FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

RECONSTRUCCIÓN FACIAL 3D MEDIANTE CÁMARA DE PROFUNDIDAD RGB-D

AUTOR

RICHARD HERRERA

AÑO

2020

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

RECONSTRUCCIÓN FACIAL 3D MEDIANTE CÁMARA DE PROFUNDIDAD RGB-D.

Trabajo de Titulación presentado en conformidad a los requisitos establecidos para optar por el título de Ingeniero en Electrónica y Redes de Información.

Profesor Guía
MSc. David Fernando Pozo Espín

Autor

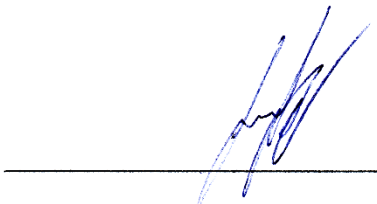
Richard Herrera Orbea

Año

2020

DECLARACIÓN PROFESOR GUÍA

"Declaro haber dirigido el trabajo, Reconstrucción Facial 3D mediante Cámara de Profundidad RGB-D, a través de reuniones periódicas con el estudiante Richard Alexander Herrera Orbea, en el semestre 202020, orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación"



David Fernando Pozo Espín
Máster en Automática y Robótica.
C.I:1717340143

DECLARACIÓN PROFESOR CORRECTOR.

“Declaro haber revisado este trabajo, Reconstrucción Facial 3D mediante Cámara de Profundidad RGB-D, del estudiante Richard Alexander Herrera Orbea, en el semestre 202020, dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de titulación”.

Wilmar Hernández Perdomo

Doctor Ingeniero en Electrónica.

CI:0151721016

DECLARACIÓN DE AUTORÍA DEL ESTUDIANTE.

“Declaro que este trabajo es original, de mi autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes”.

Richard Alexander Herrera Orbea

CI:1724002405

AGRADECIMIENTOS

Agradezco principalmente a mi familia la cual supo ser mi soporte y quien me ha apoyado en los momentos más difíciles, siempre me han brindado todo de ellos y este trabajo es para mostrar mi agradecimiento, a mis amigos, con los cuales he vivido un sin número de aventuras y que día a día hemos sobrepasado cualquier dificultad.

DEDICATORIA

A mi familia que siempre me supo guiar y ayudar en los momentos más difíciles, a mi madre que ha sido el pilar fundamental para que yo pueda seguir adelante día tras día, a mis amigos que formaron parte de la persona que soy actualmente. Gracias a todos por siempre estar ahí para mí.

RESUMEN

Los sistemas de detección y reconocimiento facial son un tipo de tecnología se encuentran en desarrollo y han encontrado gran número de aplicaciones la sociedad actual. Como ejemplo de una de sus múltiples aplicaciones se encuentran dentro los teléfonos inteligentes (smartphones) para seguridad del equipo. Otro ejemplo se encuentra presente en las cámaras de seguridad que se instalan frecuentemente en ciudades para poder vigilancia electrónica en las calles con el avance de estas tecnologías se han desarrollado cámaras que pueden darnos no solo una imagen en dos dimensiones, sino también una medida de profundidad.

En el presente proyecto de titulación se desarrolla el prototipo de un código el cual está enfocado a la detección facial, su reconstrucción de imágenes en tres dimensiones y la aplicación de filtros para eliminar los puntos de ruido dentro de las imágenes analizadas y el suavizado de las mismas. El proyecto se ha desarrollado en un ambiente digital, donde se ha utilizado el entorno Python para su programación. El uso de una cámara de profundidad Orbbec Astra Pro para la obtención de imagen con la respectiva profundidad de cada una de sus partes y así poderla representar en 3D.

Finalmente, se diseñó una interfaz gráfica y un prototipo funcional para realizar pruebas de terreno, los resultados experimentales son satisfactorios, demostrando que los algoritmos propuestos y la cámara RGB-D la cual brinda gran cantidad de puntos significativos, son una herramienta factible para el reconocimiento y reconstrucción de imagen.

ABSTRACT

Facial detection and recognition systems are a type of technology under development and have found many applications in today's society. As an example of one of its many applications are inside the smart phones (smartphones) for computer security. Another example is present in the security cameras that are frequently installed in cities to be able to electronic surveillance in the streets with the advance of these technologies have developed cameras that can give us not only an image in two dimensions, but also a measure of depth.

In the present project, a prototype of a code is being developed which is focused on facial detection, its reconstruction of images in three dimensions and the application of filters to eliminate the points of noise within the images analyzed and their smoothing. The project has been developed in a digital environment, where the Python environment has been used for its programming. The use of an Orbbec Astra Pro depth camera to obtain an image with the respective depth of each of its parts in order to be able to represent it in 3D.

Finally, a graphic interface and a functional prototype were designed to carry out field tests. The experimental results are satisfactory, demonstrating that the proposed algorithms and the RGB-D camera, which provides a large number of significant points, are a feasible tool for the recognition and reconstruction of images.

ÍNDICE

1. Introducción

1.1 Antecedentes:

La biometría es el proceso por el cual se toman ciertos datos morfológicos únicos que diferencian a un individuo de otro. Las diferentes formas de los ojos, rasgos en los dedos y tonalidades de color pueden ser utilizadas para reconocer diferente a cada persona. (Serratosa 2008)

Se practica desde hace mucho tiempo, el simple hecho de contestar una llamada y reconocer de quien es la voz son mecanismos de biometría. Dentro de las técnicas biométricas más utilizadas tenemos detección dactilar, reconocimiento del iris, reconocimiento facial.
El reconocimiento facial nace en 1960 con mediciones manuales que realizaba Woodrow Blledsoe con un dispositivo llamado tableta RAND, el cual tomaba las medidas de los ojos nariz y boca y junto con una base de datos devolvía el rostro con las características más cercanas. (Bledsoe 1959)

Turk y Pentland para el año de 1991 realizaron las primeras pruebas sobre la detección de rostros dentro de imágenes, pero su limitante fue la tecnología de aquel entonces. (Turk, M. A. 1991)

Para el año de 2009 en el condado de Pinellas, se elaboró una base de datos forense con las imágenes de los criminales, las cuales fueron tomadas por cámaras equipadas en los vehículos de los oficiales.

En la actualidad existen diversas técnicas de reconocimiento facial para imágenes en dos dimensiones y se encuentran aplicando reconocimiento facial en lugares públicos para detectar atracos y reconocer personas. (Wang, Z 2020)

1.2 Alcance

El documento tiene como propósito la adquisición de datos, procesamiento de imágenes en dos y tres dimensiones para la detección y reconstrucción un rostro en 3D. Implementación de un prototipo con una cámara de profundidad.Para esto se utilizará una cámara RGB-D, la cual cuenta con un campo centrado y ofrece una mayor calidad de profundidad por grado. Estas cámaras tienen un sensor RGB integrado los cual nos proporciona el color y se definirán las mejores opciones para realizar la detección facial y escaneo 3D.

1.3 Justificación

La modelación tridimensional del rostro humano es una tarea compleja y de gran experticia en entornos donde la calidad del modelo 3D resultante es primordial. Para este propósito se han desarrollado técnicas computacionales de reconstrucción facial 3D. Con el continuo avance tecnológico aparecen nuevos equipos, los cuales nos ayudan a mejorar las técnicas y el rendimiento de los algoritmos actuales de detección y reconstrucción facial.Este proyecto tiene como propósito brindar una alternativa de detección y reconstrucción facial basado en cámaras de profundidad para una implementación futura en controles de acceso.

1.4 Objetivo General

Realizar la reconstrucción facial de una imagen en tres dimensiones con una cámara de profundidad.

1.5 Objetivos específicos

- Investigar las posibles aplicaciones para cámaras 3D de profundidad junto con sus diferentes características.
- Implementar algoritmos existentes de detección facial y reconstrucción de rostros en tres dimensiones.
- Analizar la información obtenida en las pruebas realizadas en él prototipo.

2. Marco Teórico

2.1 Introducción

Un problema muy común dentro de los sistemas de visión artificial es la detección y reconstrucción facial cuyos datos son tomados de fotografías y videos. La cualidad más importante de esto es la definición que se obtiene con cada uno de los métodos.

GANFIT, es un método de reconstrucción de rostros con alta definición propuesto por Baris Gecer, el cual consiste en varios algoritmos de inteligencia artificial, los cuales son un tipo de aprendizaje no supervisado y comúnmente utilizado dentro de las redes neuronales. Utiliza este modelo para entrenar un generador facial de texturas, estos se alinean hacia un modelo de rostro genérico utilizando una optimización no lineal para no perder detalles. Este modelo al analizar la imagen define cual es la forma de rostro que más se parece junto con que expresión está mostrando, se realiza una convolución con los parámetros de luz y detalles de la imagen. (Gecer et al., 2019)

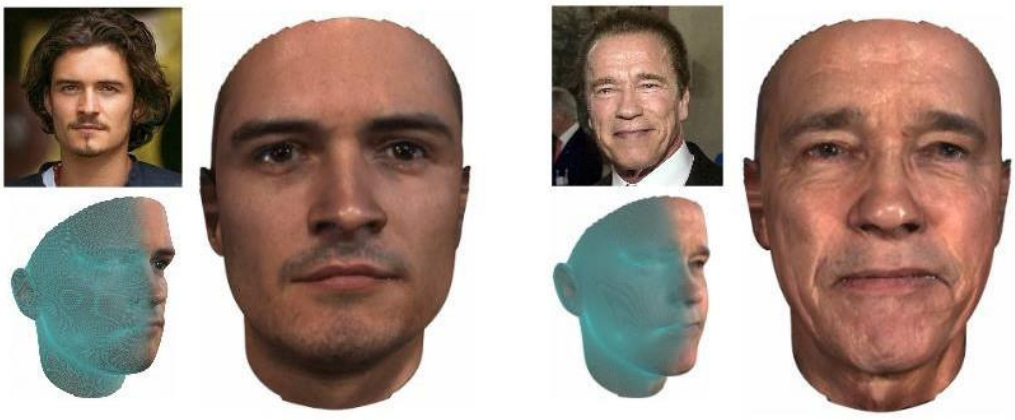


Figura 1. Reconstrucción facial por medio de GANFIT.
Tomado de (Gecer et al., 2019)

Otro método que se puede utilizar para recrear imágenes en 3D a partir de imágenes en 2D es “Stacked Contractive Autoencoder”, el cual es un framework que trabaja con Deep learning. Este mecanismo está diseñado para analizar las características de los subespacios dentro de imágenes para luego emparejarlas con un subespacio entrenado de una reconstrucción en 3D. Todo esta se lleva a cabo gracias a una capa directamente conectada a una red neuronal, usando parámetros del rostro pre entrenados se optimiza al aplicar algoritmos descendientes de gradiente. (Zhang et al., 2017)

Pengfei, proponen un método de reconstrucción basado en redes neuronales. Inspirado en reconstrucciones realizadas mediante DNN (Deep Neural Networks), así como otros métodos este se basa en reconocer cual es la forma del rostro y por medio de convoluciones obtienen dos muestras las cuales concatenan. Finalmente se filtra ambas muestras y se agregan los parámetros de luz, las muestras de los rostros con los cuales se va a realizar la comparación por medio de convoluciones siguen dos modelos faciales principalmente. AFM el cual consiste en 7,597 vértices con 14,912 triangulaciones de rostros y BFM el cual consiste en 53,490 vértices con 160,470 triangulaciones de rostros. (Pengfei et al., 2017)

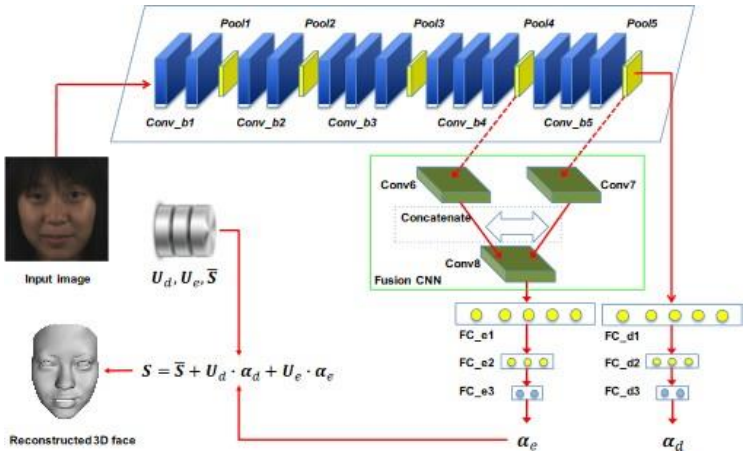


Figura 2. Diagrama de la Red Neuronal Convolucional.
Tomado de (Pengfei et al., 2017)

Como podemos observar de todos los modelos de reconstrucción, se basan mucho en la comparar con redes neuronales características principales del rostro. Para el siguiente método se enfocan en la posición del rostro, distancia entre los ojos, contorno de los ojos y la nariz para realizar una reconstrucción. Con un me todo de evaluación de la luz junto con un amplio detalle de las posiciones del rostro en un espacio UV, se logra un rápido procesamiento de la imagen de 9.8ms. Este modelo se basa principalmente en encontrar con gran precisión los detalles del rostro, para o que se utilizan modelos como AMM (Active Appearence Model), CLM (Constrained Local Models) y modelos basados en redes neuronales. (Feng et al., 2018)

Aaron S. Jackson (2017), menciona que los sistemas de reconstrucción en 3D del rostro resultan tener una alta dificultad, ya que no siempre se cuenta con múltiples imágenes del mismo rostro. Muchos desafíos que presentan las imágenes son las diferentes iluminaciones que no son uniformes, para lo que se utilizarán Redes Neuronales

Convolucionales (CNN), las cuales tienen como herramientas muchas imágenes tanto en dos como en tres dimensiones y distintos modelos faciales. Esto les permite realizar una regresión de la representación volumétrica del rostro. Esta técnica requiere solo una imagen que puede ser total o parcial, la cual será evaluada por el modelo y siempre pedirá una retroalimentación de este para su entrenamiento.

La reconstrucción de rostros en tres dimensiones puede ser principalmente realizada por una colección de imágenes del rostro, los cuales se encuentran en diferentes posiciones, gestos y condiciones de luz. Para esto se mete las imágenes a un esquema fotométrico de formulación estéreo y obtener datos de profundidad junto con la información de albedo, la cual nos muestra la incidencia de radiación de luz sobre una superficie. El modelo que usa esta reconstrucción se llama 3DMM (3D Morphable Model). (Roth et al., 2017)

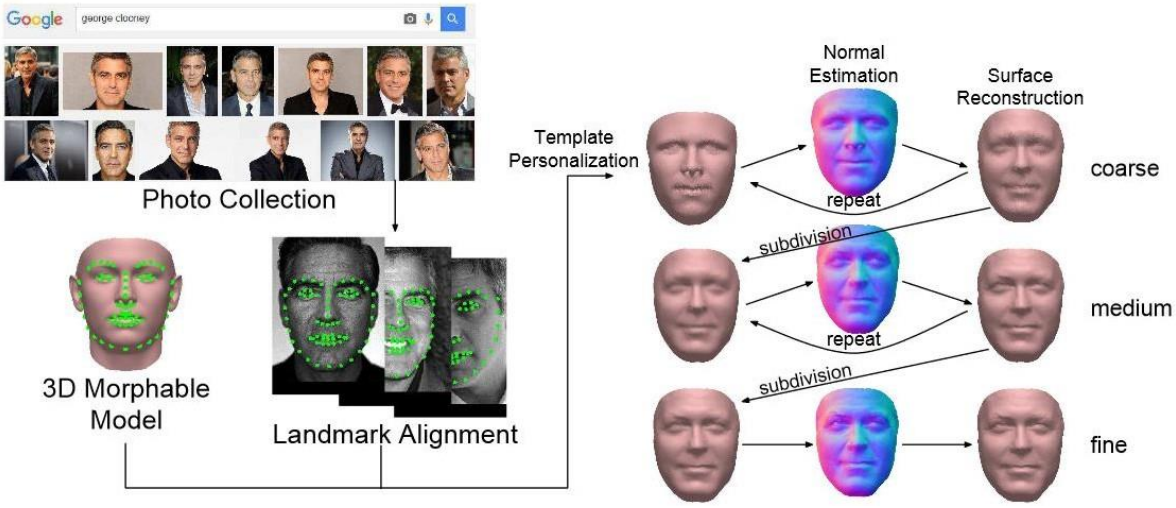


Figura 3. Diagrama de reconstrucción de rostros por medio de facciones características. Tomado de: (Roth et al., 2017)

Existen varios métodos de deconstrucción de rostros a partir de imágenes en dos dimensiones, estos tienen bastante detalle en la zona frontal, pero presentan problemas en zonas del rostro lejanas al centro o que sean obstruidas parcialmente. Extreme 3D Face Reconstruction presenta un método basado en mapeo de relieve, el cual asegura un nivel de detalle muy alto. Para realizar la estimación de las facciones del rostro se procede a separar las partes fundamentales y siguen un proceso de codificación decodificación por medio de convoluciones, lo que ayuda a obtener las aproximaciones de los mapas de relieve. (Tran et al., 2018)

Feng Luis (2018), propone una red dentro de la cual se pueda codificar y decodificar las características partículas re de los rostros y sus diferentes formas con gestos para las reconstrucciones en 3D a partir de imágenes en 2D. Mediante esta red se puede detectar simultáneamente la forma del rostro, así como su gesto, a diferencia de las reconstrucciones comunes en 3D a partir de una sola imagen, en la cual la imagen en 2D se trata separada de la imagen en 3D y se tienen valores residuales, Feng propone la utilización de clasificadores para poder detallar la imagen en 3D.

