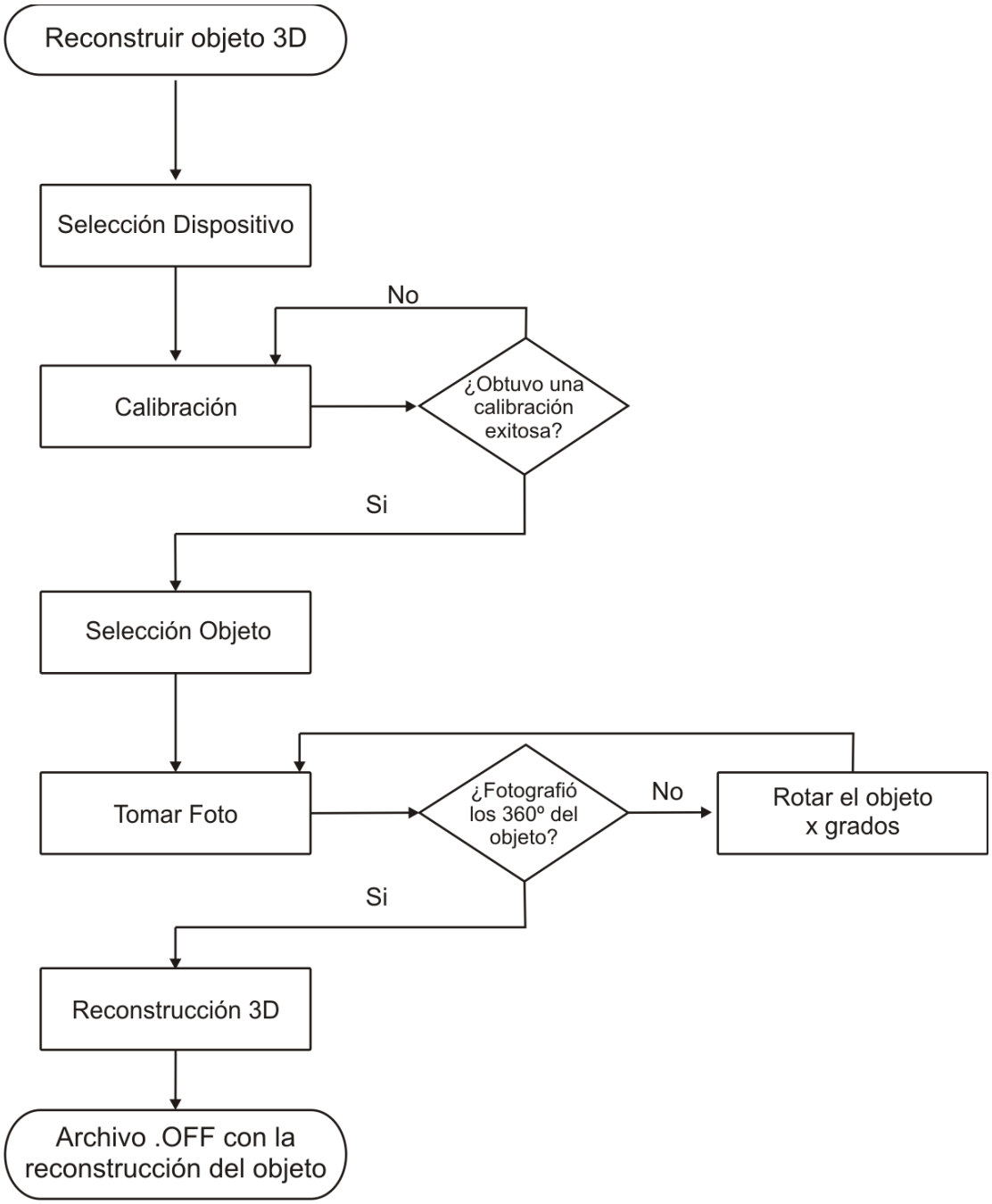


Figura 37 Pasos a seguir en la Reconstrucción 3D de un objeto

Una vez seleccionado el objeto, se procede a tomar las fotos que serán luego utilizadas en el proceso de reconstrucción. En cada instante se toman dos fotos, una con la cámara izquierda y la otra con la cámara derecha, a fin de obtener la visión estéreo mencionada. Después de cada toma, se gira el objeto una cierta cantidad de grados, previamente

determinada. En este caso, se definió que el objeto rotaría 10º en sentido horario. Una vez culminado el proceso de adquisición de imágenes, se obtiene la reconstrucción tridimensional del objeto y con ello el archivo .OFF con la estructura del mismo.

4.1.1. Diseño del Prototipo

El prototipo del escáner 3D desarrollado está constituido por un par de cámaras Web conectadas a un computador y una caja con todas sus paredes de un mismo color, con una superficie circular giratoria en su interior sobre la cual se coloca el objeto en cuestión, el diseño del mismo puede observarse en la Figura 38.

Figura 38 Diseño del prototipo empleado

Es importante que se disponga de una buena fuente de luz que sea constante y uniforme y que el centro de la distancia d entre las cámaras, coincida con el centro c de la superficie circular sobre la cual se coloca el objeto.

La caja empleada como ambiente de control tiene unas dimensiones de 28x28x28cm y un color uniforme, en este caso es de color fucsia porque no es un color común que esté presente en la mayoría de los objetos, por lo que éstos no se confundirán con el fondo.

Además, la base circular sobre la cual se coloca el objeto para poderlo girar en cada toma tiene una dimensión de 22cm de diámetro, como se puede observar en la Figura 39.

Figura 39 Ambiente de control utilizado

Por otro lado, las cámaras fueron colocadas sobre una plataforma para que alcanzaran la misma altura del objeto que se desea escanear, y fueron separadas una distancia de 5cm ó 6cm según el experimento realizado. La configuración del prototipo puede observarse en la Figura 40.

Figura 40 Configuración de la plataforma de hardware

Fueron empleadas 2 cámaras web Logitech Quickcam Pro 9000® con una resolución de imagen de 640x480 y un sensor CMOS de 2MP.

4.2. Plataforma de Software

A continuación se especifican el lenguaje de programación utilizado, así como las librerías empleadas en la implementación de la aplicación. Además, se detalla el método de calibración utilizado.

4.2.1. Lenguaje de Programación

El lenguaje de programación seleccionado para la implementación de la aplicación debe permitir la programación orientada a objetos y la posibilidad de interactuar con las diferentes API's creadas para el desarrollo de aplicaciones en computación gráfica. Por ello, la aplicación fue desarrollada bajo el lenguaje de programación C++, que permite utilizar las interfaces de programación para gráficos tridimensionales como OpenGL [34] y QT [35], así como la utilización de la librería OpenCV [36], para el

tratamiento de imágenes, y está destinada principalmente a aplicaciones de visión por computador en tiempo real. Por otro lado, también se emplearon las librerías CGAL [37] y VTK [38], para la utilización de algoritmos eficientes de geometría computacional.

La aplicación se desarrollo bajo el ambiente de desarrollo de Visual C++ 2008.

4.2.2. Librerías para el desarrollo

Como se mencionó anteriormente, para el desarrollo de la aplicación se empleó OpenGL y QT, específicamente se utilizó la clase QGLWidget, que provee funcionalidades para mostrar gráficos OpenGL integrados en una aplicación QT.

Además, se utilizó la librería OpenCV para el manejo de las cámaras, su calibración y obtención y tratamiento de las imágenes. El conjunto de imágenes del objeto que se capturan durante el proceso se alinean en un mismo sistema de coordenadas a través del método *Iterative Closest Point* (ICP), para lo cual se empleó la librería VTK. También fue utilizada para generar el mallado correspondiente mediante el algoritmo de Triangulación de Delaunay, lo que permite visualizar la forma del objeto.

Por último, la librería CGAL fue empleada para procesar la nube de puntos que fue obtenida a lo largo del proceso de reconstrucción 3D.

4.2.3. Método de Calibración

Zhang propone una técnica de calibración basada en la observación de una plantilla plana, tipo tablero de ajedrez, desde varias posiciones [10]. Este método resulta versátil ya que la elaboración de la plantilla se puede realizar fácilmente, y consiste simplemente en tomar varias imágenes de la misma desde diferentes posiciones y orientaciones. Mientras mayor cantidad de cuadros tenga el patrón de calibración, más preciso será el resultado. Se obtienen resultados favorables a partir de plantillas de $7x8$ cuadros, y se considera que el número base de imágenes que deben tomarse es de 8, en este caso, el aumentar la cantidad de imágenes no suele influir sobre el resultado del proceso.

La calibración consiste en la obtención de los parámetros del modelo de la cámara. Se distinguen dos tipos de parámetros [39]: intrínsecos, relacionados con la óptica de la cámara y, los extrínsecos relacionados con su orientación.

Se utiliza el modelo de la cámara *pinhole,* de acuerdo al cual todos los rayos de luz que provienen de la escena pasan por un punto común, el *pinhole* o centro de proyección, antes de incidir en el plano de la imagen. Se tiene un sistema de coordenadas de referencia de la escena y otro de la cámara, cuyo origen es el *pinhole*, como se observa en la Figura 41.

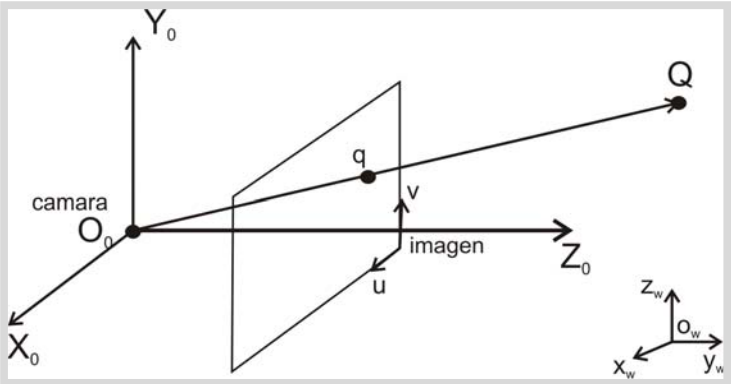


Figura 41 Modelo de la cámara pinhole

Los parámetros intrínsecos relacionan la posición de un punto en el sistema de referencia de la cámara con la posición, dada en píxeles, de su proyección en la imagen. Los parámetros extrínsecos, por su parte, relacionan los sistemas de coordenadas de la escena y de la cámara. Posteriormente se introducen coeficientes de distorsión de la lente de la cámara, para refinar la calibración.

Un punto $(x_c, y_c, z_c)^T$ en coordenadas del sistema de referencia de la cámara, tiene su proyección en el punto $(u, v)^T$ en coordenadas de la imagen. La relación entre ambos está dada por la siguiente ecuación:

$$s\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = A\begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix}$$

Donde s es un factor de escala y A es la matriz de parámetros intrínsecos dada por:

$$A = \begin{pmatrix} f_u & \gamma & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{pmatrix}$$

Donde f_u y f_v son las longitudes focales en píxeles, tomando el tamaño del píxel en la dirección de las abscisas y de las ordenadas, respectivamente, $(u_0, v_0)^T$ es el centro de la imagen y γ es un factor que describe la oblicuidad de los ejes de la imagen.

La transformación del punto $(x_w, y_w, z_w)^T$ en coordenadas de la escena a coordenadas de la cámara se realiza mediante una matriz de rotación R y un vector de traslación t (parámetros extrínsecos) utilizando la siguiente ecuación:

$$\begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix} = R \begin{pmatrix} x_w \\ y_w \\ z_w \end{pmatrix} + t$$

La distorsión radial de la lente se corrige mediante dos coeficientes, k_1 y k_2, utilizando las siguientes ecuaciones:

$$u' = u + (u - u_0)[k_1(x^2 + y^2) + k_2(x_2 + y_2)^2],$$
$$v' = v + (v - v_0)[k_1(x^2 + y^2) + k_2(x_2 + y_2)^2]$$

Donde $(u, v)^T$ son las coordenadas ideales (sin distorsión) y $(u', v')^T$ son las coordenadas reales, $(x, y)^T$ son las coordenadas del punto normalizadas, donde

$$x = \frac{x_c}{z_c} \quad \text{y} \quad y = \frac{y_c}{z_c}.$$

En la primera etapa de la calibración se utiliza un patrón plano con cuadros blancos y negros, como un tablero de ajedrez, como se observa en la Figura 42.

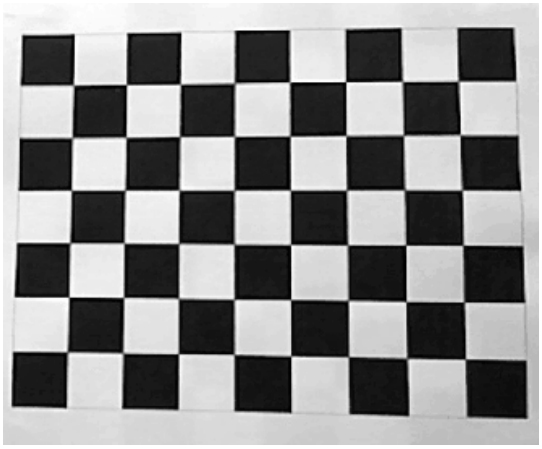


Figura 42 Patrón de Calibración que forma un tablero de ajedrez

Se toman imágenes del patrón con diferentes orientaciones y, en cada una, se determina una matriz, llamada homografía, que relaciona las posiciones de puntos de referencia en el patrón y las posiciones de los puntos correspondientes proyectados en las imágenes. El plano del patrón se toma como el plano en el sistema de coordenadas de la escena y como puntos de referencia se utilizan las esquinas de los cuadros. Al utilizar la propiedad de que una matriz de rotación es ortogonal, se pueden determinar los parámetros intrínsecos a partir del conjunto de homografías y luego, para cada orientación del patrón, se determinan los parámetros extrínsecos.

El reconocimiento del patrón, cálculo de las esquinas internas y cálculo de la matriz de la cámara fueron realizados con ayuda de un conjunto de rutinas de la librería OpenCV.

4.2.4. Diseño de la aplicación

Para alcanzar la solución deseada se crearon diversas estructuras de datos y clases que permitieran representar los objetos deseados. A continuación se da una descripción de los mismos.

4.2.4.1. Diagrama de Clases

A continuación se muestran el diagrama de clases que servirá de base para el diseño e implementación de la aplicación. Además, se plantean las funciones que deben cumplir sus clases. En la Figura 43 se muestra el diagrama mencionado.

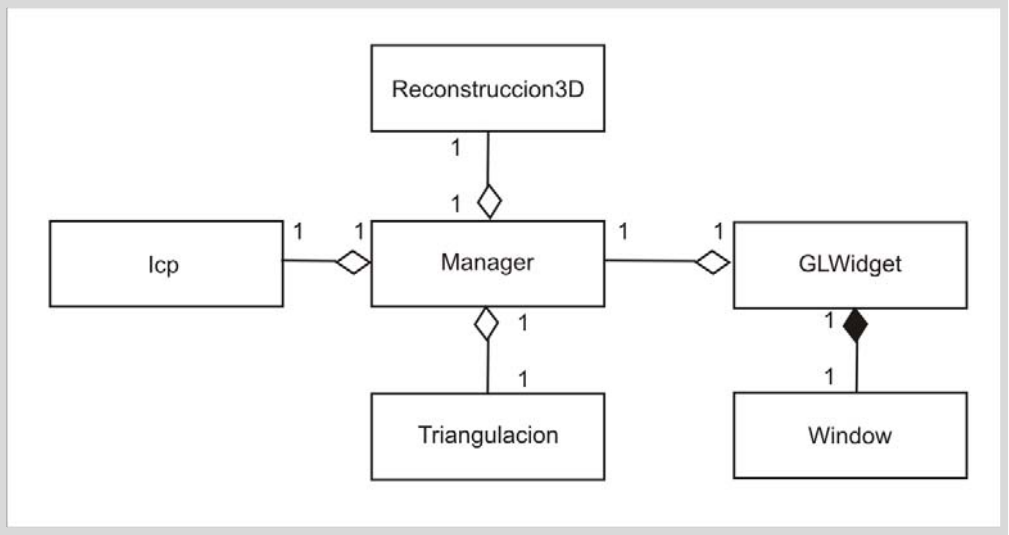


Figura 43 Diagrama de Clases

1. Clase GLWidget: Esta clase se encarga de la creación, manipulación y destrucción de las ventanas de despliegue y de su contenido dentro de la aplicación. Permite la interacción con el objeto.

2. Clase Window: Se encarga de la creación y conexión de los botones y menús con las acciones vinculadas a cada uno de ellos.

