Sistema para creación de modelos 3D con el uso de imágenes monoculares

Fabián Alejandro Ramírez Báez
Julián Camilo Ramírez Gamboa

Director
Ing. Julián Armando Quiroga, PhD.

Pontificia Universidad Javeriana
Facultad de Ingeniería
Departamento de Electrónica

1 de junio de 2018

Tabla de contenido

Tabla de figuras:

1. Introducción

La visión es la percepción de realidades físicas a través de la vista, por lo tanto, es la capacidad de reconocer y discriminar los diferentes objetos e interacciones que son perceptibles por medio del sentido de la vista, debido a esto, es una de las principales herramientas que tienen diferentes organismos para interactuar con su entorno y comprenderlo.

Dentro del tema de la visión, se encuentra un área que ha sido objeto de estudio por parte de diferentes ingenieros y científicos alrededor del mundo, que es la visión artificial, la cual es el estudio de los procesos que los organismos generalmente realizan con su visión, con el fin de entenderlos y construir máquinas con ese tipo de capacidades, de tal modo que tengan la posibilidad de interactuar con su entorno y tomar decisiones a partir de la información captada por medio de cámaras digitales. A partir de la década de los 60's, se inició el estudio de la visión artificial como un área de la computación y hoy en día sigue siendo un campo de investigación con gran cabida en el desarrollo de la robótica, sentando las bases de diferentes formas de procesamiento de imágenes, que han permitido avanzar en temas como la fotografía, la seguridad cibernética y en el mundo real, entre otros.

Con la creación de las imágenes digitales se abrió un nuevo mundo lleno de posibilidades para el futuro de la robótica y el procesamiento de estas. Al tener la posibilidad de digitalizar una representación visual del mundo real por medio del uso de sensores integrados en las cámaras digitales, que permiten transformar la imagen en el mundo real en una serie de señales eléctricas que permitirán la creación de la imagen digital.

Al tener la posibilidad de leer una imagen en un computador como una matriz se puede empezar a usar la imagen como parte de un gran número de diferentes ecuaciones con diferentes funciones, que le permitirán al computador entender los elementos presentados dentro de la imagen, y de este modo, procesarlas a voluntad según el objetivo que se tenga.

Dentro del mundo de la visión artificial se encuentra también el reconocimiento facial, en el cual las imágenes del rostro son frecuentemente utilizadas en los sistemas de seguridad biométrica [5], sin embargo, la información 2D capturada a través de cámaras monoculares resulta insuficiente en algunas ocasiones para discriminar entre individuos de manera confiable y bajo diferentes condiciones de adquisición. Por tal motivo es necesario adicionar información complementaria en la tarea de reconocimiento. El uso de la estructura tridimensional del rostro, es decir, de su geometría, se presenta como una alternativa para mejorar la confiabilidad de este tipo de sistemas biométricos. Este trabajo de grado pretende obtener y comparar modelos tridimensionales del rostro a partir de cámaras monoculares.

A medida que el estudio de la visión artificial avanza, se han ido presentando nuevas aplicaciones para la misma, hoy por hoy el uso de las imágenes captadas de un mismo objeto desde diferentes perspectivas para crear los modelos tridimensionales de estos es un área de la visión artificial que ha dado paso a muchas investigaciones que buscan posibilitar la digitalización de muchos objetos del mundo real para obtener información adicional a la que se puede obtener con las imágenes digitales tradicionales, pues no cuentan con información precisa de la profundidad a la que se encuentran diferentes rasgos de un mismo objeto sin el uso de sensores específicos para la determinación de esta información. Sin embargo, al hacer uso de sensores de profundidad en las cámaras se dan muchas limitaciones que hacen que la información captada pueda no ser precisa dependiendo del objeto que se desea recrear. Un claro ejemplo es el de los sensores infrarrojos (utilizados para obtener la profundidad de los diferentes puntos en los pixeles de la imagen) que son profundamente afectado por las propiedades de reflexión del material sobre el cual se desea hacer la medida si el material tiende a absorber parte de la onda del infrarrojo, el sensor no podrá brindar información acertada sobre la profundidad de las partes del objeto.

En la actualidad los gobiernos se encuentran invirtiendo en sistemas de seguridad biométrica a partir de información tridimensional, ya que estos resultan más seguros que los sistemas actuales. Es por esto que el reconocimiento facial preciso se hace tan necesario, ya que este es usado como medida de seguridad biométrica en la actualidad, como es el caso del sistema de reconocimiento facial implementado por Apple en el nuevo iPhone X.

Adicionalmente en la industria de los videojuegos, a medida que se crean juegos con mejores gráficas, que cada vez son más realistas y parecidas a la realidad, se utiliza el modelado 3D de rostros para facilitar el proceso de producción del juego.

En el proyecto explicado en este documento se pretende desarrollar un sistema de reconstrucción tridimensional de imágenes faciales a partir de imágenes monoculares. Esto con el fin de orientar diferentes métodos preexistentes de reconstrucción de modelos 3D a la creación automática de modelos fidedignos a la información del rostro de una persona, y así facilitar y automatizar dicho proceso.

En los documentos consultados se presentan distintas maneras de realizar el modelado 3D del rostro, en [2] presentan un modelo preciso para el reconocimiento desde el uso del sistema RGB-D con cambios en la pose de la cara, por lo cual se puede decir que su método es una mejora a los métodos actuales ya que muchos de los métodos actuales no funcionan adecuadamente sin la vista frontal del rostro.

El Kinect es una herramienta utilizada en la toma de imágenes para el modelamiento del rostro, en [3] y [4] se hace uso del Kinect, que cuenta con un sensor de profundidad y pese a que su aplicación es directa en videojuegos esta herramienta ha resultado bastante útil en el proceso de toma de imágenes para proyectos con objetivos distintos al del desarrollo de videojuegos.

En [5] se tratan mejoras para la reconstrucción, al igual que en [2], basadas en distintas poses de la cara, sin embargo, presenta fallas en el sentido de que se asume el rostro simétrico por lo que su información no es precisa, adicionalmente en [6] se presentan las diferencias entre el costo de las imágenes en 2D y las imágenes 3D en la reconstrucción del rostro.

Además, se hace énfasis en [1] donde se plantea la importancia del modelamiento en los sistemas de seguridad, buscamos que este trabajo de grado proponga la reconstrucción del rostro como modelo 3D desde el método de varias imágenes tomadas por una cámara desde distintas perspectivas con el fin de evitar la pérdida de información, para que así el sistema sea capaz de identificar el modelo del rostro frente a otros modelos existentes.

En [9] se presenta un método de reconstrucción de modelos a partir de superficies con poco soporte, creado por investigadores del instituto técnico checo en Praga, el método utiliza tetrahedrización con el fin de robustecer la reconstrucción de superficies a partir de nubes de puntos con poco soporte, esto es un avance significativo en el área de reconstrucción 3D dado que en métodos desarrollados previamente al método explicado en el articulo si no se cuenta con la suficiente información en nubes de puntos, o si se presenta mucho ruido en las mismas, se obtienen modelos de poca fidelidad para el usuario, lo que presenta un serio problema al momento de automatizar la creación de modelos, dado que, con diferentes nubes de puntos del mismo objeto no se presentaría constancia en los modelos que se desean realizar.

En [10] se presenta un método de reconstrucción de superficies a partir del uso de la calibración de la cámara con la que se toman las diferentes vistas de un objeto o estructura, como parámetro de la reconstrucción. Esto hace que el refinamiento de las superficies obtenidas sea mas robusto y confiable que en métodos previamente desarrollados, puesto que, bajo condiciones no controladas de toma de imágenes para la reconstrucción del modelo de un objeto se encontraban varios problemas de consistencia en la visibilidad de las superficies, que no permitían precisión entre modelos de estructuras de gran tamaño, para las cuales controlar el proceso de toma de imágenes era un proceso que consumía un tiempo significativo, y en el caso de muchas estructuras, simplemente no se podía realizar de manera acertada.

El método de texturizado de modelos 3D presentado en [11] utiliza información de color encontrada en las imágenes para texturizar los modelos obtenidos a partir de dichas imágenes. El método establecido texturiza cada triangulo del modelo con una imagen ortogonal a este, acercada, y con foco para dicho triangulo en especifico para darle un color base al modelo, posteriormente se ajustan los colores para prevenir disparidades en el color debido a la exposición a diferentes iluminaciones en la superficie, y el posible ruido que se pueda presentar. El siguiente paso a utilizar es el chequeo de foto-consistencia, que permite evitar que objetos no deseados, que obstruyen la vista de algunas imágenes no sean parte de la textura de la superficie del modelo, haciendo que sea un método ideal para darle textura a modelos de estructuras de gran tamaño, en

las cuales durante el proceso de toma de imágenes se pueden presentar algunas obstrucciones que no pertenecen a la estructura, este método hace que se puedan eliminar dichas obstrucciones y no sean tenidas en cuenta durante el proceso de texturizado del modelo.

En el presente documento se tratará un proyecto de visión artificial basado en la creación de modelos tridimensionales del rostro de diferentes personas a partir del uso de cámaras digitales convencionales (como las que se encuentran en los smartphones de hoy en día), que, debido a la gran gama de aplicaciones que tienen los métodos de modelado 3D a partir de imágenes 2D desde diferentes perspectivas de un objeto, no existen programas destinados específicamente a la reconstrucción del rostro en un modelo 3D acertado para su posterior procesamiento.

El documento consta de 7 partes que describen el proceso realizado para la realización del proyecto, y los resultados obtenidos de acuerdo con cada procedimiento.

Las partes presentes en este documento son:

- Introducción: se presenta el contexto en el que se dio el proyecto realizado y se dan a conocer al lector los antecedentes y las justificaciones para la realización de este proyecto.
- Marco teórico: Se presentan los conceptos y la teoría clave para entender el proyecto y su desarrollo.
- Objetivo del proyecto: se enumeran y explican los diferentes objetivos que tiene el presente proyecto.
- Desarrollo: se describe el proceso dado para la realización y finalización del proyecto.
- Protocolo de pruebas: se explican y justifican los procedimientos realizados en el transcurso del proyecto.
- Análisis de resultados: se analizan los resultados obtenidos en el desarrollo del proyecto.
- Conclusiones y recomendaciones: se presentan las diferentes conclusiones obtenidas a partir del desarrollo del proyecto y se plantean los posible alcances, mejoras y aplicaciones que se pueden dar a partir del proyecto realizado.