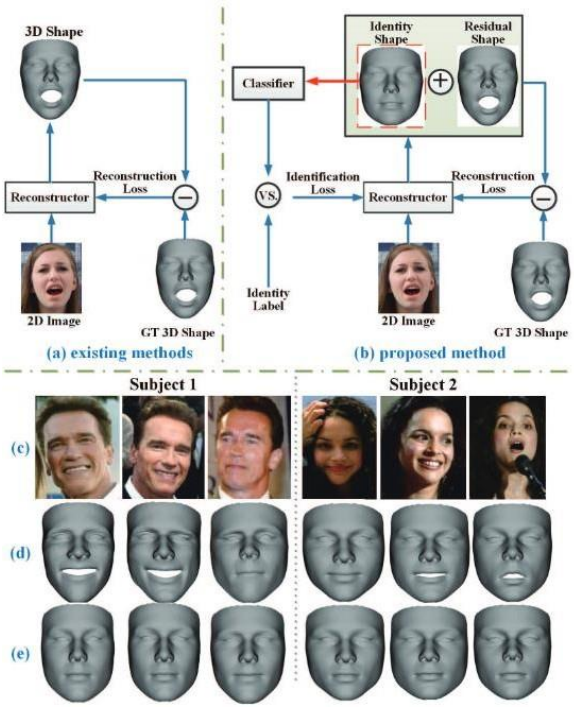


Figura 4. Propuesta de la red neuronal para reconocimiento de expresiones. Tomado de:(Liu et al., 2018)

La reconstrucción con detalles geométricos propone una alternativa a la complicada reconstrucción a partir de una sola imagen en 3D, inspirados por trabajos de animación usando cámaras RGB-D se desarrolló un método para reconstrucción sin restricciones en imágenes de dos dimensiones usando una estrategia de optimización que va desde lo más grueso a lo más fino. Primero se obtiene un modelo suavizado, generado por un modelo de rostro bilineal, luego se aplican deformaciones correctivas para así obtener un rostro medianamente reconstruido. Finalmente se usa un método de formas y sombras para obtener detalles geométricos.

2.2 Rostro Humano

Según la Real Academia de la Lengua Española el rostro o cara tiene como significado “Parte anterior de la cabeza humana desde el principio de la frente hasta la punta de la barbilla.”

El rostro es la parte frontal de la cabeza, para el caso de los humanos, este se encuentra conformado por 5 zonas características: Frente, es un segmento de piel que va desde la línea capilar hasta las cejas; Los ojos, los cuales se encuentran protegidos por los parpados y pestañas; La nariz, junto con sus fosas nasales; Las mejillas, aquellas que cubren la mandíbula (maxilar superior) y la boca, comprendida por los labios.(Keith L. Moore, Arthur F. Dalley, 2013)

Figura 5. Rostro Humano

El área fusiforme de rostro es una parte del cerebro que nos permite por medio de la visión detectar el patrón en el cual se encuentran localizados y la forma de órganos tales como los ojos, nariz y boca. Los cuales son característicos para cada persona y son los datos que permiten realizar la identificación biométrica.(Kanwisher et al., 1997)

2.3 Visión Artificial

La visión artificial es una rama de la inteligencia artificial, la cual permite que el computador tome decisiones a partir de patrones analizados en imágenes. Esto permite que los equipos puedan tener cierto grado de autonomía, permitiendo que evadan obstáculos, dentro de la industria usado para la cuantificación de objetos, y en el ámbito médico que logren interpretar parámetros de un TAC.

(García, 2002)

A diferencia de los humanos que podemos percibir el mundo con tres dimensiones y determinar ciertas características como el material, color y rugosidad de un objeto los sistemas computacionales no pueden realizarlo de la misma manera. En la visión artificial se trata de explicar al computador por medio de algoritmos y reconstrucción de propiedades tales como la forma, el color, como se puede determinar que en una imagen exista un objeto.

Para esto la visión artificial ha avanzado en varios campos y tiene muchas aplicaciones dentro de las cuales se incluye:1) OCR (Optical carácter recognition), la cual es una herramienta que usan las cámaras para determinar si dentro de la imagen existen letras y para el caso de los números (ANPR). 2) “Machine Inspection” la cual ayuda a determinar la calidad de un objeto, usado comúnmente en la visión estéreo para detectar deformaciones en piezas o con Rayos X para encontrar defectos en metales. 3) La reconstrucción de modelos en 3D a base de puntos o por medio de imágenes en dos dimensiones. (Szeliski, 2004)

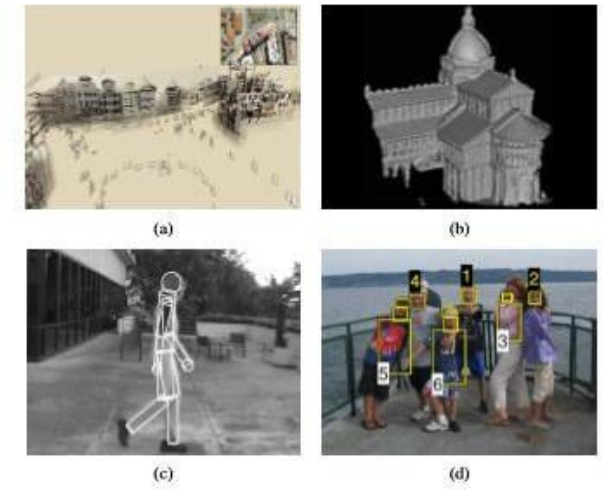


Figura 6. Aplicaciones de visión artificial. a) Procesamiento de imágenes, b) Reconstrucción en 3D, c) Tracking de movimiento corporal, d) Detección facial. Tomado de (Szeliski, 2004)

La visión artificial nace en la década de 1970, como una percepción de los equipos para imitar la inteligencia humana mediante comportamientos, los pioneros de esta tecnología se encontraban principalmente en MIT y Standford. Rosefeld en 1976 hacía experimentos de procesamiento de imagen para crear estructuran en tres dimensiones a partid de imágenes, Fischel y Elschlager en 1973 crearon arreglos elásticos de partes de estructuras pictográficas, lo cual sentó las bases para el reconocimiento de objetos. (Szeliski, 2004)

Los pasos principales de la visión artificial son los siguientes: 1) La obtención de la imagen y datos a trabajar: Esto se lleva a cabo por medio de cámaras normales, estereoscópicas o de profundidad. 2) Análisis de la imagen: Se utilizan ciertos algoritmos que nos permiten descomponer a la imagen en sus características básicas para poner como colores primarios, iluminación, bordes, etc. 3) Procesamiento y aplicación: Aquí se usarán diferentes algoritmos que en base a la información recopilada permitirá a la máquina tomar decisiones. (Basu y Li, 1993)

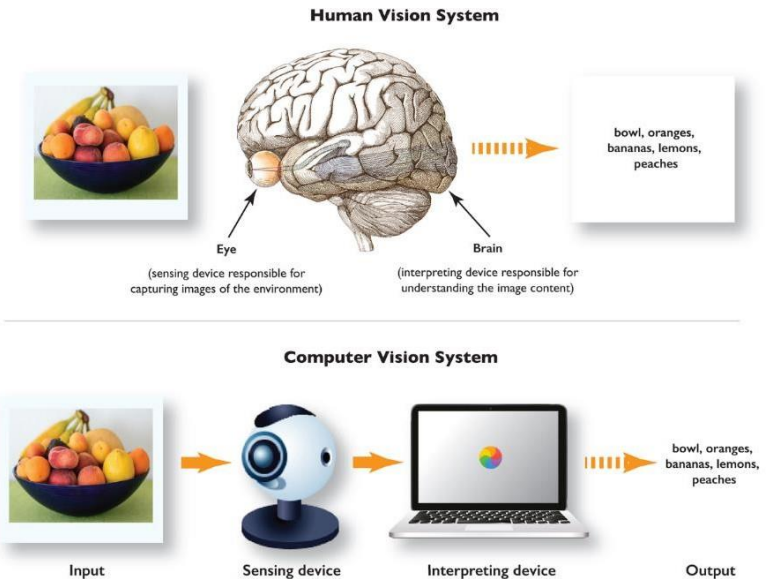


Figura 7. Comparación de Visión Artificial y Visión Humana.
Tomado de (Elgendy, 2019)

2.3.1 Imagen Digital

El ojo humano puede captar un amplio rango de colores dentro del espectro de la luz visible, las cámaras tienen la capacidad de poder captar ciertos rangos de la luz invisible tales como infrarroja y ultravioleta. Puesto que estos rangos de frecuencia no tienen un color, las cámaras cuenta con un filtro para eliminarlas.

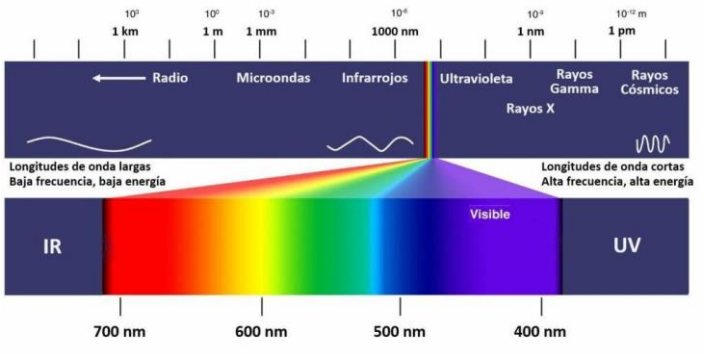


Figura 8. Espectro de luz Visible.
Tomado de (*Espectro Luz Visible*, n.d.)

Históricamente el capturar una imagen ha ido evolucionando desde el siglo XIX, uno de los más importantes hechos fue la invención de la cámara oscura, la cual inició la fotografía. Actualmente las cámaras digitales cuentan con varios sensores que les permiten no solo capturar una imagen estática sino una serie de imágenes que se repiten con una frecuencia establecida lo cual dan un efecto de movimiento(video). Tecnologías como CDD y CMOS son fundamentales para lograr una alta tasa de FPS (Recuadros por segundo). (García, 2002)

La forma en que un computador interpreta una imagen es a manera de matriz bidimensional, dentro de la cual cada posición contiene combinaciones de los tres colores primarios. Las codificaciones de los colores primarios se representan por medio de RGB (Red, Green, Blue).

Figura 9. Colores base de una imagen.
Tomado de (Peyre, 2010)

La gran mayoría de imágenes se encuentran representadas en dimensión por una proporción 4x3, siendo el primero valor el que representa al ancho de la imagen y el segundo el alto. Existen ciertos estándares en imágenes tales como: 640x480, 1024x768, etc. En los últimos años con el avance de la tecnología la definición de las cámaras ha aumentado tanto que podemos tener relaciones de aspecto de hasta 16x9, como ejemplo tenemos las imágenes HD 1920x1080. Para los sistemas computacionales se opta por trabajar con bajas definiciones tales como 640x480 ya que no se necesita tanto detalle y el procesamiento de la imagen sería demasiado pesado. (García, 2002)

Un término comúnmente utilizado dentro de las imágenes es el “Pixel” el cual está compuesto por tres valores de ocho bits, estos valores perteneces a los colores primarios y se encuentran expresados como un número que va del 0-255.

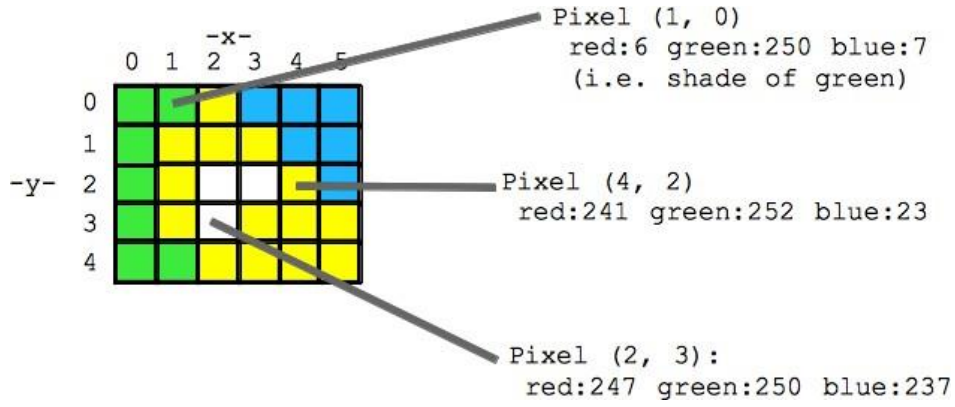


Figura 10. Estructura de un píxel dentro de una imagen RGB.
Tomado de (*Introduction to Digital Images*, n.d.)

2.3.2 Detección Facial

Un Sistema de detección facial es aquel que por medio de algoritmos determina si existe o no un rostro humano dentro de una imagen. Este procedimiento es principalmente utilizado en el área de seguridad, para en primera instancia detectar si hay una persona y después proceder al reconocimiento de esa parte de la imagen y determinar si es un usuario o no.

Uno de los métodos más populares para la detección facial es “Haar-Cascade”, un método planteado por Paul Viola y Michael Jones, el cual realiza la detección de varios rostros en una imagen con una alta velocidad de detección de objetos. Este método trabaja a nivel de pixeles, evaluado por medio de ventanas de 24x24 una serie de rectángulos usados en los algoritmos básicos de “Haar”. Estos rectángulos pueden ser rápidamente interpretados por el computador y nos permite representar a la imagen, la cual llamaremos “imagen integral”.

Después este pasa a una etapa llamada “Aprendiendo funciones de clasificación” la cual es una variación de AdaBoost, dentro de la cual la imagen integral se dividirá en ventanas más grandes y se procesará la información de cada bloque, por medio de un arreglo de seis patrones de rectángulos se determinará si dentro de cada ventana existe un rostro. (Viola y Jones, 2001)

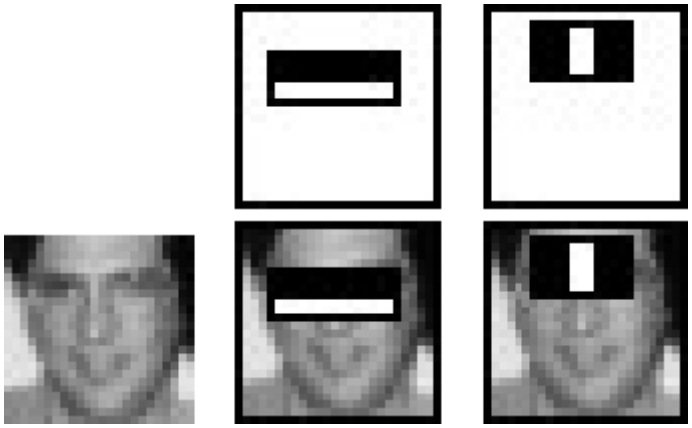


Figura 11. Clasificadores de imagen en un rostro.
Tomado de (Viola y Jones, 2001)

A esta ventana se la llamará clasificadores, los cuales devolverán valores positivos o negativos dependiendo de si se ha encontrado un rostro, siempre existirán errores por lo que se seleccionan los que tengan el mínimo error posible. El último clasificador es una suma ponderada de los clasificadores débiles, que individualmente no pueden clasificar la imagen, pero al unirlos forman un clasificador fuerte. Esta investigación asegura tener un 95% de precisión al encontrar rostros, ya que el modelo final cuenta con 6000 características entrenadas en modelo. (Viola y Jones, 2001)

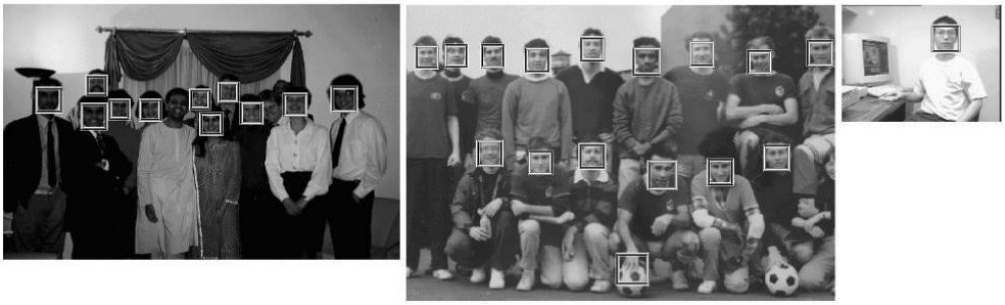


Figura 12. Detección múltiple de rostros.
Tomado de (Viola y Jones, 2001)

Rein-Lien Hsu(2002), en su artículo “Face Detection in color Images” plantea una detección facial basado en un algoritmo, el cual evalúa la variación de los tonos de los colores. Basados en técnicas de compensación de luz y transformaciones de colore no lineales encuentran posibles rostros candidatos, después se aplica un filtro para encontrar patrones de ojos y boca, determinando finalmente si existe un rostro.