3. Clase Manager: Es el núcleo de la aplicación. Se encarga de recibir las imágenes y parámetros iniciales y de servir de enlace entre el resto de las clases empleadas para obtener como resultado final la reconstrucción 3D del objeto analizado.
4. Clase Icp: Se encarga de calcular la matriz de transformación (rotación y traslación) que permitirá alinear las imágenes capturadas a lo largo del proceso.

5. Clase Triangulacion: Esta clase se encarga de calcular las coordenadas 3D de los puntos característicos que definen al objeto en cuestión. Entre sus funciones se encuentran:
   - Calcular la nube de puntos 2D por cada imagen capturada del objeto.

- Calcular los puntos correspondientes por cada par de imágenes estéreo del objeto.
- Calcular las coordenadas tridimensionales de los puntos correspondientes entre cada par de imágenes estéreo a través del método de triangulación lineal.

6. Clase Reconstruccion3D: Se encarga de generar el mallado del objeto que se quiere reconstruir, mediante la Triangulación de Delaunay.

4.2.4.2. Diseño de las Clases

A continuación se explican las clases identificadas en la sección anterior detallando sus atributos y explicando sus métodos.

Clase Manager: Esta clase es uilizada como enlace entre las clases que obtienen los resultados del modelado 3D y la clase que se encarga de desplegarlo en pantalla. En la Figura 44 se observa el detalle de la misma.

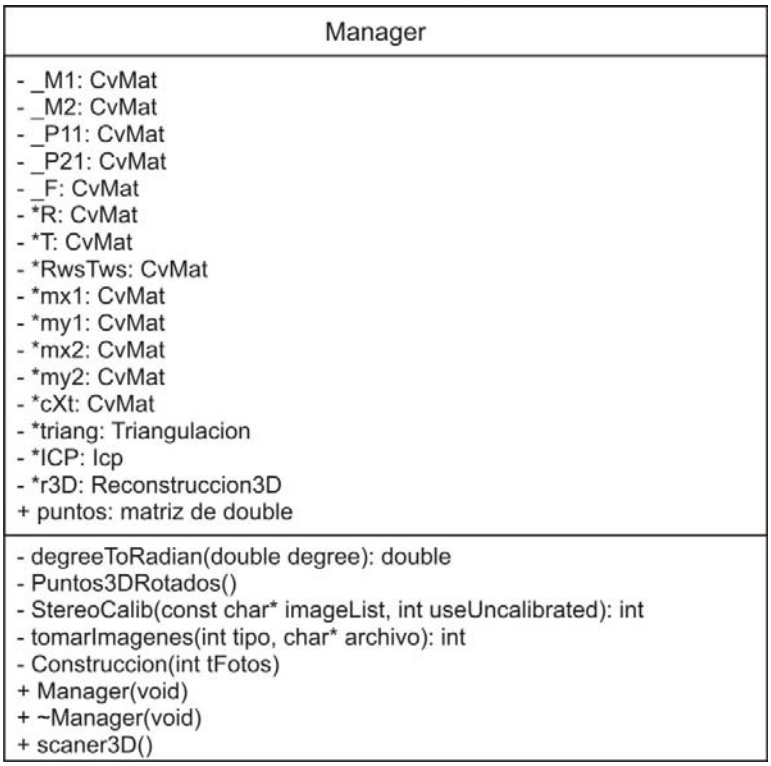


Figura 44 Clase Manager

Descripción de los métodos:

- `Puntos3DRotados`: Se encarga de aplicar la transformación obtenida por el método del ICP a los puntos 3D, previamente encontrados, de la imagen actual, de tal manera que pueda ser alineada con la anterior.
- `StereoCalib`: Se encarga de realizar la calibración de las cámaras para así obtener los parámetros intrínsecos y extrínsecos de las mismas, que serán utilizados a lo largo de todo el proceso.
- `tomarImagenes`: Esta función permite capturar las imágenes estéreo, tanto del patrón de calibración, como del objeto en cuestión.
- `Construccion`: Función que toma cada par de imágenes estéreo y, en conjunto con otras funciones, obtiene como resultado final los puntos 3D del objeto, almacenándolos en un archivo.
- `Escaner3D`: Esta función sirve de enlace entre un conjunto de funciones para obtener como resultado final el modelo 3D del objeto y un archivo de tipo .OFF con el mismo.
- `Manager`: Constructor de la clase.

- `~Manager`: Destructor de la clase.

Clase Icp: Obtiene, a partir del método *Iterative Closest Point,* la matriz de transformación necesaria para poder alinear el conjunto de imágenes obtenidas del objeto después de cada rotación, en la Figura 45 se observa el detalle de esta clase.

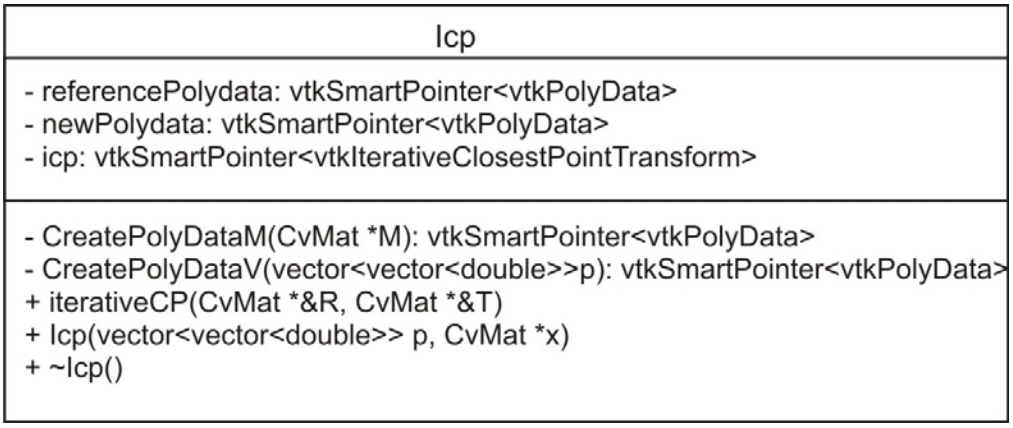


Figura 45 Clase Icp

Descripción de los métodos:

- `CreatePolyDataM`: Esta función se encarga de guardar los puntos 3D de la imagen actual del objeto en la estructura de datos correspondiente.
- `CreatePolyDataV`: Se encarga de guardar los puntos 3D de la imagen previa del objeto en la estructura de datos adecuada.
- `iterativeCP`: Esta función realiza el método ICP, obteniendo las matrices de transformación correspondiente.
- `Icp`: Constructor de la clase.
- `~Icp`: Destructor de la clase.

Clase Triangulacion: Esta clase calcula los puntos correspondientes entre un par de imágenes estéreo, para obtener como resultado final los puntos 3D que representan al objeto en cuestión. El detalle de la misma puede observarse en la Figura 46.

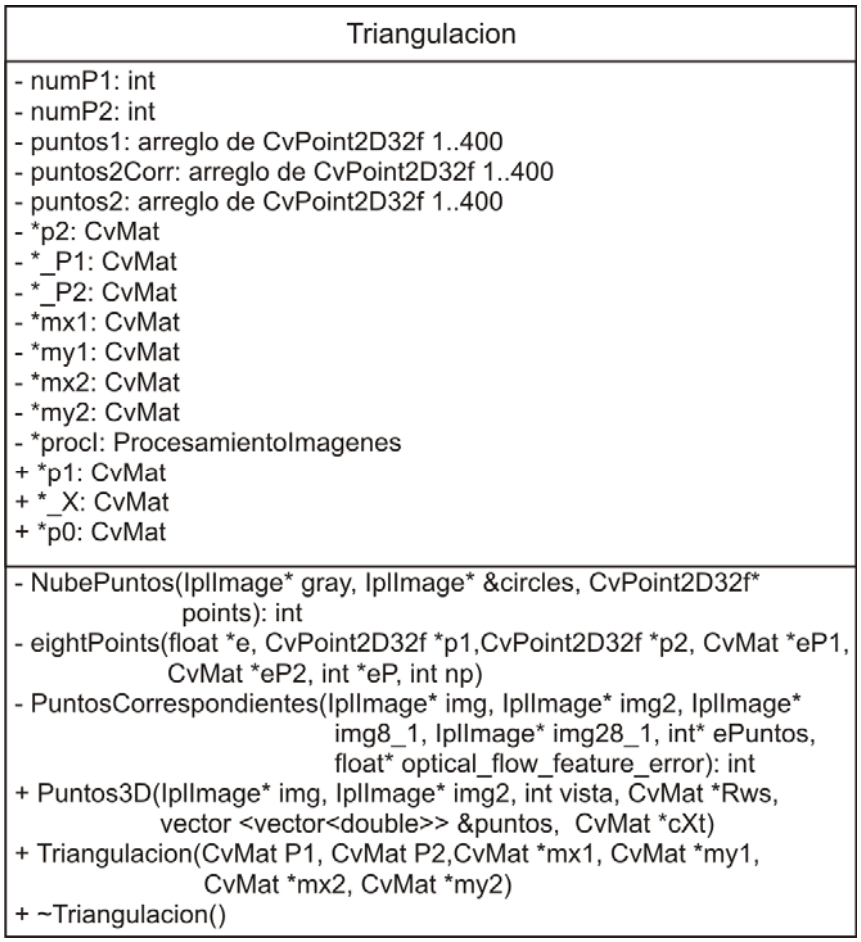


Figura 46 Clase Triangulacion

Descripción de los métodos:

- `NubePuntos`: Toma las imágenes en su estado original y aplica un detector de esquinas con el fin de resaltar los puntos más importantes para, posteriormente, obtener la nube de puntos 2D.
- `eightPoints`: Determina los puntos correspondientes que están aptos para continuar el proceso de reconstrucción del objeto.
- `PuntosCorrespondientes`: Toma un par de imágenes estéreo para obtener los puntos correspondientes 2D que existen entre ellas, a partir del algoritmo piramidal de Lucas y Kanade.
- `Puntos3D`: Esta función utiliza los puntos 2D obtenidos en la función anterior para, a partir del método de triangulación lineal, obtener los puntos 3D correspondientes.
- `Triangulacion`: Constructor de la clase.

- `~Triangulacion`: Destructor de la clase.

Clase Reconstruccion3D: Esta clase se encarga de obtener el mallado del objeto en cuestión, a partir de una nube de puntos 3D. En la Figura 47 se presenta la descripción de esta clase.

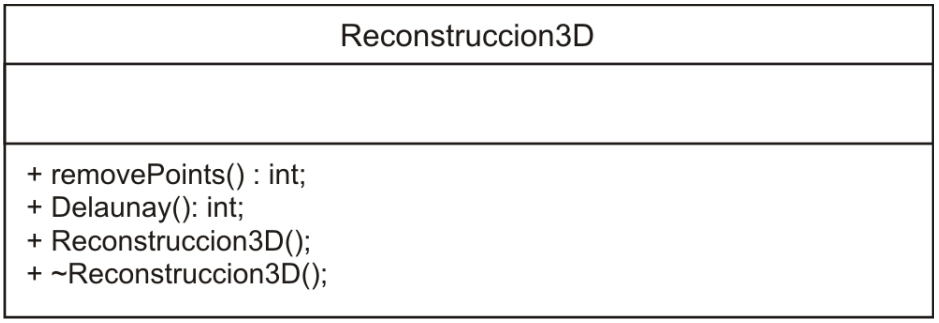


Figura 47 Clase Reconstruccion3D

Descripción de los métodos:

- `removePoints`: Esta función se encarga de eliminar un porcentaje de la nube de puntos para asi reducir el ruido que se haya podido generar durante el proceso.
- `Delaunay`: Se encarga de obtener el mallado triangular del objeto a través del método de Triangulación de Delaunay. Obtiene como resultado final un archivo .OFF con las especificaciones del mismo.
- `Reconstruccion3D`: Constructor de la clase.
- `~Reconstruccion3D`: Destructor de la clase.

4.3. Desarrollo de la Aplicación

El ciclo de vida de la aplicación se divide en tres etapas: Inicialización, ejecución y liberación. En la etapa de inicialización se obtienen los parámetros, imágenes e información inicial que debe ser suministrada por el usuario. La fase de la ejecución comprende el procesamiento de los datos e imágenes obtenidos, la ejecución de los algoritmos y el despliegue de la solución, con la cual el usuario tendrá la posibilidad de interactuar. La última fase comprende la liberación de todos los recursos utilizados en la aplicación.

4.3.1. Etapa de Inicialización

El primer paso es realizar la creación del ambiente donde se va a trabajar en OpenGL, esto se logra a través de la utilización de QT. En el siguiente código se puede observar la manera en que inicia la ejecución de la aplicación:

```
1. QApplication app(argc, argv);
2. Window window;
3. window.show();
```

El código anterior puede explicarse de la siguiente manera:

1. QApplication es una clase de QT que se encarga de gestionar el flujo de las aplicaciones que poseen interfaces gráficas, así como la configuración o ajustes principales. Es en ella donde todos los eventos desde ventanas y otras fuentes son atendidos y procesados. Por su parte, app es una instancia de esta clase, que permite inicializar el sistema de ventanas que se utilizará más adelante.
2. Window es un objeto que hereda todos los métodos y atributos de la clase QWidget, que no es mas que la clase base de todos los objetos de interfaz de usuario. Recibe todos los eventos que provienen del teclado, ratón o de la propia ventana y permite incluir en ella el contenido que se quiera mostrar por pantalla.
3. Esta acción muestra la ventana en pantalla, así como el contenido, menús o botones que se hayan incluido en ella.

Por lo tanto, es a través del objeto window que se creará la interfaz gráfica con la que interactuará el usuario a lo largo del proceso de reconstrucción 3D. El objeto windows se encarga entonces de:

- Crear la ventana de despliegue principal.
- Crear los botones o menús con los que interactuará el usuario y asignarle sus respectivas acciones.

4.3.1.1. Parámetros Iniciales

Los parámetros con los que debe contar el sistema para poder empezar el proceso de reconstrucción 3D y que deben ser ingresados por el usuario son:

- Las dimensiones del tablero de calibración que se utilizará en el proceso.

- Las tomas del patrón o tablero de calibración en diferentes posiciones y orientaciones.
- Las imágenes del objeto a reconstruir.

Para cumplir con estos requerimientos, se indica por pantalla al usuario que introduzca las dimensiones horizontales y verticales del tablero que utilizará. Una vez hecho esto, se despliegan dos ventanas donde se mostrarán las imagenes que en ese momento están captando las webcams conectadas al computador. El usuario entonces puede realizar las tomas del tablero de calibración, para ello sólo debe presionar la tecla Enter e inmediatamente serán tomadas las fotos de ambas cámaras. Cada vez que haya sido tomada una imagen, se le preguntará al usuario si desea realizar otra toma o, si por el contrario, ya culminó con el proceso. De ser así, las imágenes derecha e izquierda quedan almacenadas en el directorio y, a la vez, se crea un archivo de tipo .txt que contiene las dimensiones del tablero y el nombre de cada una de las imágenes capturadas que, por convención se decidió utilizar izqX.JPG con X el número de la foto, para las imágenes de la cámara izquierda y derX.JPG, para las fotos tomadas por la cámara derecha.

El proceso de captura de imágenes se realizó mediante la utilización de la librería OpenCV. El siguiente código muestra las funciones de esta librería utilizadas para realizar el proceso descrito:

```
1. CvCapture* capture = cvCaptureFromCAM(0);

2. cvSetCaptureProperty(capture, CV_CAP_PROP_FRAME_WIDTH, 640 );
3. cvSetCaptureProperty(capture,CV_CAP_PROP_FRAME_HEIGHT, 480 );

4. cvGrabFrame(capture);
5. img = cvRetrieveFrame(capture);
6. cvShowImage("Izquierda", img);
7. cvSaveImage(name,img);
8. cvReleaseCapture(&capture);
```

Dichas funciones ejecutan las siguientes acciones:

1. CvCapture es una estructura que permite obtener videos desde webcams o archivos AVI. En este caso se utilizó la función cvCaptureFromCAM(int index) para obtener imágenes desde las cámaras conectadas al computador.
2. cvSetCaptureProperty fue utilizada con el objeto de fijar un tamaño predeterminado para las imágenes que se capturarían, siendo éste de 640x480.
4. cvGrabFrame toma un fotograma desde la cámara web que fue inicializada con el nombre capture.
5. cvRetrieveFrame esta función toma la imagen del fotograma que fue obtenido anteriormente y la almacena en una variable llamada img que es del tipo IplImage.
6. cvShowImage muestra en pantalla la imagen que fue adquirida desde la cámara web.
7. cvSaveImage guarda la imagen obtenida en un archivo con el nombre que contenga la variable name.
8. cvReleaseCapture libera la estructura CvCapture para que el recurso quede nuevamente disponible.