2. Marco Teórico

2.1. Imágenes

Las imágenes se forman a partir del uso de sensores que se encuentran en la cámara utilizada, los cuales registran la información de luz captada por los lentes. La imagen obtenida puede verse como “una función bidimensional, donde el valor de la función corresponde a la intensidad o brillantez en cada punto de la imagen (imágenes monocromáticas, conocidas como imágenes en “blanco y negro”)” (citado de [12] pagina 3).

En el procesamiento de imágenes digitales, estas se pueden recorrer con el uso de un sistema coordenado bidimensional (x, y), en el cual el origen (punto (0,0) o (1,1)) se encuentra en el primer pixel de la esquina superior izquierda de la imagen.

En una imagen en blanco y negro, o escala de grises, cada una de las posiciones de la matriz de la imagen tienen un valor especifico dentro de un rango determinado por las propiedades de la imagen (numero de bits) según la información capturada por la cámara en ese punto en especifico.
En una imagen digital tanto el valor de la intensidad, como el valor de ubicación espacial son discretizados con respecto a la función continúa censada a partir del escenario real. Este muestreo es representado por la función:

$$f_s(x,y) = \int\int_{-\infty}^{\infty} f(x,y)\delta,(x-x_0,y-y_0),dx,dy$$

2.1.1. Proyección de la imagen

2.1.1.1. Sistemas coordenados homogéneos y euclidianos

Trabajando desde un sistema coordenado homogéneo, un punto en un espacio N-dimensional es expresado por un vector de N+1 elementos, suponiendo el elemento N+1≠0 una coordenada homogénea puede ser relacionada con su equivalente euclidiano dividiendo los primeros N elementos por su elemento N+1-esimo. Por ejemplo, si se tiene un punto:

$$P = [X \quad Y \quad Z \quad W]$$

en un sistema coordenado homogéneo, su equivalente en un sistema coordenado euclidiano es:

$$P' = \left[\frac{X}{W} \quad \frac{Y}{W} \quad \frac{Z}{W}\right]^T$$

2.1.1.2. Modelo de cámara "pinhole"

El modelo de cámara pinhole es el más común encontrado en diferentes textos al ser uno de los modelos que mejor describe el comportamiento de las cámaras reales. Este consiste en la relación entre un punto 3D (mundo real) y su correspondiente en la imagen tomada(2D). El modelo consiste en 3 componentes que son:

- La transformación que relaciona los puntos 3D del mundo real con los componentes 2D de la imagen como:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} \sim \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

  Donde R es una matriz de 3*3 que representa la orientación de la cámara, T es un vector de tres posiciones que representa la traslación de la cámara, el punto a la izquierda de la ecuación representa el punto $\tilde{X}_c$ (en el sistema coordenado de la cámara) y el punto a la derecha de la ecuación es $\tilde{X}$(en el sistema de coordenadas del mundo real).

- La transformación 3D a 2D que relaciona los puntos 3D, que relaciona los puntos $\tilde{X}_c$ con los puntos 2D $\tilde{x} \sim [x \quad y \quad 1]^T$ (en el plano de la imagen). Usando triángulos similares (ver figura 1) se obtiene:

$$x = f\frac{X_C}{Z_c}$$
$$y = f\frac{Y_c}{Z_c}$$

  f es el foco del lente. Al cambiar el valor de f se cambia la escala de la imagen, por lo tanto, al definir f=1, la relación entre la escala homogénea y la escala euclidiana es:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}$$

- Finalmente se encuentra la transformación 2D a 2D, que relaciona los puntos $\tilde{x}$ en el plano de la imagen con el sistema coordenado de pixeles $\tilde{u} \sim [u \quad v \quad 1]^T$. Esto se escribe de la siguiente manera:

$$\tilde{u} = K\tilde{x}$$

donde K es la matriz de triangulación superior de la cámara con la forma:

$$K = \begin{bmatrix} \alpha_u & s & u_0 \\ 0 & \alpha_u & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

a_u y $a_v$son factores de escala, s es oblicua, y $U_0 = [u_0 \quad v_0]^T$ es el punto principal. Estos son los parámetros intrínsecos de la cámara.

En coordenadas homogéneas, un punto 3D $\tilde{X}$ esta relacionado con su posición en el plano coordenado de pixeles como $\tilde{u}$, a partir de la siguiente relación:

$$\tilde{u} \sim P\tilde{X}$$

donde $P \sim K[R\ T]$ es una matriz de proyección de 3*4.

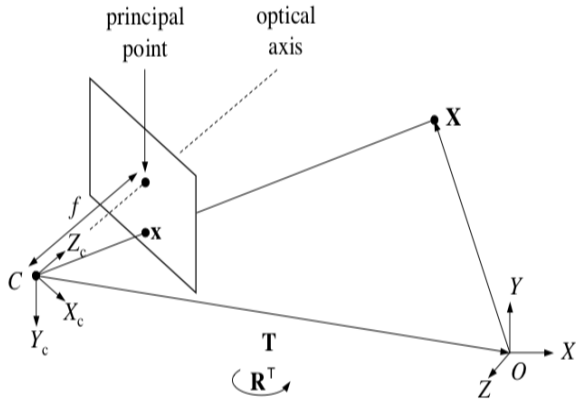

Figura 1: proyección de un punto 3D X en el plano coordenado de la imagen tomada.(imagen tomada de [12] pagina 4)

La figura 1 ilustra un punto X en un sistema coordenado 3D del mundo real, y su proyección en el plano de la imagen.

2.1.1.3. Puntos de interés:

Los puntos de interés son puntos en las imágenes que se califican como puntos que difieren de aquellos que se encuentran a su alrededor, lo cual facilita su detección y permite el procesamiento de imágenes a partir de estos. Generalmente los puntos de interés en las imágenes pueden ser vistos por el ser humano como puntos en los que hay cambios abruptos en textura o iluminación como bordes y esquinas en una estructura. Este tipo de puntos difieren de puntos en los que hay una textura constante, o que se encuentran en regiones "planas" de la imagen, que no permiten discriminar entre puntos dentro de una región de la imagen escogida (ventana). Para detectar los puntos de interés en las imágenes existen diferentes métodos, tales como:

- Hessiano y Harris.
- Laplaciano.
- EBR e IBR.
- MSER.
- SIFT (*Scale Invariant Feature Transform*).
- Muchos otros.

Realizando un estudio de los métodos utilizados en este proyecto, solo se hará énfasis en el método SIFT para encontrar y procesar los puntos de interés, pues es el método usado en varios de los diferentes algoritmos de SFM, tales como los encontrados en [14] y [19].
El método SIFT es un método utilizado para extraer puntos de interés en una imagen y computar sus descriptores. Este método es robusto ante cambios de escala de un objeto en una imagen, lo que permite que

el cambio del tamaño de un objeto dentro de la imagen no sea un problema para la detección de los puntos de interés en el mismo. Consiste en 4 pasos:

- ✓ *Scale-space extrema detection:* cuando se usa el ventaneo para encontrar los puntos de interés de una imagen, el mismo tamaño de ventana no puede ser usado para encontrar los bordes y esquinas de algunas imágenes, dado que, a pesar de que en algunos casos el tamaño de ventana es el adecuado para encerrar el borde en una esquina, en otros casos el tamaño de la ventana no es el óptimo para encontrar puntos de interés, por ejemplo, si en una misma imagen se tienen bordes gruesos y bordes delgados, un solo tamaño de ventana que tenga el tamaño especifico para encerrar un borde delgado, no tendrá el tamaño necesario para encontrar bordes de mayor grosor en la misma imagen.

  Debido a lo anterior, para evadir errores de este tipo, es necesario variar el tamaño de las ventanas usadas en el ventaneo, esto se hace filtrando la escala-espacio. Para esto es necesario encontrar el Laplaciano de la Gaussiana (LoG) de la imagen para diferentes valores de σ (valor de escala). El LoG funciona como un detector de blobs (binary large objects), que detecta blobs en diferentes escalas, usando σ como factor de escalamiento. Para el caso de un σ pequeño el tamaño de la ventana usada será pequeño, en cambio, para un σ grande el tamaño de la ventana será mayor, proporcional al cambio de σ.

  Como resultante de esta parte del proceso se obtiene una lista de valores (x, y, σ) que indican la posición (x, y) del pixel en la imagen, y el factor de escala σ en el que se encontró un potencial punto de interés.

  Una aproximación del LoG muy utilizada en el algoritmo SIFT es la diferencia de gaussianas(DoG), que requiere un menor costo de procesamiento, agilizando el proceso.

  El DoG realiza dos filtrados gaussianos de la parte de la imagen con un σ diferente para cada filtrado y posteriormente los sustrae para obtener una imagen nueva, resultante del procesamiento de la imagen original. Este proceso se realiza para diferentes octavos de la imagen dentro de la pirámide gaussiana. Una ilustración de este proceso puede ser vista en la figura 2.

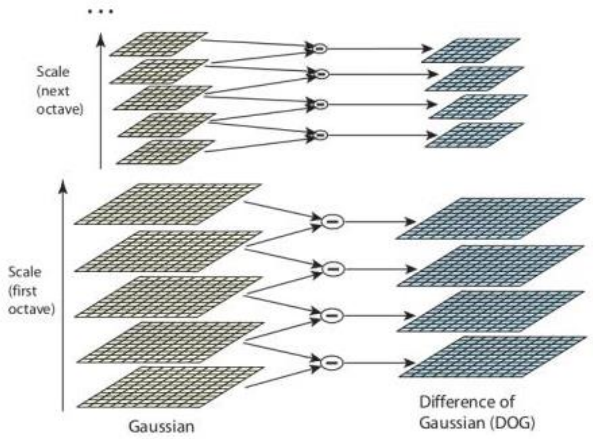

Figura 2: Ilustración del proceso DoG.(imagen obtenida [13])

  Posteriormente se buscan extremos locales, comparando cada punto con sus vecinos inmediatos en busca de un extremo local, cuando se encuentra un extremo en la imagen, se compara con la ventana centrada en el mismo punto con la escala anterior y la escala siguiente, como se puede observar en la figura 3. Si se encuentra únicamente en la escala actual el extremo, es un potencial punto de interés.