Sivaram en su artículo “DETECTION OF ACCURATE FACIAL DETECTION USING HYBRID DEEP CONVOLUTIONAL RECURRENT NEURAL NETWORK”, propone un método de detección facial basado en redes neuronales el cual toma la forma del rostro y la ingresa a una red neuronal convolucional (CNN) para obtener una estimación de la apariencia, luego se procede a comparar los segmentos de imagen, con estimaciones de rostros dentro de una base. Sivaram asegura que la búsqueda por estimaciones es mucho más exacta ya que el rostro tiene rasgos característicos y eso hace mucho más factible la detección facial. (Sivaram et al., 2019)

2.3.3 Depth Image

Una imagen de profundidad, también conocido como mapa de profundidad, es una imagen la cual contiene valores de profundidad(distancia). Estas distancias pueden ser relativas si se trabaja solo procesando una imagen o más exactas al utilizar cámaras del tipo estereoscópicas o de profundidad (RGB-D). Dentro de esta imagen comúnmente representadas a blanco y negro, cada pixel contiene un valor de distancia ayudando al computador a interpretar la forma, textura y ayudando a realizar la reconstrucción en 3D. (García, 2002)

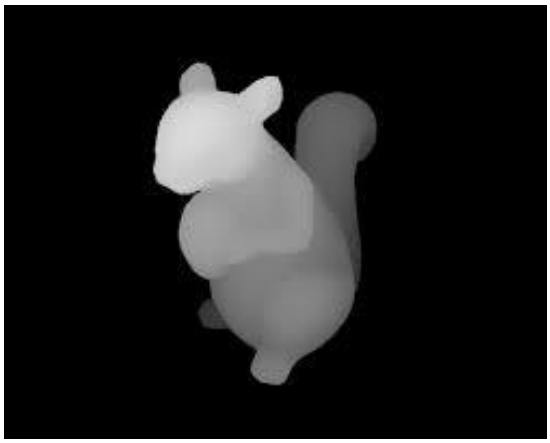


Figura 13. Imagen de profundidad de un juguete de ardilla.
Tomado de (Tjaden et al., 2019)

Las imágenes de profundidad o mapas de profundidad no son muy exactas, trabajar con ellos puede ingresar errores al procesamiento de imágenes, se debe utilizar esta imagen como referencia y determinar el porcentaje de error que introduce a la imagen original. Los mapas de profundidad pueden ser usado para encontrar objetos, ya que con su baja definición es más fácil su procesamiento y se pueden realizar comparativas con objetos cuyos mapas de profundidad son conocidos. (Malik et al., 2012)

2.3.4 Sensores infrarrojos de distancia

Un sensor infrarrojo de distancia consiste en la transmisión de un haz de luz infrarroja la cual choca con los objetos y mediante un sensor infrarrojo se determina el tiempo en que cada punto tarda en regresar, este tiempo depende de la frecuencia a la que el láser sea disparado, dándonos como resultado una distancia aproximada del objeto en el que rebotó el láser. (Bates, 2013)

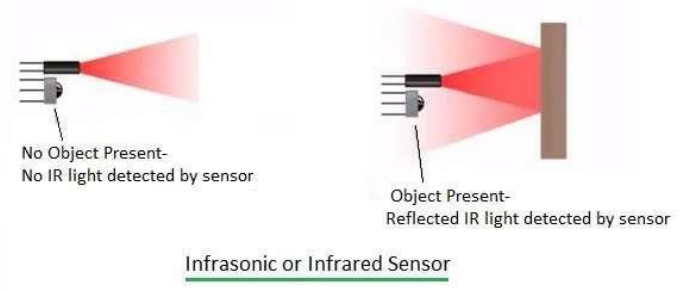


Figura 14. Funcionamiento de un sensor infrarrojo de distancia.
Tomado de (*Home of RF and Wireless Vendors and Resources*, n.d.)

En las Cámaras RGB-D se encuentra una matriz la cual contiene una gran cantidad de receptores, los cuales cada uno devuelve una distancia aproximada. Estas distancias se pueden expresar como una nube de puntos.

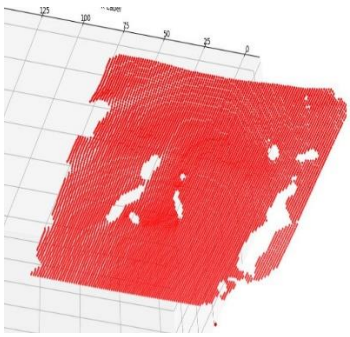


Figura 15. Rostro humano reconstruido con Orbbec Astra Pro

2.3.5 Cámaras RGB-D

Dentro de los equipos usados para el proceso de detección del rostro existe una gran variedad de cámaras que nos pueden ayudar con gran detalle, pero con un problema presente en la mayoría, se necesita una iluminación adecuada para cada cámara por lo que en ambientes de poca luz estas tienden a fallar. Para esto se ha decidido utilizar cámaras del tipo RGB-D.

Una cámara RGB-D es aquella que combina una imagen RGB con mecanismos de detección de profundidad, y como resultado tenemos los valores de los pixeles con su respectiva posición y una distancia. Un ejemplo de estas es Kinect.

Microsoft Kinect

Figura 16. Microsoft Kinect V2.
Tomado de (Asenjo, n.d.)

Asus Xtion Pro Live

Figura 17. Asus Xtion Camera.
Tomado de (ASUS, n.d.)

Orbbec Astra Pro

Figura 18. Orbbec Asta Pro.
Tomado de (ORBBEC, n.d.)

Intel Real Sense

D415

Figura 19. Intel Reals Sense Series D415.
Tomado de (Intel, n.d.)

La tabla 1, muestra una comparación entre las cámaras mostradas anteriormente, esta comparación se realiza teniendo en cuenta las siguientes características:

- Video RGB
- Video Profundidad
- Entrada de audio
- Sensibilidad.

Como podemos observar la cámara más optima resulta la Intel Realsense, por motivos ajenos, la cámara no pudo ser usada y se opto por la segunda mejor opción la Orbeec Astra.

Características	Microsoft Kinect V2	Orbbec Astra Pro	Asus Xtion Pro Live	Intel Real Sense D415
Video RGB	640x480px 30FPS.	1280x720px 30FPS	640x480px 30FPS	1920 × 1080
Video Profundidad	640x480px 30FPS.	640x480px 30FPS	320x240px 60FPS	1280 × 720
Entrada Audio	Micrófono multiarreglo	2 micrófonos	2 micrófonos	no
Sensibilidad	1,2-3,5 m	0.6m – 8m	0.8m - 3.5m	~0.16m - 10m
Interfaz	USB 2.0	USB 2.0	USB2.0/ 3.0	USB 3.0
Precio	170 USD	150 USD	900 USD	180 USD

Tabla 1. Comparativa de cámaras RGB-D

2.3.6 Point Cloud

Una nube de puntos es un conjunto de coordenadas tridimensionales, las cuales representan la parte externa de un objeto, en otras palabras, su superficie. Estas nubes de puntos pueden ser obtenidas de diferentes maneras, la más común es por medio de escáneres laser. Dependiendo de la precisión del equipo la nube de puntos puede tener mayor o menor definición, estas nubes de puntos nos ayudan a tener una noción en 3D del objeto que ha sido escaneado y por medio de técnicas de mallado podemos reconstruir el objeto. Cada punto de la nube se encuentra compuesto por las coordenadas X, Y y Z. (Rusinkiewicz y Levoy, 2000)

Las nubes de puntos son un método de mostrar las formas de los objetos de manera primitiva, estas nubes son utilizadas para en tratamiento de imágenes, puesto que

el trabajar con superficies resulta muy complicado. Marc Levoy propone un método por el cual las superficies representadas de manera geométrica se conviertan en nubes de puntos, las cuales son más fáciles y rápidas de procesar. (Levoy y Whitted, 1985)

Dentro de sus usos más comunes se presentas las reconstrucciones topográficas, pues estas se realizan por medio de mapas de profundidad, los cuales son transformados en una nube de puntos que se representará a manera de superficies. Los escáneres laser también brindan una alta precisión al generar nubes de puntos, siempre se debe remover datos erróneos de la nube tales como ruido y puntos que no tengan relación. En la siguiente imagen se presenta un análisis de superficie con una cámara infrarroja la cual obtiene aproximadamente cuatro millones de datos, estos serán filtrados y se realizara un proceso de reconstrucción para realizar una superficie topográfica. (Smith et al., 2011)

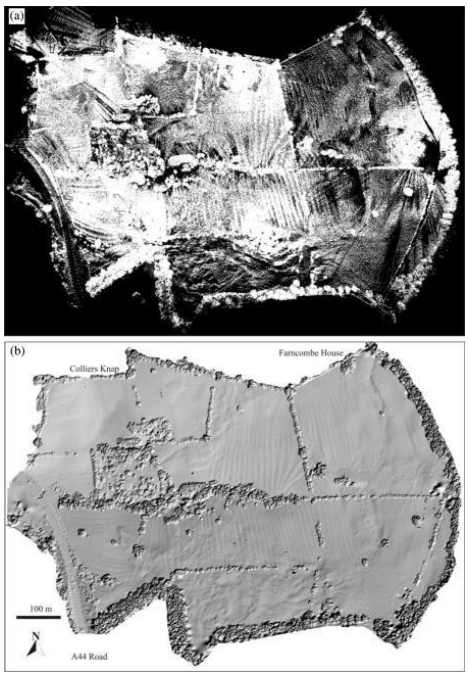


Figure 20. a) Nube de puntos de un terreno, b) reconstrucción topográfica de la superficie Tomada de (Smith et al., 2011)

2.3.7 Reconstrucción de superficies

La triangulación de superficie consiste en crear un área llena de triángulos, los cuales en sus vértices se conectarán por los puntos del mapa de profundidad. Este método pertenece a la generación de polígonos para superficies implícitas, el parámetro fundamental de la reconstrucción es el determinar cuáles son los puntos más cercanos al mismo y no deben ser necesariamente perpendiculares, conectándolos y formando figuras triangulares. (Hartmann, 2003)

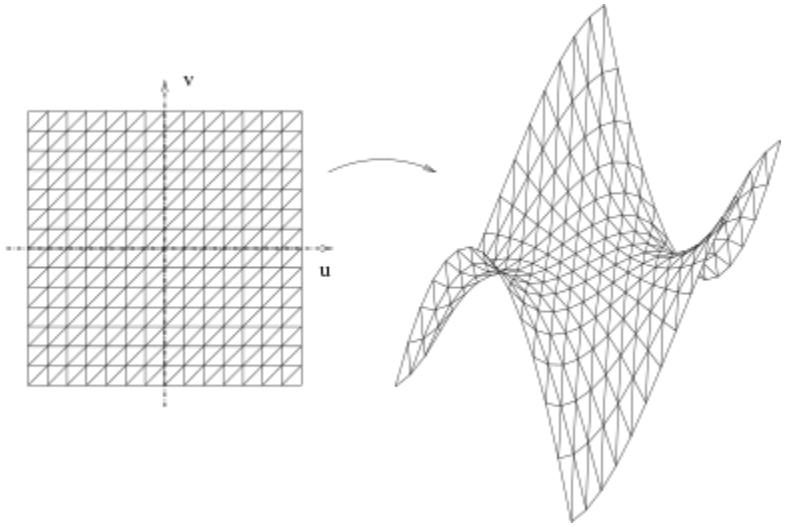


Figura 21. Superficie aplicada triangulación.
Tomada de (Hartmann, 2003)

Un problema muy común de este tipo de reconstrucción es que no realiza ninguna corrección de errores por lo que, si la nube de puntos tiene mucho ruido, se presentaran muchos picos dentro de la gráfica. Para evitar estos inconvenientes se necesita previamente una etapa de filtrado o usar otro método para graficar.

Marton Zoltan(2009), propone un método para las superficies cuyas nubes de puntos tengan gran cantidad de ruido, usando el mismo método de las propiedades geométricas genera una superficie triangular tan eficiente que puede ejecutarse en tiempo real. Al aplicar variables para corregir su densidad se obtienen superficies suavizadas y se corrigen errores de ruido que pueda tener la reconstrucción, este

método está diseñado para reconstrucciones tanto de interiores como exteriores asegura el autor.

Uno de los métodos más comunes es la triangulación de Delaunay, la cual es una derivada de la construcción de Voronoi. Delaunay puede ser construida a partir de coordenadas simples en dos dimensiones, la cual consiste en conectar a un punto con otros más cercanos de la nube de puntos, estos forman triángulos con cada tres puntos que se conectan. La principal diferencia con Voronoi es que esta reconstrucción al utilizar una célula y esta tiene una forma poliédrica, en muchas funciones para mejorar el detalle de la imagen generan primero una triangulación Delaunay para luego filtrarla y crear una nueva superficie con Voronoi.(Amenta et al., 1998)

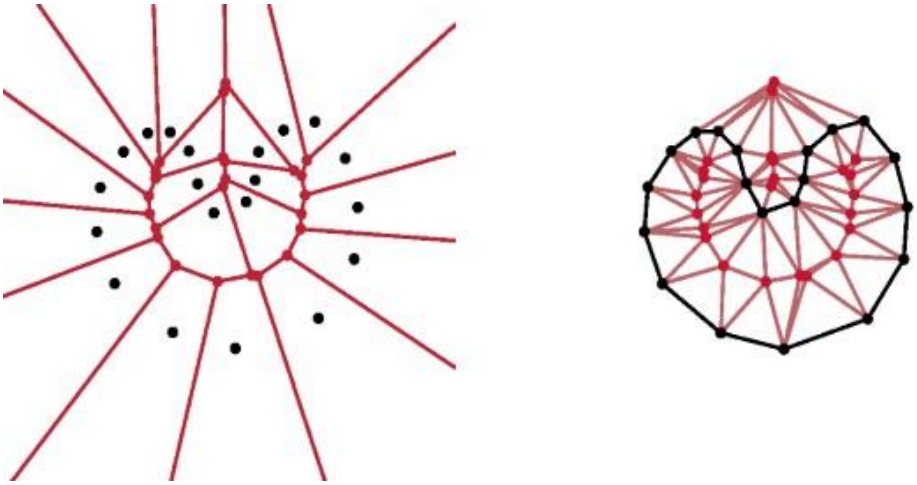


Figura 22. Creación de Polígonos por medio de Delaunay.
Tomado de (Amenta et al., 1998)

2.3.8 Filtros

Estos filtros para tratamiento de imagen nos ayudan a suavizar bordes y eliminar superficies muy planas dando un efecto de suavizado a la imagen, los filtros más comunes para el suavizado de una imagen son llamados LPF (low-pas filter), Este filtro representa a las imágenes como frecuencias y se encarga de eliminar y corregir aquellas que estén fuera de rangos muy elevados. citar

Un filtro muy común de los LPF es “Gaussian Blur” el cual suaviza a la imagen siguiendo una distribución del tipo gaussiano, esta técnica es comúnmente utilizada en el preprocesamiento de imágenes para algoritmos de visión artificial para dos dimensiones ya que ayuda implementar imágenes a diferentes escalas por el método de convolución. (Getreuer, 2013)

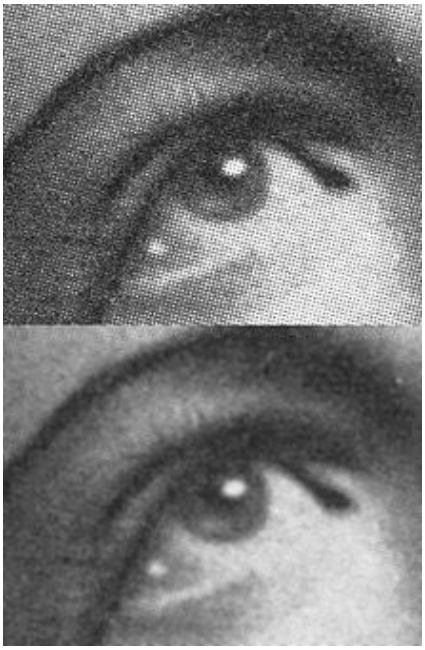


Figura 23. Suavizado de imagen mediante Gaussian Blur.
Tomado de (Getreuer, 2013).

El filtro gaussiano es un filtro de dos dimensiones con operaciones de convolución, el cual se encarga de remover el ruido de la imagen, usa ventanas(kernel) para representar
la campana característica. Su fórmula para dos dimensiones es: dentro de la cual sigma es la desviación estándar de la distribución. Dentro de esto se selecciona una ventana y se procesará los valores desde el centro hacia los extremos. (Davies, 2012)

$$\frac{1}{273}\begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

Figura 24. Kernel 5x5 con una derivación estándar de 273, Gaussian Blur.
Tomado de (Davies, 2012).

“Mean Filtering” es un método por el cual se reducen las variaciones de una imagen y evita cambios muy drásticos en la misma, dando un efecto de suavizado, este método se enfoca en crear una ventana den valores la cual recorre la matriz de la imagen generando un valor promedio desde el centro hacia los lados de la ventana. Suavizando por segmentos la imagen. (Davies, 2012)

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix}$$

Figura 25. Kernel 3x3 usando filtro de media.
Tomado de (Davies, 2012).

A diferencia con el filtro gaussiano en este método, toda la ventana obtiene un valor promedio. Por lo que, el suavizado se vuelve más uniforme y cómo podemos ver en la Figura 25, todos los valores de la imagen se dividen por la media aritmética dentro de la ventana. Esta ventana recorrerá toda la imagen.

3. Implementación

En el siguiente trabajo de titulación nos enfocaremos en el procesamiento de una imagen para obtener una reconstrucción en tres dimensiones del rostro por medio de abstracción, usando una cámara RGB-D. Se utilizó un método teórico experimental con el cual se probaron diferentes métodos de procesamiento de imagen.