Una vez que han sido adquiridas las imágenes del tablero de calibración, se inicia el proceso de calibración estéreo, con el fin de obtener las matrices de proyección de ambas cámaras que serán utilizadas más adelante. OpenCV ofrece funciones específicas para completar este proceso, a continuación se muestran las que fueron utilizadas en el desarrollo de la aplicación:

```
1.cvFindChessboardCorners( timg, cvSize(nx, ny), &temp[0],
                           &count,CV_CALIB_CB_ADAPTIVE_THRESH |
                           CV_CALIB_CB_NORMALIZE_IMAGE);

2.cvDrawChessboardCorners( cimg, cvSize(nx,ny), &temp[0], count,
result );

3.cvFindCornerSubPix( img, &temp[0], count, cvSize(11, 11),
cvSize(-1,-1),cvTermCriteria(CV_TERMCRIT_ITER+CV_TERMCRIT_EP,
30, 0.01) );
```

```
4. cvStereoCalibrate( &_objectPoints, &_imagePoints1,
&_imagePoints2, &_npoints, &_M1, &_D1, &_M2, &_D2, imageSize, &_R,
&_T,&_E,&_F,cvTermCriteria(CV_TERMCRIT_ITER+CV_TERMCRIT_EPS,100,
1e-5), CV_CALIB_FIX_ASPECT_RATIO +CV_CALIB_ZERO_TANGENT_DIST +
CV_CALIB_SAME_FOCAL_LENGTH  );

5. cvStereoRectify( &_M1, &_M2, &_D1, &_D2, imageSize,&_R,
&_T,&_R1, &_R2, &_P11, &_P21, 0,0);

6. cvInitUndistortRectifyMap(&_M1,&_D1,&_R1,&_P11,mx1,my1);
```

Las acciones que realizan dichas funciones se describen a continuación:

1. Localiza las esquinas presentes en cada una de las imágenes del patrón de calibración que fueron tomadas y almacenadas en la variable timg. CV_CALIB_CB_ADAPTIVE_THRESH transforma las imágenes a blanco y negro según un nivel de umbral fijo, y CV_CALIB_CB_NORMALIZE_IMAGE normaliza las imágenes antes de aplicar la umbralización.
2. Si la función anterior logró encontrar las esquinas en el patrón de calibración, esta función se encarga de dibujarlas en pantalla.
3. Esta función se encarga de localizar con mayor precisión la posición donde fueron encontradas previamente las esquinas.
4. Encuentra las matrices de calibración (_M1 y _M2) para ambas cámaras, así como sus coeficientes de distorsión (_D1 y _D2). También calcula los parámetros extrínsecos: matrices de rotación y traslación (_R y _T) y la matríz esencial y fundamental (_E y _F).
5. Calcula las matrices de rotación para cada cámara (_R1 y _R2), lo que permitirá que ambas queden alineadas en un mismo plano. También obtiene las matrices de proyección rectificadas (_P11 y _P21) para ambas cámaras en el nuevo sistema de coordenadas. La función debe aplicarse en cada imagen estéreo por separado.
6. Calcula el mapa de transformación (mx1 y my1) que debe ser aplicado en cada par de imágenes estéreo para que puedan ser alineadas en el eje de las y. La función debe usarse por separado en cada una de las imágenes.

A medida que se van utilizando cada una de las imágenes del tablero de calibración, se muestra en pantalla el resultado obtenido. Si se obtiene una imagen como la que se observa en la Figura 48, es porque la función logró encontrar todas las esquinas del mismo.

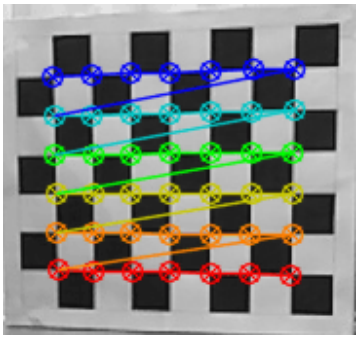


Figura 48 Calibración Correcta

Si por el contrario se obtiene la misma imagen o la imagen con puntos rojos sobre ella, como se muestra en la Figura 49, es porque no pudo distinguir las esquinas y debe realizarse el proceso nuevamente.

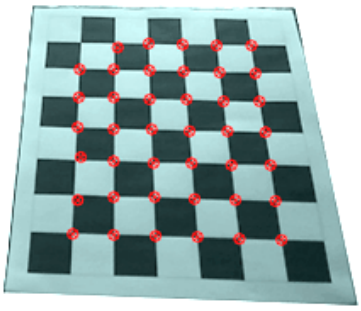


Figura 49 Calibración Incorrecta

Una vez que se han obtenido las imágenes del tablero y se ha realizado la calibración el próximo paso es obtener las imágenes del objeto en cuestión. Este procedimiento se realiza con las mismas funciones que se utilizaron en la obtención de las imágenes del patrón. El objeto ha de colocarse en el centro de la superficie giratoria y, cuando se desee tomar la foto, se presiona la tecla Enter obteniéndose así el primer par de ellas. Posteriormente, el usuario indica si desea continuar tomando fotos, de ser así,

gira la superficie una cantidad de grados predeterminados y nuevamente se presiona Enter, y así sucesivamente hasta obtener una vista de 360° del objeto.

4.3.2. Etapa de Ejecución

Una vez que se han obtenido las imágenes del objeto y se ha calibrado el sistema y calculado las matrices correspondientes, se puede iniciar la etapa de ejecución, donde se aplican un conjunto de algoritmos de manera iterativa para alcanzar la solución del sistema:

1. Detección de Bordes
2. Obtención de la Nube de Puntos 2D
3. Correspondencia de puntos entre un par de imágenes estéreo
4. Transformación de la nube de puntos bidimensional a tridimensional
5. Cálculo de las matrices de transformación para alinear las imágenes en el sistema de coordenadas de referencia

Estos 5 pasos se realizan de manera iterativa por cada par de imágenes estéreo. Al final se obtiene una matriz de puntos 3D que es utilizada por el proceso de triangulación de Delaunay para obtener el mallado 3D del objeto en cuestión.

1. Detección de bordes

Para la detección de bordes se utilizó una función de OpenCV que realiza el algoritmo de Canny y se aplica en cada una de las imágenes que fueron adquiridas:

```
cvCanny( img1r, gray, 80, 200, 3 );
```

Se establece un cierto umbral (80 – 200) sobre el cual actuará esta función con el fin de obtener los bordes más representativos del objeto.

2. Obtención de la Nube de Puntos 2D

Luego de haber aplicado el algoritmo de Canny en las imágenes se deben obtener los puntos característicos que definen la estructura del objeto. Para ello se utilizó una función llamada cvGoodFeaturesToTrack, función propia de la líbreria OpenCV.

Ésta se encarga de hallar y almacenar en un vector los puntos característicos más significativos que representan los detalles que definen la estructura del objeto,

pudiendo definir la de nsidad que se quiere obtener y si se desea utilizar el algoritmo de Harris para ello. Esta función se aplica únicamente en la imagen izquierda.

3. Correspondencia de puntos entre un par de imágenes estéreo

Una vez obtenida la nube de puntos de la imagen izquierda, se debe establecer una correspondencia entre los puntos de esta imagen y su par estéreo. Para realizar este proceso se utilizó la función de OpenCV cvCalcOpticalFlowPyrLK. Ésta se basa en el algoritmo de Lucas y Kanade Piramidal para encontrar los puntos comúnes entre ambas fotos. Recibe la nube de puntos que fue calculada previamente para la imagen izquierda y obtiene como resultado un vector que almacena los puntos correspondientes a ella de la imagen derecha.

Para determinar la correspondencia entre los puntos, la función crea un vector del mismo tamaño que el vector de puntos de la imagen izquierda denominado optical_flow_found_feature. Si encontró su punto correspondiente en la imagen derecha, entonces el arreglo en la posición del punto tendra un valor distinto de cero, en caso contrario significa que no halló correspondencia.

4. Transformación de la nube de puntos bidimensional a tridimensional

Al tener la correspondencia de puntos entre un par de imágenes estéreo, es posible calcular la coordenada Z de cada punto para así poder ubicarlo en el espacio tridimensional. Esto se logra mediante un procedimiento denominado triangulación lineal, que consiste básicamente en resolver, mediante el método de descomposición de valores singulares, la matriz

$$A = \begin{pmatrix} xp^{3T} - p^{1T} \\ yp^{3T} - p^{2T} \\ x'p'^{3T} - p'^{1T} \\ y'p'^{3T} - p'^{2T} \end{pmatrix}$$

En nuestro caso, una vez formada la matriz A se utilizó la función cvSVD de OpenCV para resolver el sistema. Dicha función devuelve la matriz diagonal D y las matrices ortogonales U y V.

La solución del sistema es la última columna de la matriz V, donde se encuentra el menor valor singular. Con ello se obtienen las coordenadas 3D de cada uno de los puntos que definen la forma del objeto.

5. Cálculo de las matrices de transformación para alinear las imágenes en el sistema de coordenadas de referencia

Con cada par de fotos estéreo se obtiene una nube de puntos tridimensional, cada una se encuentra en su propio sistema de coordenadas, pues se debe recordar que las fotos han sido tomadas despues de ir rotando el objeto una cierta cantidad de grados. Para obtener una nube de puntos general del objeto, las imágenes deben ser trasladadas a un mismo sistema de coordenadas, esto se logra utilizando el algoritmo Iterative Closest Point que fue explicado previamente.

En este sistema el algoritmo ICP fue provisto por una librería denominada VTK. En el siguiente código se muestra el desarrollo del algoritmo:

```
1.icp =
  vtkSmartPointer<vtkIterativeClosestPointTransform>::New();
2. icp->SetSource(newPolydata);
3. icp->SetTarget(referencePolydata);
4. icp->GetLandmarkTransform()->SetModeToRigidBody();
5. icp->SetMaximumNumberOfLandmarks(40);
6. icp->SetMaximumMeanDistance(0.001);
7. icp->SetMaximumNumberOfIterations(15);
8. icp->StartByMatchingCentroidsOn();
9. icp->Modified();
10. icp->Update();
11. vtkSmartPointer<vtkMatrix4x4> M = icp->GetMatrix();
```

A continuación se describe el algoritmo anterior:

1. Crea un objeto del tipo vtkIterativeClosestPointTransform, que es la que se encargará de llevar a cabo el proceso de alineación.
2. Se asigna la nube de puntos de la imagen que será alineada.

3. Se asigna la nube de puntos de la imagen que servirá como referencia para alinear la nueva imagen.
4. Establece los grados de libertad a la hora de encontrar la solución. En este caso se establece SetModeToRigidBody que sólo toma en cuenta la rotación y la traslación.
5. Fija el número máximo de puntos que se toman en cuenta a la hora de buscar la transformación correspondiente.
6. Establece la distancia media máxima entre dos iteraciones consecutivas. Si la distancia es menor la convergencia se detiene.
7. Fija la cantidad máxima de iteraciones para alcanzar la convergencia.
8. Inicia el proceso trasladando el centro de la imagen nueva al centro de la imagen que se toma como referencia
9. Le informa al objeto icp que es necesario que actualice su salida.
10. Para ejecutar el algoritmo.
11. Se obtiene la matriz de transformación que fue calculada por el algoritmo.

Una vez obtenida la matriz de transformación, debe aplicarse a cada uno de los puntos que forman parte de la imagen para ubicarlos en la posición y sistema de coordenadas adecuado.
Estos 5 pasos se repiten tantas veces como pares de imágenes estéreo hallan. Una vez finalizado, se obtiene una nube de puntos 3D que modela la forma del objeto y que se almacena en un archivo que será utilizado para conseguir el mallado del mismo.

Obtención del mallado 3D

Al obtener una nube de puntos 3D que modela la forma del objeto es posible producir el mallado que represente la superficie del mismo. Esto se logra mediante la Triangulización de Delaunay y utilizando la librería VTK para tal fin. En el siguiente código se muestran las funciones más importantes del algoritmo empleado:

```
1.  vtkSmartPointer<vtkCleanPolyData> cleaner =
     vtkSmartPointer<vtkCleanPolyData>::New();
     cleaner->SetInputConnection (reader->GetOutputPort());


 2.  vtkSmartPointer<vtkDelaunay3D> delaunay3D =
```

```
vtkSmartPointer<vtkDelaunay3D>::New();
delaunay3D->SetInputConnection (reader->GetOutputPort());
delaunay3D->SetAlpha(0.6);
```

A continuación se describen las funciones utilizadas:

1. Esta función se encarga de depurar la nube de puntos que se emplea, eliminando los puntos duplicados que puedan estar presentes en ella.
2. Con esta función se genera el mallado a partir de la nube de puntos depurada previamente. La función setAlpha permite fijar la distancia o separación que se quiere tengan los puntos de la nube de puntos de salida.

4.3.3. Etapa de Liberación

Al finalizar la aplicación es necesario liberar todos los recursos que han sido utilizados a lo largo del proceso de reconstrucción. Las estructuras de datos que contienen los puntos característicos del objeto, así como de las triangulaciones realizadas y aquéllas que permiten el manejo de las cámaras. Esta última es de gran importancia, pues si no llegara a ser liberado el recurso podría originar errores inesperados una vez finalizada la aplicación.

Al mismo tiempo, es necesario eliminar las instancias de las ventanas que fueron utilizadas para desplegar el resultado en pantalla.

Capítulo 5 Pruebas y Resultados

Una vez finalizada la etapa de diseño y desarrollo es necesario poner a prueba el sistema para evaluar rendimiento, eficiencia, precisión y determinar si se logran los objetivos. En este capítulo se presentarán las pruebas realizadas para escanear un objeto incorporando variaciones en el hardware y parámetros del sistema.

5.1. Descripción de los Escenarios

Se utilizaron dos objetos de prueba:

- Modelo #1: Un cubo de papel donde cada uno de sus lados está pintado con el objetivo de obtener algún tipo de textura, como se observa en la Figura 50.

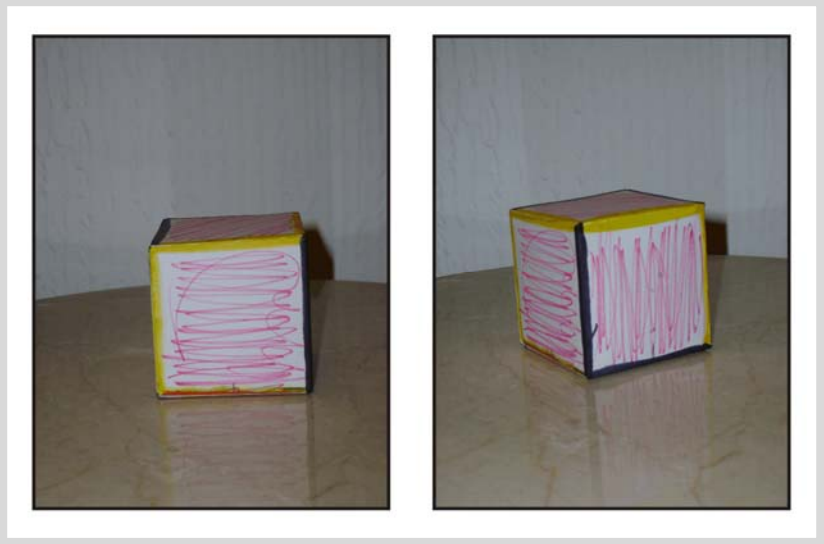


Figura 50 Modelo #1 - Cubo de Papel

- Modelo #2: Un recipiente en forma de conejo con notables relieves y texturas, tal como se observa en la Figura 51.

Figura 51 Modelo #2 - Recipiente en forma de conejo

- Modelo #3: Un cubo con estampado floral, como se muestra en la Figura 52.

Figura 52 Modelo #3 Cubo con estampado floral

Para evaluar el desempeño del sistema y determinar los parámetros y configuración del hardware que permitan obtener mejores resultados a la hora de escanear un objeto, se realizaron las siguientes pruebas:

1. Variación del número de imágenes utilizadas para la calibración del sistema.
2. Utilización de un ambiente controlado y otro libre.
3. Variaciones en la colocación del hardware utilizado: distancia entre las cámaras web y distancia de las cámaras con respecto a la base del objeto.
4. Variación de la triangulación de Delaunay.

5.1.1. Variación del número de imágenes utilizadas para la calibración del sistema

La calibración del sistema fue realizada utilizando funciones de la librería de OpenCV. Ésta utilizaron 13 imágenes para lograr una correcta calibración de las cámaras empleadas en el sistema. La distancia focal real de las cámaras empleadas es de 3,7mm y las dimensiones de su sensor son de 4,5mm de ancho y 3,4mm de alto, estos datos serán de utilidad más adelante.