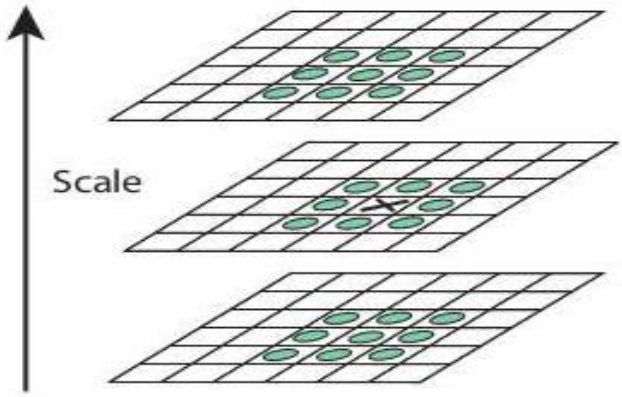

Figura 3: comparación entre ventanas de diferentes escalas.(imagen tomada de[13])

- ✓ Localización de los puntos de interés: Con el uso de la expansión en series de Taylor en los resultados obtenidos en la parte anterior del proceso se obtiene una locación mas asertiva de los puntos de interés potenciales, si la intensidad en estos puntos es menor a un threshold (llamado threshold de contraste en OpenCV) definido, este es rechazado. Ya que el DoG tiene una respuesta mayor en los bordes, los bordes son removidos (la remoción de bordes esta explicada en [13]). En resumen se eliminan los elementos con bajo contraste y los bordes, con el fin de que todos los demás puntos de interés potenciales obtenidos del procedimiento anterior sean los puntos de interés resultantes.
- ✓ Asignación de la orientación: dependiendo de la escala del punto de interés σ se toma un vecino del punto de interés y se calculan la magnitud del gradiente y la dirección de la región escogida. Se obtienen un histograma de 36 bins (perspectivas binoculares) cubriendo 360 grados.

Para una explicación más profunda del método SIFT ver [13].

2.2. Visión estereoscópica

Varios organismos, tales como los seres humanos, tienen percepción de profundidad por medio del uso del sentido de la vista, esto es gracias a la interpretación que estos organismos pueden realizar a partir de la información captada por los ojos, que le ofrecen al cerebro dos vistas del mismo objetos diferenciadas por la posición en la que se encuentra cada uno de los ojos del organismo, de modo que el cerebro del mismo realiza un proceso inconsciente de triangulación entre las diferentes vistas de su entorno. Teniendo en cuenta este principio es posible computar la información de profundidad de objetos captados por cámaras ubicadas en diferentes posiciones por medio de un método llamado visión estereoscópica, el cual esta divido en 3 partes.

2.2.1. Estimación de la matriz fundamental

Usando la proyección de un punto 3D (mundo real) en el plano de la imagen, la proyección de este mismo punto en una segunda imagen (tomada desde una vista diferente del objeto) esta restringida únicamente a su línea epipolar, esta es la proyección de un rayo que va desde el centro óptico de la imagen base, pasando por un punto de interés $\tilde{x}$ en el plano de la misma imagen y coincidiendo con la proyección del punto del plano en el mundo real, el punto X (ver figura 4). Todas las líneas epipolares de la imagen base tienen en común la proyección en el centro óptico de la segunda imagen, este punto es llamado epipolo.

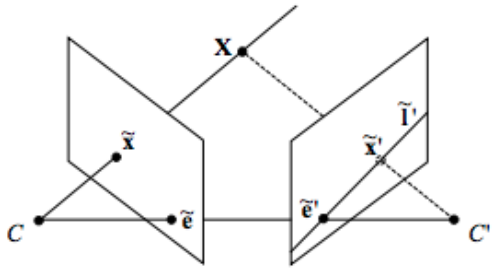

Figura 4: geometría epipolar de dos cámaras. (Tomada de: [12] pagina 10)

Como se puede observar en la figura 4 la línea epipolar saliente de C tiene incidencia en el punto $\tilde{x}$ en el plano de la imagen y tienen una proyección en la segunda imagen, que corta con $\tilde{x}'$ y el epipolo de la segunda imagen, asimismo al trazar una recta entre el centro óptico de las dos imágenes C y C' se puede observar que esta pasa por los epipolos de las dos imágenes. Además, es posible encontrar en la figura 4 que la proyección del punto X en el plano de la segunda imagen ($\tilde{x}'$) esta restringida a la línea epipolar I'.

La restricción epipolar descrita anteriormente puede ser formulada usando la matriz esencial E, que correlaciona los puntos en la vista base con sus respectivos pares en el plano de la segunda imagen. Dado un punto euclidiano X' en el sistema coordenado de C', su posición X en el sistema coordenado de C esta dado por:

$$X = RX' + T$$

Donde R es la matriz de rotación (descrita anteriormente en esta sección) y T es el vector de traslación. Multiplicando ambos lados de la ecuación anterior por $X^T[T]_X$ se obtiene:

$$X^T[T]_X RX' = X^T EX' = 0$$

Por lo tanto, se puede suponer que:

$$E \sim [T]_X R$$

Para profundizar sobre los pasos intermedios para comprobar la relación descrita en la ecuación anterior, ver [12]

2.2.2. Recuperación de matrices de proyección

Teniendo en cuenta diferentes consideraciones numéricas, descritas en [12], las matrices de proyección están directamente relacionadas con la rotación y traslación, alineando la referencia del sistema coordenado con la cámara base se obtiene:

$$P = K[I|0]$$
$$P' = K'[R|T]$$

Donde P y P' son las matrices de proyección de las dos cámaras, asimismo K y K' son las matrices de calibración de las cámaras.

2.2.3. Triangulación

Habiendo obtenido las matrices de proyección en el paso anterior, se pueden computar los puntos 3D en la reconstrucción por medio del uso de la triangulación, dado que es un método muy reconocido con diferentes aplicaciones en diversos campos de investigación no se hará énfasis en este método, sin embargo, si el lector desea puede profundizar en el tema en [19].

2.3. Structure from motion (SFM)

Teniendo un numero N de vistas de una estructura, es posible crear un modelo 3D en nubes de puntos de una estructura en particular a partir de las mismas. Para el proceso de SFM se conocen dos algoritmos de reconstrucción diferentes, que son:

- Método secuencial: donde se plantea una imagen del conjunto escogido como una imagen base, y la rotación y traslación de las demás imágenes alrededor del objeto se da con la imagen base como referencia.
- Método global: Los métodos globales son métodos que recuperan las vistas y las rotaciones de las cámaras alrededor de las estrcturas, todas al mismo tiemp, esto permite que no se cree una acumulación de errores de una comparación a otra, como es el caso de los métodos secuenciales. Generalmente los métodos de SFM global se dan en dos pasos:
  - ✓ Computo de la rotación global de cada vista.
  - ✓ Computo de las traslaciones de la cámara junto con la estructura.

  Este acercamiento tiene en cuenta la grafica epipolar completa, que tiene en cuenta los nodos de esta suponiéndolos como diferentes vistas y los bordes vinculan las vistas con suficientes puntos emparejados. Para tener una mayor claridad con respecto al proceso especifico de SFM global, ver [15], donde se describe matemáticamente la estimación de la rotación global, y la estimación de translaciones relativas a partir de tensores trifocales.

2.4. Bundle adjustment

El bundle adjustment es un método de minimización de errores de proyección, que utiliza las correspondencias y poses obtenidas del proceso de SFM. En principio se supone que el error es el cuadrado de la norma de la diferencia entre la localización del punto observado y su proyección en el punto 3D correspondiente en el plano de la cámara.
Siendo x un vector de parámetros y $f(x) = [f_1(x) \quad \cdots \quad f_k(x)]$ un vector de errores residuales para a reconstrucción 3D. Teniendo la ecuación:

$$x^* = \arg\min_x \sum_{i=1}^{k} \|f_i(x)\|^2$$

Se usa el algoritmo de Levenberg-Marquardt (LM) para resolver el problema de mínimos cuadrados, que consta de resolver una serie de aproximaciones lineales regularizadas para el problema no lineal original.

2.5. Iterative closest point

ICP es un algoritmo utilizado para alinear nubes de puntos y minimizar las diferencias entre ambas. El algoritmo esta descrito detalladamente en [17].

3. Objetivos

Para el proyecto se definieron un total de 5 objetivos diferentes, un objetivo general y 4 objetivos específicos.

3.1. Objetivo general

Desarrollar un sistema de reconstrucción tridimensional de imágenes faciales a partir de imágenes monoculares.

3.2. Objetivos específicos

Los objetivos específicos planteados para este proyecto son:

- Implementar un algoritmo de reconstrucción tridimensional a partir de varias imágenes capturadas desde distintas poses.
- Implementar un algoritmo de comparación entre dos modelos tridimensionales del rostro.
- Implementar una interfaz gráfica que permita seleccionar las imágenes y configurar los parámetros de generación y visualización del modelo tridimensional.
- Diseñar e implementar un protocolo de pruebas con 10 usuarios para evaluar la consistencia y capacidad de discriminación de la reconstrucción tridimensional.

4. Desarrollo del proyecto

Para explicar cada una de las partes del proyecto y el proceso de desarrollo del proyecto se hace uso del diagrama de flujo que se encuentra en el anexo 1 del documento, en el cual se describe bloque por bloque el proceso de creación de los modelos dentro del sistema realizado durante el transcurso de este proyecto.