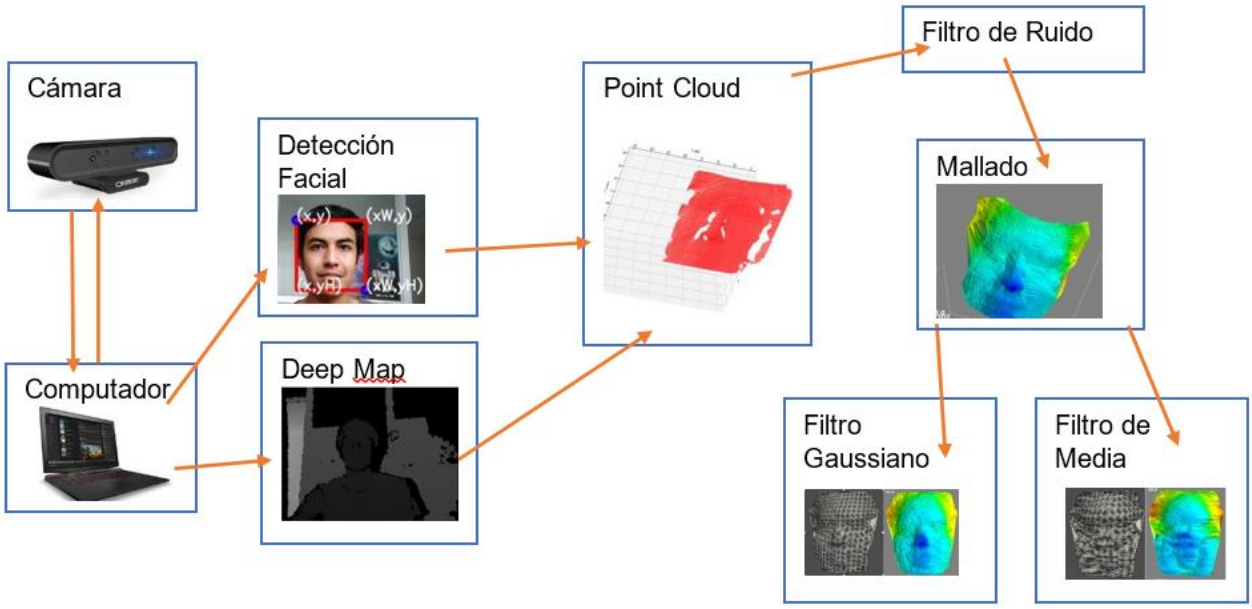


Figura 26. Diagrama de flujo de la reconstrucción del rostro.

3.1 Herramientas de Software

Para la implementación de la programación se usará Python, el cual es un lenguaje de programación que se caracteriza por su compatibilidad con otros sistemas, su fácil comprensión de los comandos y su amplia librería de paquetes que es continuamente actualizada y desarrollada por la comunidad.

Dentro de todos los entornos que Python ofrece, trabajaremos con Anaconda 3.7, la cual es una distribución de Python 3.7 que tiene muchos paquetes preinstalados, el proceso de instalación de Anaconda se encuentra en el (Anexo 1).

Como editor de texto para la elaboración del código se usará a PyCharm IDE, el cual es un entorno de desarrollo integrado creado por la empresa JetBrains. Este entorno

de desarrollo nos permite una interfaz gráfica completa y un manejo rápido de paquetes y dependencias junto con un Debuger gráfico.

3.2 Herramientas de Hardware

Para la reconstrucción en 3D se utilizará la cámara Orbecc Astra Pro por ser una de las que mejores características tiene, su precio accesible y su venta en distintos establecimientos la convierten en la opción más viable. Esta cámara tiene los siguientes requisitos:

- USB 2.0
- Windows 7/8/10

También cuenta con su propio paquete de comandos Astra SDK y tiene soporte con la librería OpenNI la cual nos permitirá realizar la conexión con Python.

El código será desarrollado en un computador con las siguientes características:

- Sistema operativo: Windows 10
- Marca: Lenovo
- Modelo: Ideapad Y700
- RAM: 16GB

3.3 Imagen de Profundidad

Como primer paso debemos tener instalado la librería de OpenCV, el paquete de OpenNI y Numpy (Anexo 2). La Interacción Natural Abierta “OpenNi” fue una organización sin fines de lucro la cual se encargaba de realizar código abierto y que tenía por objetivo mejorar la compatibilidad entre los lenguajes de programación con los dispositivos. Esta empresa fue comprada por Apple en el 2013. PrimeSense es uno de los grupos fundadores de OpenNi y todavía continúan con el trabajo que esta dejó atrás, con el nombre de OpenNI versión 2, entre sus trabajos PrimeSense fue una de las compañías que formó parte del desarrollo de Kinect para la consola Xbox

360. (PrimeSense, 2013)

Numpy por otro lado es un paquete fundamental de Python para realizar cálculos, este paquete contiene funciones que nos ayudan a creer arreglo multidimensionales y matrices, así como los caculos entre ellos. Con este paquete manipularemos la información provista por la cámara para llevarla a arreglos, los cuales nos ayudarán al filtrado y procesamiento de la imagen. (The SciPy community, 2020)

OpenCV es una librería abierta de visión artificial y machine learning, esta librería fue creada para que todos los proyectos de visión artificial tengan una estructura en común. OpenCV cuenta con alrededor de 2500 algoritmos optimizados tanto de visión artificial como machine learning. Entre sus aplicaciones más comunes tenemos, la detección e identificación tanto de objetos como rostros, el traqueo de movimientos, trabajar con cámaras estereoscópicas y nubes de puntos. Esta librería es compatible con lenguajes como C++, Java, Python y MATLAB, así como sistemas operativos como Windows, Linux y MAC OS. (OpenCV team, 2019)

Para realizar la conexión debemos descargar el kit de desarrollo de software (SDK) de OpenNI, directamente de la página de Orbbec Astra: https://orbbec3d.com/develop/ . El código para la conexión de la cámara con Python, se basará en las instrucciones de códigos (*OpenNI2 Coding Instructions*, n.d.), que vienen dentro del paquete de desarrollo como documentación de usuarios en de la carpeta Windows.

Para probar el correcto funcionamiento de la cámara se realiza la implementación del primero código, el cual mostrará en pantalla que se obtienen la imagen de profundidad por parte de la cámara.

Un error muy común de ejecución que se da, es el no determinar la ubicación de la carpeta que contiene los archivos de inicialización de OpenNI para la inicialización, a continuación, se mostrara dirección del archivo de inicialización en entornos Windows de 64 bits tomando como raíz la carpeta del SDK.

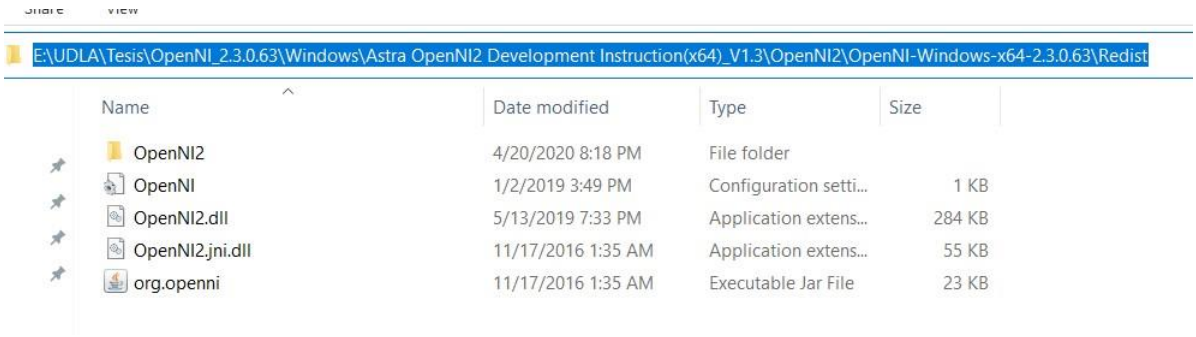


Figura 27. Ubicación de la carpeta Redist con archivos de OpenNI.

Tomando en cuenta los datos de la Tabla 1. Podemos observar que la resolución de la imagen de profundidad que encontraremos tiene una resolución de 640x480, y que su velocidad es muestreo es de 30 muestras por segundo. Dentro de las instrucciones de código de la librería OpenNI para la cámara Orbbec, observamos que cuenta con dos unidades de medida dentro de la que nos puede mostrar las medidas de profundidad, milímetros y micrómetros. Al trabajar con medidas tan pequeñas se puede tener escalas demasiado grandes complicando los cálculos, para mejor apreciación de los resultados se opta por trabajar con milímetros y evitar las transformaciones de escala.

Figura 28. a) Mapa de profundidad con escala micrométrica a una distancia de 60 centímetros, b) Mapa de profundidad con escala milimétrica a una distancia de 60 centímetros.

En las imágenes podemos ver que, al imprimir la imagen en diferentes escalas, la

primera de ellas en micrómetros se puede ver la silueta de la persona, mientras que al imprimirla en milímetros no podemos observar algún cambio. Pese a que no haya imagen alguna dentro de la imagen de la derecha, sus datos si encuentran ahí, la escala en la que se gráfica no es visible, esto se debe a la gama de colores que cada una usa para realizar la gráfica, si bien las distancias en ambos casos son las mismas, los valores de blanco y negro son diferentes.

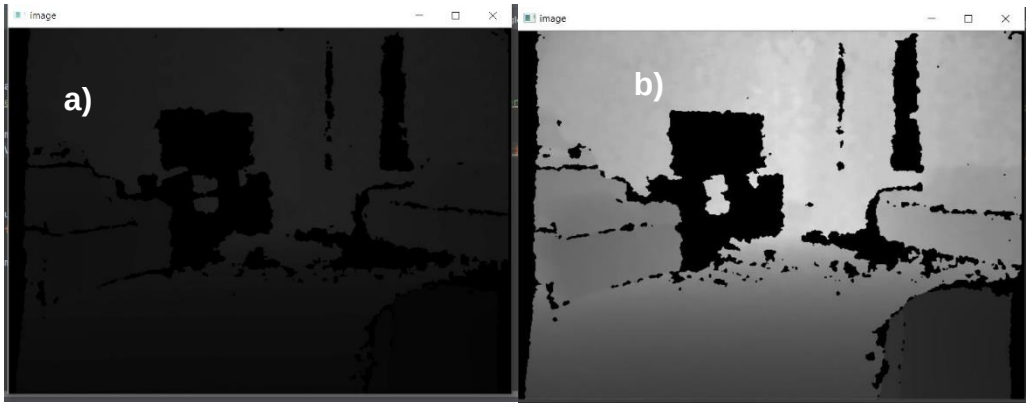


Figura 29. a) Mapa de profundidad con escala micrométrica a una distancia de 5 metros,

b) Mapa de profundidad con escala milimétrica a una distancia de 5 metros.

Tras obtener los datos de profundidad, por medio de una funciona llamada “convert-deep-to-world” podremos obtener la nube de puntos, esta función tiene como entrada una matriz de coordenadas X, Y dentro de los cuales evaluará el color en estas coordenadas devolviendo valores de posición en X, Y con una nueva referencia (centro de la imagen) y un valor de distancia Z.

3.4 Detección Facial

Para realizar la reconstrucción del rostro, primero debemos detectar si existe un rostro en la imagen. OpenCV en su documentación cuenta con un tutorial específico para la detección de rostros con el algoritmo de Haar-Cascade de Paul Viola, la ventaja de usar este código en particular es el modelo entrenado con más de 5000 rostros diferentes, lo cual brinda al algoritmo mayor precisión y dibujará un recuadro

sobre el o los rostros que encuentre, por medio de la herramienta de OpenCV.

Por medio de Haar-Cascade encontramos puntos de interés, las coordenadas X, Y desde las cuales se va a calcular el área del rostro y las distancias “h” y “w” que representan a la altura y ancho de la imagen respectivamente.

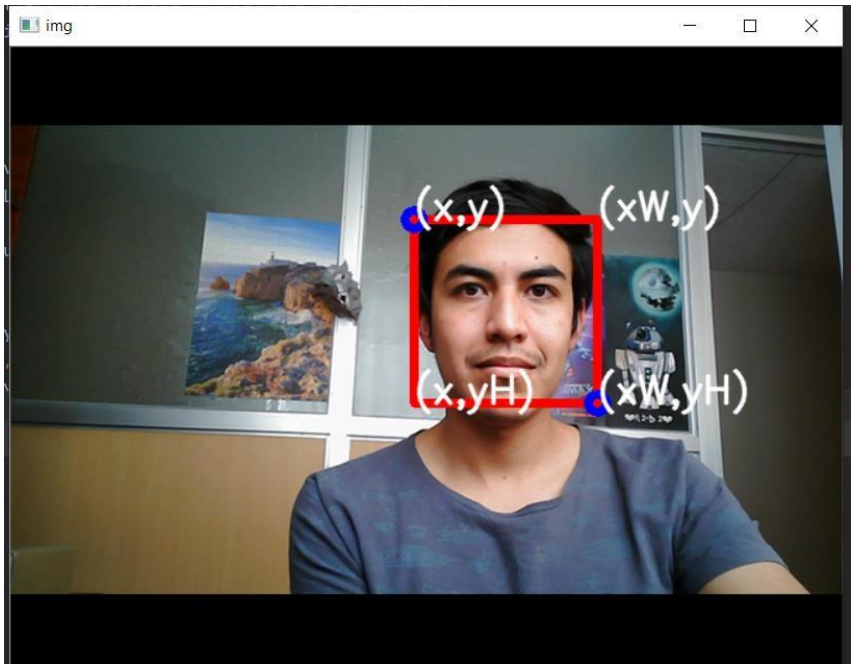


Figura 30. Detección del rostro por medio de Haar-Cascade y determinar puntos X, Y.

En la Figura 28, podemos observar la posición de los puntos que formarán la matriz de coordenadas para realizar el proceso de abstracción y enviar dicha matriz al código de transformación de coordenadas hacia una nube de puntos. Para graficar estos puntos usamos las herramientas “cv2”, librería de OpenCV, la cual nos permite realizar gráficos sobre la imagen.

En la función “detectMultiScale” podremos ajustar el parámetro que especifica cuánto se reduce el tamaño de la imagen en cada escala y la cantidad mínima de vecinos que debe tener cada rectángulo candidato para detectar si se encuentra una imagen. Mientras los valores de vecindades del rectángulo sean más altas, el rostro deberá tener una mayor cantidad de detalle para ser detectado, caso contrario si disminuimos este valor, encontraremos falsos positivos de rostros. Al igual que si la

escala del rostro es muy alta detectará más de una vez el mismo rostro. La documentación de OpenCV recomienda que se utilice una escala de 1.1 y un valor de vecindades de 3.

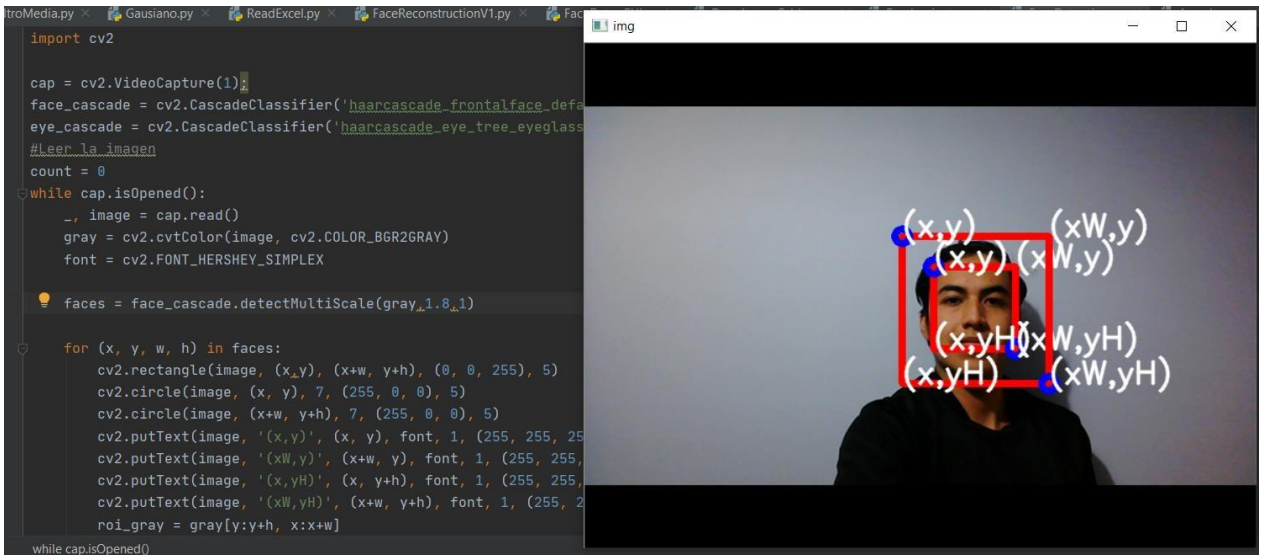


Figura 31. Detección facial con parámetros de escala 1.8 y de vecindades 1.

3.5 Nube de puntos

Para crear esta nube de puntos debemos trabajar con los datos de posición de la detección facial y generaremos una matriz, la cual representará a nuestro rostro. Llenaremos esta matriz con las posiciones de los pixeles que se encuentran dentro del rectángulo que representa al rostro, esta será la matriz que ingresemos a la función que nos devolverá los puntos de distancia esta función la podemos encontrar en la documentación de OpenNI para la cámara Orbbec Astra Pro.

(X, Y)					
	(100,100)	(101,100)	(102,100)	(103,100)	(104,100)
	(100,101)	(101,101)	(102,101)	(103,101)	(104,101)
	(100,102)	(101,102)	(102,102)	(103,102)	(104,102)
	(100,103)	(101,103)	(102,103)	(103,103)	(104,103)
	(100,104)	(101,104)	(102,104)	(103,104)	(104,104)

Tabla 2. Ejemplo de una matriz cuyas coordenadas iniciales son (X, Y) = (100,100) respectivamente.

“Matplot lib” es una librería que nos ayuda a crear gráficas en dos y tres dimensiones, esta librería se encuentra basada en Numpy(Hunter et al., 2019). Para obtener la nube de puntos uniremos ambos códigos, la detección facial y la trasformación de coordenadas.