Para probar la importancia de este proceso y determinar cuál es la cantidad de imágenes necesaria, se realizaron pruebas utilizando 4, 8 y 13 fotos. A continuación se muestran los resultados obtenidos.

En la Figura 53 se observa el resultado de utilizar sólo 4 imágenes para la calibración del sistema.

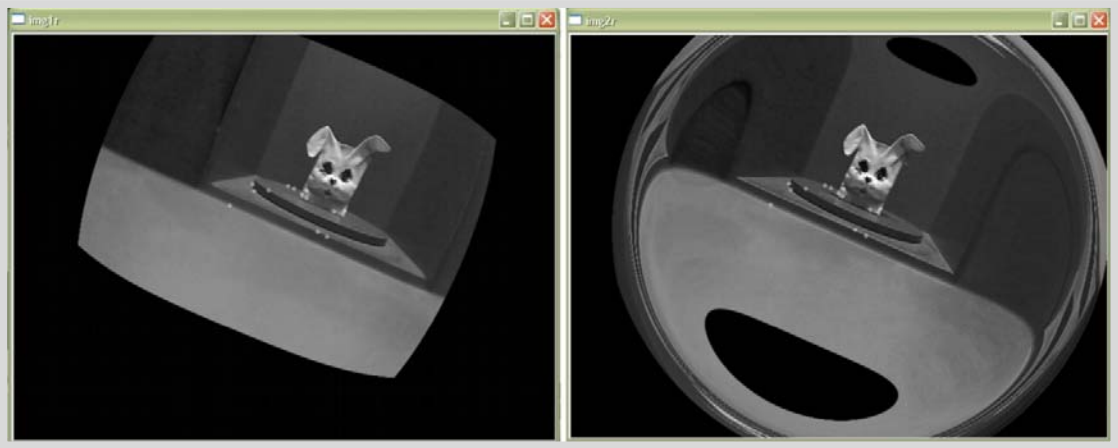


Figura 53 Calibración utilizando 4 imágenes

Se considera que el proceso de calibración no fue adecuado, ya que en los resultados obtenido se puede observar que al menos una de las imágenes sufrió serias alteraciones y, además, los pocos puntos que se obtuvieron no forman parte del objeto. Támbien, al consultar la distancia focal calculada durante el proceso f_c se obtuvo que ésta fue de 606 píxeles. Básandonos en este valor y sabiendo que las cámaras empleadas tienen una distancia focal real f de 3,7mm es posible obtener las dimensiones del sensor utilizado.

Para ello, se divide la distancia focal real f entre la distancia focal en pixeles f_c y se multiplica por el ancho de la imagen w (para obtener el ancho del sensor, s_W) y por el alto de la misma h (para obtener el alto del sensor, s_H).

$$s_W = \frac{f}{f_c}\, x\, w \qquad s_H = \frac{f}{f_c}\, x\, h$$

Siguiendo esto, según los cálculos de la calibración realizada, el sensor tiene un ancho de 3,90mm y un alto de 2,9mm lo cual dista mucho de las dimensiones reales del mismo.

Por otro lado, en la Figura 54 se observa el resultado de calibrar las cámaras con 8 imágenes.

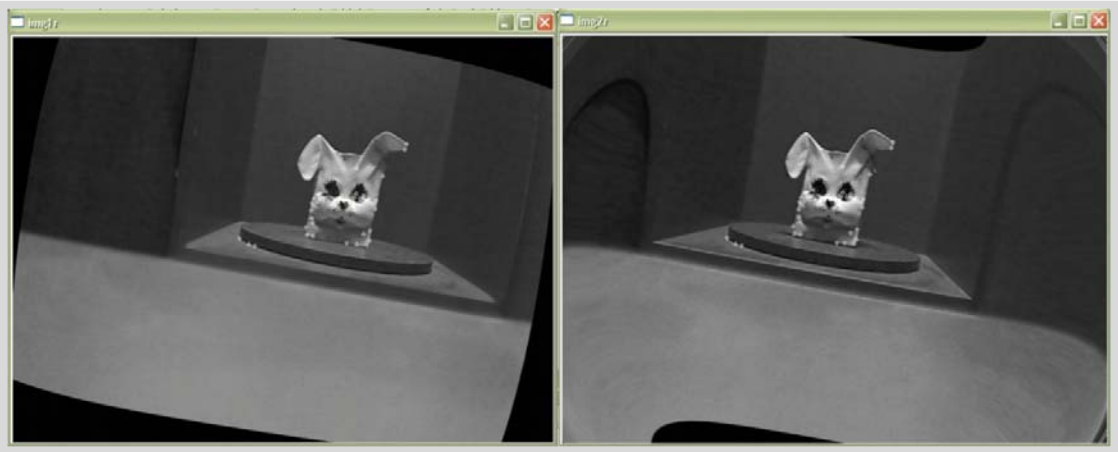


Figura 54 Calibración utilizando 8 imágenes

El proceso de calibración tampoco resultó exitoso. Basándonos en los resultados obtenidos, se puede observar como la segunda imagen presenta alteraciones y, además, los puntos obtenidos no pertenecen al objeto en cuestión. Entonces, la distancia focal obtenida fue de 631 píxeles, siguiendo los cálculos anteriores tendríamos un sensor de 3,75mm de ancho y 2,81mm de alto, que tampoco concuerda con las dimensiones reales del sensor.

Por último, en la Figura 55 se observa el resultado de aplicar la calibración con 13 imágenes.

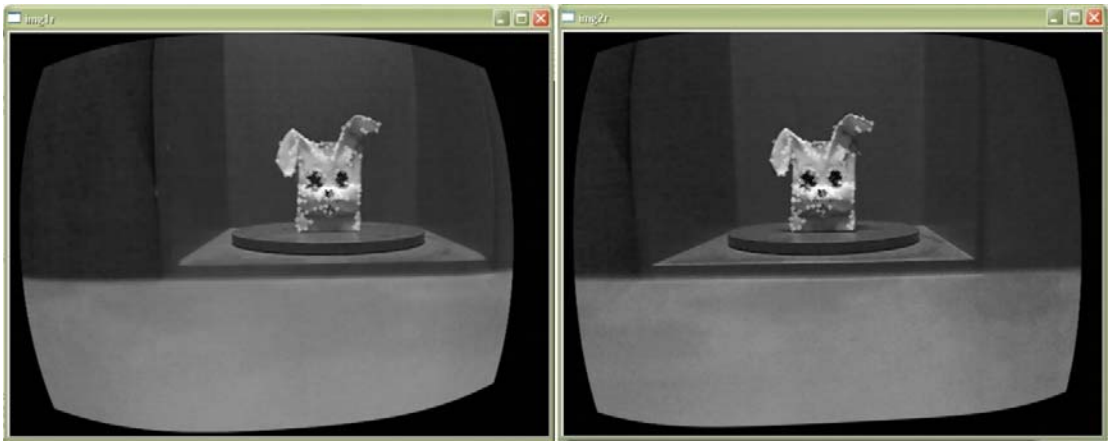


Figura 55 Calibración utilizando 13 imágenes

En este caso vemos como la imagen preserva sus características iniciales, y como los puntos característicos obtenidos pertenecen efectivamente al objeto que se desea escanear. En cuanto a la distancia focal computada, ésta fue de 548 píxeles. Al calcular las dimensiones del sensor se obtiene 4,32mm de ancho y 3,24 de alto lo que se acerca bastante a las dimensiones reales del sensor empleado.

De esta manera, se sabe que es necesario emplear al menos 13 imágenes para poder obtener una buena calibración y con ello mejores resultados durante el proceso de reconstrucción 3D.

5.1.2. Utilización de un ambiente controlado y otro libre

Se realizaron dos pruebas, ambas con una distancia cámara-cámara de 5cm y distancia cámaras-base de 48cm. Una de ellas se encontraba en un ambiente controlado y la otra no. Para el ambiente controlado, como fue explicado con anterioridad, se colocó al objeto dentro de una caja unicolor. A continuación se muestran las imágenes de los ejemplos realizados.

En la Figura 56 se puede observar como los puntos adquiridos no pertenecen únicamente al objeto que se desea escanear, puesto que, al estar en un ambiente libre, se presenta mucho ruido que interfiere con el objeto.

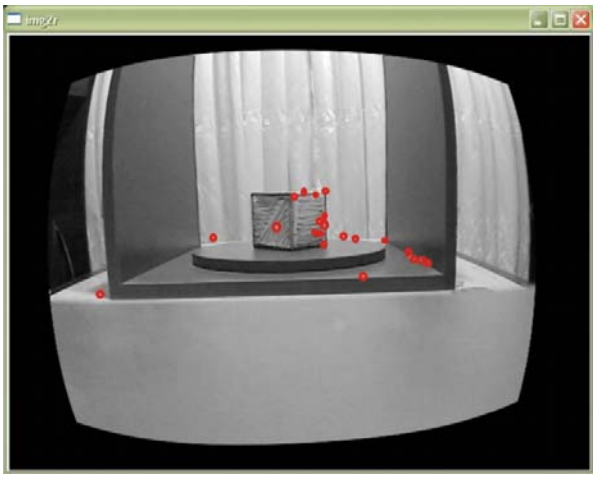


Figura 56 Calibración Ambiente Libre

Por otro lado, al realizar el proceso en un ambiente controlado, donde el foco de atención es el objeto a escanear, los puntos característicos obtenidos pertenecen al objeto en cuestión, en este caso al cubo. Tal como se observa en la Figura 57.

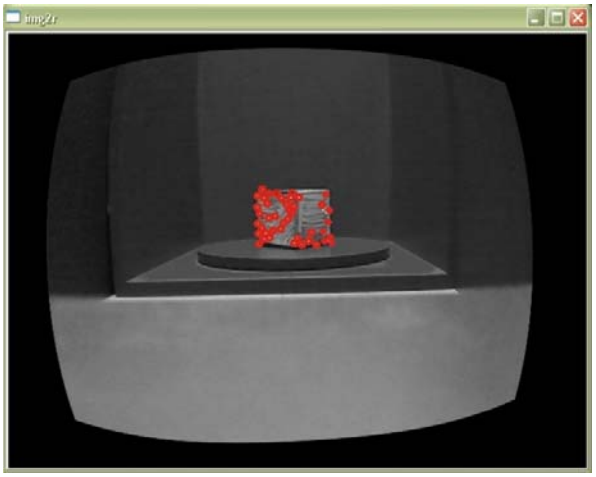


Figura 57 Calibración Ambiente Controlado

Al intentar realizar la reconstrucción bajo un ambiente libre, el proceso no se pudo completar, ya que, en algunas ocasiones el sistema no fue capaz de obtener puntos característicos en la imagen. De allí la importancia de contar con un ambiente controlado a la hora de realizar este tipo de procedimientos.

5.1.3. Variaciones en la colocación del hardware utilizado

Se realizaron 6 experimentos distintos variando la distancia entre las cámaras y la distancia de ellas con respecto a la base. El sistema estereoscópico empleado se fundamenta en el sistema de visión humano, por ello las distancias consideradas entre las cámaras fueron de 5cm y 6cm, que es aproximadamente la distancia que existe entre los ojos:

Experimento #1:

- Separación de las cámaras = 5cm
- Separación cámaras-base = 15cm

Experimento #2:

- Separación de las cámaras = 5cm
- Separación cámaras-base = 30cm

Experimento #3:

- Separación de las cámaras = 5cm
- Separación cámaras-base = 48cm

Experimento #4:

- Separación de las cámaras = 6cm
- Separación cámaras-base = 15cm

Experimento #5:

- Separación de las cámaras = 6cm
- Separación cámaras-base = 30cm

Experimento #6:

- Separación de las cámaras = 6cm
- Separación cámaras-base = 48cm

A continuación se puede observar en la Figura 58 un par de imágenes por cada distancia cámara-base, que representan la foto derecha e izquierda obtenidas en los Experimentos #1, #2 y #3.

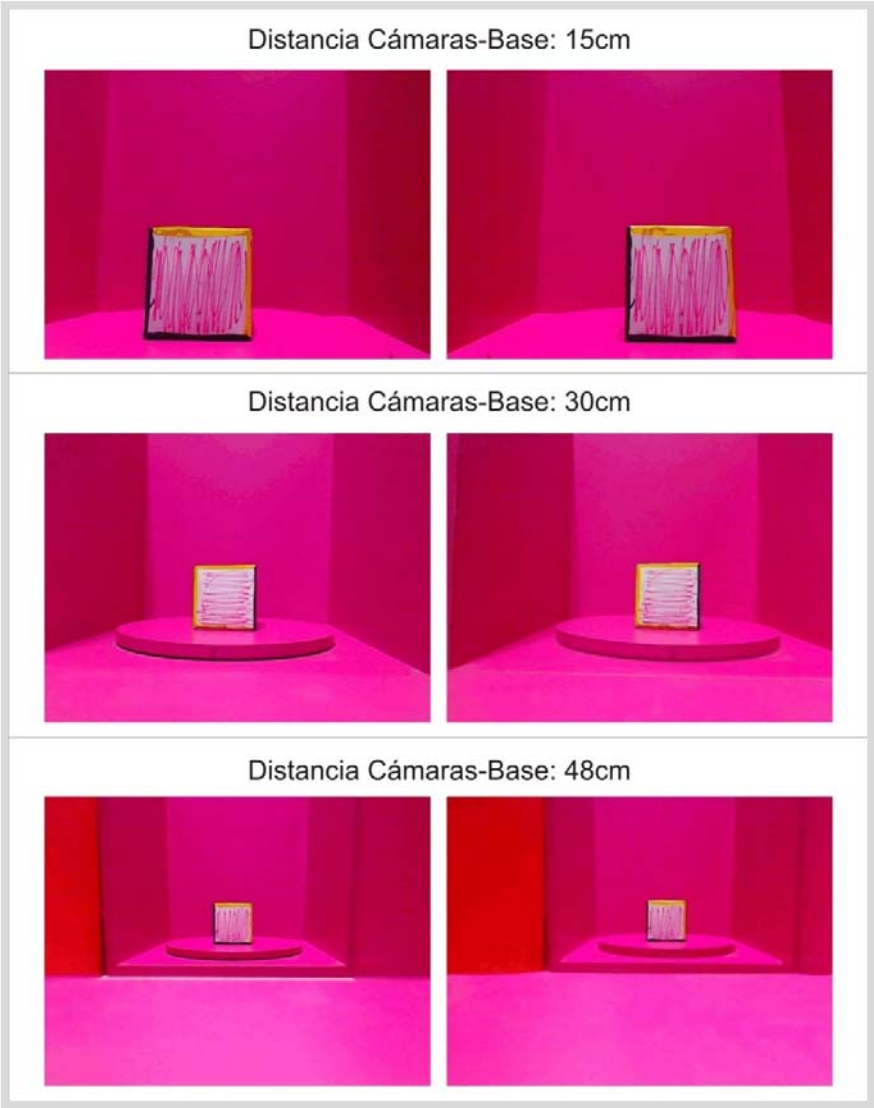


Figura 58 Imágenes estéreo distancia cámara-cámara de 5cm

Luego, se puede observar un conjunto de imágenes de ejemplo obtenidas al realizar tomas con una separación entre cámaras de 6cm, que corresponden a los Experimentos #4, #5 y #6, tal como ilustra la Figura 59.