4.1. Toma de imágenes

La toma de fotografías de cada uno de los candidatos fue realizada bajo las siguientes condiciones:

- El candidato no debe llevar puestas gafas al momento de la toma de fotografías, esto debido a que en diferentes escenarios se presentaban problemas con el algoritmo de reconocimiento facial utilizado para recortar automáticamente y no encontraba el rostro del candidato dentro de las fotografías, además de ser un obstáculo para la creación del modelo.
- El candidato debía tener barba corta, o afeitada. Esta condición era necesaria con el fin de crear un modelo sin agujeros en el rostro, puesto que, en las imágenes de personas con barba no se detectaba textura en los sitios donde había bello facial y, por tanto, el modelo resultante no era acertado en cuanto al relleno que se realizaba en los huecos que se daban el área donde había barba.
- Las fotografías de los candidatos se realizaron con dos cámaras cuyos sensores eran diferentes, un iPhone 6 con cámara frontal de 1,2MP, y un iPhone 7 con cámara frontal de 7MP, esto con la intención de tener la posibilidad de evaluar el comportamiento del sistema frente a fotos de diferentes tamaños y resoluciones, con el fin de hacer el programa mas robusto y darle un rango de operación amplio al usuario de acuerdo con la cámara que el mismo tenga disponible para la toma de fotografías.
- Se tomó un mínimo de 25 fotos por rostro de candidato. Solo se estableció un mínimo puesto que, a partir de esta cantidad de diferentes fotografías del rostro de la persona en diferentes posiciones se puede obtener un modelo aproximado a la forma del rostro del candidato con un menor numero de fotos. Para el caso de establecer un máximo, no se considero importante para la operación del sistema debido a que entre mas imágenes se tengan disponibles del rostro de una persona, mejor será el modelo.

4.2. Obtención y recorte de rostro en las imágenes

Para este proceso se utilizó un algoritmo creado por los autores del proyecto por medio del uso de un archivo de entrenamiento, el cual permite detectar rostros dentro de imágenes por medio de la detección de puntos críticos en el rostro, encerrando el contorno del rostro, boca, nariz, ojos y labios como se puede apreciar en la siguiente imagen:

Figura 5: puntos de interés superpuestos en el rostro

De este modo con los puntos encontrados se encerró el contorno del rostro en el polígono que mas se ajustara a la forma del rostro por medio de la función ConvexHull de openCV, y copió la información contenida en los pixeles encontrados dentro del contorno. Adicionalmente se obtuvo el rectángulo que mejor se acomodara al contorno. Para la creación de la nueva imagen se creó una matriz del mismo tamaño de la

fotografía original, en la cual todas las posiciones de la matriz tienen como valor 0, por lo tanto, es una "imagen negra", sobre la cual se pegó el recorte del contorno del rostro y el ruido producido por el entorno en el que se tomó la imagen es eliminado dejando únicamente la información deseada para la creación del modelo. La figura 6 muestra el resultado del recorte del contorno del rostro con respecto a la imagen original.

Figura 6: A la izquierda se encuentra una imagen tomada del rostro de una persona, y a la derecha se encuentra la imagen después del recorte del rostro

Sin embargo, la imagen resultante al tener una gran cantidad de información f (x, y) =0 el algoritmo de creación de nubes de puntos determina a la imagen como una imagen de baja calidad, que no puede ser utilizada para la creación de nubes de puntos, y por tanto modelos. Debido a lo anterior, es necesario acotar la imagen en el rectángulo previamente adquirido con el fin de poder encerrar la menor cantidad de pixeles en negro para obtener una imagen utilizable. El resultado final es:

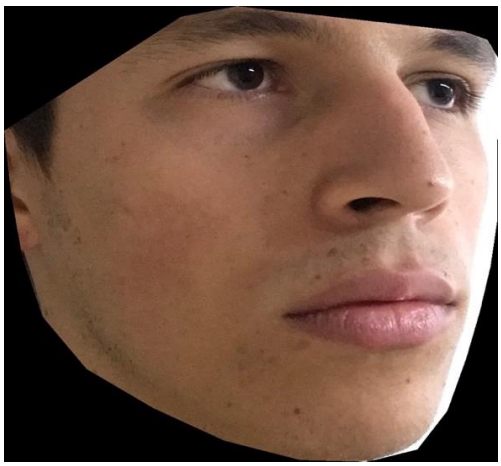

Figura 7: imagen resultante final

Como se puede observar es un recorte rectangular de la imagen mostrada a la derecha de la figura 6.
Finalmente, en el algoritmo de recortes de rostros se adiciona la información de la cámara (modelo, sensor, etc…) y foco de esta a la imagen (metadata) con el uso de la librería pexif creada por Ben Leslie (desarrollador en github con varias contribuciones) con el fin de agregar, editar o eliminar la información exiv (metadata) de las imágenes. Esto con el propósito de que las imágenes sean utilizables en el algoritmo de reconstrucción, que será descrito a continuación.

4.3. Reconstrucción de nube de puntos base

Para la reconstrucción de nubes de puntos a partir de múltiples vistas fue utilizado un algoritmo de libre uso llamado OpenMVG [14], el cual tienen un método de reconstrucción de nubes de puntos global, no secuencial, basado en [15]. El proceso de reconstrucción de nubes de puntos usado en este proyecto tiene 5 etapas:

- Estimación de la rotación global.
- Revisión de la consistencia en la rotación.

- Computo de traslación relativa.
- Registro de traslación.
- Creación final de la nube de puntos.

En la figura 13 se puede observar la nube de puntos resultante de este paso.

Posterior al proceso de creación de nube de puntos base, se crea la nube de puntos densa.

4.4. Reconstrucción de nube de puntos densa

Para el proceso de reconstrucción de nubes de puntos densa se utilizó el algoritmo OpenMVS [19], el cual es un algoritmo de uso libre, que permite crear las nubes de puntos densas, utilizadas para la reconstrucción de la malla (o "mesh") que es utilizado para la creación del modelo 3D resultante del sistema. Esta reconstrucción permite realizar un relleno mas completo y asertivo de la nube de puntos original de forma rápida, puede ser considerado como un proceso de refinamiento de la nube de puntos, que facilitaría la creación de la malla del modelo al rellenar un poco mas la nube de puntos. Posterior a la finalización de este proceso, la nube de puntos se verá de la forma descrita en la imagen de la figura 14.

Como se puede observar en la imagen, de este proceso se obtiene una nube de puntos mucho más completa y parecida al rostro de la persona que la nube de puntos original, encontrada en la tercera etapa del proceso que realiza el sistema explicado en este documento.

4.5. Reconstrucción de la malla (mesh):

Posterior a este paso se obtiene realmente la primera aproximación real al modelo 3D del rostro, los resultados de los pasos anteriores son solo precursores al modelo, donde hay varios puntos en un espacio coordenado definido, creando una forma, pero nunca conectándose entre si, el proceso de reconstrucción de la malla del modelo utilizado para este proyecto está explicado en [9], y es un proceso basado en 4 partes:

- Evaluación de prioridades de soporte en espacio libre.
- Clasificador de interface.
- Encontrar superficie por medio de la solución de un problema de optimización de graficas.
- Uso de clasificador de interface para la reconstrucción de superficies con poco soporte.

Como resultado del proceso de reconstrucción de la malla se obtiene la malla inicial (ver figura 19 a la derecha) sobre la cual se va a trabajar para su refinamiento y posterior texturizado.

Como se puede observar en la figura 19, la malla reconstruida cuenta ya con una superficie definida, y claramente se puede diferenciar la unión de los puntos con respecto a la nube de puntos densa resultante de la anterior parte del proceso.

4.6. Refinamiento

Hasta este momento en el proceso la malla obtenida sigue siendo algo rudimentaria y cuenta con una cantidad apreciable de ruido, y difícilmente cuenta con detalles pertinentes del rostro de la persona, por lo tanto, la malla debe ser refinada para obtener el resultado deseado para este proyecto. Dado que los dos probables problemas de ruido para la malla inicial es la presencia de triángulos que se presentan debido a puntos que no deberían ser tomados en cuenta, dado a que no fueron creados por puntos que hacen parte directa del rostro, o a triángulos de gran tamaño debido a huecos en la nube de puntos, por ejemplo, en las mejillas del rostro de las personas.
Para ver el resultado de este paso, ver figura 16.

4.7. Texturizado

El proceso de texturizado permite obtener el modelo final con color, el método utilizado para este paso esta explicado en [11] y el modelo resultante puede ser apreciado en la figura 17.

4.8. Comparación

La comparación se basa en el algoritmo de ICP el cual consiste en tomar una nube de puntos como base y superponerle la otra en forma rígida sobre esta, así se puede identificar la distancia entre puntos y dar un score para esta. Esto puede ser observado en la figura 24.

5. Protocolo de pruebas:

Con la intención de lograr los objetivos especificados para el proyecto se diseñó un protocolo de pruebas basado en la inexperiencia que se tenía al inicio del proyecto, con el fin de lograr una aproximación inicial a los procesos de reconstrucción de modelos ya existentes, para entender así los procedimientos y crear el sistema final. Teniendo en cuenta lo dicho anteriormente, el protocolo de pruebas diseñado para el proyecto consiste en las siguientes partes:

- Toma de imágenes con diferentes métodos.
- Creación de nubes de puntos con el programa visual SFM (un software libre de *structure from motion)* con las imágenes tomadas con los diferentes métodos de toma de imágenes.
- Creación de los modelos finales con el uso de Meshlab.
- Evaluación de los modelos obtenidos para la determinación del mejor método de toma de imágenes.
- Definición del método final para creación de modelos como un proceso.
- Desarrollo de algoritmos individuales para cada parte del proceso de creación de modelos.
- Evaluación del desempeño obtenido por los algoritmos y su respectiva reestructuración den caso de ser necesario.

Del mismo modo el criterio utilizado para la evaluación de la calidad de los modelos encontrados tiene los siguientes componentes:

- Robustez del modelo: afección del ruido ambiente y el error en el algoritmo al modelo creado.
- Calidad del modelo: comparación del modelo con la persona en la que esta basado, con el fin de determinar de manera subjetiva que tan parecido es el modelo encontrado.
- Automatización del proceso de creación: que tanto tiene que intervenir el usuario en el proceso de creación del modelo.