La matriz PointCloud que contiene las Coordenadas X, Y, Z será dividida en tres arreglos unidimensionales que facilitarán el procesamiento de los datos, puesto que la mayoría de los cambios se realizarán en el arreglo Z.

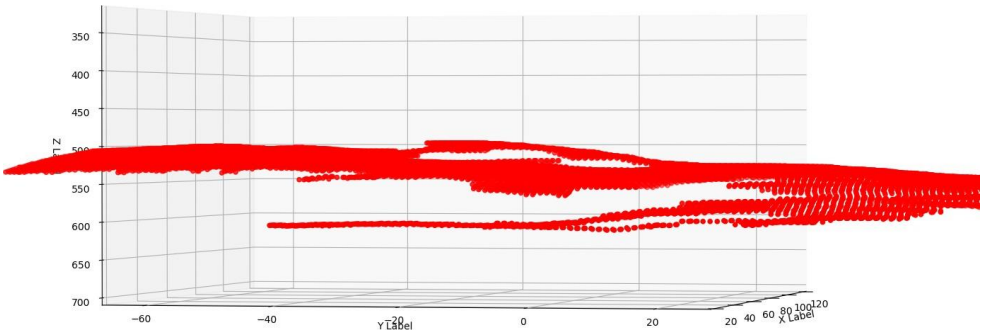


Figura 32. Nube de puntos original.

A primera vista en la imagen no podemos observar la forma de un rostro, al fijarnos en las escalas utilizadas para la gráfica, observamos que tanto en X, Y tenemos rangos de 20 medidas formado aproximadamente 120 muestras. En el eje Z tenemos rangos de 50 medidas, estos rangos evitan que se pueda apreciar la forma correcta del rostro, esto se debe a que la cámara toma medidas en rangos de (60cm-8m). Si la cámara encuentra un punto muy cerca o muy alejado del rostro, los cuales son considerados ruido, se graficará y produce una imagen plana como la mostrada en la Figura 30.

Para realizar el filtrado de la imagen, tomaremos un concepto básico de la estadística. La campana de Gauss es una distribución normal, la cual representa la probabilidad en la que puede ocurrir un evento, la deviación estándar es el valor que representa que tan dispersos se encuentran los datos del centro o también llamado promedio. La varianza por su parte nos muestra un promedio de cuanto cambia un valor con respecto a su siguiente valor. (Walker, 1931)

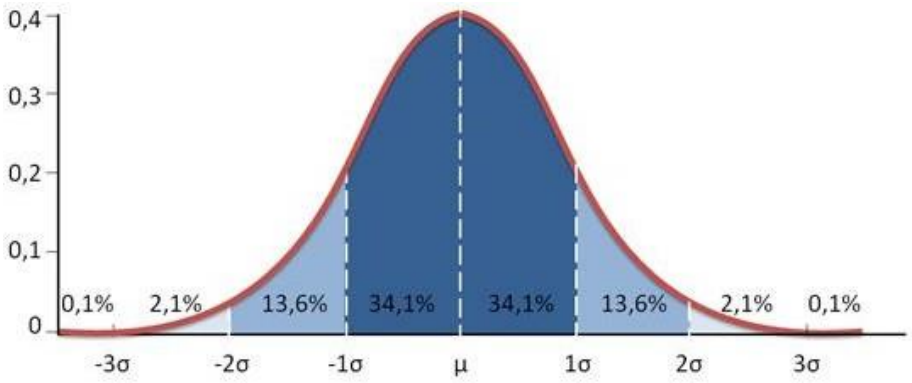


Figura 33. Campana de Gauss. Tomado de (*Campana de Gauss*, n.d.)

Como podemos ver en la imagen, dentro de una distribución normal, en 68.2% de los datos se encuentras dispersos a una distancia sigma, bajo este concepto para poder reducir la escala de la nube de puntos Figura 30, realizaremos un filtrado tomando en cuenta le promedio de los valores y determinando como valores útiles los que se encuentren dentro de la varianza. Una vez realizado este filtro realizaremos un offset en la imagen para que se pueda visualizar de mejor manera.

Las fórmulas que usaremos para los cálculos son las siguientes:

Media:

$$\bar{z} = \frac{\sum_1^n z}{n}$$

Z= Puntos en el eje Z obtenidos en la nube de puntos

N= Cantidad de puntos en el eje Z

Varianza:

$$S^2 = \frac{\sum_1^n (z_i - \bar{z})}{n-1}$$

Desviación Estándar:

$$\sigma = \sqrt{S^2}$$

Con los valores de media y desviación listos eliminaremos aquellos valores que se encuentren fuera de los rangos sobre el promedio más la desviación y datos que se encuentren por debajo del promedio menos la desviación. Debemos tomar en cuenta que si eliminamos uno de los datos de este arreglo debemos

eliminar sus datos de posición correspondiente de los arreglos X, Y para que no existan errores.

Para esto se utilizarán arreglos unidireccionales, los cuales representarán las tres variables, hay que tomar en cuenta que si eliminamos datos en uno de estos arreglos se debe eliminar sus datos correspondientes el resto de los arreglos.

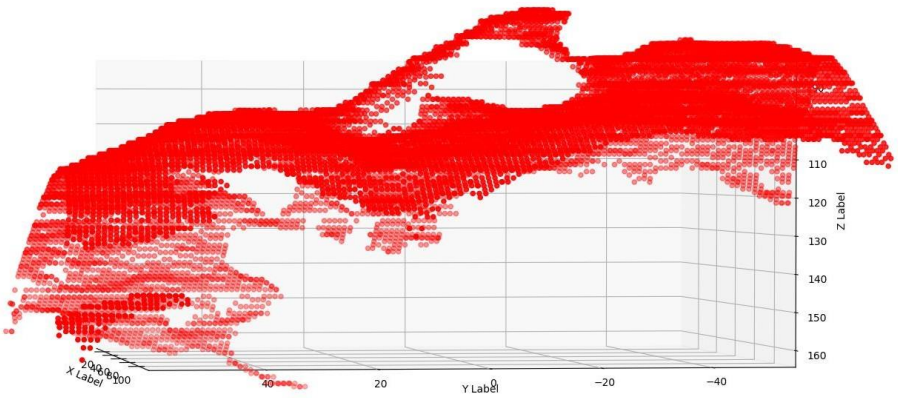


Figura 34. Nube de puntos filtrada (Perfil del Rostro).

3.6 Representación de la Imagen

Para tener una mejor visualización de la imagen se realiza una reconstrucción de la superficie del objeto, para poder llenar los espacios en blanco y mostrar a mayor detalle las facciones del rostro. Mayavi es una librería que permite el dibujar imágenes en tres dimensiones por medio de coordenadas, usada comúnmente en los mapas topográficos nos brinda una impresión por niveles la cual por defecto tiene capas de colores para representar diferentes alturas. (Enthought Inc, n.d.)

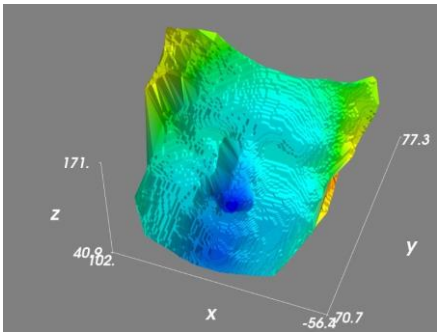


Figura 35. Reconstrucción facial por medio la librería Mayavi, con la función Points3D.

Como se puede ver en la imagen existen superficies de nivel como los mapas topográficos, esto se debe a la sensibilidad de la cámara ya que esta tomara medidas a cada milímetro por lo que se forman superficies en cada distancia de muestreo. Para obtener mayores detalles se usará la triangulación de Delaunay para crear la superficie por medio de la nube de puntos.

Como se detalló en el marco teórico una de las técnicas de mallado, la triangulación de Delaunay nos ayuda a crear superficies representadas por triángulos, lo que nos ayudará a tener una mejor representación del rostro y al momento de realizar el filtrado poder tener superficies curvas en la imagen.

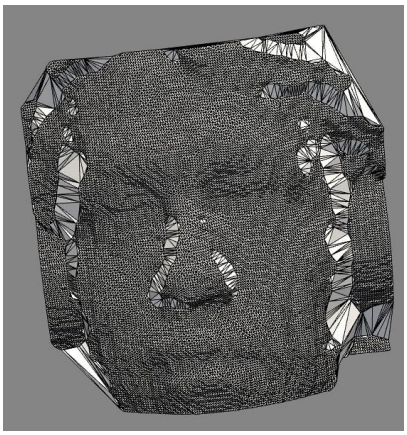


Figura 36. Rostro reconstruido por medio de triangulación Delaunay a una distancia aproximada de 60cm.

3.7 Filtros de Kernel multidimencional

Para suavizar las superficies de manera topográfica de la imagen, la cual se encuentra en milímetros, y dar valores intermedios entre estos puntos, se aplicará filtros de kernel como se menciona en el marco teórico, para esto nos ayudaremos de la librería Scipy(Instalación en el Anexo 2). Scipy es una librería de código abierto a la comunidad que continuamente se actualiza y tiene gran cantidad de algoritmos. Dentro de su lista de algoritmo utilizaremos dos principalmente:

- Gaussian Filter
- Generic Filter (Filtro de media)

Dentro de “Generic Filter” se ingresa una matriz la cual utilizará una ventana diferente que es ingresada por el usuario, para que los valores de la matriz puedan tomar valores intermedios y dar un efecto de suavizado a la imagen.

La principal diferencia entre los filtros usados son las unidades de medida que manejan y el tipo de filtro realizado, para el caso del filtro genérico acepta solo valores enteros, por su parte el filtro gaussiano acepta valores decimales ya que la variable a utilizarse es “sigma”. Sigma representa la desviación estándar que se usará.

En ambos casos mientras mayores sean los valores de la ventana de muestreo o la variable sigma, el suavizado de la imagen será más fuerte, hasta deformar o perder las propiedades originales de la misma.

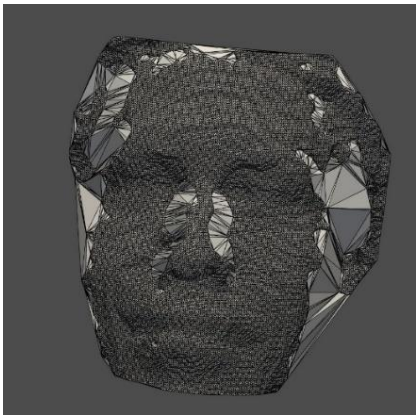


Figura 37. Reconstrucción del rostro usando un filtro gaussiano con un valor de sigma=0.4, a una distancia aproximada de 58cm.

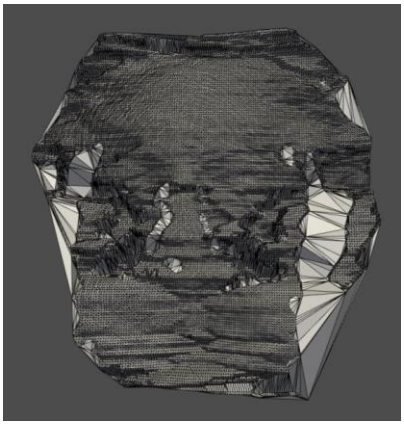


Figura 38. Reconstrucción del rostro un filtro de media con una ventana de kernel = 6.

Los valores mínimos que pueden tomar estos filtros son 0.1 para Gaussian y 1 para el filtro de media, al aplicar estos valores de filtro la ventana (tamaño de la matriz) toma el valor de 1x1 por lo que no se realizará filtrado alguno. Dentro del procesamiento las ventanas más pequeñas nos ofrecen un filtrado más rápido y eficiente pues los cálculos que realizan son menores y recorren la imagen en menor tiempo.

3.8 Interfaz de usuario

Para realizar la interacción con el usuario se presenta la interfaz detallada en la Figura 37, la cual toma el nombre usuario, se centra en la reconstrucción con los filtros de kernel gaussiado y filtro de media, permitiendo ver el impacto de los mismo sobre una misma captura de datos. Los datos recolectados se pueden guardar en archivos CSV, los cuales son archivos sencillos que se encargan de representar la información a manera de tablas.

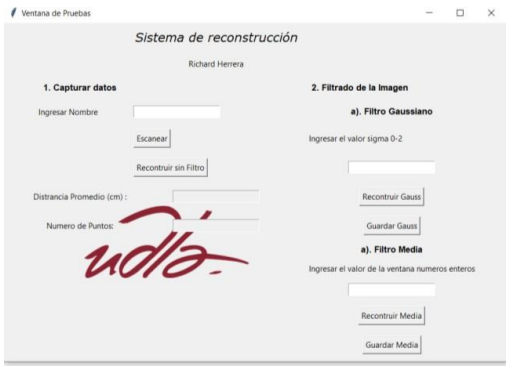


Figura 39. Interfaz de usuario.

Esta interfaz nos muestra las opciones de guardar las reconstrucciones realizadas con cada filtro y de modificar los parámetros de los filtros para revisar las imágenes de reconstrucción resultantes, también nos muestra la cantidad de puntos de la nube original y la distancia aproximada que la persona se encuentra de la cámara.

4. Análisis y resultados

Para realizar las pruebas se han planteado distintos escenarios, en los cuales se validarán distintos parámetros del prototipo implementado, así como las diferentes combinaciones que se pueden dar al realizar la recolección de datos y representación de imágenes en tres dimensiones.

4.1 Número de Puntos vs Distancia

Escenario 1:

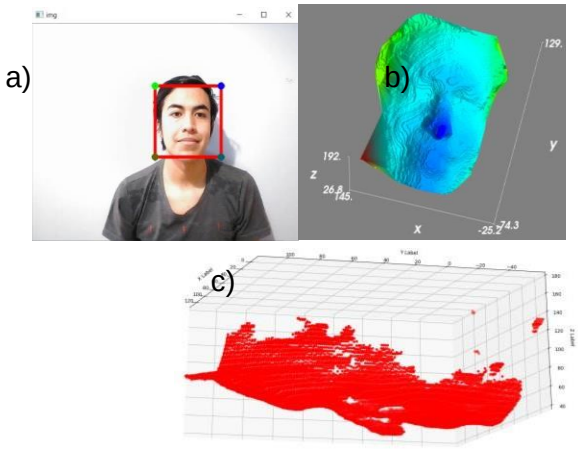


Figura 40. Reconstrucción del rostro a aproximadamente 60 cm con una pared al fondo. (a. Foto RGB, b. Reconstrucción superficie, c. Nube de Puntos).

Distancia promedio:63.22cm

Puntos originales:24649

Filtrado de puntos de ruido:17023

Escenario 2:

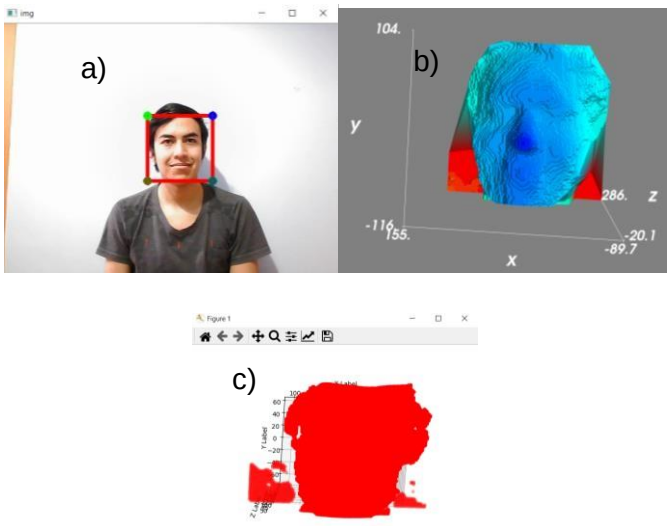


Figura 41. Reconstrucción del rostro a aproximadamente 80 cm con una pared al fondo. (a. Foto RGB, b. Reconstrucción superficie, c. Nube de Puntos).

Distancia promedio:83.609cm

Puntos originales:21025

Filtrado de puntos de ruido:14309

Escenario 3:

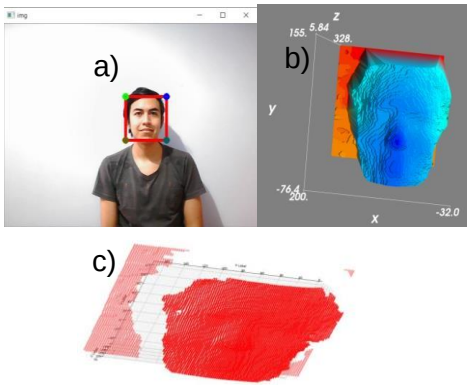


Figura 42. Reconstrucción del rostro a aproximadamente 100 cm con una pared al fondo. (a. Foto RGB, b. Reconstrucción superficie, c. Nube de Puntos).