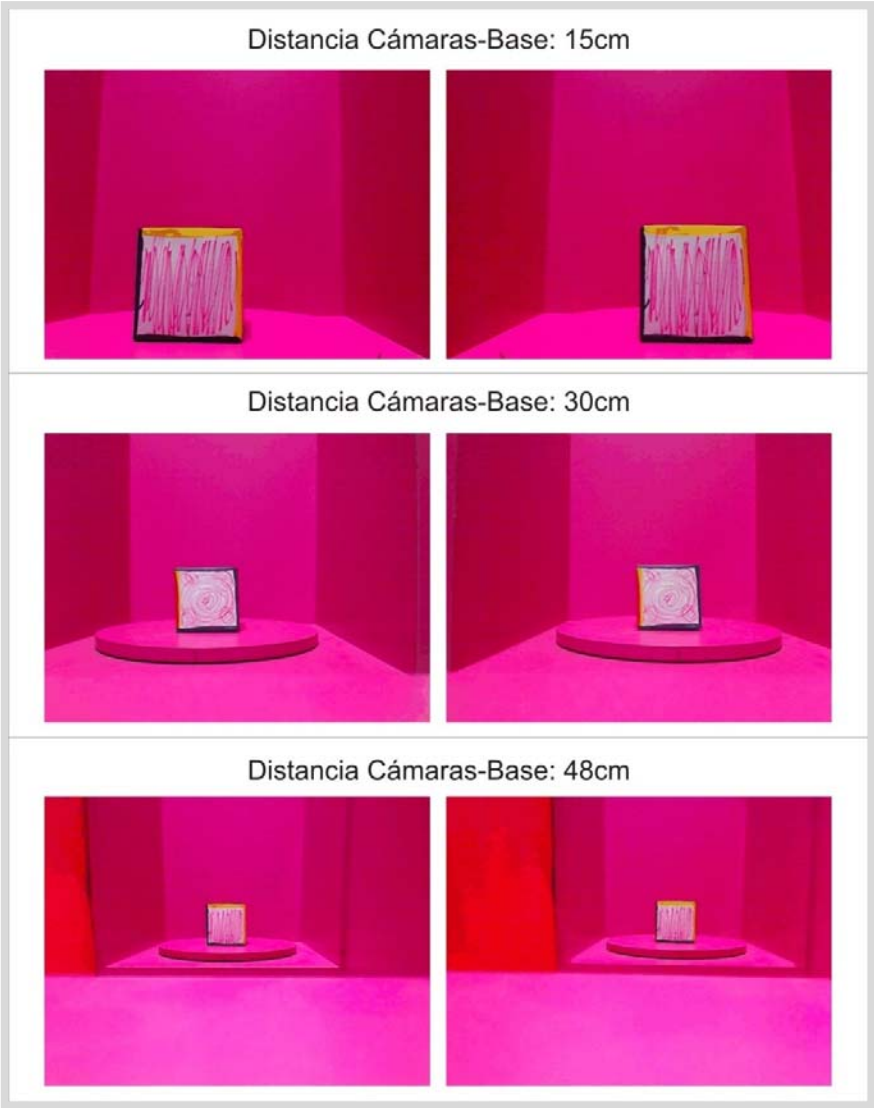
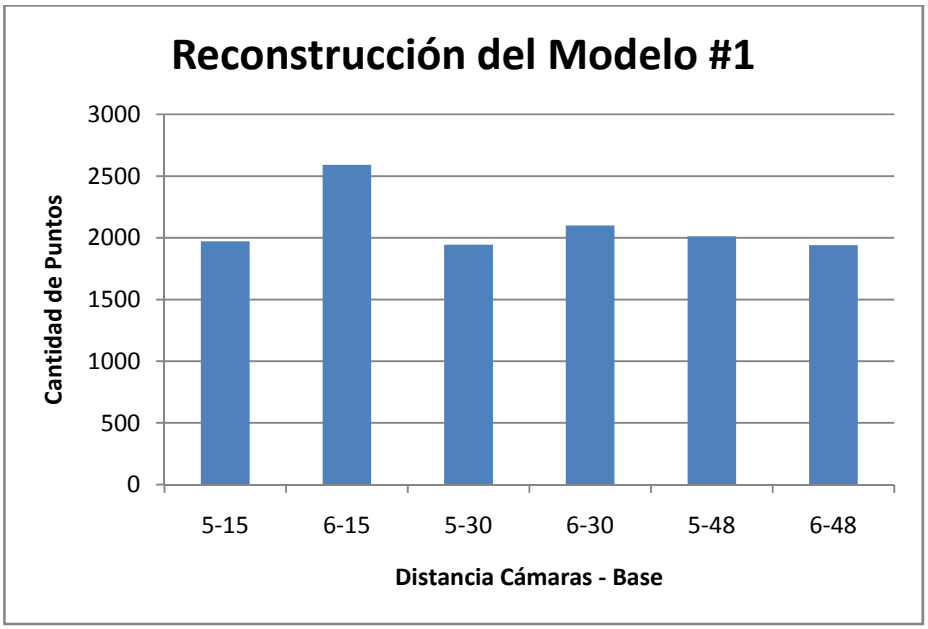


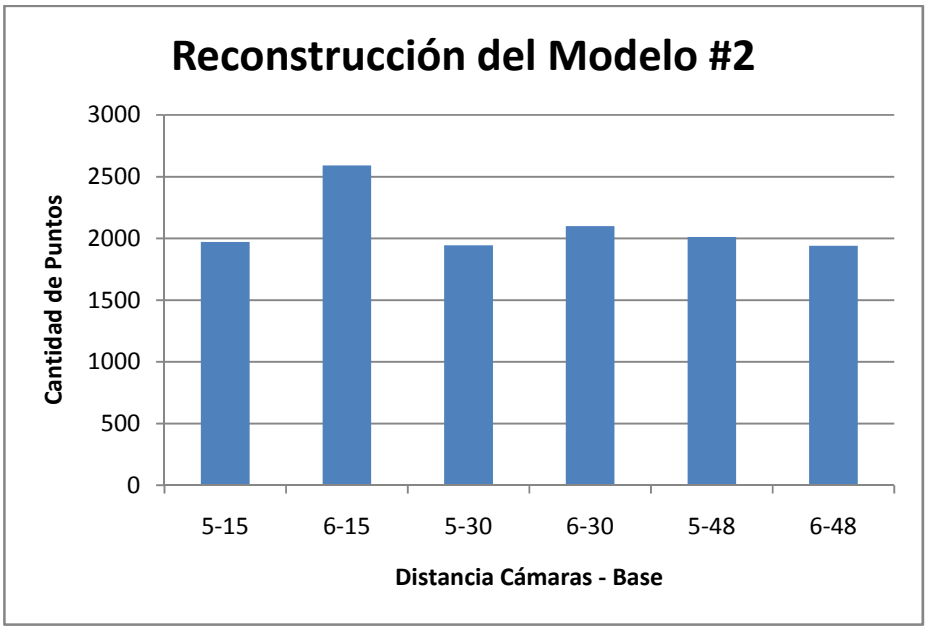
Figura 59 Imágenes estéreo distancia cámara-cámara de 6cm

Las variaciones realizadas influyen directamente sobre la cantidad de puntos característicos del objeto que es posible obtener, resultando una notable diferencia entre objetos lisos como el cubo y objetos con relieves y texturas bien definidas como el recipiente con forma de conejo.

En la gráfica que sigue se observa como a medida que se va incrementando la distancia entre las cámaras y la base la cantidad de puntos tiende a disminuir y, como el hecho de separar las cámaras un centímetro más entre ellas fue un factor importante para aumentar la cantidad de puntos obtenidos.



Por otro lado, en el caso del modelo #2, no hubo una diferencia tan marcada en la cantidad de puntos, esto debido a que presenta una estructura bastante definida y marcada en su relieve. Sin embargo, se mantiene la misma tendencia de disminución de puntos a medida que aumenta la distancia cámara-base y de aumento de puntos al incrementar la distancia entre las cámaras.



Es importante resaltar como objetos con estructuras bien marcadas y definidas como el modelo #2, mantienen un índice de puntos constante a pesar de las variaciones realizadas. Mientras que un objeto liso como el modelo #1, cuya textura está representada por dibujos en sus caras, varía notablemente dependiendo de las condiciones en que sea escaneado el mismo.

Si bien es cierto que es importante contar con una buena cantidad de puntos característicos para obtener un resultado óptimo y lo más parecido a la realidad, también es cierto que cuando la cantidad de puntos es muy grande tiende a ser redundante y los resultados obtenidos distan mucho de lo esperado. A continuación se detallan los resultados alcanzados al modelar ambos objetos bajo las distancias cámara-base y cámara-cámara antes mencionadas.

La Figura 60 representa la reconstrucción obtenida al escanear el modelo #1 con una distancia entre las cámaras de 5cm. Se puede observar como a medida que va aumentando la distancia cámaras-base la cantidad de puntos disminuye y con ello la estructura del objeto empieza a tomar forma. Sin embargo, aun no se define claramente que el objeto escaneado es un cubo.

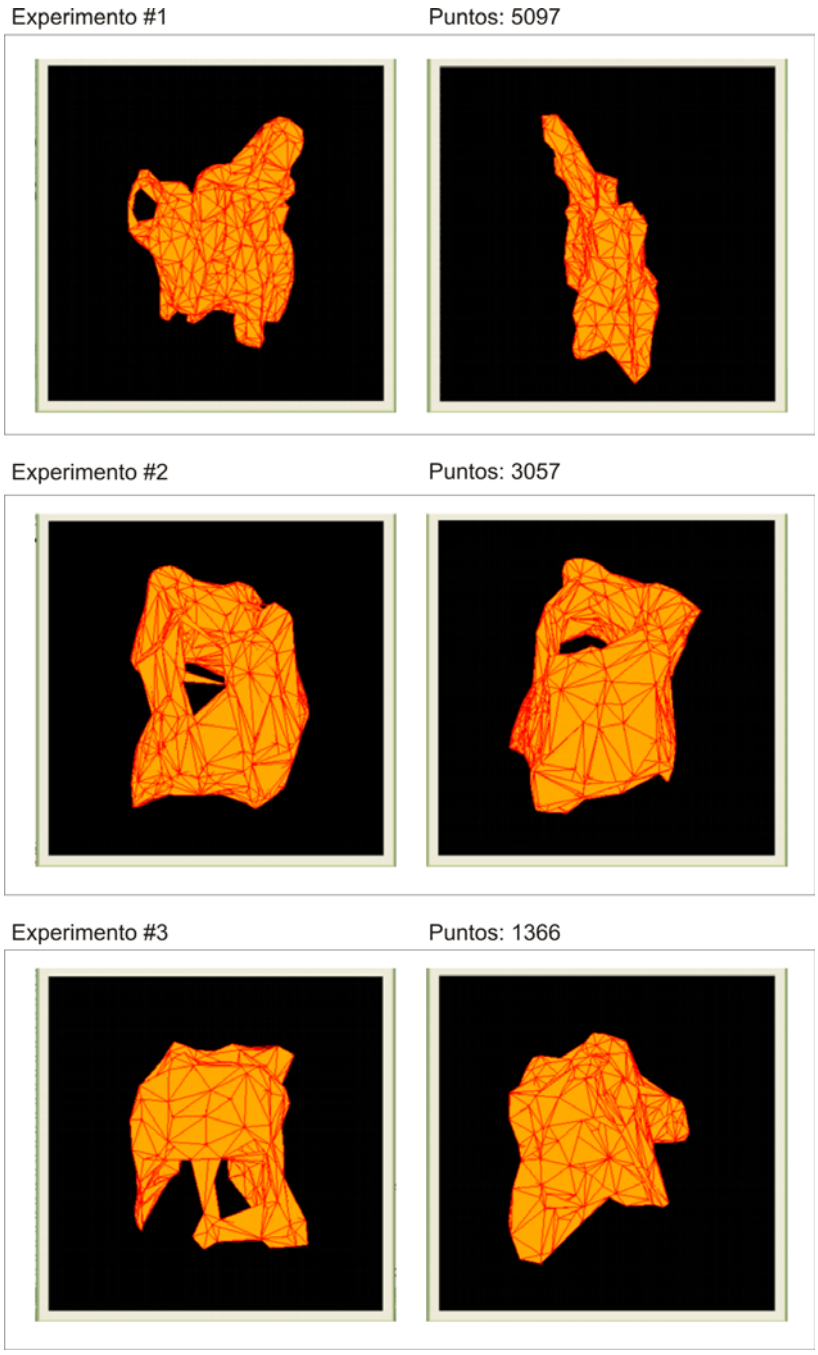


Figura 60 Reconstrucción del modelo #1 con una distancia cámara-cámara de 5cm

En la Figura 61, se muestran los resultados obtenidos con una distancia cámara-cámara de 6cm. Se puede observar como desde un principio hay mayor definición y semejanza de la reconstrucción obtenida con el objeto real que en el caso anterior y,

como la distancia cámaras-base utilizada, permitió alcanzar una mayor definición en la estructura, que se va pareciendo más al cubo escaneado.

En este ejemplo se puede confirmar lo señalado anteriormente con respecto a la cantidad de puntos característicos y el modelado del objeto. A simple vista existe una gran diferencia entre el resultado obtenido en el experimento #1 y el experimento #6. Por más que el primero triplica la cantidad de puntos del segundo, este último tiene mejor definición y semejanza con el objeto real.

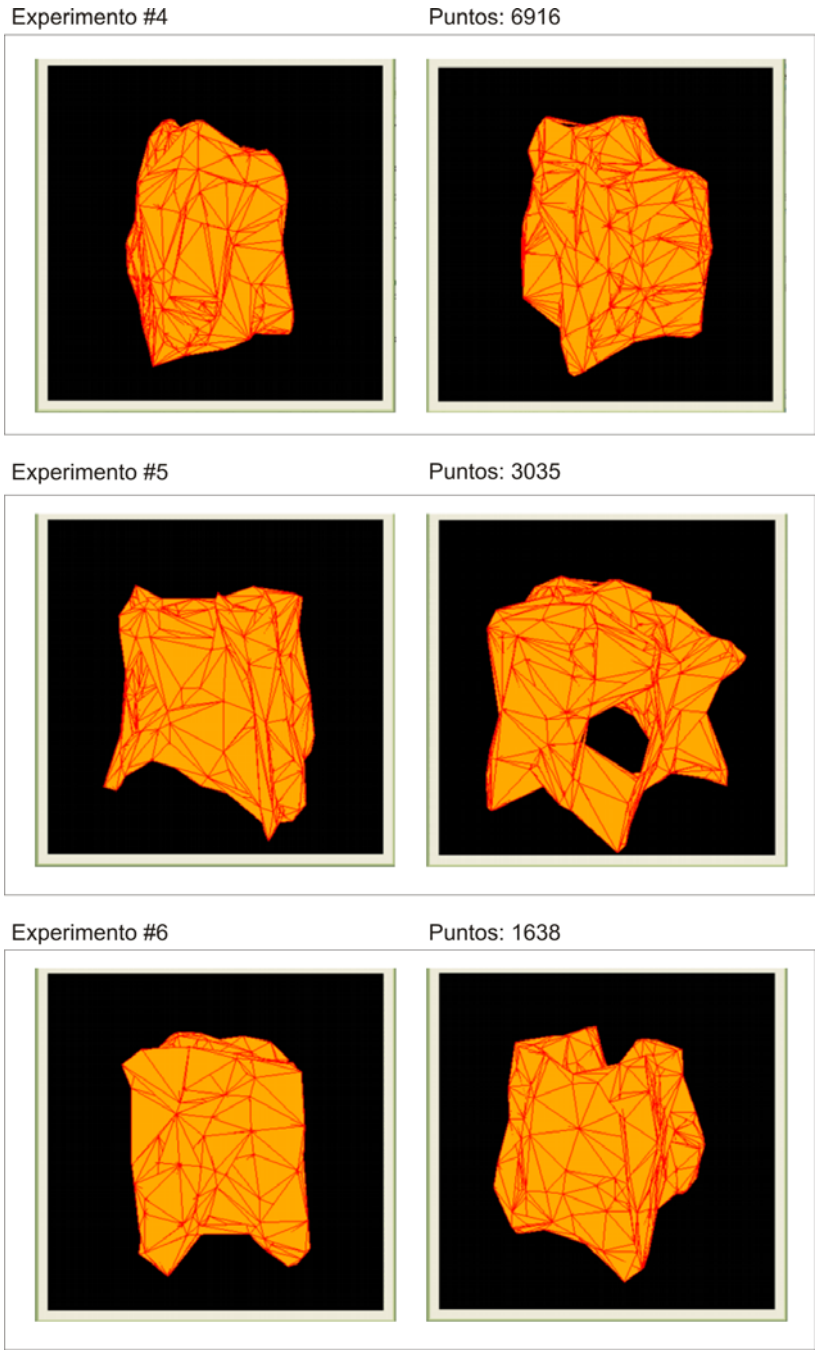


Figura 61 Reconstrucción del modelo #1 con una distancia cámara-cámara de 6cm

Por otro lado, en la Figura 62 se observan los resultados alcanzados al escanear el modelo #2 con una distancia cámara-cámara de 5cm.