5.1. Toma de imágenes:

Para la toma de imágenes se eligieron tres métodos:

- Grabación de un video en el que la cámara esta estática y el usuario solamente mueve la cabeza para obtener las diferentes vistas del rostro. Posterior a eso se separaban los cuadros (frames) de cada video para dar con las imágenes deseadas. Sin embargo, este método fue descartado inmediatamente debido a que las imágenes obtenidas tenían muy poca calidad y no era posible encontrar detalles del rostro de los candidatos incluso a la vista de los autores del proyecto.
- Toma de imágenes en secuencia con cámara estática y movimiento de la cabeza del candidato, esto se realizó por medio del uso de la función de ráfaga encontrada en diversos dispositivos móviles. A diferencia del método anterior, con este método es posible apreciar el rostro del candidato en la mayoría de las fotografías realizadas. La principal razón para el uso de este tipo de método es que, al no haber cambios en las regiones diferentes a la región de la cabeza del candidato en las imágenes, en teoría no debe haber modelo en nube de puntos en las regiones sin cambios, puesto que, no es posible realizar una triangulación entre puntos, y por lo tanto no hay información de profundidad.
- Rotación y traslación de la cámara en torno al rostro del candidato con uso de la función de ráfaga de la cámara integrada en dispositivos móviles. En teoría es el método menos robusto para la creación de nubes de puntos, dado que se crea también el modelo en nube de puntos del entorno que rodea al candidato y el cuerpo de este (hombros y cuello).

5.2. Creación de nube de puntos con visual SFM

Teniendo los sets de imágenes deseados para la creación de nubes de puntos se procede a la creación de nubes de puntos con el programa VisualSFM, programa que realiza todo el proceso de reconstrucción desde obtención de vistas hasta la reconstrucción de puntos densa, visible en la figura 8.

Figura 8: Nube de puntos densa obtenida de VisualSFM

5.3. Creación de modelos finales con Meshlab

A partir de las nubes de puntos obtenidas se creó la malla del modelo final del rostro de los candidatos con el uso del método de Poisson para la reconstrucción de superficies. Los modelos creados por el proceso tienen bastantes inconsistencias en cuanto a que hay bastantes agujeros en la frente y las mejillas de la persona, y en algunos casos hay confusiones por parte del programa entre el fondo detrás del candidato y el candidato, que hacen que haya incrustaciones de este en el modelo del rostro, un ejemplo de lo dicho anteriormente es apreciable en la siguiente imagen:

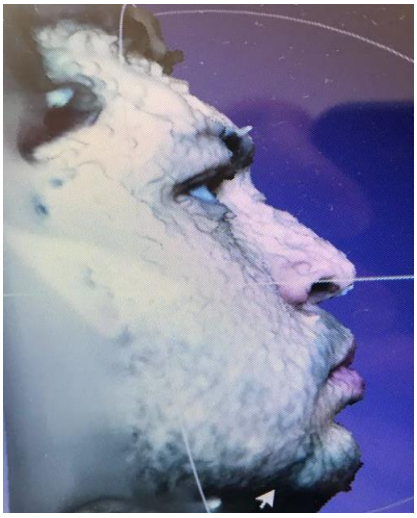

Figura 9: modelo final sin procesamiento adicional obtenido de Meshlab

5.4. Evaluación de los modelos para determinación del mejor método de toma de imágenes

Dado que los modelos encontrados que mas se asemejan al rostro de los candidatos iniciales son los modelos creados por medio del tercer método de toma de imágenes descrito en esta sección, se seleccionó dicho método como el método definitivo para la creación de los modelos del proyecto.

La solución propuesta para evitar la unión entre rasgos del entorno del candidato y el rostro de este fue crear un algoritmo en Python, con el uso de OpenCV que detecte el rostro de la persona en la foto, y lo recorte según su contorno, permitiendo la eliminación de varios factores que producen ruido en la nube de puntos, y por lo tanto en el modelo final.

5.5. Definición del método final para la creación de modelos como un proceso

Como fue descrito en la sección de desarrollo el método final definido para la creación de los modelos esta explicado como un diagrama de flujo, visible en el anexo 1 del documento.

5.6. Desarrollo de algoritmos individuales para cada parte del proceso de creación de modelos

Este paso fue descrito previamente en la sección de desarrollo del documento.

5.7. Evaluación del desempeño obtenido por los algoritmos y su respectiva reestructuración en caso de ser necesario

Los algoritmos utilizados tienen diferentes problemas, los cuales son:

- Son algoritmos de alta rigurosidad, por lo tanto, la calidad e información de las imágenes que se deben ingresar en los algoritmos son muy específicos, esto se solucionó creando el código de recortes de la manera descrita la sección de desarrollo.
- Los algoritmos de reconstrucción de nube de puntos densa y mallas no tienen soporte para todos los modelos disponibles, en el caso del código usado para la reconstrucción de puntos inicial, el formato de archivo obtenido a partir del algoritmo no estaba soportado por los códigos implementados en los siguientes pasos del proceso, por lo tanto, fue necesario agregar un código de conversión de archivos de la nube de puntos resultante, para hacerla compatible con las demás partes del programa. Lo mismo sucedió en el proceso de comparación, en el cual fue se realizó una conversión de archivos del modelo de .PLY a .PCD de modo que fuera posible realizar comparaciones entre modelos.
- Para poder visualizar el modelo texturizado (final) es necesaria la información de imágenes, este problema no se solucionó, debido a que solo es necesario tener el modelo en la misma carpeta de las imágenes para poder visualizarlo.
- No fue posible encontrar un algoritmo de comparación que entregara un valor porcentual como medida de similitud entre los dos modelos, sin embargo mediante el algoritmo ICP se pudo obtener un criterio de correspondencias entre los modelos, y una medida de transición para poder alinear los dos modelos a comparar. Esto se explica mejor en la sección de análisis y resultados, donde se muestra la salida obtenida por el código del algoritmo.
- Los modelos a pesar de ser precisos y similares al rostro del candidato requieren de procesamiento manual para obtener un modelo final, utilizable para otras aplicaciones, esto se puede realizar con programas especializados, como meshlab o blender. Esto no se soluciono, debido a que la intención del proyecto era encontrar los mejores resultados por medio de métodos autónomos, y la edición manual de modelos no lo es.
- Los diferentes algoritmos utilizados son de diferentes bibliotecas, por lo tanto, al tener problemas de compatibilidad, estos se resolvieron en la unificación de los códigos para la creación del sistema final.

6. Análisis y Resultados

A continuación, se presentan los resultados obtenidos por el proceso llevado a cabo en el trabajo de grado con su respectivo análisis frente a lo que se esperaba obtener. Para esto se presentarán los modelos obtenidos por cada algoritmo en una prueba realizada con 36 imágenes. Para posteriormente observar otras pruebas realizadas por el programa frente a otras personas con menor cantidad de imágenes, y además de imágenes sin filtrar.

Se inicia con la reconstrucción de la nube de puntos sparse realizada por la biblioteca de código abierto OpenMVG, mediante el método de reconstrucción global.

Los resultados obtenidos por los algoritmos de este modo de reconstrucción SfM son los siguientes:

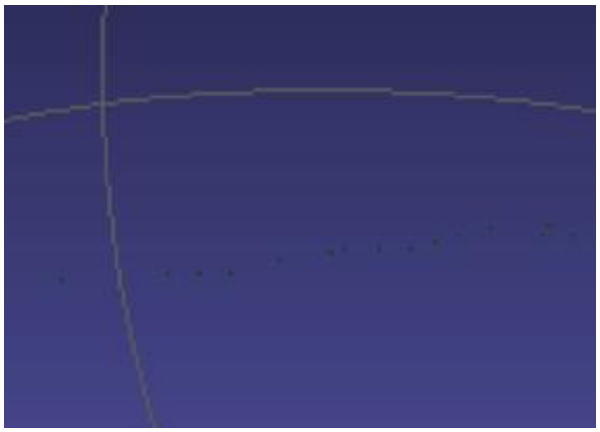

Figura 10: traslación de la cámara

En la figura 10 se observan puntos verdes que representan las distintas poses que tomó la cámara en la toma de imágenes para la construcción del modelo.

de SfM realiza un proceso extenso donde se crean alrededor de 6 modelos, sin embargo, solo se mostrarán los tres modelos finales entregados por el código.

Figura 11: modelo robusto del sparse

En la figura 11 se ve el modelo robusto del sparse, como se puede apreciar este no presenta información de color por lo que se hace necesario mejorarlo.

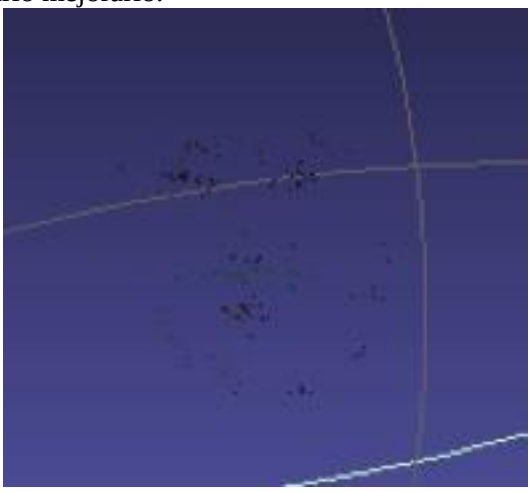

Figura 12: modelo colorized

En la figura 12 se observa el modelo colorized, sin embargo, este no es el último modelo que nos presenta el código de OpenMVG.

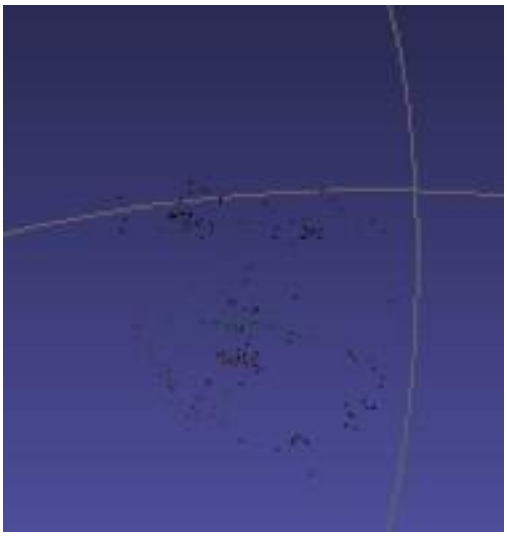

Figura 13: modelo robust colorized

El modelo final que entrega el código SfM es el modelo robust colorized, aun así, se puede observar que el modelo sparse no cuenta con la suficiente información para el manejo de modelos 3D por lo que se hace necesario el uso de la siguiente librería OpenMVS.