Distancia promedio:98.97cm

Puntos originales:10201

Filtrado de puntos de ruido:7569

Escenario 4:

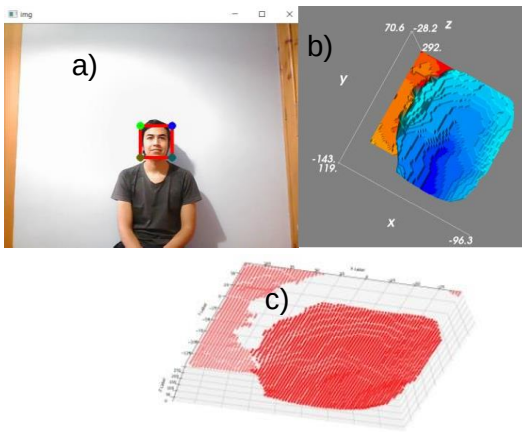


Figura 43. Reconstrucción del rostro a aproximadamente 150 cm con una pared al fondo. (a. Foto RGB, b. Reconstrucción superficie, c. Nube de Puntos)

Distancia promedio:162.36cm

Puntos originales:7946

Filtrar los puntos de ruido:4476

Escenario 5:

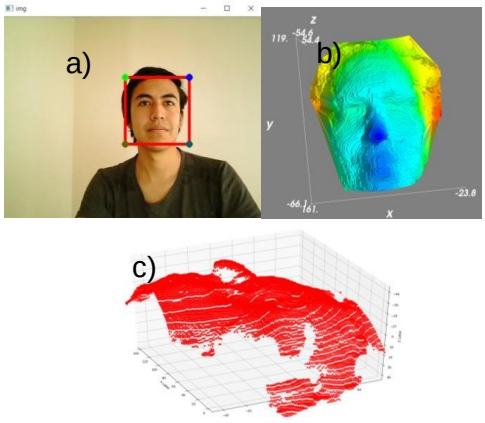


Figura 44. Reconstrucción del rostro a aproximadamente 60 cm con una pared al fondo. (a. Foto RGB, b. Reconstrucción superficie, c. Nube de Puntos).

Distancia promedio:61.71 cm

Puntos originales:24649

Filtrado de puntos de ruido:15889

Escenario 6:

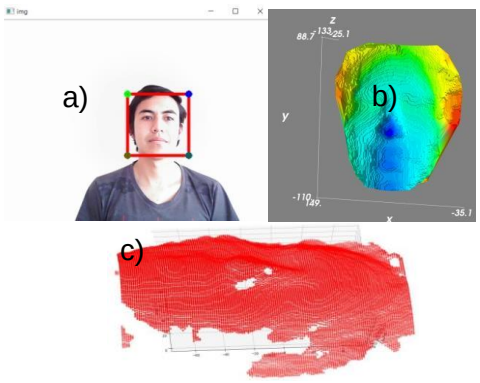


Figura 45. Reconstrucción del rostro a aproximadamente 80 cm sin pared de fondo. (a. Foto RGB, b. Reconstrucción superficie, c. Nube de Puntos).

Distancia promedio:82.88 cm

Puntos originales:14400

Filtrado de puntos de ruido:8414

Escenario 7:

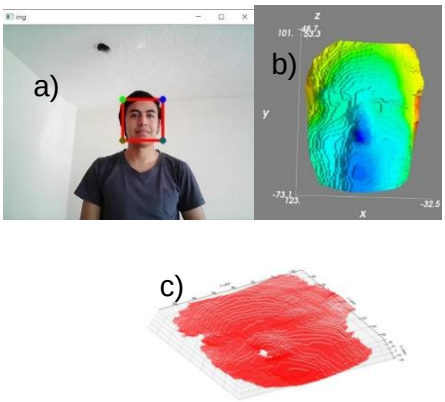


Figura 46. Reconstrucción del rostro a aproximadamente 100 cm sin pared de fondo. (a. Foto RGB, b. Reconstrucción superficie, c. Nube de Puntos).

Distancia promedio:98.91 cm

Puntos originales:10609

Filtrado de puntos de ruido:9604

Escenario 8:

Reconstrucción de rostro sin filtros de kernel a una distancia de 150cm sin pared de fondo cercana

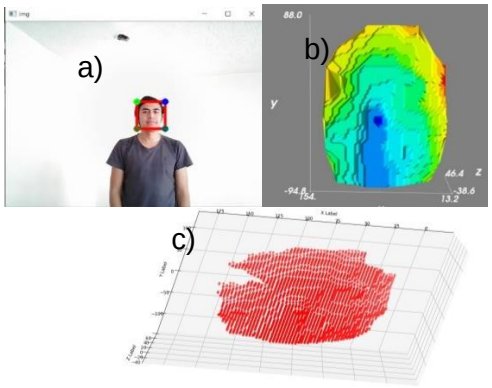


Figura 47. Reconstrucción del rostro a aproximadamente 150 cm sin pared de fondo. (a. Foto RGB, b. Reconstrucción superficie, c. Nube de Puntos).

Distancia promedio:149.76 cm

Puntos originales:7569

Filtrado de puntos de ruido:3721

En la Tabla 3, podemos observar un resumen de los puntos que conformaron la imagen en comparación con las distancias a las que fueron tomados. También tenemos las diferencias entre los puntos originales de la imagen.

	Fondo lejano		Fondo Cercano	
Distancia	Puntos Originales	Puntos Filtrados	Puntos Originales	Puntos Filtrados
60 cm	22649	15889	24469	17023
80 cm	14400	8414	21025	14309
100 cm	10609	9604	10201	7569
150 cm	7569	3721	7946	4476

Tabla 3. Resultados de distancias.

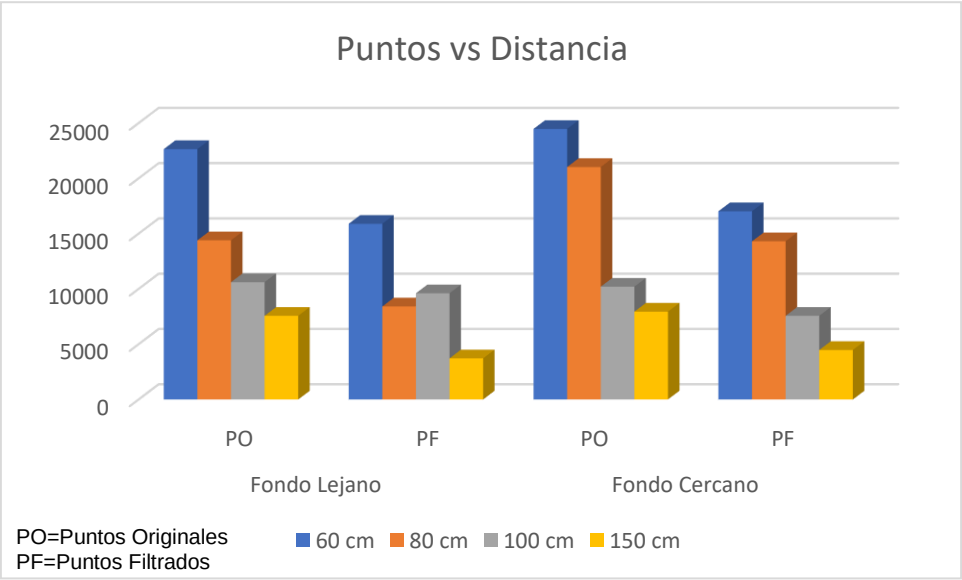


Figura 48. Histograma de puntos.

En los primeros cuatro escenarios notamos que la definición disminuye inversamente proporcional a la distancia por la cantidad de puntos tomados, y esto podemos darnos cuenta en la cantidad de puntos que se han recolectado del rostro. En las últimas distancias podemos observar que se nota parte de la pared, esto se debe a que al alejar la imagen cambia la varianza y esto hace que cada vez valores más cercanos al rostro aparezcan en la imagen.

En los siguientes cuatro escenarios podemos ver que si la persona se encuentra a distancia de un objeto en el fondo este puede ser filtrado, y que al igual que con las primeras pruebas la distancia es inversamente proporcional al detalle del rostro.

4.2 Aplicación de Filtros

En los siguientes casos vamos a probar los filtros de Kernel con las diferentes opciones de ventanas, para lo que nos centraremos en la capacidad de suavizado de la imagen sin perder rasgos característicos o deformación del rostro. Como se observó en las pruebas anteriores a menor distancia, hay mayor cantidad de puntos por lo que se usarán dos distancias:

- 60 cm
- 80 cm

Escenario 1.

4.3 Filtro Gaussiano 60cm

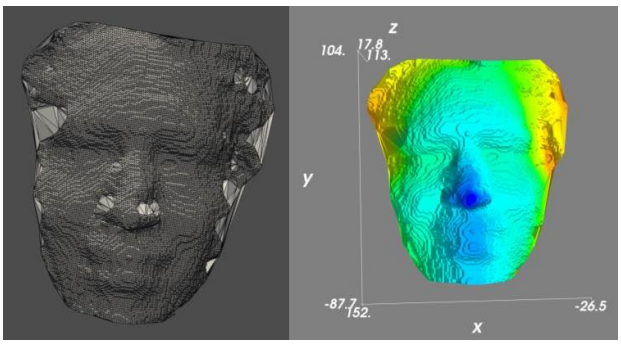


Figura 49. Reconstrucción del rostro a aproximadamente 60 cm (Sin filtro).

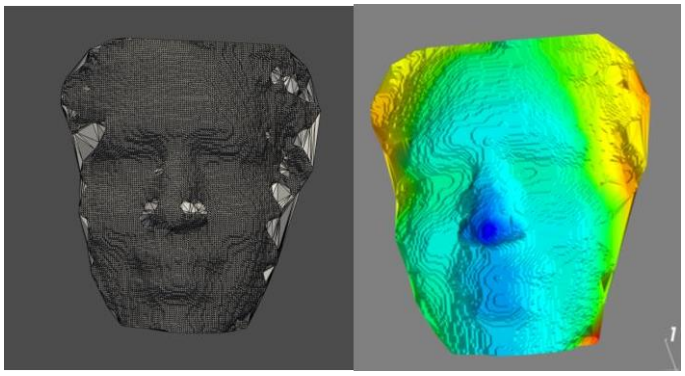


Figura 50. Reconstrucción del rostro a aproximadamente 60 cm con filtro gaussiano, sigma=0,3.

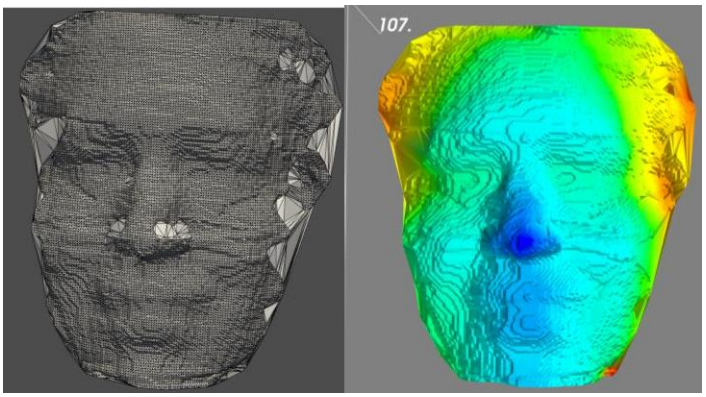


Figura 51. Reconstrucción del rostro a aproximadamente 60 cm con filtro gaussiano, sigma=0,4.

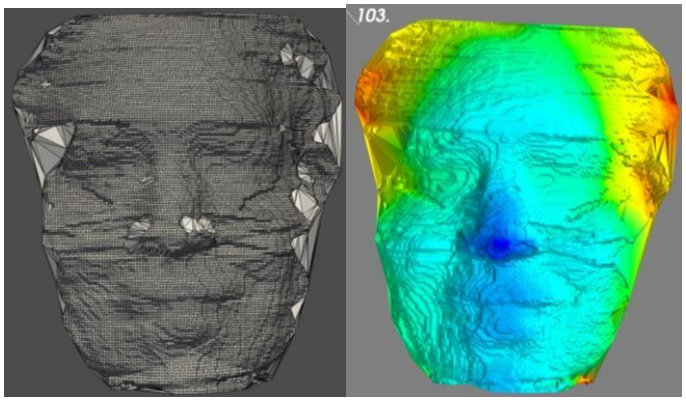


Figura 52. Reconstrucción del rostro a aproximadamente 60 cm con filtro gaussiano, sigma=0,5.

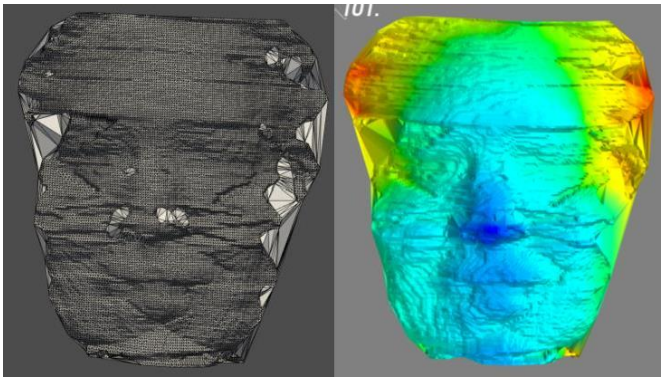


Figura 53. Reconstrucción del rostro a aproximadamente 60 cm con filtro gaussiano, sigma=0,6.

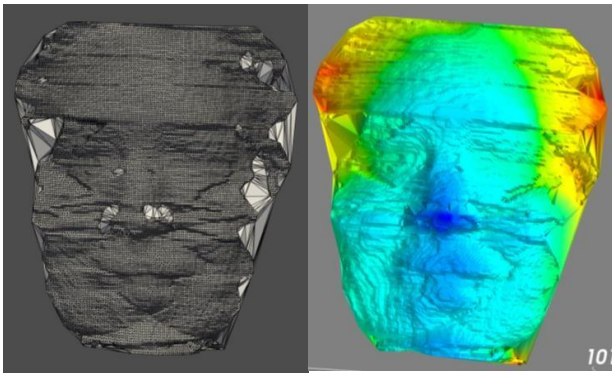


Figura 54. Reconstrucción del rostro a aproximadamente 60 cm con filtro gaussiano, sigma=0,7.

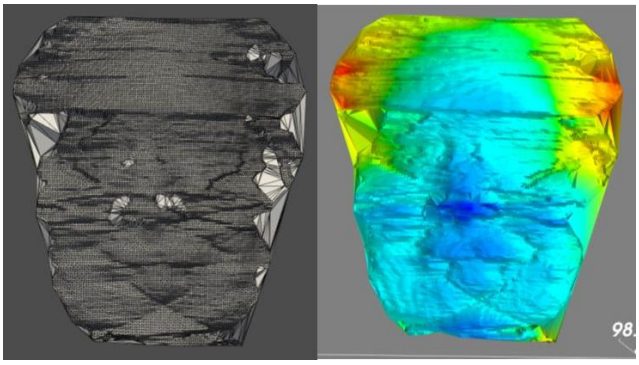


Figura 55. Reconstrucción del rostro a aproximadamente 60 cm con filtro gaussiano, sigma=0,8.

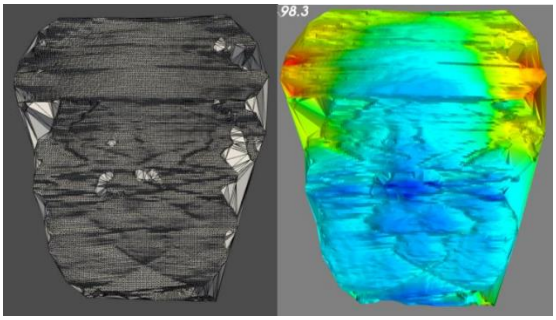


Figura 56. Reconstrucción del rostro a aproximadamente 60 cm con filtro gaussiano,
sigma=1.

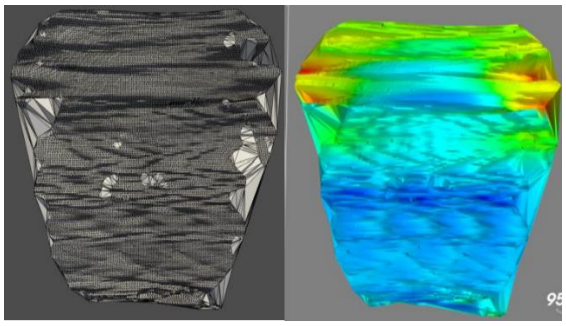


Figura 57. Reconstrucción del rostro a aproximadamente 60 cm con filtro gaussiano, sigma=2.

4.4 Filtro de Media a 60 cm

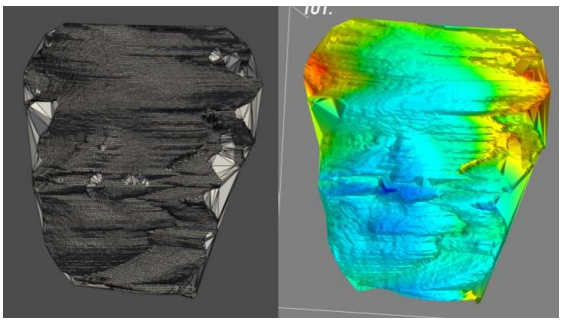


Figura 58. Reconstrucción del rostro a aproximadamente 60 cm con filtro de media, ventana=2.

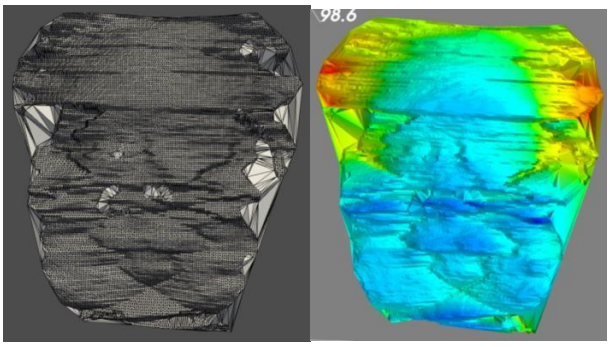


Figura 59. Reconstrucción del rostro a aproximadamente 60 cm con filtro de media,
ventana=3.