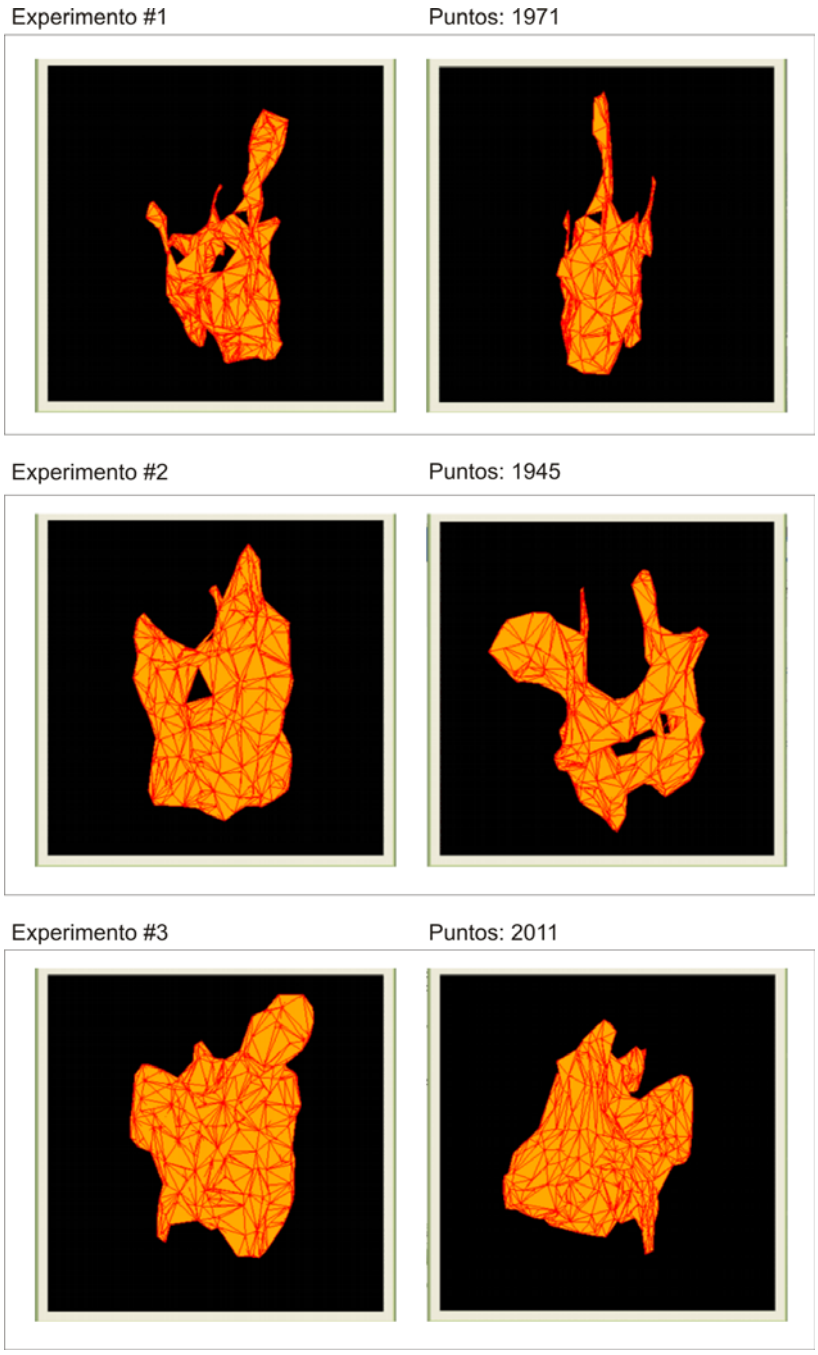


Figura 62 Reconstrucción del modelo #2 con una distancia cámara-cámara de 5cm

La definición alcanzada es muy pobre, incluso se puede apreciar como la cara trasera y delantera del objeto no logran unirse para formar las 4 paredes que componen al recipiente.

Por otra parte, cuando se compara con los resultados obtenidos al procesar el objeto con una distancia cámara-cámara de 6cm se observa una gran diferencia. En la Figura 63 se presentan estos resultados.

Poco a poco, con cada experimento el modelo se va pareciendo más a la realidad. Nuevamente, se obtienen mejores resultados con una menor cantidad de puntos característicos y con una mayor distancia entre las cámaras y la base donde se encuentra el objeto.

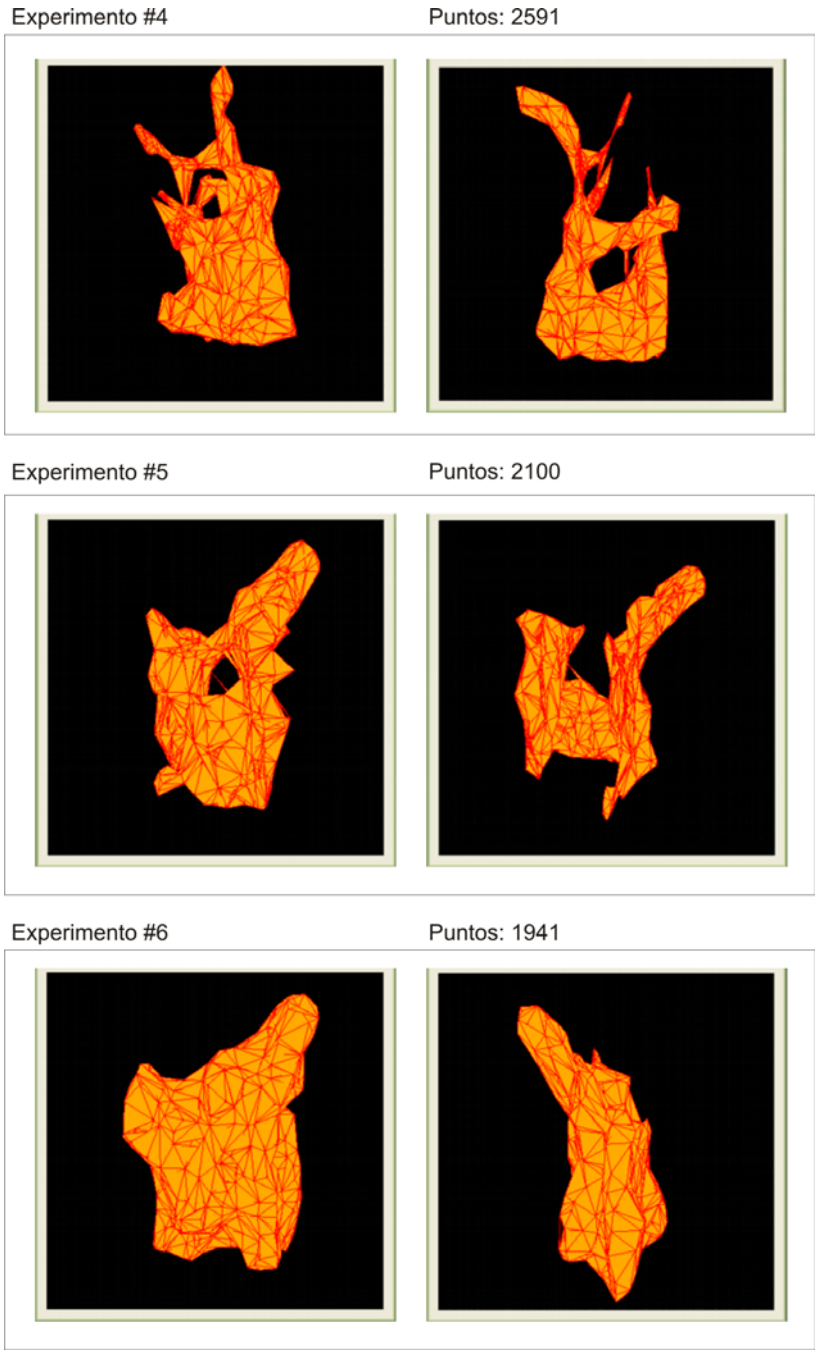


Figura 63 Reconstrucción del modelo #2 con una distancia cámara-cámara de 6cm

La distancia entre las cámaras y la base del objeto determinan la diferencia entre el par de imágenes, a menor distancia mayor es la diferencia que se obtiene entre el par de imágenes estereoscópicas. Todas estas variaciones influyen de manera importante en un fenómeno denominado oclusión, donde una característica de una

imagen puede ocultarse en la otra del par estereoscópico. También, al ser captadas las fotos desde diferentes posiciones o ángulos, las condiciones de iluminación pueden ser ligeramente diferentes o incluso pueden aparecer reflejos en una imagen que están ausentes en otras, sobretodo en objetos que reflejen la luz como es el caso del modelo #2.

5.1.4. Variación de la triangulación de Delaunay

La librería empleada para realizar la triangulación de Delaunay fue VTK, como se mencionó anteriormente. Ésta permite que se manipule un parámetro conocido como *alpha*, que representa el tamaño del círculo que circunscribe a cada triángulo y con ello se modifica tanto la cantidad de triángulos presentes como su tamaño.

Se realizaron 5 experimentos a cada objeto, con un *alpha* distinto en cada caso:

- Experimento #1: alpha = 0.0, valor por defecto de la función
- Experimento #2: alpha = 0.25
- Experimento #3: alpha = 0.5
- Experimento #4: alpha = 0.75
- Experimento #5: alpha = 1.0

En la Figura 64 a Figura 68 se pueden observar los resultados al realizar cada experimento sobre el modelo #1.

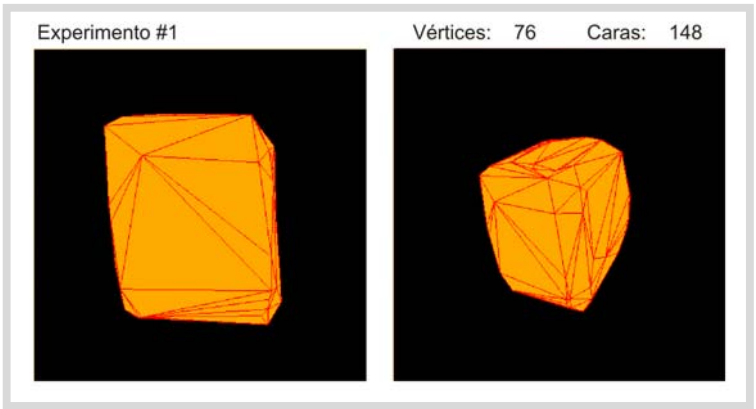


Figura 64 Modelo #1 con una triangulación de Delaunay con alpha = 0

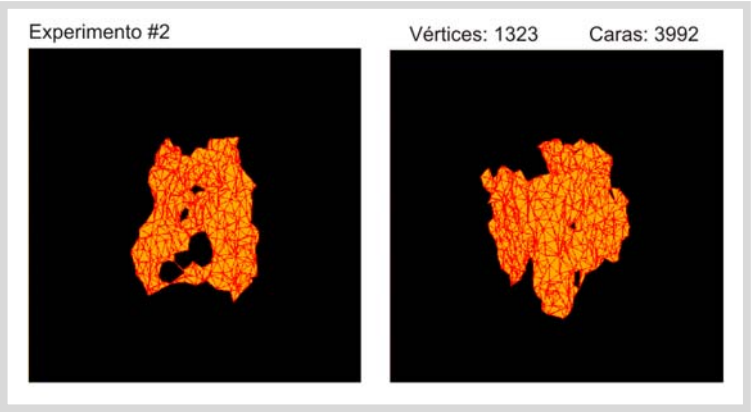


Figura 65 Modelo #1 con una triangulación de Delaunay con alpha = 0.25

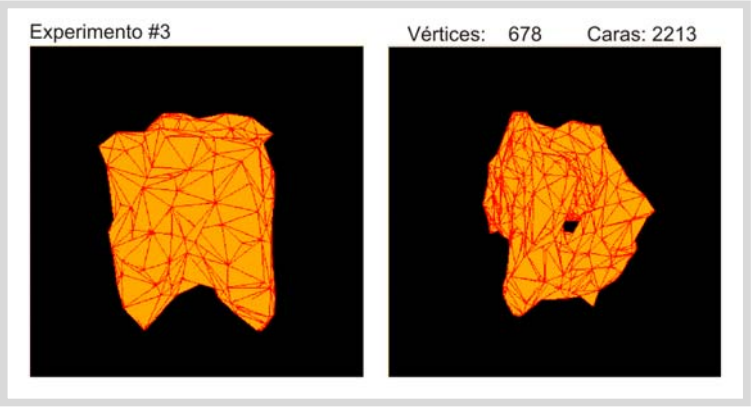


Figura 66 Modelo #1 con una triangulación de Delaunay con alpha = 0.5

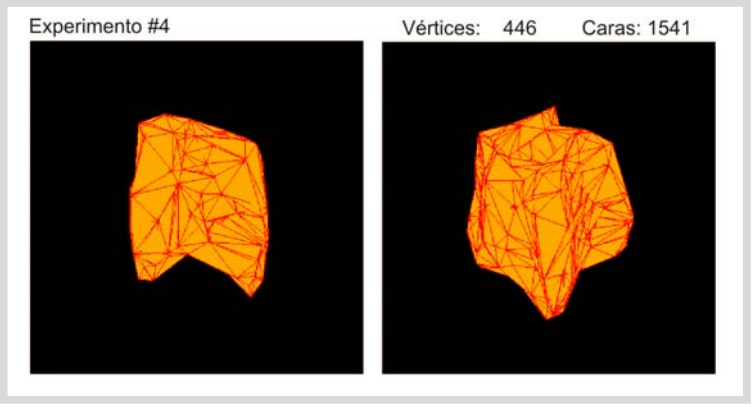


Figura 67 Modelo #1 con una triangulación de Delaunay con alpha = 0.75

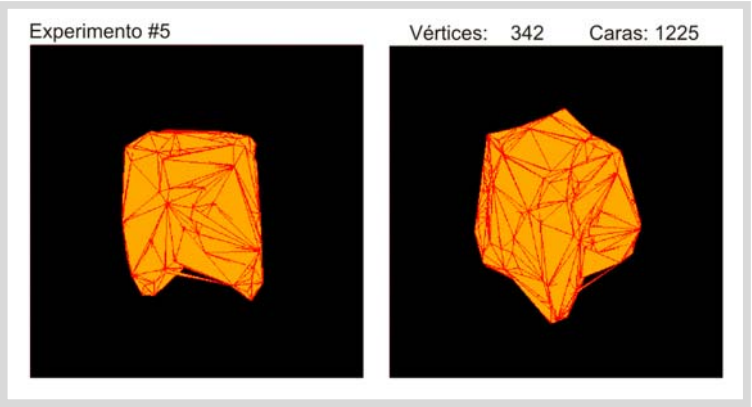


Figura 68 Modelo #1 con una triangulación de Delaunay con alpha = 1.0

Como se puede observar, el experimento #1, que usa el valor por defecto con el que viene la función, obtuvo el mejor resultado. Con éste se dibujan todos los triángulos sin importar cuán separados se encuentren sus vértices, lo que da como resultado triángulos de diversos tamaños, desde muy grandes a muy pequeños.

Por otro lado, al analizar el resto de los experimentos, se observa como a medida que se incrementó el *alpha* la estructura se fue pareciendo más al objeto real. Esto se debe a que con un *alpha* de 0.25 solo los triángulos cuya circunferencia circunscrita sea menor a dicho valor serán dibujados, excluyéndose el resto de los vértices y obteniendo una reconstrucción tridimensional errónea. Un objeto liso y sin mayor relieve ni estructura genera menor cantidad de puntos que aquél que tiene una estructura bien definida, como se pudo apreciar en los experimentos anteriores, esto también influye en los resultados obtenidos.

A continuación, en la Figura 69 a Figura 73 se pueden observar los resultados obtenidos al aplicar los experimentos sobre el modelo #2.