Adicionalmente cabe aclarar que para realizar el modelo sparse se debía contar con información de cámara para las imágenes, ya que sin el foco de la cámara la librería no podía realizar los modelos.

Para realizar el modelo de nube de puntos densa se usó la librería OpenMVS la cual soporta los modelos sparse creados en OpenMVG. Sin embargo, se presentaron varios inconvenientes a la hora de realizar el modelo denso debido a que el código de OpenMVS exigía imágenes calibradas.

El resultado del modelo denso se muestra a continuación:

Figura 14: reconstrucción de nube de puntos densa

En la figura 14 se puede observar el modelo denso que entrega el primer aplicativo de OpenMVS, se puede apreciar que es por mucho más completo que el modelo sparse entregado por OpenMVG, sin embargo, este aun presenta huecos en sí.

El siguiente paso es crear el mesh, este se muestra en la figura 15.

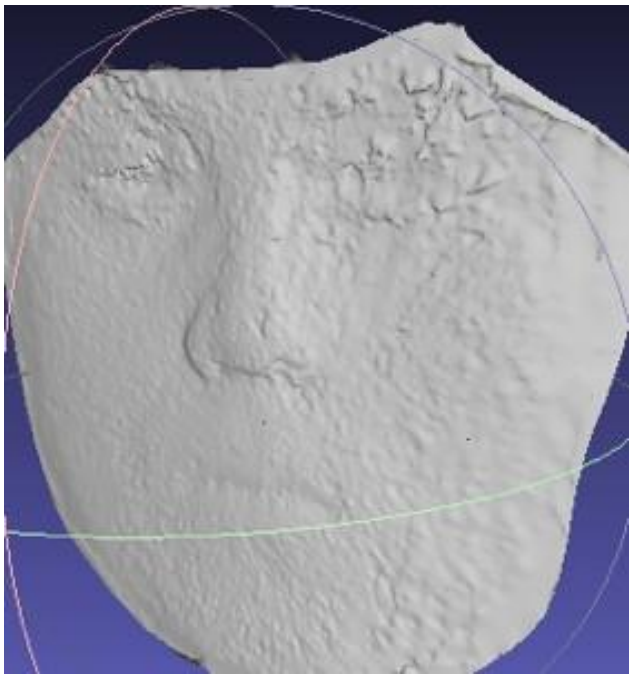

Figura 15: Modelo mesh

Se puede observar que la textura del mesh es rugosa y no se asemeja al modelo real del rostro humano, por lo cual el siguiente paso es probar el algoritmo de refinamiento.

El resultado obtenido de este es el siguiente.

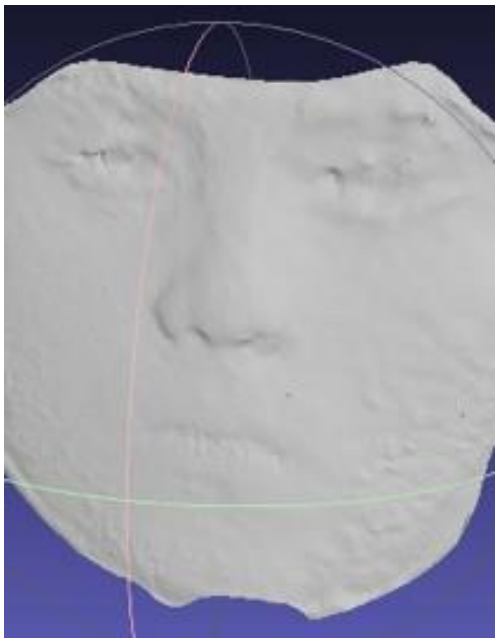

Figura 16: modelo mesh refinado

al comparar la figura 16 con la figura 15 se puede apreciar que se suavizan algunas secciones del modelo con el fin de que se vea más real.

Finalmente, para completar el modelo se procede con el texturing del mesh, siendo este el último algoritmo que nos brinda OpenMVS, el resultado obtenido es el siguiente:

Figura 17: modelo final

En la figura 17 se hace notorio que el suavizado con la textura y el color generan un modelo estable y similar al rostro en realidad. Sin embargo, se hace notorio que el modelo presenta problemas en la profundidad desde el modelo sparse.

A continuación, se presentan los modelos tomados al mismo sujeto, sin filtrar las imágenes para que solo se presentaran los puntos de interés.

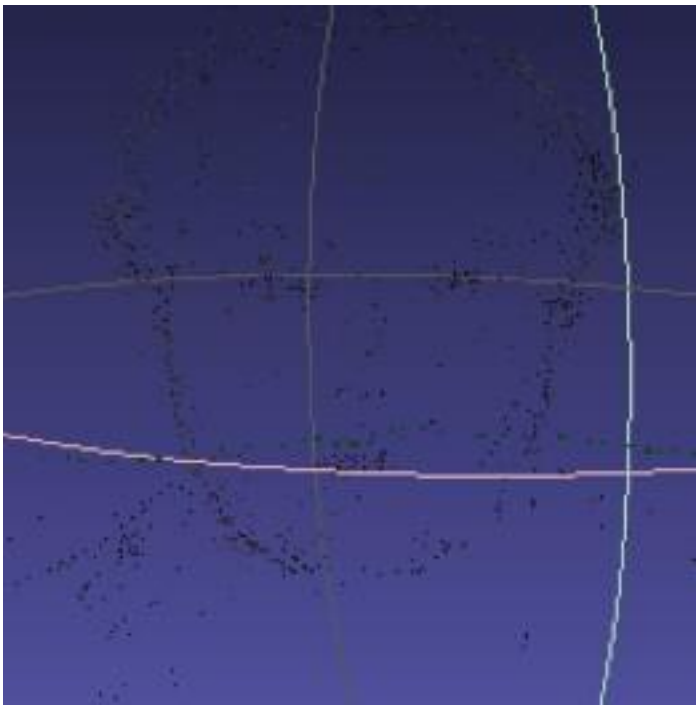

Figura 18 robust colorized sparse point cloud

Como se puede observar en la figura 18, OpenMVG toma muchísimos más puntos que en el modelo creado con las imágenes filtradas, más sin embargo muchos de estos puntos son innecesarios para el proyecto.

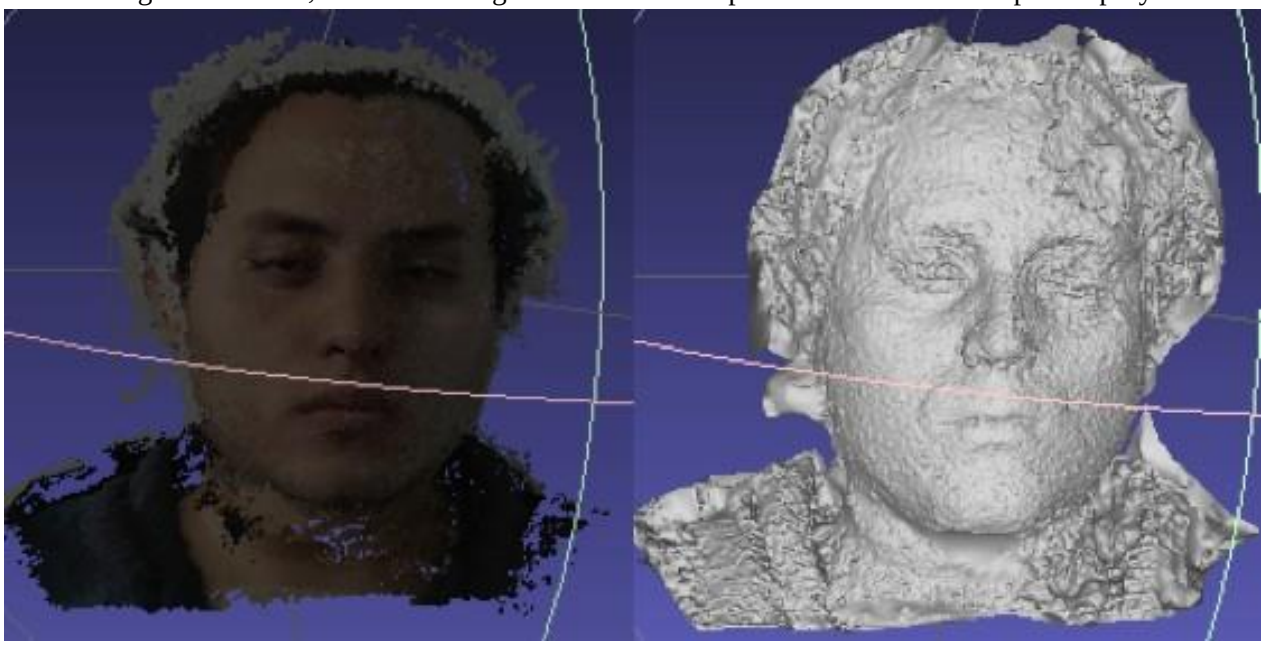

Figura 19:Dense point cloud (izquierda) y mesh(derecha)

En la figura 19, se observa que el modelo tomó puntos del fondo innecesario, adicionalmente se puede apreciar que a la hora de crear el mesh los huecos en el modelo denso son más problemáticos de rellenar.

Figura 20: texturized mesh

Finalmente, en la figura 20 se puede apreciar mejor el problema que tuvo en el mesh y se hace muchísimo más notorio el fondo que no brinda información al proyecto.

En cuanto a costos computacionales es más eficiente para el tiempo realizar los modelos con imágenes filtradas, ya que mientras realizar un modelo con imágenes sin filtrar toma alrededor de 1 hora y media para la cámara del iPhone 6, puede llegar a incluso 8 horas para la imágenes tomadas por la cámara del iPhone 7, con las imágenes filtradas el modelo estaría listo en un tiempo cercano a los 40 minutos, independientemente de la cámara, esto se debe a la cantidad de pixeles en las imágenes. Para el iPhone 6 las imágenes tienen dimensiones de 960 pixeles de ancho y 1280 pixeles de alto, las imágenes del iPhone 7 tienen dimensiones de 2320 pixeles de ancho y 3088 pixeles de alto. Finalmente las imágenes filtradas o poseen dimensiones fijas para todas las imágenes debido a la forma del rostro, sin embargo estas son menores a 900 pixeles de ancho y de alto, lo cual hace más fácil su manejo en el código.