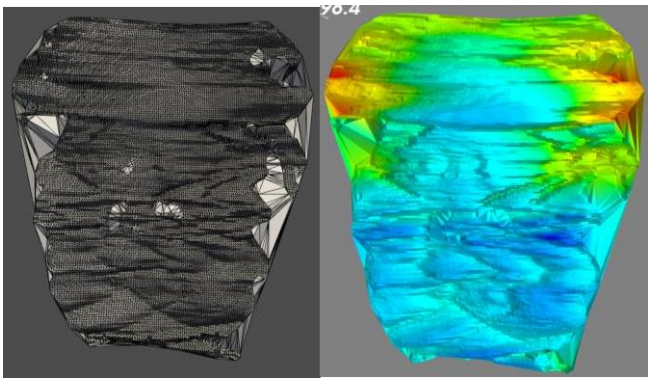


Figura 60. Reconstrucción del rostro a aproximadamente 60 cm con filtro de media, ventana=4.

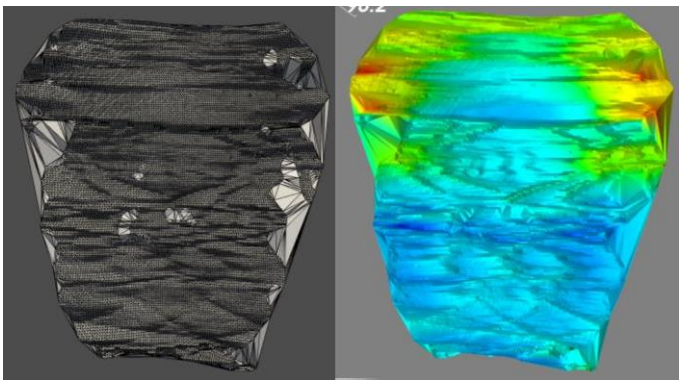


Figura 61. Reconstrucción del rostro a aproximadamente 60 cm con filtro de media, ventana=5.

4.5 Filtro gaussiano 80 cm

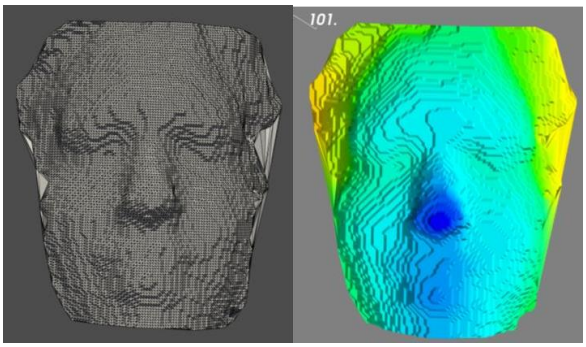


Figura 62. Reconstrucción del rostro a aproximadamente 80 cm sin filtro.

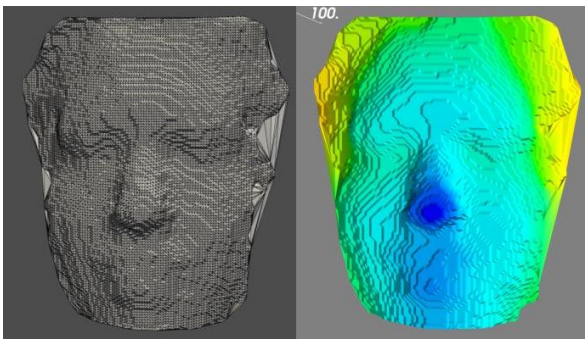


Figura 63. Reconstrucción del rostro a aproximadamente 80 cm con filtro gaussiano, sigma=0.3.

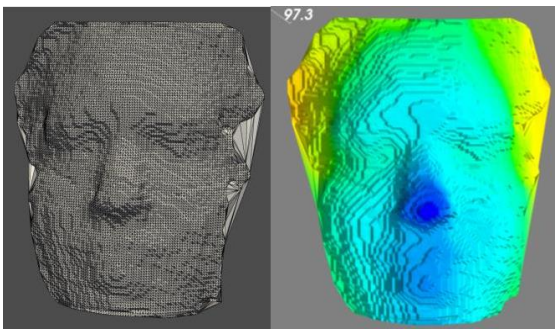


Figura 64. Reconstrucción del rostro a aproximadamente 80 cm con filtro gaussiano, sigma=0.4.

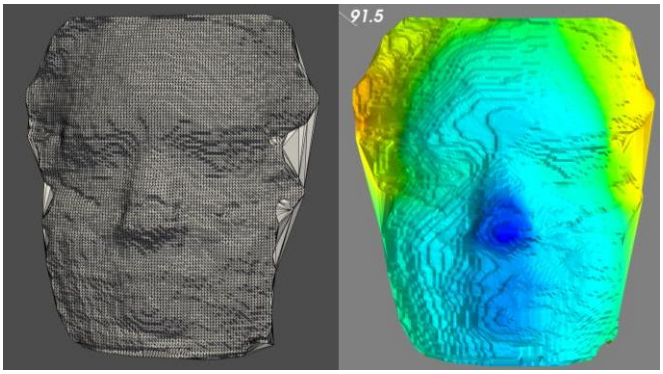


Figura 65. Reconstrucción del rostro a aproximadamente 80 cm con filtro gaussiano, sigma=0.5.

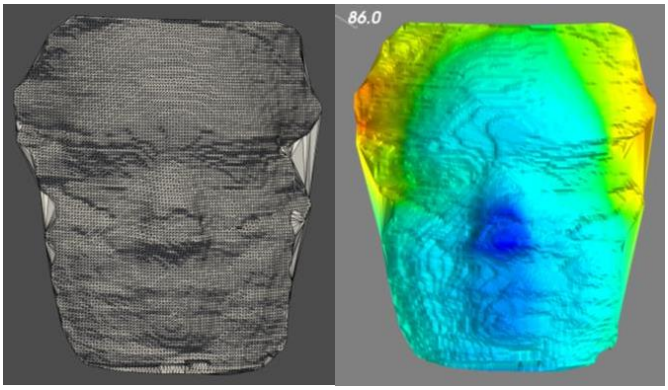


Figura 66. Reconstrucción del rostro a aproximadamente 80 cm con filtro gaussiano, sigma=0.6.

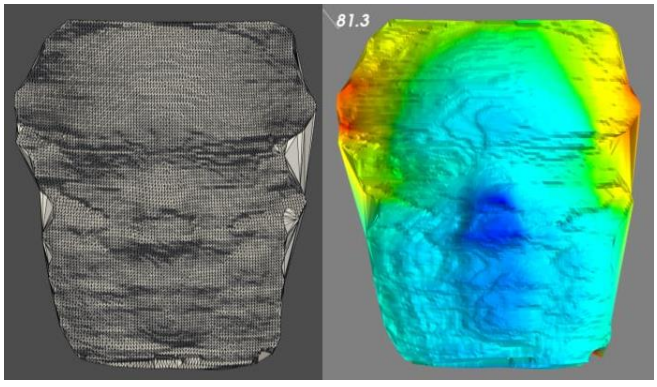


Figura 67. Reconstrucción del rostro a aproximadamente 80 cm con filtro gaussiano, sigma=0.7.

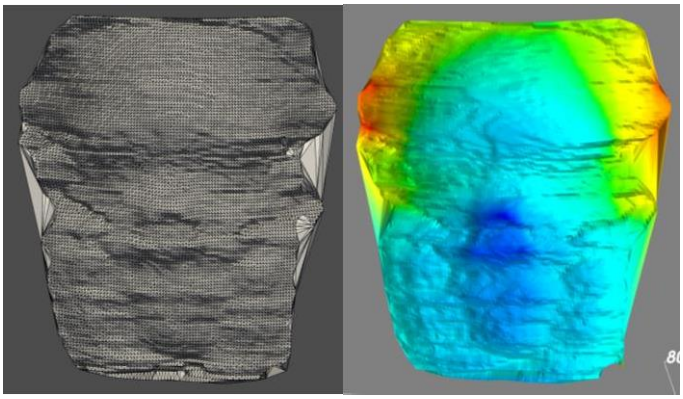


Figura 68. Reconstrucción del rostro a aproximadamente 80 cm con filtro gaussiano, sigma=0.8.

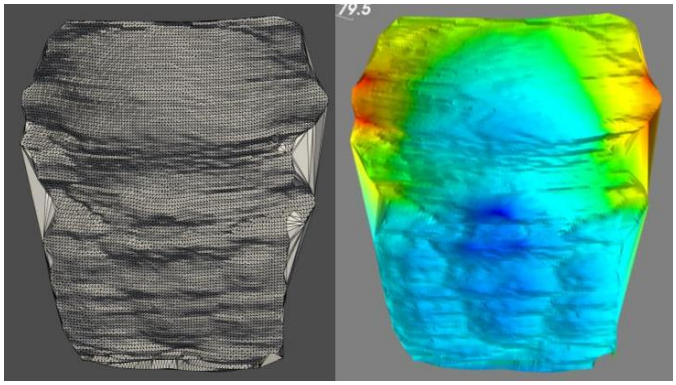


Figura 69. Reconstrucción del rostro a aproximadamente 80 cm con filtro gaussiano, sigma=1.

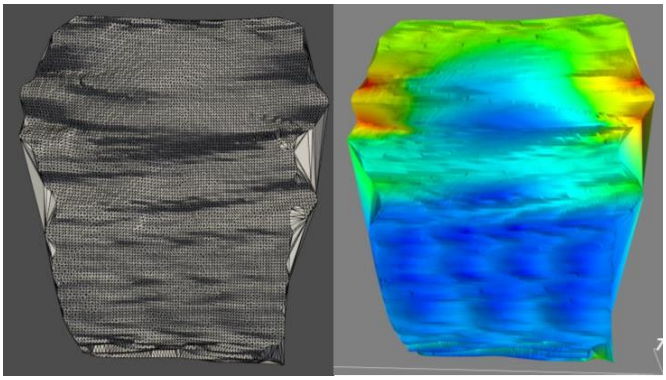


Figura 70. Reconstrucción del rostro a aproximadamente 80 cm con filtro gaussiano, sigma=2.

4.6 Filtro de Media 80 cm

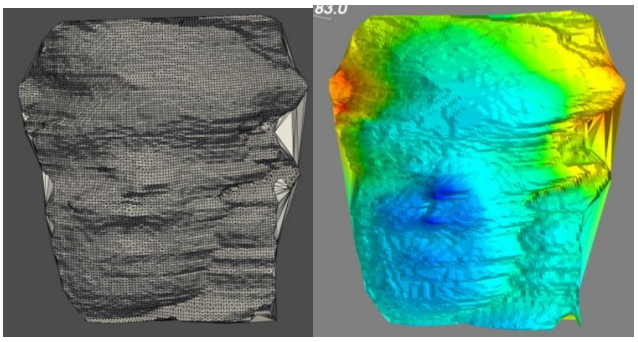


Figura 71. Reconstrucción del rostro a aproximadamente 80 cm con filtro de media,

ventana=2.

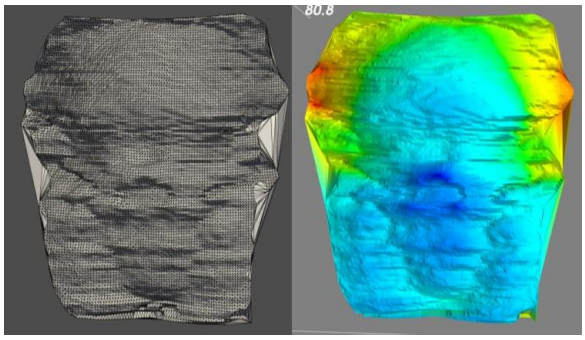


Figura 72. Reconstrucción del rostro a aproximadamente 80 cm con filtro de media, ventana=3.

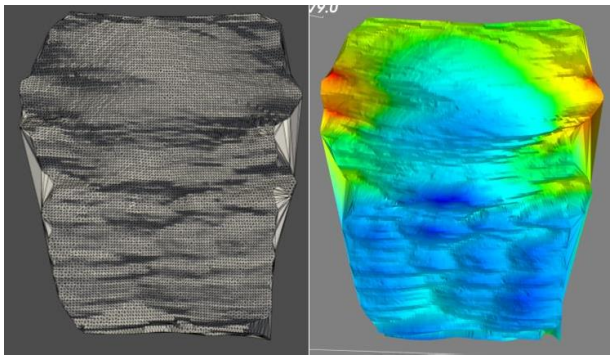


Figura 73. Reconstrucción del rostro a aproximadamente 80 cm con filtro de media, ventana=4.

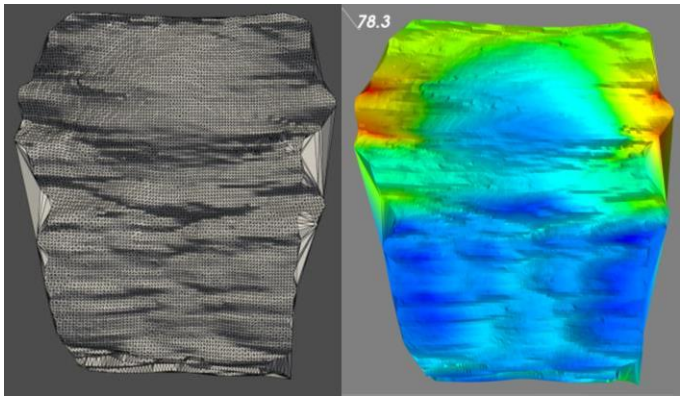


Figura 74. Reconstrucción del rostro a aproximadamente 80 cm con filtro de media, ventana=5.

En las figuras 47-71 podemos observar los cambios de los diferentes filtros y cómo se comportan al cambiar los parámetros de filtrado, claramente observamos que el filtro gaussiano al trabajar con valores decimales y muy pequeños genera un suavizado leve. Por otro lado, el filtro de media no logra suavizar las texturas sin perder la forma de la imagen.

5. Conclusiones y Recomendaciones

5.1 Conclusiones

Se concluye que la reconstrucción facial 3D por medio de cámaras RGB-D tiene un excelente procesamiento y poco uso de recursos, pues las cámaras actuales entregan la información procesada en cortos periodos de tiempo. A diferencia de las reconstrucciones a partir de imágenes estas necesitan una infraestructura muy compleja, con las cámaras RGB se pueden realizar la reconstrucción en un computador promedio.

El algoritmo de haar-cascade tiene un gran desempeño en sistemas de detección facial pues funciona con alta precisión en tiempo real, lo que permite que se realice de manera eficiente la abstracción del rostro dentro de la imagen para su reconstrucción.

El filtrar la imagen usando una campana de gauss nos ayuda a determinar cuáles son los segmentos en los que existe mayor densidad de datos dentro de la nube de puntos y podemos eliminar los puntos de ruido o falsos positivos que puede dar la cámara, así como realizar el desface de la imagen para poder observarla con claridad.

El filtro Gaussiano nos ayuda a suavizar los bordes de la imagen ya que, al poder asignar valores de desviación estándar bajos, filtra la imagen sin cambiar o eliminar las características principales del rostro. Por otra parte, este filtro no devuelve una superficie muy lisa y en valores altos de desviación estándar, el rostro pierde forma.

El filtro de Media “Mean Filtering” devuelve superficies más suavizadas y elimina las capas de niveles creada por el muestreo de la imagen, pero altera completamente la imagen puesto que el rostro no es una superficie regular. Por lo que no se recomienda utilizar en reconstrucciones del rostro.

La cámara de profundidad Orbbec Astra Pro tiene una gran precisión al capturar las nubes de puntos y su rápido procesamiento puede devolver una nube con gran cantidad de información. Su sensor RGB no tiene un enfoque rápido en

exteriores, en cambios extremos de luz se puede perder totalmente la imagen RGB por lo que no se recomienda para uso en exteriores con mucha luz.

La reconstrucción facial por medio de cámaras RGB-D es una herramienta que será utilizada a futuro para sistema de seguridad, puesto que no necesita un alto procesamiento y las cámaras que existen en el mercado siguen evolucionando, disminuyendo su tamaño y aumentando su capacidad de procesamiento, lo que las hace fáciles de implementar en cualquier lugar.

5.2 Recomendaciones

No utilizar la cámara Orbbec Astra Pro en ambientes de extrema luz o con fondos reflectivos, puesto que, en extrema luz la imagen RGB se va a perder y con ello no existirán puntos de coordenadas del reconocimiento facial poder realizar la abstracción del rostro en 3D. Los fondos reflectivos como espejos causan el fenómeno de multi trayecto del sensor infrarrojo, esto puede afectar a la medición creando muchos puntos de interferencia o falsos positivos dentro de la imagen

Revisar los paquetes de dependencias de las librerías antes de instalaras, existen varias librerías como es el caso de “Mayavi” la cual, al instalarlo en un Python totalmente limpio no va a funcionar puesto que tienen varias dependencias. Anaconda es la alternativa más eficiente para este tipo de librerías, pues se encuentra precargada con varios paquetes que ayudarán a que la instalación de librerías sea mucho más sencilla.

Revisar la documentación de “Openni” para la conexión de la cámara con Python, dentro de esta librería se encuentran todos los comandos que se pueden utilizar y ciertos ejemplos que ayudarán a la solución de errores.

Referencias

Amenta, N., Bern, M., & Kamvysselis, M. (1998). A new voronoi-based surface reconstruction algorithm. *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1998*, 415–422. https://doi.org/10.1145/280814.280947

Asenjo, S. (n.d.). Microsoft permitirá a los desarrolladores publicar apps para Kinect en la Windows Store. https://wwwhatsnew.com/2014/10/23/microsoft-permitira-a-los- desarrolladores-publicar-apps-para-kinect-en-la-windows-store/

ASUS. (n.d.). Xtion PRO LIVE. https://www.asus.com/3D-Sensor/Xtion_PRO_LIVE/ Basu, A., & Li, X. (1993). Computer Vision: Systems, Theory and Applications. WORLD SCIENTIFIC. https://doi.org/10.1142/2021

Bates, M. (2013). Interfacing PIC Microcontrollers (2nd ed.). Newnes.