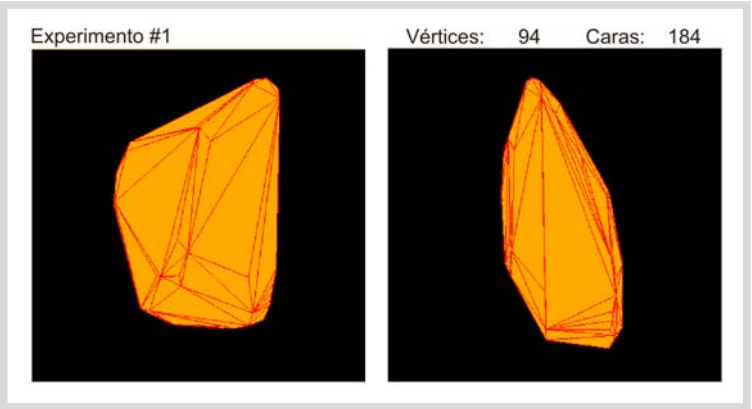


Figura 69 Modelo #2 con una triangulación de Delaunay con alpha = 0

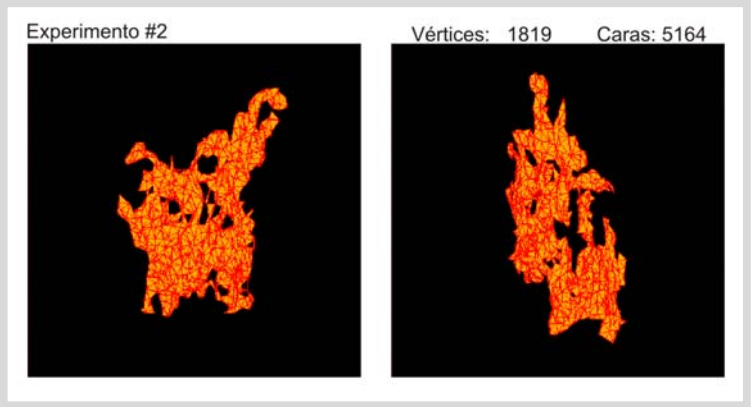


Figura 70 Modelo #2 con una triangulación de Delaunay con alpha = 0.25

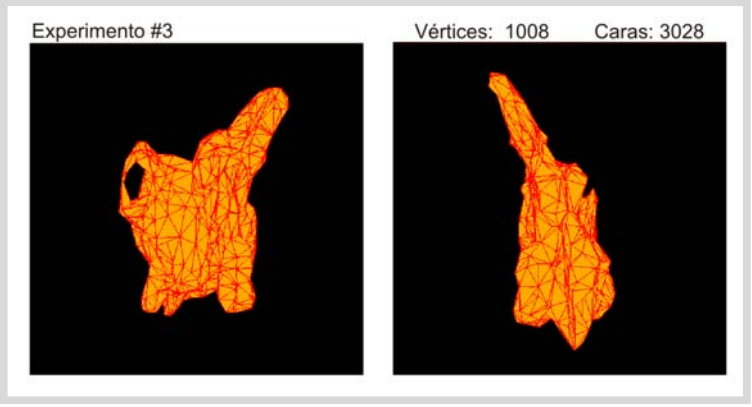


Figura 71 Modelo #2 con una triangulación de Delaunay con alpha = 0.5

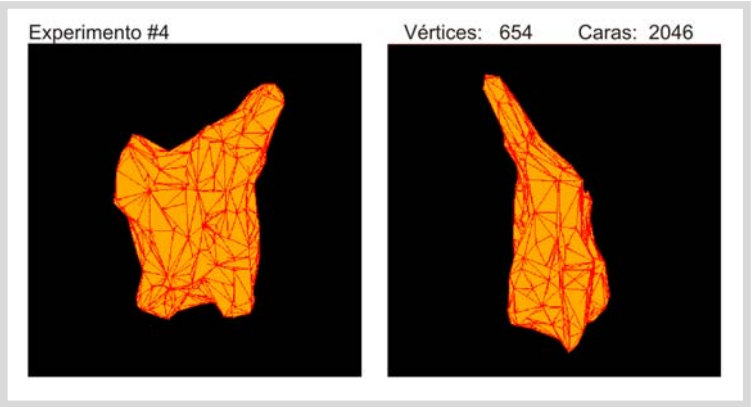


Figura 72 Modelo #2 con una triangulación de Delaunay con alpha = 0.75

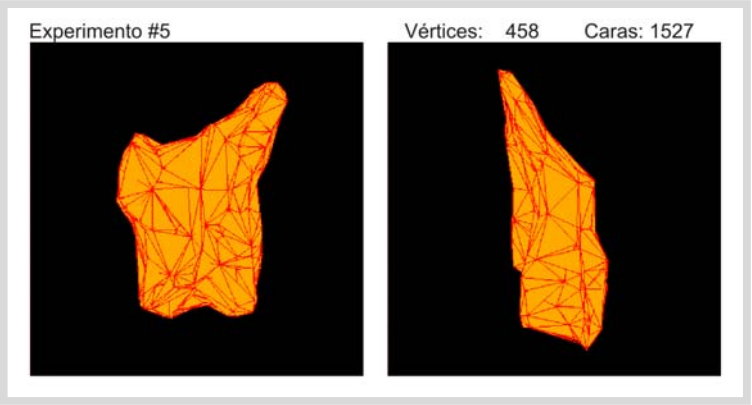


Figura 73 Modelo #2 con una triangulación de Delaunay con alpha = 1.0

En este caso, el resultado obtenido con el experimento #1 no resultó ser el mejor pues, como se puede apreciar, no existe definición en el relieve ni estructura del objeto modelado.

La reconstrucción 3D resultó más adecuada cuando el *alpha* tomó los valores de 0.75 y 1.0 y los comprendidos entre ellos. Al tratarse de un objeto con relieves y estructura marcada, es necesario que los triángulos sean más pequeños para que la correspondencia sea más precisa y el detalle sea mayor. En este caso, se logró definir las orejas del conejo, así como el relieve que presenta en la cara frontal.

Por otro lado, se realizó un último experimento sobre el modelo #3, con la configuración de la cámara que arrojó mejores resultados para el modelo #1 (distancia entre cámaras de 6cm y distancia cámara-base de 48cm) y con los distintos valores del *alpha* de la triangulación de Delaunay usados anteriormente. Los resultados pueden observarse en la Figura 74 a Figura 77.

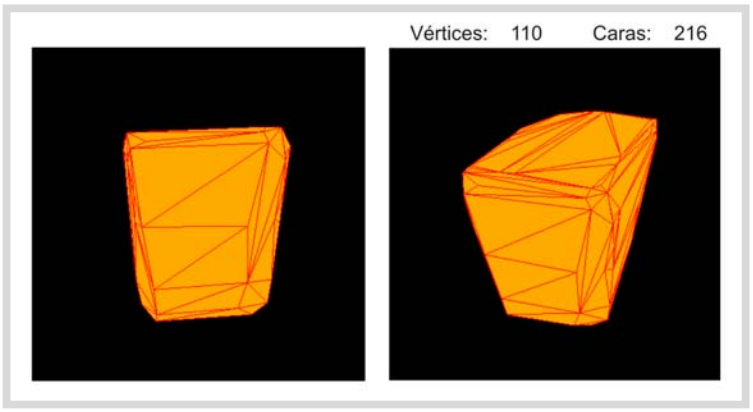


Figura 74 Modelo #3 con una triangulación de Delaunay con alpha = 0

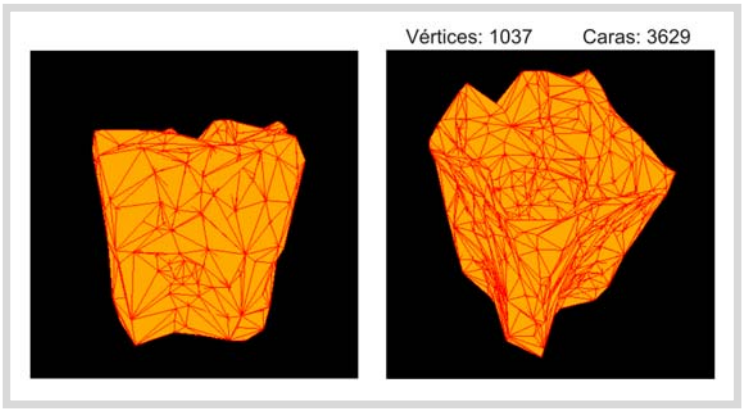


Figura 75 Modelo #3 con una triangulación de Delaunay con alpha = 0.5

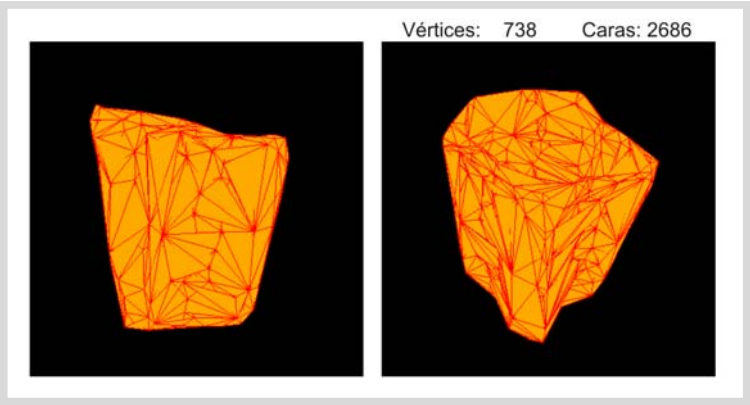


Figura 76 Modelo #3 con una triangulación de Delaunay con alpha = 0.75

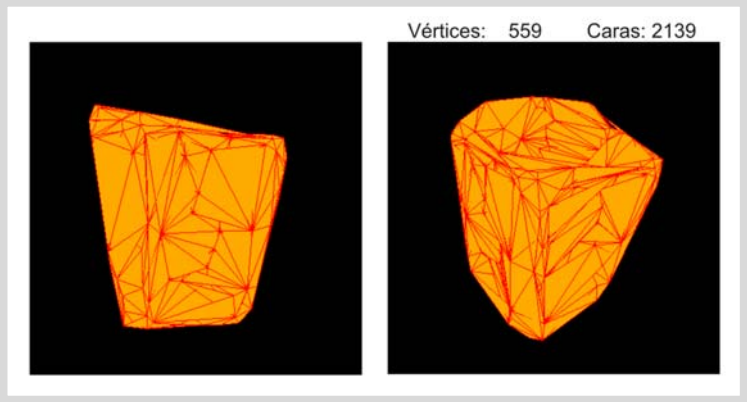


Figura 77 Modelo #3 con una triangulación de Delaunay con alpha = 1.0

Los mejores resultados se obtienen, tanto en la triangulación de Delaunay con *alpha* de 0.5, como con el *alpha* de valor 0, el que utiliza la función por defecto. La cantidad de puntos totales en cada experimento es mayor si se compara con los obtenidos en el modelo #1, lo que también permitió una definición más acertada de la reconstrucción 3D obtenida.

La configuración que obtuvo mejores resultados en todos los casos fue aquella en donde la separación entre las cámaras era de 6cm y la distancia cámara-base era de 48cm. Para el modelo #1 el mejor resultado se obtuvo al utilizar como *alpha* el valor 0, obteniéndose como resultado la reconstrucción que se observa en la Figura 78.

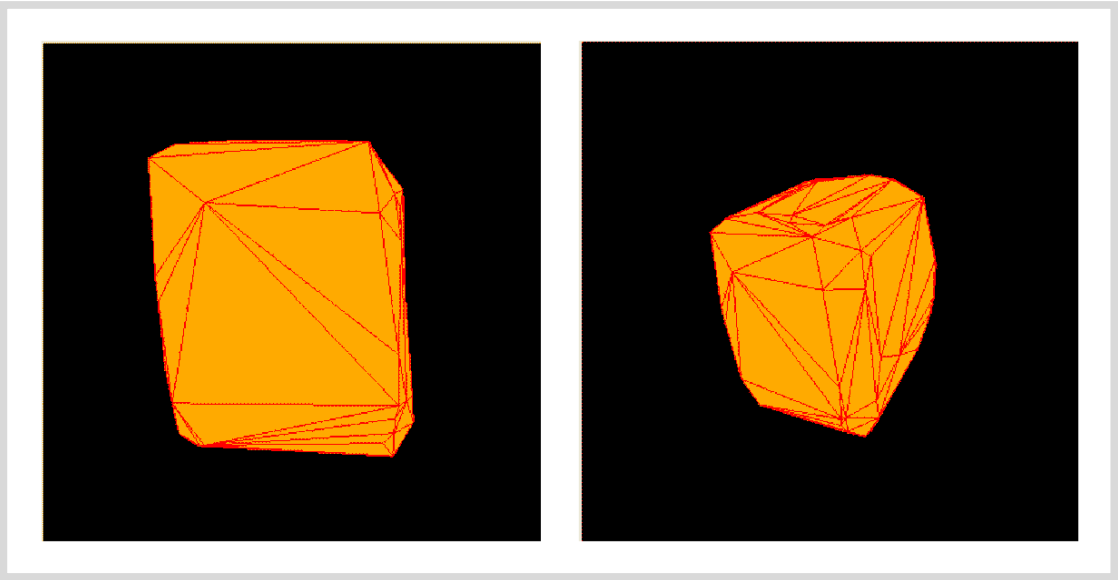


Figura 78 Reconstrucción 3D modelo #1

El modelo #2 obtuvo el mejor resultado al emplear como *alpha* los valores 0.75 y 1.0 y los comprendidos en ese rango, dando como resultado final la reconstrucción que se observa en la Figura 79 y Figura 80 respectivamente.

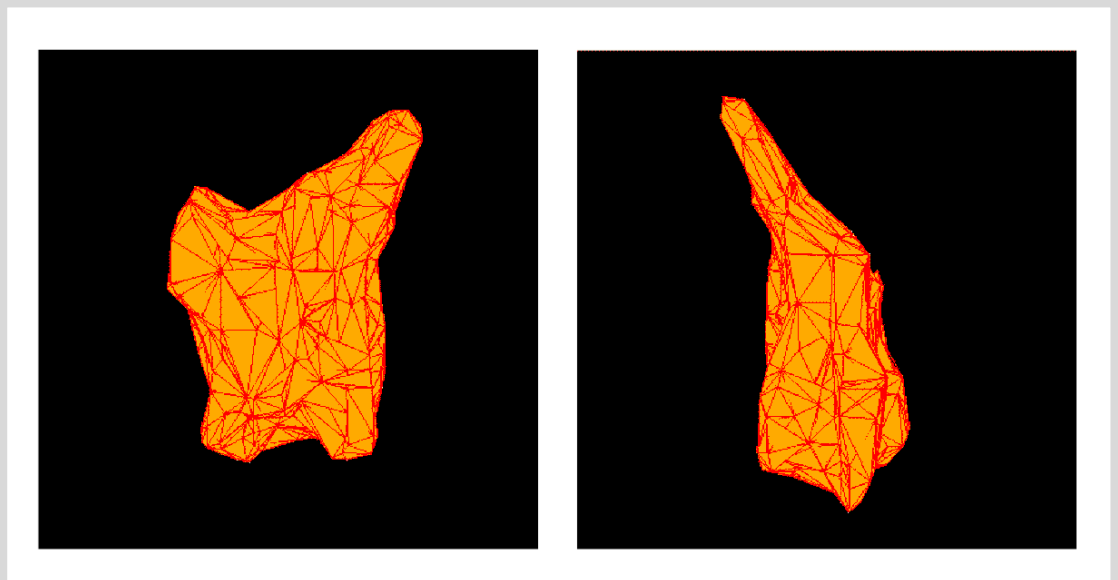


Figura 79 Reconstrucción 3D modelo #2 con alpha 0.75

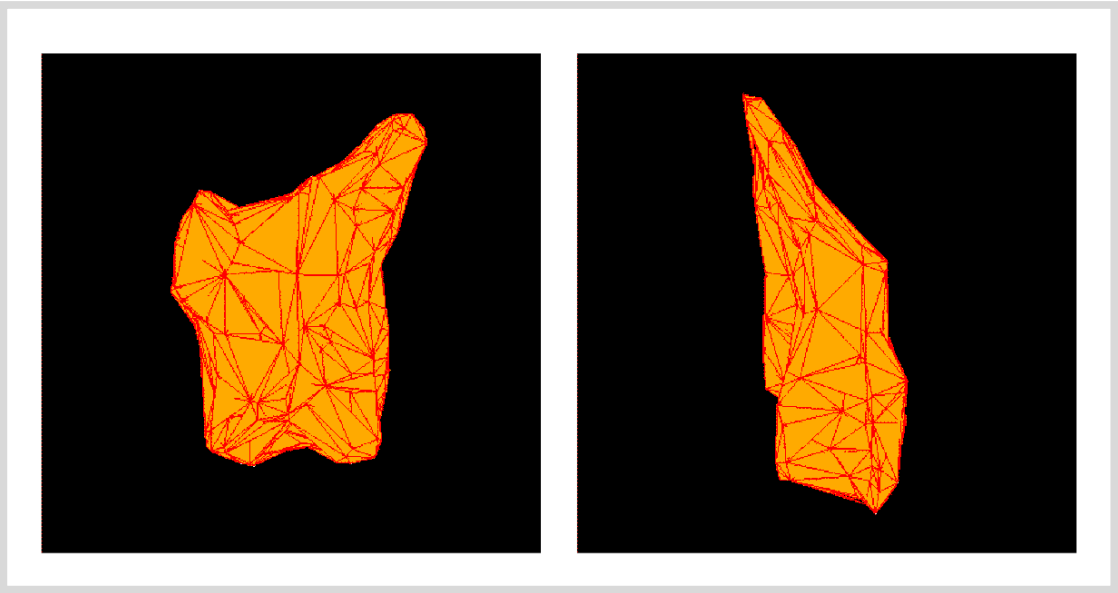


Figura 80 Reconstrucción 3D modelo #2 con alpha 1.0

Por último, en el caso del cubo con motivo floral, el mejor resultado se obtuvo al emplear como *alpha* los valores 0 y 0.5 como se observa en la Figura 81 y la Figura 82.