Si comparamos los modelos creados con imágenes sin filtrar con los modelos creados por las imágenes filtradas, se puede apreciar

Ahora se procede a observar modelos de otros sujetos de prueba. para este caso los modelos se realizaron con 46 imágenes

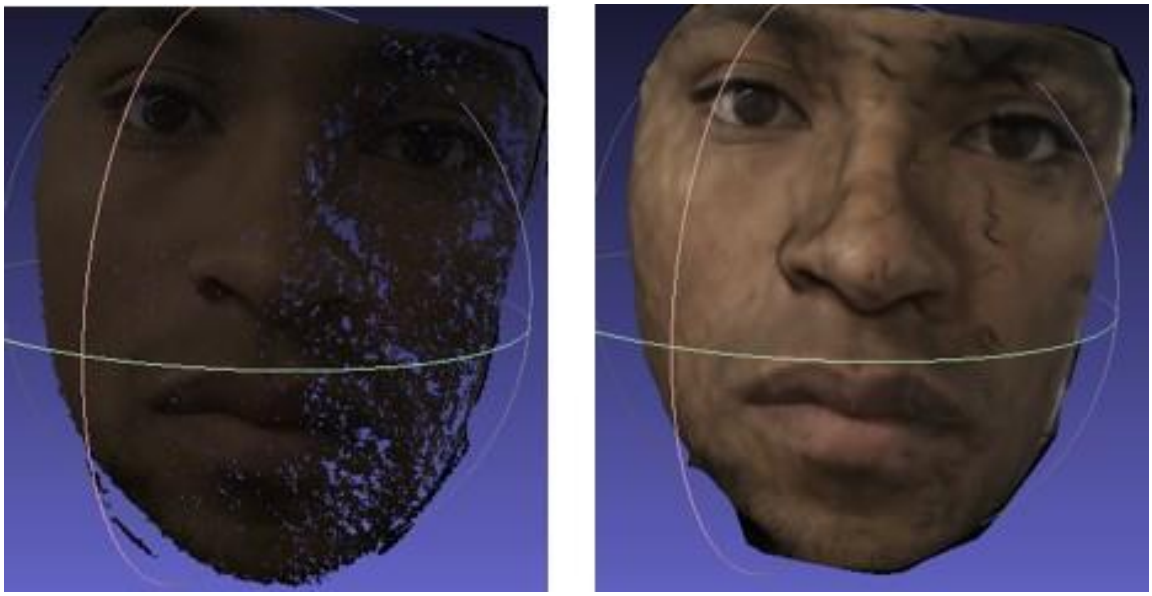

Figura 21: Dense point cloud (izquierda) y mesh texturizado (derecha)

En la figura 21 podemos notar un modelo exitoso realizado con imágenes filtradas, que, aunque presenta algunos problemas de suavizado por el mesh, es un modelo aceptable.

El siguiente sujeto que observar también cuenta con más de 40 imágenes para realizar su modelo, sin embargo su modelo presento inconvenientes. Estos se muestra a continuación.

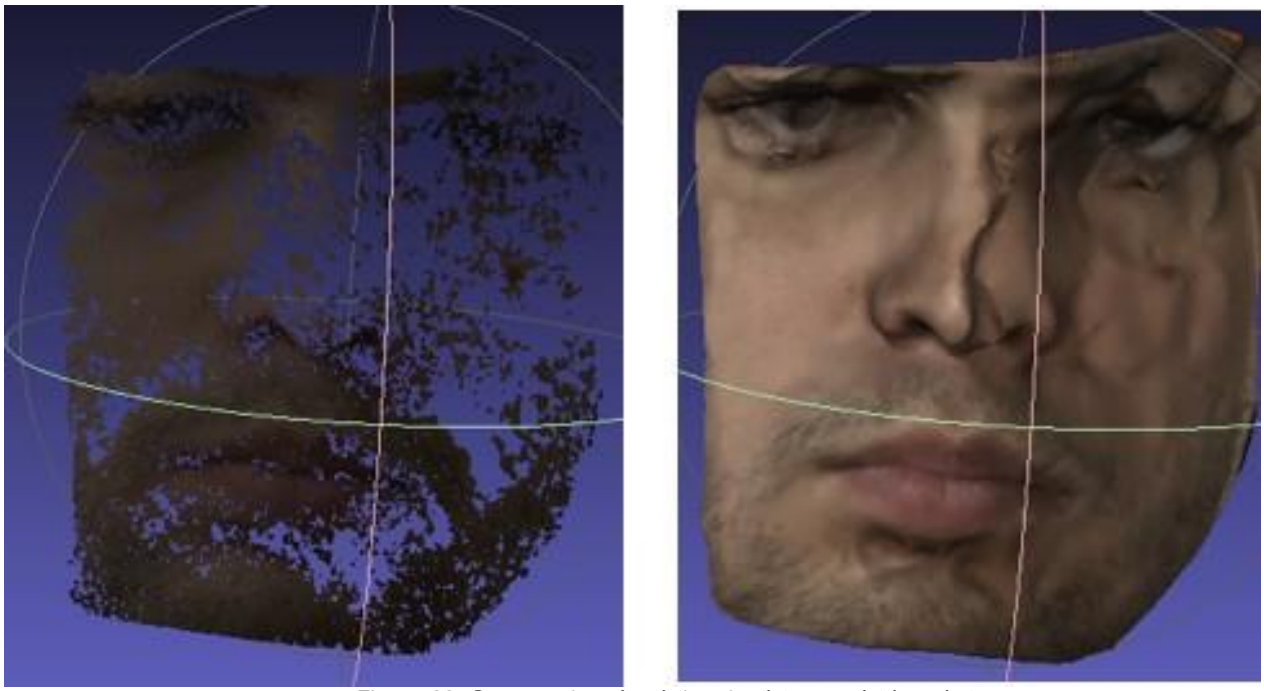

Figura 22: Dense point cloud (izquierda) y mesh(derecha)

Aunque las imágenes fueron tomadas con la misma cámara, se hace notorio que existen muchos factores externos que pueden perjudicar la construcción del modelo además de la cantidad de imágenes capturadas.

Ahora bien, adicionalmente se buscaba comparar modelos, para esto se realizaron modelos subdividiendo las imágenes contenidas para la creación del modelo del sujeto. En el documento solo trataremos una comparación. Para esta usamos los modelos finales.

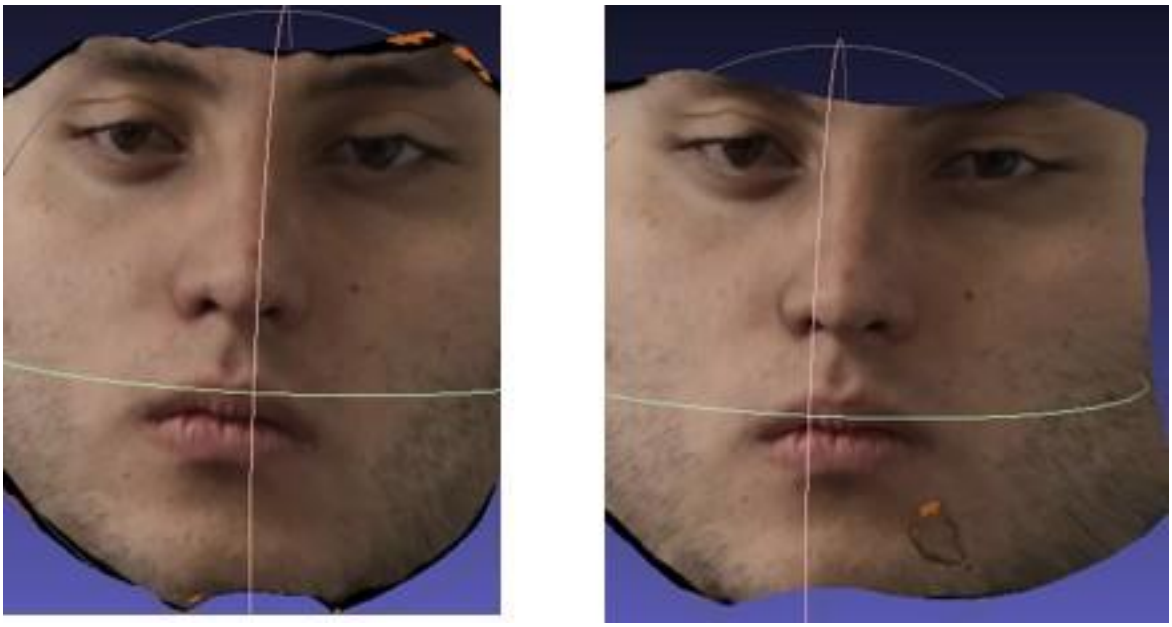

Figura 23: modelos a comparar

Siendo la imagen a la izquierda de la figura 23 la base y la imagen a la derecha la que comparamos mediante el procedimiento del algoritmo ICP y distancias entre puntos. El resultado obtenido es:

```
Using IterativeClosestPoint
1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
saving result to modelo1.pcd
  0.996933 -0.0175331  0.0762874  -0.123728
 0.0185427   0.999754 -0.0125587  0.0166033
-0.0760475  0.0139347   0.997009 -0.0325619
         0          0          0          1
saving result to modelo2.pcd
```

Figura 24: Resultados de comparación

Al comparar el modelo a la izquierda de la figura 23 con sigo mismo obtenemos una matriz identidad de 4x4, esta es la transformación rígida del ICP, y quiere decir que sus puntos hacen match en un 100%, la segunda matriz se obtiene de comparar el modelo base con el modelo de la figura 18. Finalmente, entre más cercana sea la matriz generada a la matriz identidad más similar es el modelo.