Bledsoe, W. W., & Browning, I. (1959, Dcember). Pattern recognition and reading by machine.In Papers presented at the December 1-3, 1959, eastern joint IRE-AIEE-ACM computer conference (pp. 225-232).

Campana de Gauss. (n.d.). https://www.master-malaga.com/master-mba/campana-de-gauss/ Davies, E. R. (2012). Computer and Machine Vision:Theory, Algorithms, Practicalities (4th ed.).

Elgendy, M. (2019). Deep Learning for Vision Systems.

Enthought Inc. (n.d.). Mayavi: 3D scientific data visualization and plotting in Python. https://docs.enthought.com/mayavi/mayavi/

Espectro Luz Visible. (n.d.). https://www.experimentoscientificos.es/espectro-electromagnetico/espectro-luz-visible/

Feng, Y., Wu, F., Shao, X., Wang, Y., & Zhou, X. (2018). Joint 3d face

reconstruction and dense alignment with position map regression network. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 11218 LNCS, 557–574. https://doi.org/10.1007/978-3-030-01264-9_33

García, E. M. i. (2002). Visión Artificial. Inteligencia Artificial, 115.

Gecer, B., Ploumpis, S., Kotsia, I., & Zafeiriou, S. (2019). Ganfit: Generative adversarial network fitting for high fidelity 3D face reconstruction. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2019-June, 1155–1164. https://doi.org/10.1109/CVPR.2019.00125

Getreuer, P. (2013). ASurvey of Gaussian Convolution Algorithms. Image Processing On Line, 3, 286–310. https://doi.org/10.5201/ipol.2013.87

Hartmann, E. (2003). Geometry and Algortihms for Computer Aided Design. Journal of Computational Physics, 212(October), 393–399. http://www.sciencedirect.com/science/article/pii/S0021999105003736%5Cnhttp://www.mat hematik.tu-darmstadt.de/~ehartmann/cdgen0104.pdf

Home of RF and Wireless Vendors and Resources. (n.d.). https://www.rfwireless- world.com/Terminology/Advantages-and-Disadvantages-of-Infrared-Sensor.html

Hunter, J., Dale, D., Firing, E., & Droettboom, M. (2019). Matplotlib Release 3.1.1 John Hunter, Darren Dale, Eric Firing, Michael Droettboom and the. https://matplotlib.org/Matplotlib.pdf

Intel. (n.d.). Depth Camera D415. https://www.intelrealsense.com/depth-camera-d415/

Introduction to Digital Images. (n.d.).

https://web.stanford.edu/class/cs101/image-1-introduction.html

Jackson, A. S., Bulat, A., Argyriou, V., & Tzimiropoulos, G. (2017). Large Pose 3D Face Reconstruction from a Single Image via Direct Volumetric CNN Regression. Proceedings of the IEEE International Conference on Computer Vision, 2017-Octob, 1031–1039. https://doi.org/10.1109/ICCV.2017.117

Levoy, M., & Whitted, T. (1985). The Use of Points as a Display Primitive Marc Levoy Turner Whitted. North, 1–13.

Liu, F., Zhu, R., Zeng, D., Zhao, Q., & Liu, X. (2018). Disentangling Features in 3D Face Shapes for Joint Face Reconstruction and Recognition. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 5216–5225. https://doi.org/10.1109/CVPR.2018.00547

Malik, A. S., Choi, T. S., & Nisar, H. (2012). Depth Map and 3D Imaging Applications. IGI Global. https://doi.org/10.4018/978-1-61350-326-3

Marton, Z. C., Rusu, R. B., & Beetz, M. (2009). On fast surface reconstruction methods for large and noisy point clouds. 2009 IEEE International Conference on Robotics and Automation, 3218–3223. https://doi.org/10.1109/ROBOT.2009.5152628

Mullen, T. (2009). Mastering Blender (1st ed). Wiley Publishing. OpenCV team. (2019). About OpenCV. https://opencv.org/about/ OpenNI2 Coding Instructions. (n.d.). 1–14.

ORBBEC. (n.d.). Astra Series. https://orbbec3d.com/product-astra-pro/

Pengfei, D., Shishir, K. S., Ioannis, & A, K. (2017). End-to-end 3D face reconstruction with deep neural networks. 78–87.

Peyre, G. (2010). Color Image Processing. http://www.numerical-tours.com/matlab/multidim_1_color/

PrimeSense. (2013). PrimeSenseTM 3D Sensors. 4. http://www.i3du.gr/pdf/primesense.pdf Rein-Lien Hsu, Abdel-Mottaleb, M., & Jain, A. K. (2002). Face detection in color images. IEEE

Rein-Lien Hsu, Abdel-Mottaleb, M., & Jain, A. K. (2002). Face detection in color images. IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(5), 696–706. https://doi.org/10.1109/34.1000242

Roth, J., Tong, Y., & Liu, X. (2017). Adaptive 3D Face Reconstruction from Unconstrained Photo Collections. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(11), 2127–2141. https://doi.org/10.1109/TPAMI.2016.2636829

Rusinkiewicz, S., & Levoy, M. (2000). QSplat. Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH ’00, 343–352. https://doi.org/10.1145/344779.344940

Sivaram, M., Porkodi, V., Mohammed, A. S., & Manikandan, V. (2019). Detection of accurate facial detection using hybrid deep convolutional recurrent neural network. 1844–1850. https://doi.org/10.21917/ijsc.2019.0256

Smith, M., Paron, P., & Griffiths, J. (2011). Geomorphological Mapping, Volume 15.

Szeliski, R. (2004). Computer vision. Computer Vision: Algorithms and Applications, 43-1-43– 23. https://doi.org/10.4324/9780429042522-10

The SciPy community. (2020). What is NumPy? https://numpy.org/doc/stable/user/whatisnumpy.html

Tjaden, H., Schwanecke, U., Schomer, E., & Cremers, D. (2019). A Region-Based Gauss- Newton Approach to Real-Time Monocular Multiple Object Tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence, 41(8), 1797–1812.

https://doi.org/10.1109/TPAMI.2018.2884990

Tran, A. T., Hassner, T., Masi, I., Paz, E., Nirkin, Y., & Medioni, G. (2018). Extreme 3D Face Reconstruction: Seeing Through Occlusions. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 3935–3944. https://doi.org/10.1109/CVPR.2018.00414

Turk, M. A., & Pentland, A. P. (1991, January). Face recognition using eigenfaces. In Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (pp. 586-587).

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1. https://doi.org/10.1109/cvpr.2001.990517

Walker, H. (1931). Studies in the History of the Statistical Method.

Wang, Z., Zhang, X., Yu, P., Duan, W., Zhu, D., & Cao, N. (2020). A New Face Recognition Method for Intelligent Security. Applied Sciences, 10(3), 852.

Zhang, J., Li, K., Liang, Y., & Li, N. (2017). Learning 3D faces from 2D images via Stacked Contractive Autoencoder. Neurocomputing, 257, 67–78. https://doi.org/10.1016/j.neucom.2016.11.062

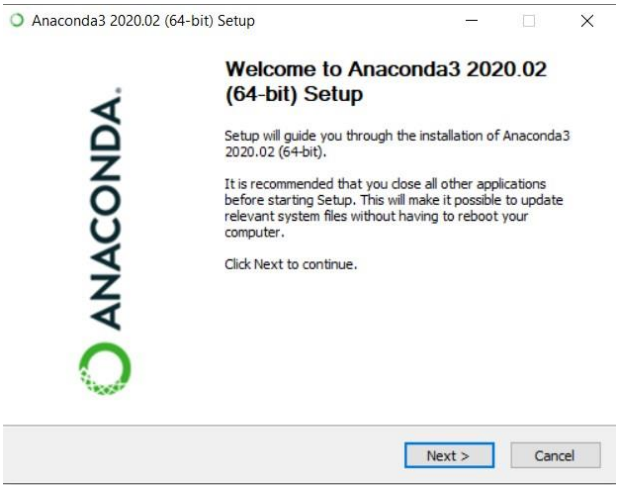
ANEXOS

Anexo 1

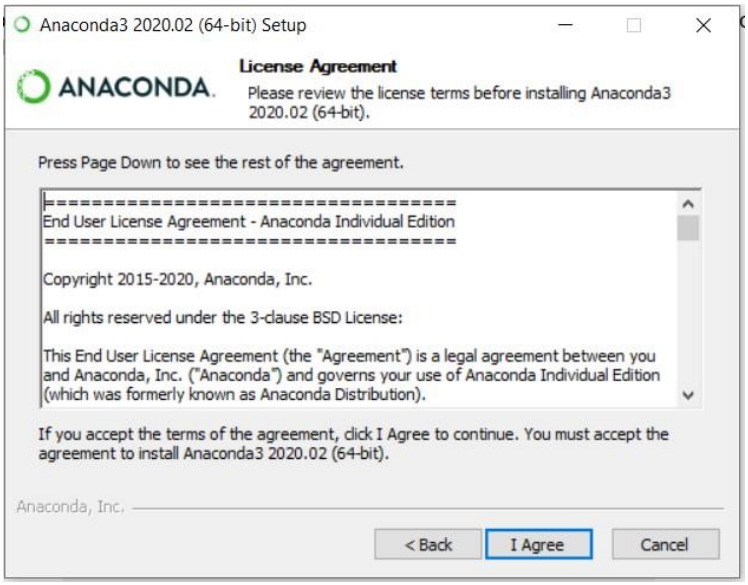
Instalación de Python Anaconda 3.7

1) Ingresar a la Página Web: https://www.anaconda.com/products/individual

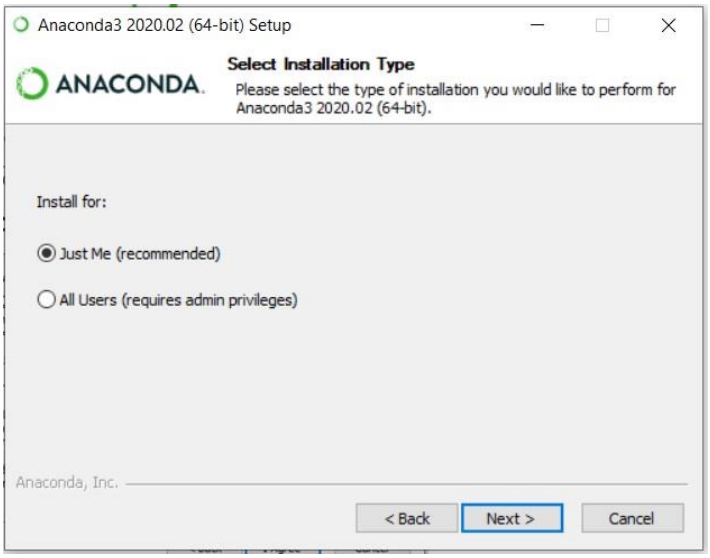
2) Seleccionaremos la distribución dentro de la cual instalaremos Python, para este caso será Windows de 64 bits.



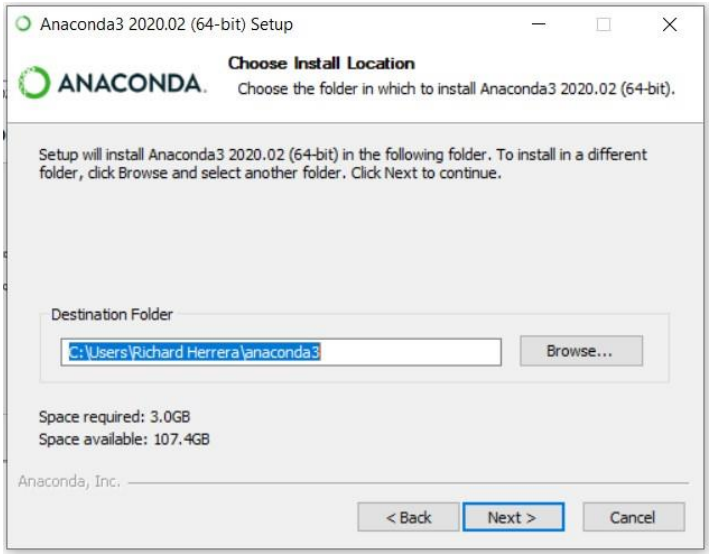
3) Una vez descargado el archivo, ejecutamos el instalador y aceptamos la licencia



4) Instalamos para uno o varios usuarios del equipo.

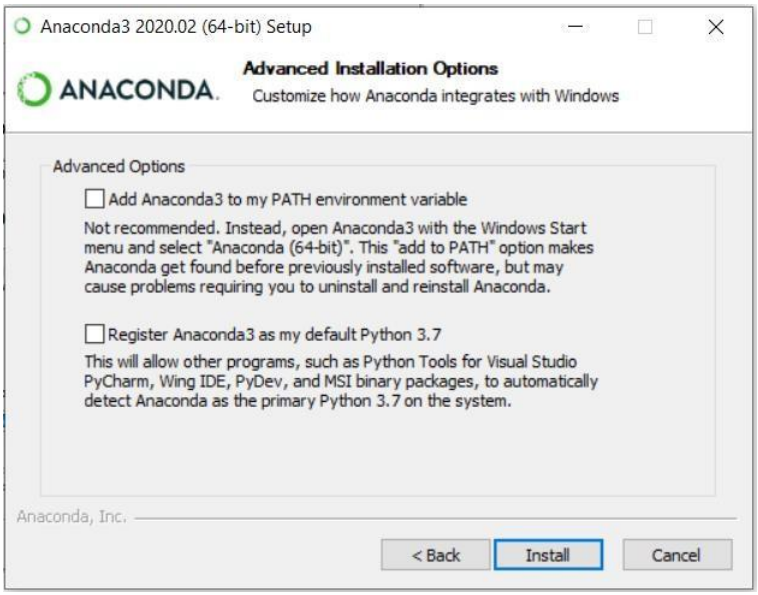


5) Seleccionamos el destino donde se instalará Anaconda



El paquete tiene un peso mínimo de 3GB por lo que se debe verificar si existe espacio suficiente en el disco.

6) Seleccionamos a Anaconda como el editor por defecto de Python y procedemos a la instalación



Anexo 2

Lista de Paquetes:

a) OpenCV
b) Numpy
c) PrimeSense
d) ScyPy

e) Pyvista

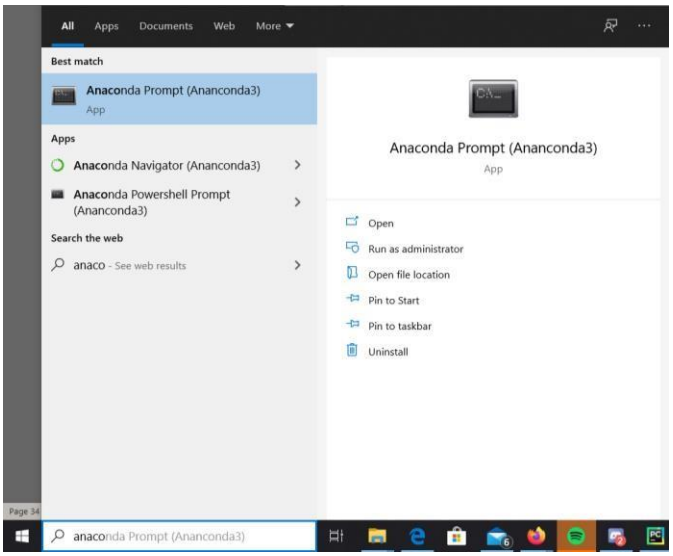
f) Matplotlib

g) Mayavi

Instalación de Paquetes:

a) OpenCV

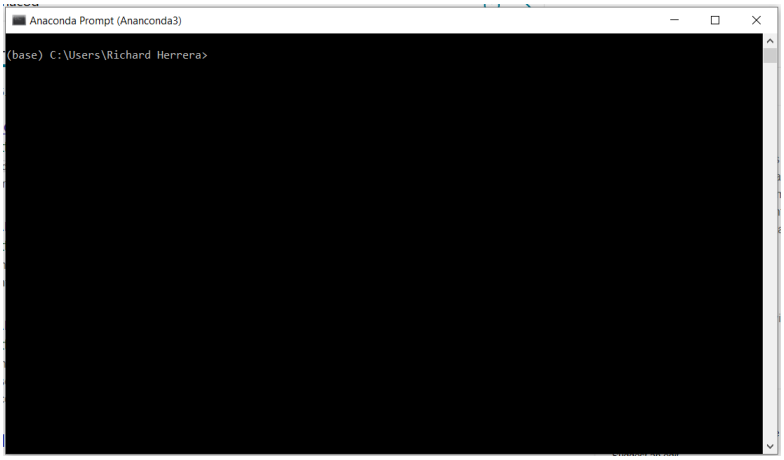
1) Abrimos Anaconda Prompt



2) Escribimos los siguientes comandos:

Pip install opencv-python==3.4.2.17

pip install opencv-contrib-python==3.4.2.17



b) Instalar Numpy

1) Dentro del Prompt de anaconda escribimos el siguiente comando:

conda install -c anaconda numpy

c) Instalar Primesense

   1) Dentro del Prompt de anaconda escribimos el siguiente comando:
      pip install primesense

d) Instalar ScyPy

   1) Dentro del Prompt de anaconda escribimos el siguiente comando:
      conda install -c anaconda scipy

e) Instalar Pyvista

   1) Dentro del Prompt de anaconda escribimos el siguiente comando:
      conda install -c conda-forge pyvisa

f) Intalar Matplotlib

   1) Dentro del Prompt de anaconda escribimos el siguiente comando:
      pip install -U pip
      pip install -U matplotlib