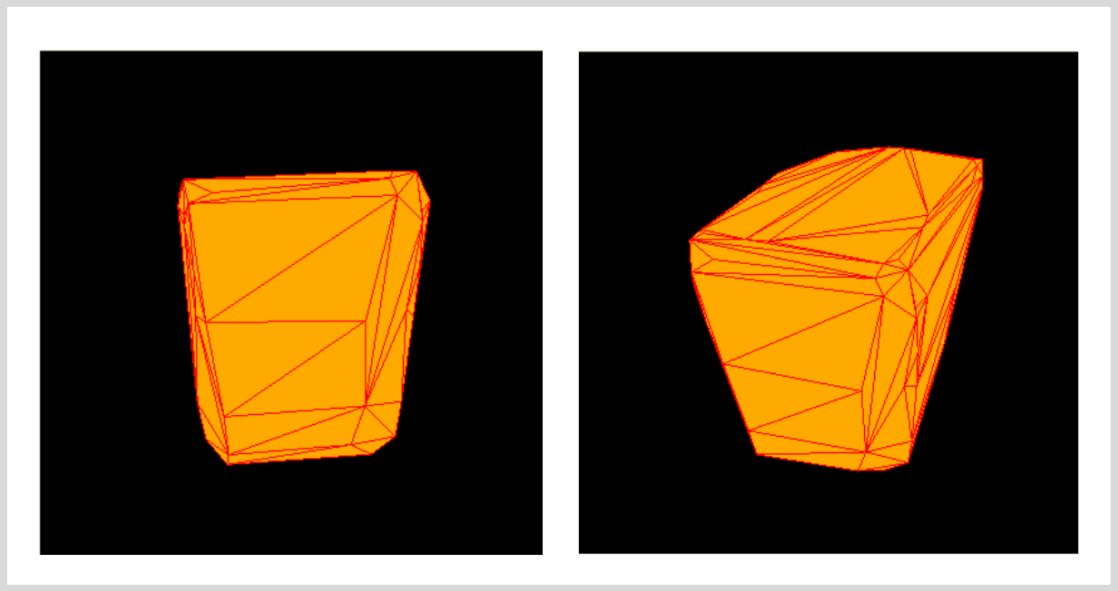


Figura 81 Reconstrucción 3D modelo #3 con alpha 0

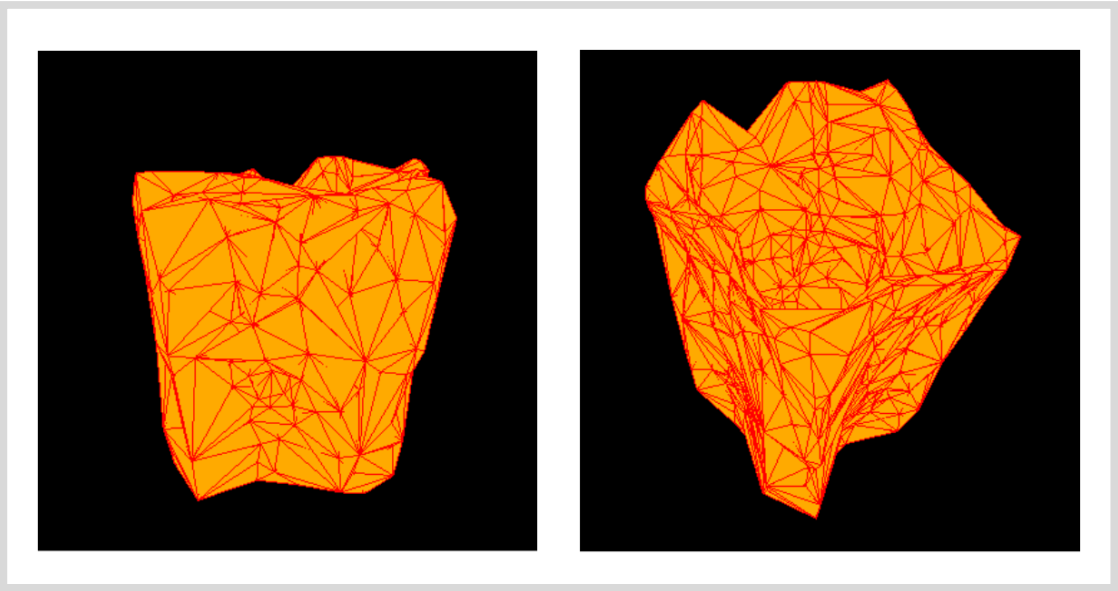


Figura 82 Reconstrucción 3D modelo #3 con alpha 0.5

Capítulo 6 Conclusiones

Durante el desarrollo del Trabajo Especial de Grado fue posible implementar un Escáner 3D que cumple con los objetivos establecidos y, al evaluar y analizar los resultados obtenidos a lo largo de la etapa de pruebas, se obtuvieron las siguientes conclusiones:

- Al emplear dos cámaras web para realizar el escaneado del objeto, separadas únicamente por una distancia horizontal, se está simulando el sistema de visión estereoscópico biológico.

- El módulo de calibración es capaz de deducir y calcular la matriz de proyección de cualquier dispositivo de captura logrando que la aplicación sea independiente del dispositivo a utilizar.

- Es importante que el dispositivo de captura provea imágenes de buena calidad y resolución para obtener mejores resultados.

  El resultado obtenido al escanear el objeto está estrechamente vinculado con el ambiente en el que se lleva a cabo el proceso y con la etapa de calibración del sistema.

  Si no contamos con un ambiente controlado, el ruido generado afectará drásticamente el desarrollo del proceso, pudiendo incluso impedir que se consiga un resultado final. Un ambiente uniforme permite que solo el objeto destaque y que, por tanto, los puntos característicos obtenidos sean correctos.

  Si la calibración no es exitosa, los puntos característicos, que son los que definen la estructura y forma del objeto, son erróneos por lo que tampoco se obtienen resultados favorables.

- Se pueden emplear objetos de color mate y textura lisa así como objetos con relieves y estructuras bien definidas. Debido a que la correspondencia entre las imágenes obtenidas por la cámara izquierda y la derecha se basan en la tonalidad del color, es importante que el ambiente presente una luz uniforme, que los objetos no sean

reflectores de luz, pues la calidad del resultado puede verse afectada, y que sean de color distinto al fondo empleado para que no se confunda con éste.

- Otro factor de importancia para obtener resultados exitosos al escanear un objeto es la distancia que debe existir entre la base del objeto y las cámaras y entre una cámara y otra. Mientras las cámaras se encuentren más cerca del objeto la diferencia que habrá entre la imagen de una cámara y otra será mayor, por lo que la superficie del objeto que tendrán en común será menor y con ello se presentarán problemas de oclusión y de cálculos erróneos en la generación de la coordenada de profundidad de los puntos. La combinación ideal se obtuvo al colocar las cámaras con una separación de 6cms. entre ellas y de 48cms. entre ellas y la base del objeto.

- La diferencia entre una imagen y otra es de 10º, por lo que se obtuvieron 36 imágenes izquierdas y 36 imágenes derechas para lograr una vista de 360º del objeto. A mayor distancia entre una imagen y otra menor superficie en común ocurriendo el mismo fenómeno de oclusión mencionado anteriormente. A menor distancia mayor redundancia en los puntos obtenidos.

Los objetivos planteados inicialmente fueron alcanzados, obteniéndose como resultado final un escáner 3D compuesto, básicamente, por dos cámaras web, y un software con el cual el usuario puede interactuar para manipular el modelado 3D del objeto en cuestión.

Capítulo 7 Trabajos Futuros

El trabajo realizado representa un prototipo donde es posible agregar nuevas características o mejorar las ya existentes, entre ellas:

- Emplear cámaras con mayor calidad de imagen para obtener mejores resultados.
- Utilizar un método de correspondencia de puntos que no dependa de la tonalidad del color o que incorpore otras características a la hora de realizarla. Por ejemplo, se puede realizar la correspondencia en base a las líneas epipolares.
- Incluir una fuente de luz que garantice una iluminación uniforme.
- Automatizar el proceso de captura de fotos del objeto.
- Soporte para objetos de mayor tamaño y dimensiones.
- Mejorar la interfaz del sistema para que sea más usable y amigable.
- Incorporar más utilidades en la aplicación destinada al usuario, de manera que sea mayor la interacción del mismo, por ejemplo la capacidad de incorporar texturas, manipulación de los puntos y mallado, manipulación de parámetros de la triangulación de Delaunay, entre otras.

El sistema fue desarrollado en módulos, donde la salida de uno es la entrada de otro, por lo que su funcionamiento tiene cierta independencia, lo que permite crear o mejorar un modulo ya existente sin interferir con el funcionamiento de otro. Esto también facilita el trabajo a la hora de agregar nuevas funcionalidades.

Glosario

CAD (*computer-aided design)* Diseño asistido por computador	Consiste en usar las computadoras para diseñar productos, permitiendo a los usuarios elaborar modelos tridimensionales "sólidos" con características físicas como peso, volumen y centro de gravedad. Estos modelos pueden rotarse y observarse desde cualquier ángulo. El computador puede evaluar el desempeño estructural de cualquier parte del modelo.
Cadencia	Es la frecuencia de fotogramas por segundo o fps a la que se pasan las imágenes.
Centroide	Es un punto que define el centro geométrico de un objeto
Contraste	Propiedad que permite diferenciar visualmente objetos plasmados en una imagen. Está relacionado con la diferencia de los niveles de gris claros y oscuros**.**
Digitalizar	Acción de convertir información analógica a una representación digital. Es convertir cualquier señal de entrada continua (analógica) en una serie de valores numéricos.
Dpi (*dots per inch*) Puntos por pulgada	Unidad de medida de resolución de una imagen.
Eje óptico	Es la línea imaginaria que recorre los centros de un sistema óptico y forma ángulo recto con el plano de la imagen.
Fotograma	Cada una de las fotografías que forman una película.
Frame	Es un fotograma o cuadro, una imagen particular dentro de una sucesión de imágenes que componen una animación.
Gradiente	El gradiente de un campo escalar es un campo vectorial que indica en cada punto del campo escalar la dirección de máximo incremento del mismo.
Máscara de Convolución	Matrices de tamaño kxk (donde $k = 2N + 1$ con $N \in \mathbf{N}$) que se utilizan al aplicar operaciones de convolución a una imagen.
.OBJ	Es un formato 3D creado por *Wavefront* para su producto

	Advanced Visualizer™. Estos archivos están basados en texto y soportan geometrías de formas libres y poligonales.
Píxel	Acrónimo de *picture element*. Menor unidad posible con la que se compone cualquier imagen digital en una computadora.
Tetraedro	Es un sólido constituido por cuatro planos o caras.

Bibliografía

1. Fluid Scanner. [En línea] http://fluidscanner.moviesandbox.net/.

2. **Dari, Enzo.** *Construcción de Mallas Delaunay en Tres Dimensiones.* Río de Janeiro, 1994.

3. **Monedero, Javier.** *Teoría de la Imagen Digital. Conceptos Básicos.* Barcelona, 2008.

4. **Rodríguez, Hugo.** *Imagen digital. Conceptos básicos.* Cataluña : Marcombo, 2005.

5. **Pajares, Gonzalo y M. de la Cruz, Jesús.** *Vision por Computador: Imágenes digitales y aplicaciones.* Madrid : RA-MA, 2001.

6. **Rice, Patrick.** *Master Guide for Professional Photographers.* Amherst Media, Inc, 2006.

7. **Ortega Cantero, Manuel, Bravo Rodríguez, José y Ruiz Hernández, Julián.** *Informática Industrial.* Univ de Castilla La Mancha, 1997.

8. **Viala, Carlos Ricolfe.** *Caracterización y optimización del proceso de calibrado de cámaras basado en plantilla bidimensional.* Universidad Politécnica de Valencia. Valencia, 2006.

9. **R., Tsai.** *A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-self TV camera lenses.* IEEE Journal of Robotics and Automation. 1997. págs. 323-344.

10. **Zhengyou, Zhang.** *A flexible new technique for camera calibration.* Redmond, 2008.

11. **Aldalur, B. y Santamaría, M.** *Realce de Imágenes: Filtrado Espacial.* Departamento de Ingeniería - Departamento de Matemática, Universidad Nacional del Sur. Bahía Blanca, 2002.

12. **Pertusa, José.** *Técnicas de análisis de imagen: aplicaciones en biología.* Universitat de València. Valencia, 2003.

13. **Rebaza, Jorge Valverde.** *Detección de bordes mediante el algoritmo de Canny.* Universidad Nacional de Trujillo. Trujillo.

14. **Ballesta, Mónica, y otros.** *Evaluación de Destectores de Puntos de Interés para SLAM Visual.* Departamento de Ingeniería de Sistemas Industriales, Universidad Miguel Hernández. Alicante.

15. **Harris, Chris y Stephens, Mike.** *A combined corner and edge detector.* Plessey Research Roke Manor. 1988. págs. 147-151.

16. **García Martín, Jorge.** *Integración de objetos sintéticos en Imágenes y Vídeo real.* Universidad Politécnica de Madrid. Madrid, 2007.

17. **Bouguet, Jean-Yves.** *Pyramidal Implementation of the Lucas Kanade Feature Tracker.* Intel Corporation. 2000.

18. **Recasens Urzi, Bruno A.** *El Cálculo de la Correspondencia Densa y sus aplicaciones al reconocimiento facial.* Universidad de Murcia. Murcia, 2007.

19. **Sánchez Pérez, Javier.** *Visión Tridimensional.* Facultad de Informática, Universidad de las Palmas de Gran Canaria.

20. **H. Press, William.** *Numerical recipes: the art of scientific computing.* Cambridge University Press, 2001.

21. **Jiménez Moscoso, José Alfredo.** *Algebra lineal II con aplicaciones en estadística.* Univ. Nacional de Colombia. Bogota, 2004.

22. **Valencia Gutiérrez, Jairo Iván.** *Registro de Imágenes de Rango para la Construcción de Modelos Faciales.* Manizales, 2004.

23. **Cuartas Morales, Ernesto.** *Optimización de la Representación con Superficies NURBS de imágenes de rango mediante el algoritmo de Levenberg-Marquardt.* Facultad de Ingeniería y Arquitectura, Universidad Nacional de Colombia. Manizales, 2004.

24. **Hitschfeld, Nancy.** *Mallas geometricas y aplicaciones.* Departamento Ciencias de la Computacion, Universidad de Chile. Santiago de Chile.

25. **D.F.Watson.** *Computing the n-dimensional Delaunay tessellation with applications to Voronoi.* 1981.

26. **Cerrolaza, Miguel y Flórez-López, Julio.** *Modelos Matemáticos en Ingeniería moderna.* CDCH UCV, 2000.

27. **Vicedo, Jordi y Linares, Jordi.** *Escaneado de Objetos Tridimensionales en el ITI.* Valencia, 2008.

28. **Hurbain, Philippe.** Philo's Home Page. [En línea] http://philohome.com/scan3d/scan3d.htm.

29. **Pan, Qi, Reitmayr, Gerhard y Drummond, Tom.** *ProForma: Probabilistic Feature-based On-line Rapid Model Acquisition.* Cambridge.

30. 3DSOM Pro Features. [En línea] http://www.3dsom.com/features/process.html.

31. **Winkelbach, Simon.** David 3d Scanner. [En línea] http://www.david-laserscanner.com/.

32. **Gil, P., y otros.** *Reconstrucción Tridimensional de Objetos con Técnicas de Visión y Luz Estructurada.* Alicante.

33. **Lorensen, William y Cline, Harvey.** *Marching Cubes: A high resolution 3D surface construction algorithm.* 1987.

34. OpenGL. The Industry Standard for High Performance Graphics. [En línea] http://www.opengl.org/.

35. QT. A cross-platform application and UI framework. [En línea] http://qt.nokia.com/.

36. OpenCV. [En línea] http://opencv.willowgarage.com/wiki/.

37. CGAL. Computational Geometry Algorithms Library. [En línea] http://www.cgal.org/.

38. VTK. The Visualization Toolkit. [En línea] http://www.vtk.org/.

39. **Vera, Pablo y Salas, Joaquin.** *Cálculo de la Incertidumbre en la Medición de los Parámetros de un Péndulo de Foucault.* Querétaro, 2008.

40. **Guzmán, Carlos, González, Erick y Chávez, Patricia.** *Análisis e Implementación de algoritmos para el mejoramiento de Imágenes.* Facultad de Ingeniería Eléctrica y Computación, Escuela Superior Politécnica del Litoral. Guayaquil.

41. **Leal Narváez, Esmeide, Leal Narváez, Nallig y William Branch, John.** *Segmentación de superficies de forma libre a partir de nubes de puntos ruidosos.* Universidad Simón Bolívar - Universida Nacional de Colombia. Barranquilla.