La información de la matriz son las coordenadas del modelo frente al modelo base, y el valor numérico es la rotación necesaria para que los puntos correspondientes queden alineadas, es decir que al obtener la matriz identidad se indica que los puntos de correspondencias entre los modelos están perfectamente alineados.

Adicionalmente, se realizó la prueba de que al comparar modelos diferentes ICP no termina de iterar por lo que no haya una comparación o puede indicar que no hay correspondencias entre los modelos y por lo tanto no entregar la matriz.

Finalmente se presenta la interfaz gráfica realizada, para facilitar el uso de estos códigos para el usuario.

Figura 25: inicio de la interfaz grafica

En la figura 25 observamos el inicio de la interfaz donde el primer botón "Visualizar un modelo" crea una ventana emergente para visualizar los modelos creados. Mientras que el botón Directorio de entrada carga el directorio que contiene las imágenes para trabajar con los códigos.

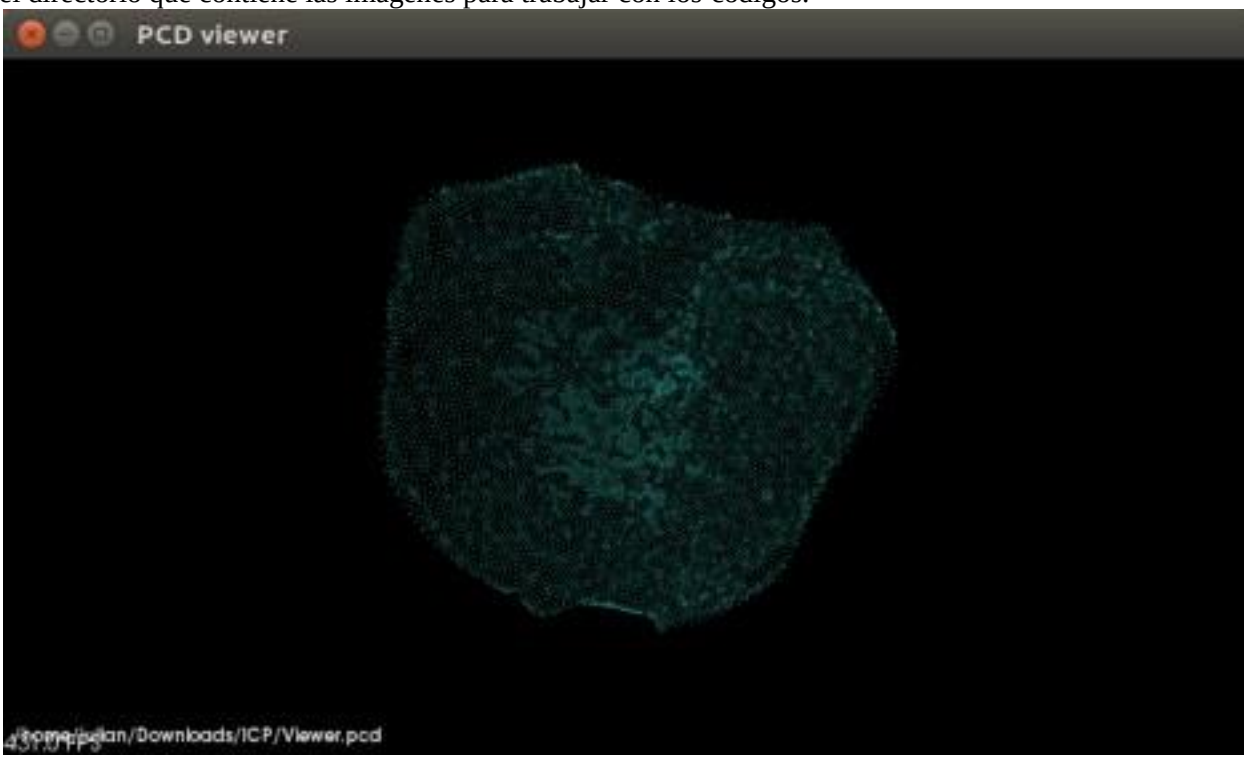

Figura 26: viewer

Como se observa en la figura 26, la ventana permite ver los modelos, pero no los muestra con el color adecuado como lo hace el programa Meshlab.

Figura 27: interfaz con todos los códigos

En la figura 27 se puede observar la interfaz gráfica una vez ha corrido todos los pasos, cada paso se habilita una vez se haya completado el paso anterior.

7. Conclusiones, recomendaciones y posibles avances

El proyecto tiene como objetivo general el desarrollo de un sistema de reconstrucción tridimensional de modelos faciales a partir de imágenes monoculares, este fue cumplido en términos generales, ya que como se puede apreciar en la sección de análisis y resultados, los modelos obtenidos en general por los códigos usados, son aceptables visualmente y adicionalmente se puede apreciar la consistencia entre modelos tomados a la misma persona. Sin embargo mediante el uso de softwares externos pueden ser mejorados.

- Se tenía como objetivo la implementación de un algoritmo de reconstrucción tridimensional a partir de varias imágenes capturadas desde distintas poses del rostro de una persona. Este objetivo se logró satisfactoriamente, usando los códigos libres OpenMVG y OpenMVS, los cuales generan el modelo tridimensional del rostro mediante la fotogrametría. Como se aprecia en la sección 5. Los modelos creados por el código resultan muchísimo más agradables visualmente que los generados por el software de visualSFM.
- Como segundo objetivo específico esta la implementación de un algoritmo de comparación entre dos modelos tridimensionales, este con el fin de evaluar objetivamente la consistencia de los modelos obtenidos. Para esto implementamos el algoritmo ICP, sin embargo este no da un score claro de la consistencia de los modelos, por lo cual no se obtiene un valor cuantitativo de la consistencia de los modelos.
- Se logró implementar una interfaz gráfica, con el fin de facilitar el uso de los códigos utilizados en nuestro trabajo de manera efectiva, además de permitir la visualización de los modelos creados por el código.

Según lo visto en la sección 6. Imágenes de mayor tamaño implican un modelo con mayor calidad, sin embargo también implica un mayor tiempo de procesamiento, llegando a incluso más de 8 horas. Debido a los largos tiempos de procesamiento, se recomienda al usuario ser paciente en la creación de modelos.

Como posibles mejoras al trabajo se recomienda la implementación de un protocolo para la toma de imágenes, con el fin de obtener un modelo bueno para futuros trabajos. Además de un posible refinamiento extra en softwares externos como Blender y Meshlab.

Bibliografía

[1] Andrea F. Abate, Michele Nappi *, Daniel Riccio and Gabriele Sabatino, "2D and 3D face recognition: A survey," Pattern Recognition Letters 28(14):1885-1906 · October 2007.
[2] D. Kim, J. Choi, J. T. Leksut and G. Medioni, "Accurate 3D face modeling and recognition from RGB-D stream in the presence of large pose changes," 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, 2016, pp. 3011-3015.
[3] J. Li, Y. Zhang, P. Xu, S. Lan and S. Li, "3D Personalized Face Modeling Based on KINECT2," 2016 9th International Symposium on Computational Intelligence and Design (ISCID), Hangzhou, 2016, pp. 197-201.
[4] Y. Saleh and E. Edirisinghe, "3D face reconstruction and recognition using the overfeat network," 2017 8th International Conference on Information and Communication Systems (ICICS), Irbid, 2017, pp. 116-119.
[5] S. Naveen, K. P. Rugmini and R. S. Moni, "3D face reconstruction by pose correction, patch cloning and texture wrapping," 2016 International Conference on Communication Systems and Networks (ComNet), Thiruvananthapuram, 2016, pp. 112-116.
[6] J. Yun, J. Lee, D. Han, J. Ju and J. Kim, "Cost-efficient 3D face reconstruction from a single 2D image," 2017 19th International Conference on Advanced Communication Technology (ICACT), Bongpyeong, 2017, pp. 629-632.
[7] JPEG, International Standard ISO/IEC 10918, 1994.
[8] ISO C++, International Standard ISO/IEC 1488, 2014.
[9] *Exploiting Visibility Information in Surface Reconstruction to Preserve Weakly Supported Surfaces* M. Jancosek et al. 2014
[10] H. H. Vu, P. Labatut, J. P. Pons and R. Keriven, "High Accuracy and Visibility-Consistent Dense Multiview Stereo," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 5, pp. 889-901, May 2012.
[11] *Let There Be Color! - Large-Scale Texturing of 3D Reconstructions* M. Waechter et al. 2014.
[12]C.Roberto.(2008).Structure From Motion [online] encontrado en: http://mi.eng.cam.ac.uk/~cipolla/publications/contributionToEditedBook/2008-SFM-chapters.pdf
[13]OpenCV, Team. "Introduction to SIFT" [online] encontrado en: https://docs.opencv.org/3.3.0/da/df5/tutorial_py_sift_intro.html
[14] https://github.com/openMVG
[15] P. Moulon, P. Monasse and R. Marlet, "Global Fusion of Relative Motions for Robust, Accurate and Scalable Structure from Motion," *2013 IEEE International Conference on Computer Vision*, Sydney, NSW, 2013, pp. 3248-3255.
[16] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239-256, Feb. 1992. doi: 10.1109/34.121791

[17] Bill Triggs, Philip Mclauchlan, Richard Hartley, Andrew Fitzgibbon. Bundle Adjustment – A Modern Synthesis. Bill Triggs and Andrew Zisserman and Richard Szeliski. Inter- national Workshop on Vision Algorithms, Sep 2000, Corfu, Greece. Springer-Verlag, 1883, pp.298–372, Lecture Notes in Computer Science; Vision Algorithms: Theory and Practice. <http://www.springerlink.com/content/plvcrq5bx753a2tn/>. <10.1007/3-540-44480-7 21>. <inria-00548290>

[18]R.Hartley y P.Sturn.(1997, Nov 2) Triangulation (vol 68)[online] encontrado en: https://perception.inrialpes.fr/Publications/1997/HS97/HartleySturm-cviu97.pdf

[19] https://github.com/cdcseacave/openMVS

Anexos

Para ver anexos visitar el siguiente link: https://drive.google.com/open?id=1Zi68ysUSHRkFlLpWzW_HDZQnr5G2ZQoI